Вятский государственный гуманитарный университет

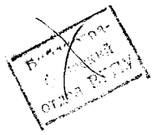
ВЕСТНИК ВЯТСКОГО ГОСУДАРСТВЕННОГО ГУМАНИТАРНОГО УНИВЕРСИТЕТА

Информатика

Научно-методический журнал

Киров
2002

ББК 32.81



Выпуск готовили: С.М. Окулов, В.А. Онегов, М.В. Петухова, Е.Н. Колодкина

Материалы выпуска подготовлены при поддержке
Министерства образования Российской Федерации, проект Г00-4.1-60

Адрес редакции: 610002, г. Киров, ул. Ленина, 111,
тел.: (8332) 678-860 (научный отдел), (8332) 673-674 (издательство)

Редакторы: Т. Котельникова, Ю. Болдырева, О. Коробкова
Компьютерная верстка: О. Редькина

Подписано в печать 8.07.2002 г. Формат 60x84 1/8.
Бумага офсетная. Гарнитура Таймс. Печать офсетная.
Усл. печ. л. 13,95. Тираж 500. Заказ 2062.

Отпечатано с готового оригинал-макета
в КОГУП "Кировская областная типография"

ISBN 5-93825-025-0

СОДЕРЖАНИЕ

Окулов С.М. Об особенностях подготовки учителей информатики 5

ИНФОРМАТИКА В ВУЗЕ

Окулов С.М. Методологические проблемы образовательной информатики (полемические заметки) 7
Окулов С.М. Образовательные стандарты высших учебных заведений по информатике 15
Онегов В.А. Теоретическая информатика и математика 20
Бушмелева Н.А. Курс «Вычислительная геометрия с элементами компьютерной графики» 23
Бабушкина И.А. Проблемы внедрения достижений Computer Science в области технологий программирования в учебный процесс 24
Чеканов Д.А. Специализация «Компьютерные сети» в программе обучения информатике 26
Разова Е.В. Курс «Теория чисел и ее приложения» 28
Ашихмина Т.В. Методика преподавания темы «Динамические структуры данных» 31
Онегов В.А., Овсянникова М.В. Исследование операций. Проблемы и особенности современного преподавания 32
Петухова М.В. Преподавание информатики на гуманитарных факультетах педвуза 34
Волченков С.Г. Две задачи по информатике из теории чисел 37
Корчемкина О.Н. Организация учебной практики по информатике студентов экономических специальностей 38
Ямбарышева С.Ю. Использование информационных технологий при изучении курса «Численные методы» 40
Андреева Т.Н. Информационные технологии обработки табличных данных 45

ИНФОРМАТИКА В ШКОЛЕ

Юферев В.В., Корякина Г.К. Компьютерная среда, новые горизонты и новые измерения. Образовательная информатика в физико-математическом лицее г. Кирова 47
Ведерникова Е.В.. Преподавание информатики в физико-математическом лицее г. Кирова 49
Андреева Е.В., Фалина И.Н. Преподавание информатики в СУНЦ МГУ 51
Белиовская Л.Г. Эвристические технологии в работе с интеллектуально одаренными детьми лицея № 1557 г. Москвы 56
Васенина Е.А. Цели изучения информатики в школе и учебники «новой волны» 59
Матюхин В.А. Автоматизация проверки решений школьников при изучении программирования 65

ИНФОРМАТИКА И ЯЗЫК

Колодкина Е.Н. Экспериментальное параметрическое исследование конкретности и образности в структуре значения слова 68
Вахрушев А.С. Особенности преподавания интегрированных дисциплин по информатике на английском языке 70
Кипрская Е.В. Экспериментальное исследование стратегий идентификации русскоязычных эвфемизмов 71
Колодкина Е.Н., Христолюбова Н.С. Психолингвистическое исследование паронимии 73
Свищова А.А. Эпитет как эмоционально-экспрессивное средство в романе В.С. Моэма «Луна и грош» 74
Бардовская А.И. Психолингвистическое исследование синестетических прилагательных современного английского языка 76
Шейдуллина Г.К. Стимулирование речевой деятельности студентов, развитие профессионально-педагогических навыков и умений (на примере темы «Traits of character. Difficult Children»)..... 78
Федоровская Т.Н., Павлова Н.В. Лексические архаизмы и их стилистические функции (на примере произведений Э.А. По) 81

СТУДЕНЧЕСКАЯ НАУКА

Иванов С.Ю. Компьютерная арифметика: извлевение квадратного корня из многоразрядного числа 84
Ронжин А.Н. Методы поиска ошибок в программном коде с использованием директив компилятора 85
Шихов В.В. Системы управления сетью 88
Козвонина А.В. Об использовании информационных технологий в преподавании курса «Теория вероятностей и математическая статистика» 91
Калганников Е.Ю., Калганников Ю.Е. Система автоматизации документооборота школы 98
Бояринцев Д.В., Московкин А.А., Окулов С.М. О методике тестирования программных решений 101
Перевозчиков М.В. Об одной задаче по информатике 108

Сведения об авторах 111

О факультете информатики ВГТУ 112

С.М. Окулов

ОБ ОСОБЕННОСТЯХ ПОДГОТОВКИ УЧИТЕЛЕЙ ИНФОРМАТИКИ

Информационные технологии, внедряемые во все сферы жизни, определяют уровень и перспективы развития постиндустриального общества. При этом, естественно, роль образовательной информатики как структуры, обеспечивающей подготовку кадров для такого общества, становится просто огромной, если так можно выразиться, основополагающей, от которой зависит, быть государству «со своим лицом» или «тащиться» вслед за развитыми социальными структурами.

Становление системы подготовки кадров по информатике для школы М.П. Лапчик разбивает на три этапа.[1] Первый этап (до введения курса ОИВТ в школы) связывается с работой на математическом факультете Свердловского педагогического института. Под руководством В.Г. Житомирского (1934-1988) был выполнен пионерский проект по разработке концепции многофункционального специалиста – организатора учебного процесса на базе ЭВМ. На втором этапе (до введения стандартов и первое поколение стандартов) подготовка специалистов для школы осуществлялась на основе учебных планов специальностей «Математика, с дополнительной специализацией информатика» и «Физика, с дополнительной специализацией информатика». Решающий вклад в их разработку внес М.П. Лапчик. Первый выпуск учителей информатики в Омском педагогическом институте был сделан в 1988 году. Заложенные в этой системе подготовки учителей информатики идеи послужили основой для разработки стандарта первого поколения. В связи с информатизацией общества уже с начала 90-х годов прошлого столетия велись исследования по совершенствованию профессиональной подготовки учителей информатики (М.П. Лапчик, Э.И. Кузнецов, С.А. Жданов, И.П. Моисеев, М.Н. Швецкий, А.В. Могилев, Н.И. Пак и др.). Введение профильной подготовки по специальности «Информатика» и, соответственно, создание и развитие в педвузах профильных факультетов информатики – результат этой работы. В 1994 году специальность «Информатика» впервые была включена в область знания «Образование» утвержденного Госкомвузом России Классификатора направлений и специальностей высшего профессионального образования.[2] Третий этап – второе поколение стандартов, принятых в 2000 году. Сравнение стандарта по специальности «*030100 – Информатика*» (квалификация выпускника – учитель информатики) с основополагающими документами (стандартами) из других областей знания и отчетом по учебному плану информатики, разработанным специальной объединенной комиссией, образованной Ассоциацией *ACM (Association of Computing Machinery)* и профессиональным объединением *IEEE (Institute of Electrical and Electronics Engineers)*, показывает, что первые по содержанию подготовки не уступают, а во многом и превосходят. Разумное использование часов из блоков «Дисциплины специализации» и «Курсы по выбору» позволяет придать подготовке требуемые динамизм и качество, соответствующие образовательной области «Информатика».

Первый выпуск учителей математики с дополнительной специальностью учитель информатики был сделан в ВятГПУ в 1991 году. В 2001 году на факультете информатики ВятГПУ из университета вышли молодые люди со специальностью «учитель информатики и английского языка». *Особенности подготовки.* Блок дисциплин по информатике дополнен (за счет дисциплин специализации и курсов по выбору) значительным количеством часов по сетевым технологиям и программированию под Windows. Это позволило значительно усилить уровень социальной защищенности нашего выпускника. Он с успехом, как показало распределение выпускников, может работать не только в образовательной сфере, но и других сферах деятельности, например системным администратором в различных организациях, оснащенных компьютерами. Блок филологических дисциплин по второй специальности позволяет эффективно работать нашему выпускнику в сфере внеклассной и внешкольной работы «на стыке» двух предметных областей (например, организация работы школьников в учебно-исследовательских проектах, реализуемых через глобальные телекоммуникационные сети, организация интернациональных компьютерных клубов и т.п.). Если наш выпускник не будет востребован образова-

[1] Лапчик М.П. Структура и методическая система подготовки кадров информатизации школы в педагогических вузах. Диссертация в виде научного доклада на соискание степени доктора педагогических наук. – М., 1999.

[2] Государственный образовательный стандарт высшего профессионального образования. Классификатор направлений и специальностей высшего профессионального образования (приложение к приказу Госкомвуза России от 5 марта 1994 г. №180).

нием, то он может с успехом выполнять работу, например, менеджера по международным связям любой организации. Проблемы языкового барьера нет, а совокупность дисциплин психолого-педагогической направленности, естественно, обеспечивает и коммуникабельность, и умение общаться и т.д. Кроме того, в информатике подавляющее число терминов и понятий построены на английской лексике и, более того, семантически связаны с английским языком, само сочетание «информатика – английский» оказывает значительное влияние на эффективность изучения как одной, так и другой предметной области. Можно считать, что такое сочетание естественнонаучного и гуманитарного направлений подготовки есть образование XXI века или возврат к классическому университетскому образованию XIX века.

В данном выпуске в основном отражена работа сотрудников факультета по совершенствованию учебного процесса на факультете (первый раздел). В статьях из второго раздела рассматривается школьная информатика. Она представлена, как работами преподавателей факультета, так и трудами коллег из других вузов и школ. Третий раздел состоит из исследований преподавателей кафедры германских языков. И, наконец, в четвертом разделе («Студенческая наука») – статьи студентов факультета. Статьи разноплановые, но они в полной мере отражают разностороннюю подготовку наших студентов.

ИНФОРМАТИКА В ВУЗЕ

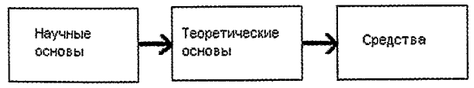
С.М. Окулов

МЕТОДОЛОГИЧЕСКИЕ ПРОБЛЕМЫ ОБРАЗОВАТЕЛЬНОЙ ИНФОРМАТИКИ (полемические заметки)*

В обществе на данном этапе его развития происходят глубокие изменения, связанные с глобальной информатизацией всех сфер человеческой деятельности. При этом, несмотря на все проблемы становления, роль образовательной информатики как механизма подготовки человека к этой деятельности с каждым годом возрастает. Перед образовательной информатикой стоят новые задачи. Ряд из них.

1. Реальная (не образовательная) информатика стремительно развивается. Появляются новые компьютерные специальности, информатикой охватываются все новые сферы деятельности человека. Отражение этих процессов в образовательной информатике – открытый вопрос.

2. Статус и роль информатики в общеобразовательной школе по-прежнему не ясны. Почему? Если не рассматривать управленческие решения, то причина может быть в том, что специалисты по образовательной информатике не могут найти единого понимания научных и теоретических основ информатики. Подтверждением этого факта является содержание большинства современных учебников по информатике. В основном они посвящены средствам информатики, а последние являются лишь третьим составляющим звеном в логической цепочке любого предмета.



3. Если значимость, например, таких общеобразовательных предметов, как математика и физика, в развитии и становлении мышления человека определена, то место информатики (компьютера) в этом ряду не ясно. Задерживает она или развивает мышление школьника? Если развивает, то как, есть ли специфические особенности в этом процессе?

Попытаемся проанализировать, как наука («теория и методика обучения информатике») отвечает на этот вызов, и начнем его с обзора исследований последних лет.

1. *Е.В. Баранова* исследует «теорию и практику объектно-ориентированного проектирования содержания обучения средствам информационных технологий» [1].

Логика работы.

• Анализируются «информационные, социальные и педагогические основы применения формальных методов для проектирования процесса обучения информационным технологиям».

• Разрабатываются теоретические основы «методологии объектно-ориентированного проектирования содержания обучения средствам информационных технологий».

• Проводится объектно-ориентированное проектирование содержания обучения информационным технологиям (Excel, Access, Delphi), приводятся практические результаты «использования объектных моделей средств информационных технологий в процессе обучения».

2. *А.Г. Гейн* считает, что «изучение информационного моделирования» является средством «реализации межпредметных связей информатики с дисциплинами естественнонаучного цикла» [4].

Логика работы.

• Рассматриваются главные методологические положения, являющиеся основой обучения информатике, обеспечивающие реализацию межпредметных связей с дисциплинами естественнонаучного цикла. Делается вывод о том, что «реализация общеобразовательного потенциала курса информатики, включающего в себя целый ряд мировоззренческих и методологических аспектов, требует при его разработке учета следующих положений: освоение учащимися основ информационного и, в частности, компьютерного моделирования является одним из ведущих компонентов курса; информационное моделирование служит принципиальным связующим звеном между информатикой и другими школьными дисциплинами, в первую очередь, естественнонаучного цикла; изучение информационного моделирования не исключает, а поддерживает линию освоения алгоритмического мышления при общей тенденции к снижению доли элементов программирования в общеобразовательном курсе информатики».

• «Конструируется модель содержания курса информатики», состоящего из пяти важных разделов, частично перекрывающих друг друга по времени изучения (в рамках 10-летней школы): «компьютерная грамотность (начальная школа); алгоритмизация (ориентировочно с 3-го по 7-й классы); объектно-ориентированный подход к информационному моделированию (ориентировочно с 6-го по 8-й классы); логический подход к информационному моделированию (ориентировочно в 8-9-х классах); системы компьютерного моделирования (начиная с 9-го класса)».

• На основе разработанной концепции конструируется содержание курса информатики. Считается, что «для эффективной реализации межпредметных связей необходимо создание единой информационно-моделирующей среды, овладение которой (в том числе как средством обучения) является одной из состав-

ляющих курса информатики». Одним из компонентов единой информационно-моделирующей среды целесообразно иметь «иерархическую систему развивающихся исполнителей, обеспечивающих освоение учащимися базовых понятий информатики для всех трех подходов: алгоритмического, объектно-ориентированного и логического».

• Разрабатывается методика обучения, вытекающая из таких иерархически выстроенных компонентов системы обучения, как цели, содержание и средства обучения.

3. *И.Б. Готская* исследует «методическую систему обучения информатике студентов педвузов в условиях рыночной экономики» [6].

Логика работы.

• Дается анализ современной концепции преподавания информатики в педагогических университетах и теоретические основы ее реализации. Обосновывается необходимость в формировании «авторской концепции обучения информатике студентов педвузов в условиях рыночной экономики на основе комплексного использования методологий системного и маркетингового подходов».

• Определяется «процедура теоретических исследований методической системы обучения информатике в условиях рыночной экономики». Выделяются и описываются три главные группы потребностей обучаемых в знаниях по информатике (социально-индивидуальные, корпоративно-индивидуальный и индивидуальные). Рассматривается логическая закономерность *потребность* ⇒ *цель* ⇒ *исполнительная система* ⇒ *результат*, выполняющая в дальнейшем структурирующую функцию для вычленения компонентов, подсистем, элементов методической системы обучения информатике.

• Дается послойное (на трех уровнях) описание методической системы обучения информатике, а именно: морфологическое, макро- и микрооописание.

• Осуществляется маркетинговое проектирование методической системы обучения информатике.

4. *Т.В. Добудько* исследует «формирование профессиональной компетентности учителя информатики в условиях информатизации образования» [7].

Логика работы.

• Выявляются факторы, обусловливающие содержание и особенности профессиональной деятельности учителя информатики. Считается, что информационные технологии являются доминирующим фактором, непосредственно влияющим на специфику труда учителя информатики. Они выступают на «уроках информатики в качестве объекта изучения; инструмента педагогической деятельности; орудия учебно-познавательной деятельности учащихся».

• Строится «деятельностная модель учителя информатики».

• Анализируется «сущность и структура профессиональной компетентности» учителя информатики. Приводится характеристика выпускника педагогического вуза, успешно завершившего обучение по программе специальности 030100 – Информатика (квалификация – учитель информатики).

• Предлагается «методика формирования профессиональной компетентности учителя информатики. В математической подготовке курс «элементы дискретной математики» заменяется на «математическую логику», а «исследование операций» – на «методы математического моделирования». Структура учебных дисциплин по информатике состоит из введения в информатику; программного обеспечения ЭВМ; элементов теории алгоритмов и теории графов; языков и методов программирования; вычислительной техники и практикума по решению задач.

5. *А.Р. Есаян* исследует «теорию и методику обучения алгоритмизации на основе рекурсии в курсе информатики педагогического вуза» [9].

Логика работы.

• Определяется методологическая и теоретическая роль рекурсии в системе научного знания. Создается научная концепция обучения алгоритмизации на основе рекурсии. «Под рекурсией понимают прием последовательного сведения решения некоторой задачи к решению совокупности более простых задач такого же класса и получение на этой основе решения исходной задачи». Рекурсия в исследовании понимается более широко: «Важнейшей особенностью объекта, которая позволяет назвать его рекурсивным, является наличие в нем некоторых базовых компонентов (составляющих, элементов, подобъектов), доступных для непосредственного изучения (описания, задания, определения, вычисления), и наличие внутренних связей (переходов, преобразований) в множестве всех компонентов, которые позволяют любой из них по некоторой единой схеме (правилу, предписанию, алгоритму) конструктивно выразить через один или несколько базовых компонентов».

• Проектируется и разрабатывается содержание обучения алгоритмизации на основе рекурсии.

• Разрабатывается учебный Web-сайт «Рекурсия в информатике».

• Описывается методика проведения педагогического эксперимента.

6. *А.В. Могилев* исследует «развитие методической системы подготовки по информатике в педагогическом вузе в условиях информатизации образования» [16].

Логика работы.

• Определяется понятие методического контекста обучения информатике, «включающего:

- организационно-методическое обеспечение обучения предмету;

- содержание знаний и актуальной деятельности предметной области, аппаратно-программные средства и технологии информатизации;

- социальный запрос к образованию в форме требований к подготовке по предмету и критериев обученности, а также комплекса мотивов;

- специальную и методическую подготовку педагогических кадров, их методическую поддержку;

- начальную подготовку учащихся и студентов по предмету, элементы информационной культуры, освоенные обществом».

• Пересматривается понятие методической системы обучения информатике. Она считается «мягкой, открытой системой с нечеткими границами, погруженной в методический контекст обучения и интенсивно взаимодействующей с его компонентами».

• Считается, что «цели и содержание обучения информатике должны обеспечивать формирование у учащихся стремления и способностей адаптироваться к быстро меняющейся информационной среде деятельности».

• Обосновываются принципы отбора и структурирования содержания обучения информатике.

• Обосновывается «переход от понятия "средства обучения" к понятию "учебно-профессиональная среда", существенно связанному с целями обучения, а также выступающему в интеграции с организационными формами и методами обучения в структуре методической системы обучения: установлена доминирующая роль этого компонента в методической системе».

• Разрабатывается «организационная форма и метод обучения, отвечающие современным информационным и телекоммуникационным технологическим средам обучения – метод квазивиртуального класса в соревновательных телекоммуникационных проектах».

7. *Н.И. Пак* исследует «нелинейные технологии обучения в курсах информатики и информационных технологий» [17].

Логика работы.

• Рассматриваются тенденции развития образования в условиях информатизации. Определяется понятие открытого образовательного пространства, формирование которого происходит по трем направлениям: создание и развитие телекоммуникационных и компьютерных сетей; информационное наполнение; совместная деятельность по региональным и федеральным проектам.

• Вводится понятие нелинейной технологии обучения и нелинейной модели знаний курсов информатики и информационных технологий. «Нелинейные технологии обучения – это совокупность системных методов моделирования и реализации деятельности преподавателя и деятельности ученика в личностно ориентированной системе образования, базирующихся на принципах активного способа обучения в условиях информатизации и глобальной коммуникации». «Информационные нелинейные образовательные технологии нацеливаются на нелинейную структуризацию информации и знаний в виде гипертекстов, гипермедиа, распределенных баз и банков данных и знаний, размещающихся на серверах и распространяющихся, как правило, на CD-rom».

• Рассматриваются вопросы создания и использования компьютерных средств обучения для нелинейных технологий обучения. Излагаются новые идеи к построению эволюционных, открытых обучающих программно-методических средств в телекоммуникационных системах.

• Нелинейные формы организации обучения с использованием информационных технологий основаны на методе учебных телекоммуникационных проектов. Обобщением практического опыта организации учебного процесса по нелинейным технологиям является разработка модели «Школы будущего» – открытого распределенного лицея информационных технологий (структуры, управления, форм и методов организации учебного процесса).

8. *Н.И. Рыжова* исследует «развитие методической системы фундаментальной подготовки будущих учителей информатики в предметной области» [18].

Логика работы.

• Строится модель развития методической системы. «Развитие системы является следствием ее взаимодействия со средой. Составляющие развития черпаются из среды и "усваиваются" системой после соответствующей обработки, превращающей компоненты среды в компоненты, ценные для системы. Другими словами, любая методическая система обучения функционирует на определенном социальном и культурном фоне, оказывающем на нее решающее воздействие... Принципы, связанные с внутренним строением системы, отражают тот факт, что изменение одного или нескольких компонентов методической системы влечет необходимость изменения всей системы в целом».

• Определяется понятие методической теории, «основными элементами которой являются: эмпирический базис, концептуальный базис, концептуальный каркас, логика теории, содержательная надстройка и интерпретация методической теории».

• Формулируется концепция фундаментализации образования в области информатики. Ее суть – в выделении математических оснований информатики, к которым следует отнести: «основы семиотики, основания математики, основания математической логики, иерархию формальных языков». Утверждается, что в рамках «фундаментальной подготовки будущего учителя информатики при обучении элементам теоретической информатики (а именно ее математическим основаниям) лежит проблема обучения формальному языку».

• Строится содержательная надстройка и интерпретация методической теории обучения информатике в виде содержания курса «теоретические основы информатики» и системы упражнений по математическим основаниям информатики.

9. *И.В. Симонова* исследует «концептуальные модели обучения практико-ориентированных учащихся в условиях интернет-образования» [20].

Логика работы.

• Проводится теоретическое обоснование проблемы исследования, а именно: анализируются подходы к моделированию обучения в условиях интернет-образования; выделяется категория практико-ориентированных учащихся (имеющих трудности в обучении) как феномена дифференциации в обучении; вводится понятие информационного дидактического пространства как источника информации и инструмента формирования системы понятий практико-ориентированных учащихся.

• Строится «модель основных элементов обучения практико-ориентированных учащихся, включающая: моделирование информационного взаимодействия учащегося и учителя; модель формирования синонимических и полисемантически связной системы понятий у практико-ориентированных учащихся в курсе информатики; разработку подхода к построению моделей задач для обучения информатике практико-ориентированных учащихся».

• Дается описание и анализ результатов педагогического эксперимента.

10. *Р. Р. Фокин* исследует «метамодель обучения информатике в высшей школе» [23].

Логика работы.

• Отмечается несоответствие динамизма развития современной информатики и методики обучения информатике. Обосновывается «целесообразность применения метамоделирования на основе теории открытых систем к методике обучения информатике в высшей школе для повышения динамизма последней».

• Разрабатывается «стратифицированная метамодель обучения информатике в высшей школе и содержание ее верхних уровней». Метамодель обучения информатике в высшей школе состоит из четырех уровней: уровня 1 (метауровня); уровня 2 (целевого); уровня 3 (содержательного); уровня 4 (процессуального). Разработано содержание метауровня. «Уровни 1 и 2 считаются верхними как наиболее абстрактные, а уровни 3 и 4 – нижними. Метауровень содержит эталонную модель (структурированное множество понятий педагогики, информатики, их взаимосвязей, обоснование четырехуровневой структуры метамодели). Таким образом, производится адаптация теории открытых систем к методике обучения информатике».

• Разрабатывается содержание нижних уровней метамодели обучения информатике: целевого («Зачем учить»?), содержательного («Чему учить»?) и процессуального («Как учить»?). Метамодели обучения на каждом уровне состоят из трех главных тематических профилей (brainware, software, hardware) и «теоретических основ открытых спецификаций для этих профилей».

• Экспериментально обосновывается эффективность практического применения разработанной метамодели. «Путем качественного анализа разработанных на основе метамодели спецкурсов (обладающих постулируемыми свойствами открытых систем – расширяемость / масштабируемость, мобильность / переносимость, интерперабельность, дружественность), показана их способность придать динамизм методике обучения информатике. Количественный анализ результатов педагогического эксперимента с использованием методов математической статистики также это подтвердил».

Рассмотрим принципы данных исследований (дисциплинарная методология). Категориальный строй перечисленных выше работ (проблема, тема, актуальность, объект и предмет исследования, задачи, гипотеза, задачи и защищаемые положения, значение для науки, значение для практики), выступающий как характеристика педагогического исследования в процессе его проведения, безукоризнен (точнее традиционен). Так, формулировка проблемы как ответ на вопрос «Что надо изучить из того, что раньше не было изучено?» практически идентична – противоречие между существующей системой подготовки учителей информатики или обучением информатике в школе и уровнем информатизации общества (в частности, самого образования). В темах исследований, за редким исключением, присутствует понятие или развития, или фундаментализации. Таким образом, требование отражения в теме момента «от достигнутого в науке к новым результатам» выполняется. При обосновании актуальности тем исследований происходит зачастую «плавный переход» на обоснование актуальности направления исследований, а она не вызывает сомнений. Объектом исследований («что рассматривается, аспекты и функции рассмотрения») является или методическая система обучения информатике, или процесс обучения информатике, что, на наш взгляд, не имеет принципиальных отличий. Предмет исследований, как это и положено, вычленяет из объекта (уход от попытки «объять необъятное») определенные свойства и отношения в соответствии с темой исследования.

Рассмотрение и по другим категориям также подтверждает мысль об их выполнении в соответствии с общепризнанной парадигмой (гр. – образец, пример), и это совершенно нормальное явление. Понятие парадигмы в современном представлении было введено Т. Куном в связи с историко-научными исследованиями и стало достаточно распространенным для характеристики «совокупности теоретических и методологических предпосылок, определяющих конкретное научное исследование, которая воплощается в научной практике на данном этапе» [22], или «исходная концептуальная схема, модель постановки проблем и их решения, методов исследования, господствующих в течение определенного исторического периода в научном сообществе» [21]. По Т. Куну, наука, развивающаяся в рамках общепризнанной парадигмы, является «нормальной»: «исследование в нормальной науке направлено на разработку тех явлений и теорий, существование которых парадигма заведомо предполагает» [13]. До тех пор, пока парадигма

выступает как надежный инструмент решения научных проблем, в науке происходит такой процесс накопления знания, при котором работа ученого заключается в совершенствовании самой парадигмы и решении проблем, называемых Т. Куном «головоломками», так как для них существует гарантированное решение, которое можно получить некоторым предписанным путем. Затем начинает расти число проблем, которые нельзя решить средствами существующей парадигмы – «аномалии» [3].

Попробуем выделить ряд проблем, для которых, на наш взгляд, не может быть получен результат в рамках существующий парадигмы.

1. *Содержание образовательной информатики.* Образовательная информатика в содержательном плане, при всей своей специфике, является, в определенной своей части, «срезом» информатики как науки, так и реальной деятельности (производственной) значительной части нашего общества. Мы должны учить реальной (настоящей) информатике, а не нечто созданному специально и называемому информатикой. Так, время, когда весь образовательный курс информатики строился на основе понятий ИСПОЛНИТЕЛЯ, видимо, ушло. Действительно, абстрагируясь, можно считать любой компьютер, любую программную среду неким ИСПОЛНИТЕЛЕМ. Но спрашивается, с какой целью создаются различные типы ИСПОЛНИТЕЛЕЙ. Биология не будет изучена лучше, если развитие растения в некой теплице показывается на экране компьютера. Межпредметные связи? Но разве в этом они состоят? На наш взгляд, это является не чем иным, как доведением до абсурда содержательных образовательных идей С. Пейперта и А. Кушниренко. Продолжим. На уровне констатации факта – «да, то-то произошло в реальной информатике» и то, что произошло, на чей то взгляд, так-то должно отрабатываться образовательной информатикой, рассуждать вряд ли возможно, точнее малопродуктивно. Например, да, произошло повсеместное внедрение компьютерных сетей, этот процесс идет по экспоненциальной нарастающей. Но только этого факта, без знания внутреннего механизма работы сетей, логики становления, развития и функционирования этих сетей, недостаточно для того, чтобы говорить об их профессиональном использовании (хотя бы) в учебном процессе, тем более о новых образовательных технологиях с использованием этой возможности. Так, «разнесение» в пространстве и во времени образовательного процесса с помощью сети Интернет вряд ли относится к синергетике, и называть это разнесение, даже с учетом использования технологий гипертекста, мультимедиа и т.д. для создания пособий, нелинейной технологией, весьма затруднительно. Еще пример. Программирование, как таковое, практически исключается из оборота в образовательной информатике (прошедший этап в ее развитии). Однако обратимся к программированию в реальной информатике. Оно, если так можно выразиться, «костяк», «позвоночник» этой информатики. В нем синтезируются все ее достижения. Программирование само развивается по весьма не простым закономерностям, оно поставляет различные технологии другим областям деятельности человека, например экономической. Вырывать из этого развития отдельный момент – речь идет об объектно-ориентированной подходе (кстати, это есть одна из информационных технологий) – и использовать его, например, для «проектирования содержания обучения средствам информационных технологий», разумеется, допустимо, но возникает естественный вопрос: «А почему используется для этих целей не тот же CASE-ориентированный подход. Трудно, очень трудно согласиться с утверждением К.К. Колина [10] о том, что технологии программирования являются даже не базовыми, а прикладными информационными технологиями. Практика и «расстановка сил» в реальной информатике говорят о другом. Через программирование учащийся совершенно по-другому осваивает и начала общей теории информации, и математические основы информатики (это из теоретической информатики [10]), и технические средства информатизации, и программные средства информатизации (это из средств информатизации [10]). Хочется предложить сторонникам этой позиции провести простейший педагогический эксперимент – посадить рядом двух школьников. Первый проучен теоретической информатике и средствам информатизации. Второй в совершенстве работает в одной из сред программирования, но не проходил обучение по вышеприведенной схеме. И тот, и другой не знают того, что называют базовыми информационными технологиями. Обоим необходимо разобраться с одной из базовых информационных технологий. Сравним время и глубину освоения. Результат будет поразительный (такие эксперименты проводились), опровергающий многое в современной образовательной информатике. Количественные характеристики отличаются, как минимум, на порядок. А нас учили: теоретические результаты (построения) обязаны подтверждаться практикой, иначе они не соответствуют действительности. Если работа [10] посвящена, в основном, обоснованию введения нового предмета «социальная информатика» (что действительно разумно и необходимо) в систему подготовки специалиста по информатике, то в работе А.Л. Семенова [19], на наш взгляд, идет полное выхолащивание предметной области информатики. Предлагается заменить термин «информатика» на «информатика и информационные технологии». Эта «мягкая» замена позволяет затем говорить о том, что все или почти все сводится «к освоению моделей деятельности», а объектами деятельности являются «результаты измерения и моделирования, тексты, изобразительная информация, звуковые образы»... И в конечном итоге «основными тенденциями становятся: трансформация роли учителя информатики в роль координатора информационных технологий (!!!); ...переход от «знаниевой» к «деятельностной» парадиг-

ме (!!!) ...». Откуда «растут рога» этой абсурдной точки зрения, достаточно очевидно. Была и есть образовательная область, связанная с подготовкой учителей труда. В эту деятельность вовлечено огромное количество людей, написано огромное количество работ. Тихо и мирно готовились учителя по токарному, слесарному, столярному и т.д. делам. Потребность в таких специалистах исчезла, а система подготовки (инфраструктура) осталась. Естественно, чтобы выжить, необходимо перестроиться. Если система подготовки специалистов-педагогов перестроится в этом направлении (думается, что сама жизнь не допустит подобного), то мы получим специалистов по молоткам (а не по информатике), умеющих говорить о двух частях этого инструмента и о том, что при забивании гвоздей держаться необходимо за одну из частей, и не более. Система подготовки специалистов по естественнонаучному циклу (а информатика относится именно к этому циклу, так же как и математика) как в школе, так и в вузах была национальным достоянием нашего государства. Идет, в силу ряда обстоятельств, медленное (из-за здорового консерватизма системы образования) ее разрушение. И данная работа из этой серии.

Еще один пример. Пишется и говорится о том, что необходимо учить работе в среде ПРОЛОГ, учить «логическому подходу к информационному моделированию» и т. д. Нет, разумеется, никто не против логического компонента в содержании обучения информатике. Он был, есть и будет. Однако в реальной информатике в этой среде не сделана ни одна сколько-нибудь известная разработка. Почему?

Из этих рассуждений напрашивается вывод: решить вопрос о содержании обучения информатике силами самой образовательной информатики, без учета реального развития в реальной информатике очень сложно, вероятно, невозможно. Требуется систематизация реальной информатики, вычленение в ней ключевых, содержательных линий, исследование их взаимосвязей и затем построение их «среза» в образование, естественно, с учетом специфики образования. И если подводить общий знаменатель под перечисленные выше исследования, то в большинстве из них берется некий факт (положение, достижение) из внешней среды, а затем он «прокручивается» по общепризнанной парадигме. Попытки построения, если так можно выразиться, диаграммы «профессиональной компетентности учителя информатики в условиях информатизации образования» с помощью инструментария педагогики и методики сводятся в реальности к незначительным правкам содержательного блока подготовки специалиста. Они вряд ли отражают реальные изменения в информатике и обеспечивают должный уровень его социальной защищенности, а значит, и профессиональной компетентности.

2. *Фундаментализация.* Многие исследования посвящены усилению фундаментальной составляющей как в подготовке учителей информатики, так и в общеобразовательных курсах. Зачастую она сводится к усилению математической составляющей. Безусловно, «пересечение» и взаимосвязь математики и информатики огромны. И если на предыдущих этапах развития информатика рассматривалась как элемент прикладной математики, то в настоящее время, в частности, после введения понятия «компьютерная математика»[12], на очереди исследования и обратного процесса – «как информатика влияет на математику, как преподавать математику с использованием информатики, какая математика должна преподаваться специалистам по информатике». От упрощенного понимания информатики ушли, однако сведение фундаментализации только к выделению «математических оснований информатики» является, на наш взгляд, очередным витком развития этого упрощения. В целом этот аспект фундаментализации должен вестись в направлении разработки «компьютерной математики». Естественным подходом при этом является:

- новая интерпретация классической математики (алгебры, математического анализа, геометрии), т.е. выделение тех ее составляющих, преподавание которых возможно с использованием компьютера и которые необходимы для подготовки специалистов по информатике (два аспекта);
- «новое чтение» курсов, которые традиционно относят к прикладной математике (исследование операций, теория графов, теория игр и т.д.);
- постановка новых курсов типа компьютерная алгебра, компьютерная геометрия и «увязка» их как с традиционными курсами, так и соответствующими информационными технологиями, т.е. выстраивание содержательных линий в подготовке специалистов.

А понятие фундаментальной составляющей образования по информатике четко и ясно определено А.А. Кузнецовым [11]. Требуются лишь дальнейшие исследования в этом направлении, однако, на наш взгляд, в рамках существующей парадигмы весомые результаты, требующие совместных усилий специалистов из различных областей знания, вряд ли будут получены. Пример тому – современные учебники по информатике. Они, в основном, посвящены средствам (см. рис.). Нет работ по теоретическим основам информатики, тем более – по научным основам информатики. Как работать с текстовым редактором Word, описывают все или почти все. Теоретические аспекты работы с информацией, в частности, с текстовой информацией, не приводятся почти никем, точнее, их нет в работах. Разработка направления «компьютерная математика» – еще один наглядный тому пример. Попытки исследований в этом направлении в рамках существующей парадигмы приводят при методологической строгости (отмечалось выше) к тому, что некое понятие абсолютизируется, и из него выводится чуть ли не вся схема преподавания по определенной содержательной линии образовательной информатики (пример исчерпанности очередной прокрутки парадигмы на определенных проблемах).

3. *«Мышление (творческое мышление) и компьютер».* Проблема мышления изучается различными науками (психологией, логикой, лингвистикой, кибернетикой, эпистемологией и т.д.). Естественно, что каждая наука абстрагируется от тех его сторон, механизмов, которые не существенны с точки зрения ее предмета и не могут быть исследованы с помощью присущих ей методов анализа. Например, психология исследует главным образом общие закономерности мышления, обнаруживающиеся при решении задач самого различного содержания. При этом творческое мышление рассматривается только как частный случай проявления этих общих закономерностей. В логике создан мощнейший аппарат, позволяющий эффективно оперировать уже имеющимися знаниями. Больших успехов добились логики в изучении механизмов выводного знания. Но соблюдение законов и правил логики не гарантирует успеха в решении многих задач, ибо решение не может быть найдено путем прямого логического вывода и не определяется совокупностью заранее заданных алгоритмов (точных предписаний мыслительных действий, ведущих от некоторых начальных данных к искомому результату). Можно продолжить, однако резюмируем. На наш взгляд, решение этой проблемы может быть получено в результате интеграции достижений эпистемологии (М. Мамардашвили, Э. Гуссерль, М. Вертгеймер и т.д.), когнитивной психологии и информатики. При этом упрощение проблемы до «процессов отражения внешнего мира в нашем сознании», а именно на этом постулате выстраиваются, в основном, дидактические схемы, приведет к упрощенным и не продуктивным решениям.

Исследование данной проблемы позволит:

- Однозначно определить место и роль обучения информатике в развитии учащихся.
- Провести «разделительную черту» между образовательными областями «информатика» и «технология». Попытки последней «съесть» значительный кусок информатики принимают гигантские размеры, при этом идет выхолащивание предметной области, сведение ее до пользовательского уровня. По образному сравнению А.А. Кузнецова, «можно ли представить себе специалиста по химическим технологиям, не знающего химии»? Трудно, очень трудно. В информатике это происходит сплошь и рядом. Специалисты, освоившие работу с несколькими программами, судят об информатике на уровне концепций и т.д. Если для подростка это выглядит нормальным явлением – возраст, то для ученого мужа это не что иное, как затянувшееся ребячество или остаточные явления реализации тезиса классика о возможности кухарки управлять государством.
- Целенаправленно продолжить работу над созданием современной методической системы обучения информатике (например, исключить схоластические споры о содержании: программирование или прикладное программное обеспечение).

На наш взгляд, прогнозировать положительный результат решения данной проблемы «мышление – компьютер» в рамках существующей парадигмы затруднительно.

4. *Моделирование и информатика.* Достаточно часто в исследованиях используется понятие модели. «...В термине "модель" все же имеется какое-то весьма оригинальное семантическое зерно, заставляющее бесчисленных авторов все же пользоваться этим термином и выставлять на первый план, несмотря на его путаность и противоречивость... То обстоятельство, что современное научное мышление еще не пришло к единообразию в этом вопросе и что каждый раз возникают все новые и новые оттенки этого понятия и этого термина, ни в каком случае не может приводить нас в уныние и уже тем более не должно приводить нас к попытке обойтись без этого термина и без этого понятия» [15]. Впервые понятие употреблено математиками в теориях математической логики. Это математическое значение модели было совершенно противоположным тому значению, которое она получила в дальнейшем. В работе [24] приводится 27 значений термина «модель». А.Ф. Лосев [15] доводит эту цифру до 34. Однако и эта цифра не исчерпывает всего терминологического разногласия. По В.А. Штоффу [25], под моделью «в широком смысле понимается мысленно или практически созданная структура, воспроизводящая ту или иную часть действительности в упрощенной (схематизированной или идеализированной) и наглядной форме» (классическое определение понятия в духе материализма. – *С.О.*). Данная терминология проникла во все науки и везде приобрела тот или иной специфический оттенок. Не обошла она и информатику. По А.П. Ершову [8], «информатика как отдельная наука вступает в свои права, когда для изучаемого фрагмента мира построена так называемая информационная модель... Информационная модель – это то сопряжение, через которое информатика вступает в отношения с частными науками, не сливаясь с ними и в то же время не вбирая их в себя». В.М. Глушков в 1963 г. писал: «Информационная модель представляет собой просто описание строения и закономерностей поведения моделируемого объекта... Средством фиксации любой конкретной информационной модели являются языки, причем не только те человеческие языки, которые изучаются традиционным языкознанием, но и любые искусственные языки, строящиеся в процессе накопления и передачи знаний...» (просто и замечательно. – *С.О.*). В.К. Белошапка придерживается похожей точки зрения [2]: «Объект, явление, процесс и прочее любой природы становится доступным компьютерной обработке только после представления его в виде текста над конечным алфавитом. Такое представление объекта называется информационной моделью... Компьютер имеет дело только с текстами, т.е. конечными последовательностями букв. Все прочие объекты, процессы, явления и т. д. попадают в сферу влияния компьютера лишь после представления в виде текста,

т.е. после информационного моделирования в виде информационных моделей». Вернемся к В.М. Глушкову [5], к его описанию структуры информационной модели: «Информационная модель, с которой оперирует каждая наука, может быть в первом приближении разделена на собственно информационную и так называемую исчисленческую часть. Информационная часть включает в себя фактический материал, классифицированный тем или иным способом. Исчисленческая часть, или просто исчисление, есть не что иное, как совокупность правил вывода, позволяющих получать дедуктивным путем те или иные следствия из совокупности основных научных положений».

Классики четко и ясно формулируют свои мысли. Однако если в одной работе встречаются несколько различных трактовок понятия: «...Информационная модель представляет собой последовательность символов некоторого алфавита, поэтому является традиционной моделью...», «информационная модель путем выполнения трансляции может быть преобразована в программное средство, поэтому является компьютерной моделью...», «принципиальной основой возможности применения компьютера для программной реализации математических моделей в вычислительном эксперименте является информационный характер математических моделей», – то вслед за Л. Маллори хочется сказать: «Беспорядочная мысль делает с нашим умом то же, что сделал бы с нашим домом беспорядочный человек, приглашенный пожить у нас», или, по А.С. Пушкину, «описывай, не мудрствуя лукаво...».

Итак, более чем тридцатилетний опыт использования понятий «модель», «моделирование» в информатике создал предпосылки для продолжения исследований «моделирование, информатика и философия», при этом, разумеется, должны учитываться достижения философской мысли (рассмотрение этого понятия не только в рамках «образной» теории сознания), которые не приветствовались во времена В.А. Штоффа [25]. Однако эти исследования в очередной раз не «укладываются» в существующую парадигму. Отсутствие фундаментальных работ с таким содержанием приводит к ситуациям, когда традиционную методическую систему обучения называют моделью для описания других моделей, а в результате получают известные курсы, но разработанные по этой сверхмодели и независимо от количества введенных страт («гора родила мышь»).

В заключение отметим, что целью этой работы является, несмотря на определенную эмоциональность (стиль автора), не критический анализ названных исследований. Автор пытался найти ответы на сформулированные проблемы и понять, почему их нет в существующих публикациях по методике обучения информатике. Может быть, это следствие неполноты исходных данных или непонимания, поэтому конструктивное обсуждение перечисленных проблем будет принято с признательностью (okulov@vspu.kirov.ru).

Литература

1. *Баранова Е.В.* Теория и практика объектно-ориентированного проектирования содержания обучения средствам информационных технологий: Автореф. дис. ... д-ра пед. наук. СПб., 2000.
2. *Белошапка В.* Информатика как наука о буквах // Информатика и образование. 1992. №1.
3. *Богуславский В.И., Извозчиков В.А., Потемкин М.Н.* Наука в педагогическом университете: Вопросы методологии, теории и практики / Под ред. В.И. Богословского. СПб.: Изд-во СПбГУ, 2000.
4. *Гейн А.Г.* Изучение информационного моделирования как средство реализации межпредметных связей информатики с дисциплинами естественнонаучного цикла: Автореф. дис. ... д-ра пед. наук. М., 2000.
5. *Глушков В.М.* Кибернетика. Вопросы теории и практики. М.: Наука, 1986.
6. *Готская И.Б.* Методическая система обучения информатике студентов педвузов в условиях рыночной экономики: Автореф. дис... д-ра пед. наук. СПб., 1999.
7. *Добудько Т.В.* Формирование профессиональной компетентности учителя информатики в условиях информатизации образования: Автореф. дис. ... д-ра пед. наук. М., 1999.
8. *Ершов А.П.* Информатика. Предмет и понятие // Наука в Сибири. 1983. № 32.
9. *Есаян А.Р.* Теория и методика обучения алгоритмизации на основе рекурсии в курсе информатики педагогического вуза: Автореф. дис. ... д-ра пед. наук. М., 2001.
10. *Колин К.К.* О структуре и содержании образовательной области «информатика» // ИНФО. 2000. №10.
11. *Кузнецов А.А.* О концепции содержания образовательной области «информатика» в 12-летней школе // ИНФО. 2000. №7.
12. *Кук Д., Бейз Г.* Компьютерная математика. М.: Наука, 1990.
13. *Кун Т.С.* Структура научных революций. М., 1975.
14. *Лаптев В.В., Швецкий М.В.* Метод демонстрационных примеров в обучении информатике студентов педагогического вуза // Педагогическая информатика. 1994. №2.
15. *Лосев А.Ф.* Введение в общую теорию языковых моделей. М., 1968.
16. *Могилев А.В.* Развитие методической системы подготовки по информатике в педагогическом вузе в условиях информатизации образования: Автореф. дис. ... д-ра пед. наук. Воронеж, 1999.
17. *Пак Н.И.* Нелинейные технологии обучения в курсах информатики и информационных технологий: Автореф. дис. ... д-ра пед. наук. Красноярск, 2000.
18. *Рыжова Н.И.* Развитие методической системы фундаментальной подготовки будущих учителей информатики в предметной области: Автореф. дис. ... д-ра пед. наук. СПб., 2000.
19. *Семенов А.Л.* Роль информационных технологий в общем среднем образовании // ИНФО. 2001. №2.
20. *Симонова И.В.* Концептуальные модели обучения практико-ориентированных учащихся в условиях интернет-образования: Автореф. дис. ... д-ра пед. наук. СПб., 2000.
21. Советский энциклопедический словарь. 4-е изд. М., 1988.
22. Философский словарь. 5-е изд. М., 1987.
23. *Фокин Р.Р.* Метамодель обучения информатике в высшей школе: Автореф. дис. ... д-ра пед. наук. СПб., 2000.
24. *Чжао Юань-жень* «Модели в лингвистике и модели вообще» // Математическая логика и ее применение. М., 1965. С. 281-293.
25. *Штофф В.А.* Моделирование и философия. Л.: Наука, 1966.

С.М. Окулов

ОБРАЗОВАТЕЛЬНЫЕ СТАНДАРТЫ ВЫСШИХ УЧЕБНЫХ ЗАВЕДЕНИЙ ПО ИНФОРМАТИКЕ*

Перед образовательной информатикой стоит проблема: как не отстать, а идти чуть впереди процесса информатизации общества так, чтобы готовить специалистов не только для текущего момента, но и с перспективой на будущее. Дискуссия о том, какой должна быть общеобразовательная информатика, идет с момента ее внедрения в школы. После некоторых исследований [1, 2] наметился прогресс в этом вопросе. В данной работе предлагается взглянуть на обучение информатике с точки зрения высшей школы, ибо там этот процесс более устоявшийся и регламентированный образовательными стандартами.

Предметом анализа являются Российские образовательные стандарты:

• «010200 – Прикладная математика и информатика». Квалификация выпускника – математик, системный программист. Рассматривается, в качестве примера, образовательная программа специализации «01.02.09 – Математическое и программное обеспечение вычислительных машин»;

• «654600 – Информатика и вычислительная техника». Квалификация выпускника – инженер. Рассматривается, в качестве примера, образовательная программа специализации «220400 – Программное обеспечение вычислительной техники и автоматизированных систем».

• «030100 – Информатика». Квалификация выпускника – учитель информатики.

Теоретическими основами проводимого анализа являются положения работы В.С. Леднева [3].

Итак, стандарты классического, технического и педагогического университетов: на первый взгляд, отличия и идеи, заложенные в названных стандартах, должны быть принципиальными. Выясним, так ли это. Для сравнения взят отчет по учебному плану по информатике за 1991 г., 2001 г., разработанный Специальной Объединенной комиссией / Рабочей группой, образованной Ассоциацией ACM (Association of Computer Machinery) и профессиональным объединением IEEE (Institute of Electrical and Electronics Engineers) [4,5].

Общая характеристика Российских образовательных стандартов и отчета ACM/IEEE. Образовательные стандарты включают следующие циклы предметов:

• ГСЭ – общие гуманитарные и социально-экономические дисциплины (не рассматриваются в работе, 1800 часов в стандартах классического и технического университетов и 1500 часов в стандарте педагогического университета);

• ЕН – общие математические и естественнонаучные дисциплины;

• ОПД – общепрофессиональные дисциплины;

• СД – специальные дисциплины, включая дисциплины специализации (в стандартах называются по-разному: специальные дисциплины, дисциплины специализации, дисциплины предметной подготовки);

• ФТД – факультативные дисциплины.

В стандарте классического университета дается общая формулировка уровня подготовки специалиста – «для организаций, использующих методы прикладной математики и компьютерные технологии в своей работе». Стандарт для педагогических университетов оговаривает сферу деятельности строго – «для работы в образовательных учреждениях различного типа». Выпускник технического университета может работать, в частности, инженером-программистом. Заслуживает упоминания в работе трактовка понятия информатики в этом стандарте (из-за ее практической направленности) – «Информатика и вычислительная техника – это область науки и техники, которая включает в себя совокупность средств, способов и методов человеческой деятельности, направленных на создание и применение: ЭВМ, систем и сетей; автоматизированных систем обработки информации и управления; систем автоматизированного проектирования; программного обеспечения вычислительной техники и автоматизированных систем». Это понимание информатики соответствует разделу «Средства» Национального доклада Российской Федерации на II международном конгрессе ЮНЕСКО «Образование и информатика» (Москва, 1996) [6].

Отчет ACM/IEEE предлагает 2 уровня подготовки: учебный минимум (общие требования) и дисциплины углубленного изучения информатики. Учебный минимум состоит из математических дисциплин и девяти предметных областей по информатике: алгоритмы и структуры данных; архитектура; искусственный интеллект и робототехники; базы данных и информационный поиск; взаимодействие «Человек – Машина»; символьные и числовые вычисления; операционные системы; языки программирования; методология разработки программного обеспечения и проектирования.

Каждая предметная область разбивается на модули. Так, предметная область «алгоритмы и структуры данных» состоит из следующих модулей: «базисные структуры данных»; «абстрактные типы данных»; «рекурсивные алгоритмы»; «анализ сложности»; «классы сложности»; «сортировка и поиск»; «вычисляемость и неразрешимость»; «методология решения проблем»; «параллельные и распределенные алгоритмы». Отчет не дает точного количества часов, отводимых на проведение дисциплин, как в образовательных стандартах Российской Федерации. Однако для составления учебного плана, а его роль, как подчеркивает В.С. Леднев [3], в организации учебного процесса огромна, даются взаимосвязи модулей из различных предметных областей, чего нет в наших стан-

* Работа выполнена при поддержке Министерства образования Российской Федерации, проект Г00-4.1-60

дартах. Эти взаимосвязи позволяют организовать процессы планирования и распределения модулей во времени изучения. Так, например, модуль «базисные структуры данных»:

• предшествует (или необходим как условие) модулям: «Абстрактные типы данных» (предметная область «Алгоритмы и структуры данных»), «Представление типов данных» (предметная область «Программирование»), «Фундаментальные концепции решения задач» (предметная область «Методология решения проблем» и «Дискретная математика» (предметная область «Математика»);

• связан с модулями «Аппаратное управление» (предметная область «Операционные системы») и «Управление памятью во время выполнения» (предметная область «Программирование»).

К Российским образовательным стандартам прилагается сокращенный вариант программ изучения предметов и проект учебного плана.

Определение шкалы измерения. Для проведения сопоставительного анализа стандартов требуется единая шкала измерения. Естественным первым шагом является разбивка дисциплин по блокам (предметным областям) и подсчет количества часов, отводимых для их изучения. Первоначально предполагалось, что исходной точкой анализа должна быть структура образовательной области «ИНФОРМАТИКА», представленная в Национальном докладе Российской Федерации на II международном конгрессе ЮНЕСКО «Образование и информатика» (Москва, 1996) [6]. В докладе указано, что ему образовательная область «ИНФОРМАТИКА» содержит следующие основные компоненты:

* фундаментальные основы информатики;
* теоретическую информатику;
* средства информатизации:
  - технические:
    - обработки данных;
    - передачи данных;
  - программные:
    - системные:
      - программные средства межкомпьютерной связи;
      - операционные системы, сервисные оболочки системы пользовательского интерфейса;
      - системы и языки программирования;.
    - реализация технологий:
      - универсальных;
      - профессионально-ориентированных.

Информационные технологии включают:

• ввод, вывод, сбор, хранение, передачу и обработку информации;

• подготовку текстовых и графических документов;

• программирование, проектирование, моделирование, обучение, диагностику, управление (объектами, процессами, системами).

Примечание. *Внесенные изменения «Системы и языки программирования» выделены отдельно, а не в составе «Операционных систем».*

Однако данная попытка оказалась непродуктивной. Причины, видимо, кроются в том, что понимание информатики, заложенное в образовательных стандартах, не в полной мере согласуется с положениями доклада, с другой стороны, некоторые положения доклада не однозначно трактуются. Так, например, текстовый процессор должен изучаться и как средство информатизации, и как информационная технология.

Для анализа стандартов дисциплины блоков ЕН, ОПД и ДС и отчета ACM/IEEE классифицируются по принципу (или разбиваются на следующие предметные области), приведенному на рис. 1. Можно дискутировать по этому поводу, однако примем такую разбивку для достижения цели, поставленной в работе, – «выявление общих закономерностей обучения информатике в высших учебных заведениях». На рис. 2, 3 приводится содержание блоков с их отнесением к выбранной системе классификации.

Название	Условное обозначение
Математика	**М**
Компьютер	**К**
Теоретическая информатика	**ТИ**
Операционные системы	**ОС**
Программирование, технологии программирования	**П**
Информационные технологии	**ИТ**
Сети, коммуникации	**С**
Прочие	**ПР**

Рис. 1

Первичный анализ. В стандарте «Прикладной математики и информатики» в блоках ЕН и ОПД дисциплин, связанных с «Информатикой», не так много (выделены курсивом на рис. 2). Полностью отсутствуют дисциплины по техническим средствам. При выборе специализации не «01.02.09 – Математическое и программное обеспечение ЭВМ», а, например, «01.02.06 – Исследование операций и системный анализ» возможность изучения информатики еще более уменьшается. В целом стандарт отражает точку зрения на информатику как на раздел прикладной математики. Она не нова, еще на заре появления кибернетики в нашем государстве был аналогичный взгляд и на суть последней. Стандарт «Информатика и вычислительная техника» более взвешенный по составу дисциплин, традиционно относящихся к информатике, и наиболее близкий к позициям отчета ACM/IEEE. Явно просматривается практическая направленность образовательного процесса, компоновка всех математических дисциплин в один курс вряд ли выдерживает критики. Стандарт «Информатика» для педагогических университетов занимает среднее положение между вышеназванными документами, ибо нет таких крайностей.

		010200-Прикладная математика и информатика				654600–Информатика и вычислительная техника		
ЕН	1.	Математика (алгебра, анализ, геометрия)	1173	М	1.	Математика (алгебра, анализ, геометрия, дискретная математика, логика и теория алгоритмов, вычислительная математика, теория вероятностей и математическая статистика)	960	М
	2.	*Информатика*	153	ТИ	2.	*Информатика*	140	ТИ
	3.	Физика	306	ПР	3.	Физика	402	ПР
	4.	Концепции современного естествознания	204	ПР	4.	Экология	70	ПР
	5.	НРК и КВ	240	ПР	5.	НРК и КВ	340	ПР
		Итого:	***2076***			Итого:	***1912***	
ОПД	1.	Дифференциальные уравнения	204	М	1.	Начертательная геометрия. Инженерная и компьютерная графика	240	ПР/ИТ
	2.	Дискретная математика	153	М	2.	Электротехника и электроника	250	ПР
	3.	Теория вероятностей и математическая статистика	204	М	3.	Метрология, стандартизация и сертификация	110	ПР
	4.	Уравнения математической физики	204	М	4.	Безопасность жизнедеятельности	100	ПР
	5.	*Языки программирования и методы трансляции*	153	П	5.	*Алгоритмические языки и программирование*	260	П
	6.	*Системное и прикладное программное обеспечение*	102	ОС	6.	Основы теории управления	120	ПР
	7.	*Практикум на ЭВМ*	400	П	7.	*Организация ЭВМ и систем*	140	К
	8.	Методы оптимизации	102	М	8.	*Операционные системы*	140	ОС
	9.	Численные методы	153	М	9.	*Базы данных*	140	ИТ
	10.	Теория игр и исследование операций	51	М	10.	*Сети ЭВМ и телекоммуникации*	170	С
	11.	*Базы данных и экспертные системы*	102	ИТ	11.	*Методы и средства защиты информации*	110	ТИ
	12.	НРК и КВ	450	М/П/ ОС/ИТ	12.	Организация планирования производства	80	ПР
					13.	НРК и КВ	340	ПР/К/ОС/П
		Итого:	*2278*			Итого:	*2200*	
ДС или СД или ДПП	Специализация 01.02.09 Математическое и программное обеспечение ЭВМ				Специализация 220400 - Программное обеспечение вычислительной техники и автоматических систем			
	1.	Языки программирования	482	П	1.	Структуры и алгоритмы обработки информации	210	ТИ
	2.	Технологии программирования	482	П	2.	Функциональное и логическое программирование	140	П
	3.	Методы системного программирования	482	П	3.	Объектно-ориентированное программирование	140	П
	4.	Методы трансляции	482	П	4.	Теория языков и методы трансляции	140	П
	5.	Базы данных	482	ИТ	5.	Теория вычислительных процессов	100	ТИ
	6.	Системы символьных преобразований	482	ИТ	6.	Архитектура вычислительных систем	100	К
	7.	Системы искусственного интеллекта	482	ТИ	7.	Технология разработки программного обеспечения	170	П
	8.	Программное обеспечение машинной графики	482	ИТ	8.	Человеко-машинное взаимодействие	100	П
	9.	Объектно-ориентированное программирование	482	П	9.	Дисциплины специализации	800	ТИ/П/К/С
	10.	Параллельное программирование	482	П				
	11.	Операционные системы	482	ОС				
	12.	Компьютерные сети	482	С				
		Итого:	*1428*			Итого:	*1900*	
		Сумма по блокам:	**5782**				**6012**	

Рис. 2

	030100 - Информатика				Отчет ACM/IEEE		
ЕН	1.	Математика (алгебра, анализ, геометрия)	382	М	1.	Математика: дискретная математика; теория исчислений; вероятность; линейная алгебра; математическая логика	М
	2.	Физика	324	ПР	2.	Естественные науки	ПР
	3.	Химия	72	ПР			
	4.	Биология с основами экологии	72	ПР			
	5.	НРК и КВ	150	ПР			
		Итого:	1000				
ОПД	1.	Психология	300	ПР	1.	Алгоритмы и структуры данных	ТИ
	2.	Педагогика	300	ПР	2.	Архитектура	К
	3.	Основы специальной педагогики и психологии	72	ПР	3.	Искусственный интеллект и робототехника	ТИ
	4.	Теория и методика обучения информатике	320	ТИ	4.	Базы данных и информационный поиск	ИТ
	5.	Возрастная анатомия, физиология и гигиена	72	ПР	5.	Взаимодействие "Человек - Машина"	П
	6.	Основы медицинских знаний	72	ПР	6.	Символьное и числовое вычисления	П
	7.	Безопасность жизнедеятельности	72	ПР	7.	Операционные системы	ОС
	8.	Технические и аудиовизуальные средства обучения	72	ПР	8.	Языки программирования	П
	9.	НРК и КВ	320	ПР/ТИ	9.	Методология разработки и проектирования программного обеспечения	П
					10.	Исторический и социальный аспекты информатики	ПР
		Итого:	1600				
ДС или СД или ДПП	1.	Математическая логика	130	М	1.	Углубленный курс по операционным системам	ОС
	2.	Дискретная математика	130	М	2.	Углубленный курс по программированию	П
	3.	Элементы абстрактной и компьютерной алгебры	130	ТИ	3.	Анализ алгоритмов	ТИ
	4.	Теория алгоритмов	130	ТИ	4.	Искусственный интеллект	ТИ
	5.	Теория вероятностей и математической статистики	160	М	5.	Комбинаторные алгоритмы и алгоритмы на графах	ТИ
	6.	Численные методы	260	М	6.	Вычислительная сложность	ТИ
	7.	Уравнения математической физики	130	М	7.	Курс по сетям	С
	8.	Теоретические основы информатики	144	ТИ	8.	Компьютерная графика	ИТ
	9.	Исследование операций	144	М	9.	Интерфейс "Человек - Машина"	П
	10.	Основы искусственного интеллекта	144	ТИ	10.	Компьютерная безопасность	ПР
	11.	Компьютерное моделирование	190	ТИ	11.	Базы данных и поиск информации	ИТ
	12.	Основы микроэлектроники	104	К	12.	Теория информации	ТИ
	13.	Архитектура компьютера	144	К	13.	Моделирование	ТИ
	14.	Программирование	366	П	14.	Численные методы	М
	15.	Программное обеспечение ЭВМ	390	ОС	15.	Параллельная и распределенная обработка	ТИ
	16.	Информационные системы	130	ИТ	16.	Принципы компьютерной архитектуры	К
	17.	Компьютерные сети, интернет, мультимедиа технологии	190	С	17.	Принципы языков программирования	П
	18.	Информационные и коммуникационные технологии в образовании	100	С	18.	Трансляция языка программирования	П
	19.	Практикум по решению задач на ЭВМ	214	П	19.	Робототехника и искусственный интеллект	ТИ
	20.	Дисциплины специализации	500	П/С	20.	СБИС проектирование	К
	21.	НРК и КВ	500	П/С			
		Итого:	4330				
		Сумма по блокам:	6930				

Рис. 3

Уточнение шкалы измерения. Для сравнения стандартов необходимо принять ряд допущений. Во-первых, блок ГСЭ (Общие гуманитарные и социально-экономические дисциплины) не рассматривается, ибо он практически совпадает во всех стандартах (в стандарте педагогического университета на изучение дисциплин этого блока отводится на 300 часов меньше), и его нет в отчете ACM/IEEE. Во-вторых, блок дисциплин специализации «01.02.09 – Математическое и программное обеспечение ЭВМ» не содержит часовой разбивки, дается только общее количество часов. Для нашего анализа общее количество часов равномерно распределяется между дисциплинами блока. В-третьих, часы, отведенные для национально-регионального компонента (НРК) и курсов по выбору (КВ), в каждом блоке распределяются равномерно, по типам дисциплин блока. В-четвертых, отчет ACM/IEEE не содержит часовой разбивки между дисциплинами. Для выявления общих закономерностей принят равномерный принцип разбивки. Общее количество часов – среднее значение по часам из Российских стандартов – делится на количество дисциплин.

После уточнения шкалы измерения появляется возможность для сравнительного анализа. Его результат приведен на рис. 4, 5.

		"010200"	"654600"	"030100"	Отчет
Компьютер	К	0	525	248	510
Математика	М	2356	960	1336	1020
Теоретическая информатика	ТИ	272	760	1227	1700
Сети	С	119	370	690	170
Операционные системы	ОС	333	225	195	340
Программирование, технологии программирования	П	1379	1235	1080	1190
Информационные технологии	ИТ	452	260	325	510
Прочие	ПР	750	1677	1738	510
Сумма по блокам		5661	6012	6839	5950
В целом по стандарту		8032	8262	8884	

Рис. 4

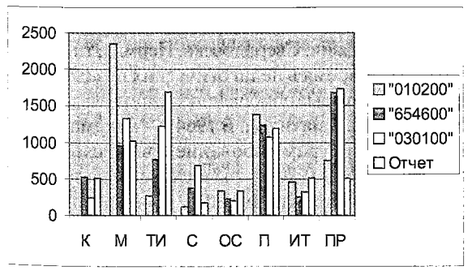


Рис. 5

Выводы. 1) Несмотря на кажущиеся различия по всем рассматриваемым образовательным стандартам, изучаются, в принципе, одни и те же дисциплины. 2) Количество часов, выделяемых на предметную область «Программирование», практически совпадает. Выравнивание по стандарту «030100 – Информатика» происходит за счет часов из дисциплин специализации и курсов по выбору (региональный аспект). Просматривается явное несоответствие между ролью этой предметной области, представленной в национальном докладе по «ИНФОРМАТИКЕ», и долей часов, отводимых на ее изучение в образовательных стандартах. 3) Если суммировать количество часов, выделяемых на предметные области «Математика» и «Теоретическая информатика», то стандарты классического и педагогического университетов незначительно отличаются друг от друга. Показатель у технического стандарта по этим предметным областям несколько ниже. 4) Предметная область «Информационные технологии» не является основополагающей. Ей уделяется значительно меньше внимания, чем в школьных учебниках по информатике последних лет. 5) Увеличение количества часов на предметную область «Сети» не следует из стандарта, а достигается за счет часов из блока дисциплин специализации в Вятском педагогическом университете (региональный аспект).

Пути дальнейшего совершенствования учебного процесса. 1) Необходимо выделить некую «сердцевину» образовательной области «ИНФОРМАТИКА», которая должна быть обязательной, независимо от типа образовательного стандарта. 2) Требуется выстроить содержательные линии по каждой предметной области. При этом, используя методы сетевого планирования, на основе учебных программ установить взаимосвязи дисциплин как одной предметной области, так и между предметными областями. 3) Пересмотреть методику преподавания традиционных математических дисциплин на предмет использования при проведении практических занятий информационных технологий.

Литература

1. *Кузнецов А.А.* О концепции содержания образовательной области «ИНФОРМАТИКА» в 12-летней школе // Информатика и образование. 2000. №7.

2. *Кузнецов А.А., Бешенков С.А., Лыскова В.Ю., Ракитина Е.А.* Системообразующая роль информатики в содержании школьного образования // Стандарты и мониторинг в образовании. 2000. №2.

3. *Леднев В.С.* Содержание образования: сущность, структура, перспективы. 2-изд., перераб. М.: Высш. шк., 1991. 224 с.

4. Отчет ACM/IEEE. http:/www.computer.org/education/cc1991/

5. Проект отчета ACM/IEEE. http:/www.cs. swarthmore.edu/~eroberts/cc2001/

6. Концепция информатизации сферы образования Российской Федерации // Бюллетень 3-4 (13-14)*1998 «Проблемы информатизации высшей школы». М., 1998.

В.А. Онегов

ТЕОРЕТИЧЕСКАЯ ИНФОРМАТИКА И МАТЕМАТИКА

Для того чтобы понять место математики в подготовке специалиста по информатике (в том числе и учителя информатики), необходимо еще раз вспомнить определение этой науки и проследить, как велась подготовка специалистов, которые в настоящее время называются информатиками.

Необходимость возвращения к определению самого содержания науки «информатика» возникает из-за того, что в связи с широким проникновением компьютеров во все сферы человеческой деятельности (вплоть до быта) сам термин «информатика» оказался чрезвычайно размытым. Кроме того, существует масса организаций, за некоторую плату достаточно быстро готовящих людей, которым выдаются свидетельства о том, что они получили квалификацию информатика. Любой, имеющий какое-то отношение к компьютерной технике, с известной долей обоснованности может называть себя (и называет) информатиком. В этом отношении технический работник, набирающий тексты в системе Microsoft Word (секретарь-машинистка по-старому), – информатик. Школьник, который круглосуточно играет в компьютерные игры, – информатик. Любитель специфических новостей, использующий для этого всемирную сеть Internet – информатик. Конечно, специалистов из названного ряда не возьмут на работу не только Билл Гейтс или Питер Нортон, но и любая уважающая себя сельская средняя школа.

Еще один важный аспект. Существуют целые псевдонаучные направления, которые эксплуатируют слово «информация» и его производные (например, Международная академия информациологии).

Каким образом готовились информатики как специалисты, которые занимались вопросами информатики до возникновения информатики как самостоятельной науки?

В середине 50-х гг. советская промышленность начала серийно выпускать ЭВМ первого поколения (Урал-1, Урал-2, Минск-2, Стрела и т.п.). Эти ЭВМ в первую очередь направлялись в оборонную промышленность и в НИИ оборонного направления. Одновременно встал вопрос о подготовке специалистов, которые могли бы ограниченные ресурсы этих ЭВМ использовать с наибольшей отдачей. Поскольку ЭВМ – сложный и дорогостоящий аппарат, допуск к работе с ним для дилетантов был закрыт. Первые исследования, проводимые на ЭВМ, относились к атомной и космической тематике. Решение задач, возникающих при этом, требует глубоких знаний по математике (умение математически формулировать задачу из соответствующей области) и, особенно, методов численной реализации сформулированной математической задачи (методы вычислений). Поэтому первых пользователей ЭВМ стали готовить на математических факультетах университетов (классических). Так, в Ленинградском государственном университете с 1955 до 1970 г. проводился прием студентов по специализации вычислительная математика на математико-механическом факультете в количестве до трех групп (по 25 человек). Первые три выпуска (1960, 1961, 1962 гг.) были полностью распределены в оборонные НИИ (Реутов, Подлипки, Арзамас, Челябинск, Красноярск и т.п.). Студентам этих выпусков Министерство обороны выплачивало специальную повышенную стипендию. Кстати, первым заведующим кафедрой вычислительной математики был будущий академик АН СССР и многих других академий мира, лауреат Нобелевской премии Л.В. Канторович.

Аналогичная ситуация имела место на механико-математическом факультете МГУ им М.В. Ломоносова и в возникшем в это время Московском физико-техническом институте. Крупный научный центр по подготовке математиков-вычислителей был создан в городе Новосибирске в 1958 г. (университет и вычислительный центр).

В конце 50-х в СССР на волне хрущевской оттепели реабилитировали кибернетику. Были переведены и изданы книги основателей кибернетики Н. Винера и А. Розенблата, У.Р. Эшби, а также английских кибернетиков супругов К. Бута и Э. Бут, К. Шеннона (теория информации). В это же время издаются первые труды по кибернетике советских авторов (В.М. Глушков и др.). В начале 70-х гг. дисциплины, связанные с кибернетикой, повсеместно (не только в университетах, но и в ведущих технических вузах) вводятся в процесс обучения; в классических университетах СССР открываются факультеты прикладной математики (вычислительная и прикладная математика плюс кибернетика). В разных университетах эти факультеты называются по-разному. Под руководством члена-корреспондента АН СССР В.И. Зубова в ЛГУ открылся факультет прикладной математики – процессов управления. В МГУ при непосредственном участии академика АН СССР, лауреата Ленинской премии, Героя Социалистического труда А.Н. Тихонова открывается факультет вычислительной математики и кибернетики. Аналогичные факультеты тогда же были открыты в университетах Новосибирска, Горького, Казани, Свердловска, Перми и т.п.

В это же время в педагогических вузах вводятся предметы, которые обобщенно можно назвать вычислительной математикой. В 1964 г. – математические машины и программирование с вычислительным практикумом. В 1970 г. – вычислительная математика и программирование. В 1979 г. – вычислительная математика и программирование с числовыми системами. В 1985 г. в программу средней школы был введен предмет «Основы информатики и вычислительной техники». Сначала на математических, а вскоре и на физических факультетах педагогических вузов начала подготовка учителей математики и физики с дополнительной специальностью – учитель информатики. В 1989 г. в учебных планах педагогических институтов появляются наука вычисления и алгоритмизация, основы информатики и вычислительной техники, численные методы и ряд специальных курсов, связанных с информатикой.

Математический энциклопедический словарь за 1988 г. дает следующее определение информатики как науки: находящаяся в становлении наука, изучающая законы и методы накопления, передачи и обработки информации с помощью ЭВМ.

Название «информатика» подчеркивает ключевую роль информации как основного понятия (объект науки), образованного в соответствии с традициями латинского словообразования и введенного в ряде европейских стран одновременно в середине 60-х гг. XX в. (французское – informatique, немецкое – Informatik). В англоязычных странах эта же наука получила название «computer science» («вычислительная наука»), в котором выделяется вторая составляющая - вычисление, понимаемое в широком смысле как любое преобразование информации (алгоритм). Неправильный перевод «computer science» как «компьютерной науки» может привести к недоразумениям в понимании содержания информатики как науки. В обоих определениях информатики выделяется объект (информация), вычислительный процесс (обработка информации) и предполагается наличие технического средства (ЭВМ).

Приведенные фрагменты исторического развития информатики дают некоторое представление о том, что предмет информатики существовал и развивался до оформления информатики в науку, и даже до массового появления ЭВМ. Возникновение ЭВМ, их стремительное развитие и применение актуализировало проблематику информатики, выяснило ее своеобразие, придало ее проблемам масштаб и специфичность. Информатика сложилась в недрах математики и кибернетики, системотехники и электроники, логики и лингвистики. Основные научные направления информатики образуют такие дисциплины, как теоретические основы вычислительной техники, теория информации, теория вычислительного эксперимента, алгоритмизация и программирование. Языком каждого из названных направлений является математика.

Информация, являясь в своей всеобщности скорее общенаучной категорией, нежели конкретным термином, входит в информатику через ее контекстные спецификации. В этом отношении как фундаментальная наука информатика связана с философией – через учение об информации (общенаучной категории) и теорию познания; с математикой – через понятие математической (информационной) модели, математическую логику и теорию автоматов; с лингвистикой и математикой – через учение о формальном языке и знаковых системах.

Важнейшими методологическими принципами информатики является изучение природного явления или поведения объекта как процесса обработки информации, а также признание единства законов обработки информации в искусственных, биологических и социальных системах.

Основными видами интеллектуальной деятельности человека, изучаемыми в информатике, являются: математическое моделирование (фиксация результатов познавательного процесса в математическом виде); алгоритмизация (реализация причинно-следственных связей и других закономерностей в виде процесса обработки информации по формальным правилам); программирование (реализация на ЭВМ); выполнение вычислительного эксперимента (получение новых знаний с помощью вычислений на ЭВМ); решение конкретных задач, относящихся к кругу объектов и явлений, описываемых исходной моделью.

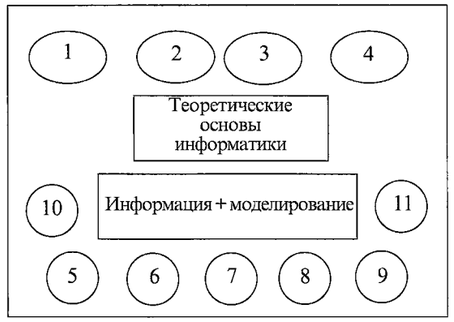
Таким образом, объектом информатики является информация, а методологией – моделирование (математическое, информационное, компьютерное) и проведение вычислительного эксперимента. Теоретическая подготовка информатика и круг его математических представлений должны соответствовать различным аспектам, связанным с объектом и методологией информатики.

С другой стороны, в связи с массовым применением ЭВМ предметом информатики становится изучение закономерностей взаимодействия человека и ЭВМ во всех видах его деятельности. Результаты этого изучения воплощаются в так называемых информационных технологиях, то есть в систематических методах и приемах применения ЭВМ в производственных процессах, управлении, образовании, научной работе, проектировании, сфере обслуживания и т.п. Практическое применение информатики постепенно формирует новый сектор народного хозяйства – индустрию информатики.

В момент становления информатики вопрос о соотношении информатики и математики в системе подготовки разрешался очевидным образом: информатика – часть математики и кибернетики. По мере возникновения новых направлений в информатике соответствующие предметы вводились в учебные планы (сначала в форме специальных курсов). И в определении информатики, и в ее методологии, и в истории возникновения явно отслеживается родство информатики и различных разделов математики. Сколько должно быть математики в подготовке специалистов по информатике? Ответ прост, но не реален – чем больше, тем лучше. В этом отношении наиболее продуктивен симбиоз математики и информатики в учебном плане подготовки специалиста «информатик, системный программист» специальности «010200 – «Прикладная математика и информатика...». Учебный план специальности «030100 – «Учитель информатики...» значительно меньше часов отводит предметам классической и прикладной математики, поэтому минимум часов, отданный этим предметам, должен быть

использован наиболее продуктивно. В первую очередь необходимо выделить основные направления и идеи, охватывающие все математические и прикладные математические науки и разделы информатики.

В учебный план подготовки «учителя информатики» в педагогических университетах включены следующие математические предметы:



1) математическая логика; 2) дискретная математика; 3) теория алгоритмов; 4) элементы абстрактной и компьютерной алгебры; 5) теория вероятностей и математическая статистика; 6) уравнения математической физики; 7) численные методы; 8) исследование операций; 9) компьютерное моделирование; 10) искусственный интеллект; 11) программирование. На рисунке схематически представлены все перечисленные предметы (номером 12 обозначена информационная технология).

Программирование (11) занимает промежуточное положение между прикладными математическими науками и информационными технологиями и выступает в роли важного связующего звена между всеми теоретическими и прикладными науками данного цикла. Искусственный интеллект можно считать специфическим разделом моделирования (логическое моделирование).

В предлагаемой схеме в верхнем ряду расположены предметы, которые объединяемые в некоторых рассмотрениях одним понятием – дискретная математика. Они являются теоретической основой информатики. Каковы отличительные черты этого направления математики? Первое, что необходимо отметить, – это дискретный характер всех объектов, рассматриваемых в этом разделе, – отсюда отказ от понятия предела и как следствие неприменимость методов классического математического анализа. Второе – конструктивный подход к любой проблеме.

Конструктивность – направление в математике, связывающее утверждение о существовании математических объектов с возможностью их построения, отвергающих в силу этого такие утверждения классической математики, как чистые теоремы существования, а также абстракцию актуальной бесконечности и ограниченность закона исключенного третьего. В качестве уточнения понятий конструктивности и эффективности выступает математическое понятие алгоритма. Развитие конструктивной математики, связанное в нашей стране с именем академика А.А. Маркова, характеризуется следующими чертами: 1) предметом изучения являются конструктивные процессы и возникающие в результате их выполнения конструктивные объекты (слова в некоторых алфавитах); 2) рассмотрение конструктивных процессов и объектов производится в рамках абстракции потенциальной осуществимости с полным исключением идеи актуальной бесконечности; 3) интуитивное понятие эффективности связывается с точным понятием алгоритма; 4) используется специальная, учитывающая специфику конструктивных процессов и объектов конструктивная логика. Таким образом, в этой части математической подготовки информатика основным является строгое понятие алгоритма над конструктивными объектами.

Вторая группа предметов (нижний ряд) представляет прикладные науки классического направления математики, без которых невозможно построение математических моделей изучаемых явлений или систем. Численные методы при этом играют роль мостика между непрерывными математическими моделями и их дискретными аналогами (дискретизация), для которых становится возможным создание алгоритмов.

Еще раз вернемся к программированию. Выделим основные аспекты, связанные с процессом программирования: 1) освоение языков программирования; 2) освоение программных сред; 3) отработка приемов и методов алгоритмизации; 4) решение конкретных задач, связанных с идеями как верхнего ряда предметов, так и нижнего.

В заключение следует отметить, что учебный план подготовки учителя информатики в теоретической части (классическая и прикладная математика и теоретические основы информатики) обладает внутренней логикой и единством подхода. Однако существующие стандарты программ по отдельным дисциплинам не всегда отражают это. Остается большой простор для варьирования наполнения конкретным содержанием почти всех рассмотренных предметов.

Появившиеся в последнее время публикации, посвященные вопросам соотношения математических и информационных предметов в подготовке учителей информатики и введения в учебные планы средней школы математических предметов, теоретически поддерживающих школьный курс информатики, требуют, по крайней мере, осторожной реакции. Попытки вводить в употребление новые термины и понятия, за которыми в науке уже закрепились определенные смысловые значения (например, конструктивная математика как часть математики, нужная информатикам), ни к чему хорошему (кроме путаницы!) не приведут.

Конечно, учитель информатики должен представлять себе соотношение теоретической информатики и математики и вводить в структуру школьной информатики те или иные элементы математики, необходимые для полноценного и осмысленного освоения предмета.

И наконец, нашим глубоким убеждением (подтверждаемым практикой) является необходимость проведения всех практических занятий по теоретической информатике и прикладной математике в компьютерных классах.

Н.А. Буимелева

КУРС «ВЫЧИСЛИТЕЛЬНАЯ ГЕОМЕТРИЯ С ЭЛЕМЕНТАМИ КОМПЬЮТЕРНОЙ ГРАФИКИ»

Одним из средств познания мира является компьютерное моделирование, содержательную основу которого должны составлять наиболее типичные компоненты информационной деятельности человека.

В последние годы одним из наиболее распространенных компонентов информационной деятельности специалистов различных профессиональных сфер является компьютерная графика. Изучение лежащей в ее основе вычислительной геометрии способствует более глубокому пониманию предмета и, следовательно, этого вида человеческой деятельности. Кроме этого «исторически и генетически геометрическая деятельность является первичной интеллектуальной деятельностью человечества в целом и каждого человека в отдельности. Геометрия – это не только раздел математики, школьный предмет, это прежде всего феномен общечеловеческой культуры, являющийся носителем собственного метода познания мира» [1, с. 8].

Исходя из этого представляется целесообразным изучение элементов вычислительной геометрии и компьютерной графики.

На факультете информатики Вятского государственного педагогического университета ведется преподавание курса «Вычислительная геометрия с элементами компьютерной графики» для студентов третьего года обучения.

Предметом изучения в курсе являются методы геометрического моделирования различных объектов, методы отображения графической информации.

Курс призван помочь студентам выработать правильное представление о роли и месте математики и информатики в современной цивилизации, показать применение компьютерных технологий во всех сферах человеческой деятельности, закрепить и углубить уже имеющиеся знания по информатике, привить практические навыки использования ЭВМ как инструмента учебной и практической деятельности.

Кроме того, курс должен способствовать повышению математической культуры студентов, пониманию того, что общественный прогресс невозможен без математического моделирования и точных количественных методов исследования, без широкого применения средств вычислительной техники и что сама математики является универсальным аппаратом исследования, средством решения прикладных задач, а также интегратором разных наук.

Объем дисциплины в виде учебной работы

Общая трудоемкость (по ГОС ВПО) – 144 ч.

Аудиторные занятия – 72 ч. Лекции – 36 ч.

Лабораторная работа – 36 ч. Самостоятельная работа – 72 ч.

Также предполагаются и другие виды работы (зачеты, НИРС, курсовые, рефераты).

Модульная структура курса «Вычислительная геометрия с элементами компьютерной графики»

Название модуля	Содержание модуля	
1. Математический аппарат	Элементы векторной алгебры	Векторы и их координаты, сложение, вычитание векторов, умножение вектора на число, скалярное и векторное произведения, угол между векторами на плоскости
	Аналитическая геометрия	Уравнения точек, прямых и др. на плоскости и в пространстве. Пересечение и объединение геометрических объектов. Операции над матрицами
	Дифференциальная геометрия	Касательная к кривой, ее уравнение. Нормальный вектор кривой и его уравнение
	Аффинные преобразования плоскости	Системы координат, связь между различными системами координат, аффинные преобразования, движения плоскости, подобие, композиция преобразований. Матричный вид формул аффинных преобразований. Технология использования композиций аффинных преобразований для изображения плоских фигур на экране компьютера
	Аффинные преобразования пространства	Системы координат, связь между различными системами координат, параллельный перенос, поворот вокруг прямой. Матричный вид формул аффинных преобразований пространства. Изображение пространственных фигур на экране компьютера
	Проективные преобразования плоскости и пространства	Параллельная и центральная проекции, перспектива
2. Основные геометрические алгоритмы	Выпуклость	Определение факта выпуклости заданного многоугольника, построение выпуклых оболочек множества точек и многоугольника
	Геометрический поиск	Локальный и региональный геометрический поиск и их приложения
	Ближайшие точки	Поиск ближайшей пары точек в заданном множестве, поиск всех ближайших соседей. Метод локусов в задачах о ближайших точках
	Геометрия прямоугольников	Периметр и площадь объединения прямоугольников. Пересечения прямоугольников и связанные с ними задачи

3. Алгоритмические основы компьютерной графики	Алгоритмы визуализации	Построение графиков функций одной и двух переменных. Алгоритмы удаления невидимых линий, отсечения отрезка и многоугольника
	Алгоритмы обработки изображений	Масштабирование изображений. Перспективы развития систем обработки изображений
Система фоновых заданий творческого характера		

Лабораторный практикум

Работа №1. Геометрические преобразования плоскости и пространства: поворот, перенос, масштабирование, композиция преобразований.

Работа №2. Изображение кривых, заданных параметрическими уравнениями. Построение графиков функций одной и двух переменных.

Работа №3. Удаление невидимых линий методом плавающего горизонта.

Работа №4. Отсечение отрезка прямоугольным окном. Алгоритм Коэна-Сазерленда и метод разбиения отрезка средней точкой.

Работа №5. Отсечение отрезка прямоугольным окном. Алгоритм Кируса-Бека. Задача отсечения многоугольника.

Работа №6. Построение выпуклой оболочки множества точек.

Работа №7. Построение выпуклой оболочки заданного многоугольника.

Работа №8. Определение факта принадлежности точки заданному многоугольнику.

Работа №9. Контрольная работа.

Работа №10. Реализация алгоритмов геометрического поиска.

Работа №11. Задачи нахождения ближайших точек и методы их решения.

Работа №12. Задачи о минимальном остовном дереве и триангуляции многоугольника.

Работа №13. Дерево отрезков и работа с ним. Построение, просмотр, вставка и удаление элементов.

Работа №14. Вычисление меры и периметра объединения прямоугольников.

Работа №15. Пересечения: плоские и трехмерные приложения.

Работа №16. Исследование и оценка сложности решения рассмотренных задач вычислительной геометрии (пересечение отрезков, принадлежность точки многоугольнику, построение выпуклых оболочек и др.).

Работа №17. Решение олимпиадных задач из области вычислительной геометрии.

Работа №18. Контрольная работа.

При введении данного курса в практику вузовского обучения будущих учителей информатики в качестве программных средств реализации рассматриваемых алгоритмов решаемых задач выбрана система программирования Turbo Pascal, так как язык программирования Pascal является, с одной стороны, достижением образовательной информатики, а с другой – основой современных профессиональных систем программирования, например Delphi.

Предложенная программа курса не заменяет и не повторяет классический курс геометрии. Предполагается, что геометрия изучается с опережением или параллельно данному курсу.

Разработанный курс служит средством реализации исследования предметной области вычислительной геометрии, интегрирующим звеном между теоретической подготовкой в области информатики и практической составляющей обучения студентов педагогического вуза по специальности «Информатика».

Эксперимент показал, что включение в учебный процесс педагогического вуза курса «Вычислительная геометрия с элементами компьютерной графики» позволяет совершенствовать информационные и математические знания студентов и подготовить их к разработке трехмерных приложений с помощью существующих пакетов и библиотек.

Программирование трехмерной графики является логическим продолжением курса «Вычислительная геометрия с элементами компьютерной графики». Современные библиотеки позволяют дизайнеру, конструктору, разработчику игр и видеоклипов и т.п. сосредоточить их внимание на том, что рисовать, используя заложенный в этих библиотеках собственный механизм визуализации и создавать на экране компьютера статические и динамические трехмерные изображения.

Литература

Шарыгин И.Ф., Ерганжиева Л.Н. Наглядная геометрия. М.: МИРОС, 1995.

И.А. Бабушкина

ПРОБЛЕМЫ ВНЕДРЕНИЯ ДОСТИЖЕНИЙ COMPUTER SCIENCE В ОБЛАСТИ ТЕХНОЛОГИЙ ПРОГРАММИРОВАНИЯ В УЧЕБНЫЙ ПРОЦЕСС

Computer Science, или компьютерные науки, – это группа учебных, научных и практических дисциплин, связанных с прикладной математикой, информационными технологиями, компьютерами и с несколькими смежными областями, в которых активно используется вычислительная техника. Наряду с такими дисциплинами, как архитектура вычислительных систем, операционные системы, базы данных, сети, информационная безопасность, компьютерная графика, мультимедиа, в состав Computer Science входит дисциплина «Программирование».

Программирование – дисциплина, которая занимается вопросами разработки эффективных и устойчивых программных комплексов, практическими сторонами теории алгоритмов, проектирования программ, теорией ошибок и методами оптимизации

и т.д. Программа подготовки будущих учителей информатики содержит дисциплину программирования. Наряду с изучением структурной и объектно-ориентированной парадигмы программирования, включенной в Государственный образовательный стандарт высшего профессионального образования по специальности «030100 – Информатика», в программу подготовки учителей информатики необходимо включить и знакомство с перспективными технологиями программирования, такими, как визуальное программирование, COM-технологии и другими.

На наш взгляд, система подготовки студентов педагогического вуза в области программирования должна иметь следующую структуру:

- Основы программирования (структурное программирование, программирование с использованием абстрактных типов данных, модельное программирование);
- Объектно-ориентированное и визуальное программирование.
- COM-технологии, включая OLE и ActiveX.
- Программирование баз данных.
- Программирование для Internet/Intranet и мультимедиа-технологии.

Выбранная последовательность изучения технологий программирования обоснована их взаимосвязью с другими дисциплинами (см. рисунок). Так, для изучения программирования баз данных по технологии клиент-сервер необходимо знание теории баз данных, умение администрировать Microsoft SQL Server, которое в свою очередь требует знаний TCP/IP и умения администрировать Windows NT (Windows 2000). Курс «Программирование в Microsoft Office (VBA)» преследует несколько целей: во-первых, позволяет максимально использовать возможности офисных программ, во-вторых, является подготовительным для использования VBA при дальнейшем изучении дисциплин предметной подготовки (например, курса Численные методы) и требует умения работать с программами пакета Microsoft Office, а также знаний в области объектно-ориентированного программирования.

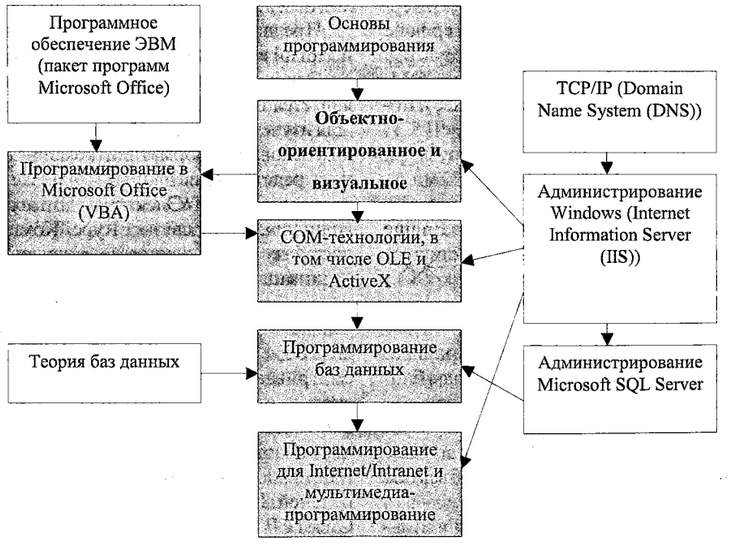
Выбор языка программирования. В последнее время для учебных целей используют два языка программирования Basic (QBasic, Visial Basic) и Pascal (Borland Pascal, система программирования Delphi). Анализируя положительные и отрицательные стороны этих языков, отметим некоторые преимущества языка программирования Pascal, способствующие развитию общей культуры программирования:

- строгая типизация языка Pascal – ни один объект не может использоваться без предварительного объявления;
- средства типизации и структуризации данных и программ имеют более развитой и концептуально завершенный характер – программист может объявлять новые типы данных для заданной предметной области, программы и процедуры обладают подобной структурой, процедуры и функции могут быть нескольких уровней вложенности.

Рассмотрим подробнее каждую из перечисленных выше дисциплин.

1. Основы программирования

Основные алгоритмические конструкции. Простые и составные типы данных. Структурное программирование. Процедуры и функции. Рекурсия. Ссылочный тип данных. Модульное программирование.



Взаимосвязь курсов подготовки учителей информатики

2. Объектно-ориентированное и визуальное программирование

Понятие класса и объекта. Инкапсуляция, наследование и полиморфизм. Раннее и позднее связывание. Виртуализация. Типовая информация периода исполнения. Жизненный цикл объекта: конструкторы и деструкторы.

Методы. Статические, виртуальные и динамические методы. Таблица динамических и виртуальных методов. Внутреннее представление объекта.

Понятие сообщения. Типы сообщений. Принципы работы системы сообщений Windows. Обработка сообщений. Языковые средства программирования, ориентированного на события. Поддержка механизма событий в Delphi. Событийно-ориентированное программирование пользовательских интерфейсов.

Этапы разработки компонента: выбор класса предка; создание модуля компонента; добавление в новый компонент свойств, методов и событий; тестирование; регистрация компонента в среде Delphi; создание файла справки для компонента.

Обработка исключительных ситуаций. Реализация механизма многотоковых приложений в Delphi.

3. Программирование в Microsoft Office (VBA)

Объекты и их семейства. Иерархия объектов. Основные объекты VBA, их свойства и методы.

4. COM-технологии

Основы COM (Component Object Model, модель многокомпонентных объектов). Компонентный класс, COM-объект. OLE-объект, OLE-контейнер, OLE-сервер. ActiveX-документ. Автоматизация, сервер автоматизации. Потоковые модели COM: однопоточная, раздельная, свободная.

Object Pascal и COM. Понятие интерфейса. Глобально-уникальный идентификатор (GUID). Интерфейс IUnknown. Фабрики классов. Интерфейс IClassFactory. Внутренние и внешние COM-серверы.

Агрегирование. Распределенная модель COM. Автоматизация. Интерфейс IDispatch. Раннее и позднее связывание.

Сервер транзакций Microsoft (MTS). Структура MTS. Пакеты. Технология MTS в Delphi.

5. Технология «клиент-сервер»

Процессор баз данных Borland DataBase Engine (BDE). Преимущества архитектуры BDE. Составные части BDE.

Иерархия объектов доступа к базам данных. Компоненты отображения данных.

Архитектура «клиент/сервер». Бизнес-правила. Модели клиент/сервер. Защита данных. Методы блокировки записей. Целостность данных. Транзакции. Язык SQL и его роль в технологии «клиент/сервер». База данных InterBase. Создание таблиц. Хранимые процедуры. Привилегии и права доступа к объектам баз данных.

6. Программирование для Internet/Intranet и мультимедиа-технологии

Обмен сообщениями. Динамический обмен данными. Именованные каналы (named piped). Сокеты. Схема работы с сокетами в Delphi-приложениях. Сокетный поток. Посылка/прием сложных данных. Алгоритм работы сокетного сервера.

Протокол HTTP. Язык HTML как средство создания информационных ресурсов Интернет. Язык JavaScript (VBScript) как средство создания интерактивных ресурсов.

Понятие мультимедиа. Мультимедиа как средство и технология. Создание мультимедийных приложений. Мультимедиа и Интернет.

Интерфейс Common Gateway Interface (CGI, общий интерфейс маршрутизации). Интерфейсы Internet Server Application Programming Interface (ISAPI) и Netscape Application Programming Interface (NSAPI).

Создание апплетов Java.

Построенная таким образом система изучения программирования обеспечит фундаментальную подготовку выпускника педагогического вуза в области технологий программирования.

Д.А. Чеканов

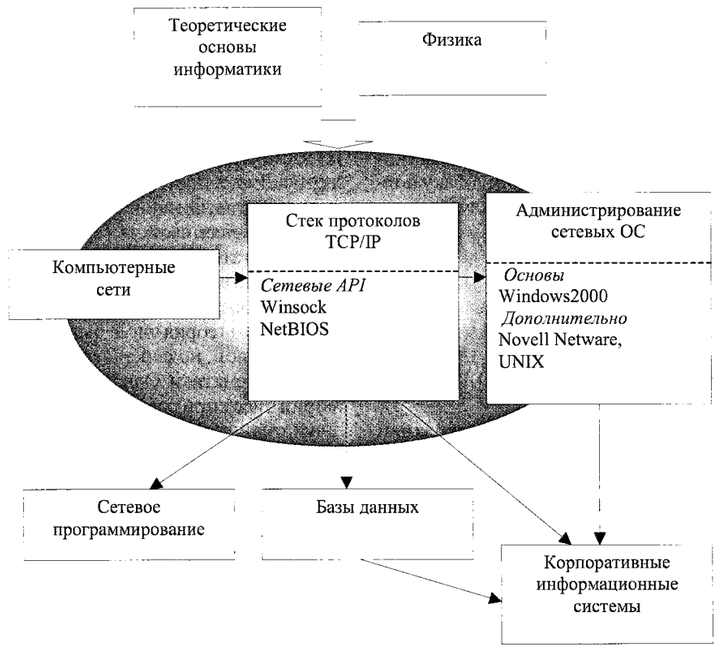
СПЕЦИАЛИЗАЦИЯ «КОМПЬЮТЕРНЫЕ СЕТИ» В ПРОГРАММЕ ОБУЧЕНИЯ ИНФОРМАТИКЕ

Курс по сетевым технологиям преподается во многих педагогических вузах. Но чаще всего методическое содержание такого курса имеет прикладной характер. Курс зачастую сводится к серии лабораторных работ по Интернету, к которым пытаются привязать теоретическую часть. Как мы считаем, такой подход неверен.

Предлагается организовать изучение сетевых технологий в вузе по схеме (см. рисунок).

Основой специализации по компьютерным сетям является теоретический курс «Компьютерные сети». Для изучения курса требуются знания по теоретическим основам информатики (комбинаторика, дискретное представление данных) и физике (область «электричество», темы «Электромагнитные волны», «Проводники», «Радиосвязь»). Курс «Компьютерные сети» дает базу для последующего изучения конкретных реализаций компьютерных сетей через стеки протоколов. Стек протоколов – это иерархически упорядоченная совокупность протоколов. Протокол – это правила обмена данными в компьютерной сети. Мы рассматриваем стек TCP/IP в качестве базового стека в силу его стандартизации, повсеместного распространения и открытости. Стек TCP/IP назван по имени двух основных протоколов, входящих в него. TCP – transmission control protocol, протокол управления передачей. IP – internet protocol, протокол интерсети. Стек TCP/IP разрабатывался в 80-х гг. Министерством обороны США.

В курсе «Стек TCP/IP» изучаются программные



интерфейсы (API) Winsock и NetBIOS. Winsock (Windows Sockets) – основной сетевой API в системе Win32 (Windows 9x/Me, Windows NT/2000/XP). NetBIOS – второй сетевой API, используется в Windows 2000/XP для совместимости с предыдущими версиями Windows в части разрешения имен.

С помощью этих интерфейсов возможно программирование под стек TCP/IP, следовательно, и изучение курса по сетевому программированию.

Знание стека TCP/IP также требуется для изучения курсов «Базы данных» и «Корпоративные информационные системы».

Все последние сетевые операционные системы в своей работе опираются на стек TCP/IP. Поэтому курс «Стек протоколов TCP/IP» дает базовую основу для изучения администрирования сетевых операционных систем. Под администрированием следует понимать управление операционной системой, учетными записями пользователей, серверными программами и сетью. За основу курса «Администрирование сетевых ОС» взята операционная система Windows 2000. В этом курсе изучаются основы сетевого администрирования и его реализация под Windows 2000. Следует отметить недавно особо выделившиеся направления в этом курсе: служба каталогов Active Directory и безопасность операционной системы. После изучения Windows 2000 реализуются практические курсы-дополнения по администрированию Novell Netware и нескольких видов Unix (Linux, BSD и т.д.) Зная основы сетевого администрирования Windows 2000, студенты смогут реализовать полученные знания под любой операционной системой. Курс по компьютерной безопасности является обязательным компонентом курсов «Компьютерные сети» и «Стек протоколов TCP/IP». Но только в ходе рассмотрения администрирования операционной системы дается основа курса компьютерной безопасности: безопасность операционной системы.

Курсы «Компьютерные сети», «Стек протоколов TCP/IP», «Администрирование сетевых ОС» предшествуют курсам «Сетевое программирование», «Базы данных» и «Корпоративные информационные системы».

Рассмотрим планирование курсов и особенности их изучения.

Курс «Компьютерные сети» является полностью теоретическим. Ориентировочная продолжительность – 60-70 часов. К ним добавляется 15 часов контрольных тестов.

Курс «TCP/IP» предусматривает около 40 часов лекционного материала, 10 часов контрольных тестов, 20 часов лабораторных работ.

Курс «Администрирование ОС на основе Windows 2000» предусматривает 60-70 лекционных часов, 15 часов контрольных тестов, 20-30 часов лабораторных работ (в зависимости от технических возможностей).

В ходе этих курсов студенты сдают контрольные

тесты на английском языке. Основой для этих тестов являются профессиональные сертификационные тесты таких корпораций, как Microsoft, Novell, Cisco и IBM. Для перечисленных выше курсов эти тесты были проанализированы и изменены с учетом методического содержания курсов.

В Вятском государственном педагогическом университете при подготовке специалистов на факультете информатики обучение строится по схеме, приведенной выше на рисунке.

Е.В. Разова

КУРС «ТЕОРИЯ ЧИСЕЛ И ЕЕ ПРИЛОЖЕНИЯ»

В последние годы в условиях информатизации общества сформировалась новая концепция структуры содержания общего среднего образования, произошло переосмысление роли и места учебной дисциплины информатики в системе школьных общеобразовательных дисциплин, вследствие чего в методической системе обучения информатике наблюдаются значительные изменения, связанные с переходом к многоэтапной структуре обучения этой дисциплине.

В проекте Федерального компонента Государственного образовательного стандарта для образовательной области «Информатика» [7] выделены три обязательных этапа обучения информатике: пропедевтический, базовый и профильный. Предполагается, что на третьем этапе должна произойти дифференциация обучения информатике по объему и содержанию материала в зависимости от профилей учебного заведения. Так, например, для учебных заведений и классов математического и естественнонаучного профиля предлагается осуществить углубленное изучение программирования и методов вычислительной математики [4, 7]. В качестве одного из таких профильных курсов информатики на старшей ступени школы для классов естественнонаучного и математического профилей может быть выбран курс «Теория чисел и ее приложения».

Теория чисел – старейшая область математики, занимающаяся изучением свойств целых, ее теория, методы и алгоритмы используются во многих разделах математики, информатики и их приложениях. Кроме того, для решения большинства задач теории чисел могут быть разработаны алгоритмы. Компьютерная же разработка эффективных алгоритмов требует модификации и совершенствования известных алгоритмов, что в свою очередь требует более глубокого понимания теоретико-числовых проблем.

Содержание этого раздела математики весьма обширно. К основным вопросам теории чисел относятся следующие [2].

1. *Теория делимости*: наибольший общий делитель (НОД), наименьшее общее кратное (НОК), простые числа, разложение числа на множители, алгоритм Евклида, линейное диофантово уравнение, основная теорема арифметики, непрерывные (цепные) дроби, подходящие дроби, их применение к решению уравнения $ax + by = c$, их связь с алгоритмом Евклида.

2. *Арифметические функции*: целая [x] и дробная {x} часть числа, каноническое представление числа n!, свертка Дирихле, мультипликативные функции, совершенные числа, функция Мебиуса, функция Эйлера.

3. *Сравнения*: свойства сравнений, китайская теорема об остатках, полная система вычетов, приведенная система вычетов, теоремы Эйлера и Ферма, китайская теорема об остатках.

Когда-то теория чисел была классическим примером красивой, но совершено бесполезной области чистой математики. Однако многие известные задачи теории чисел при рассмотрении с алгоритмической точки зрения превращаются в глубокие и притягательные открытые проблемы, причем нерешенной является не проблема вычисления, а проблема скорейшего вычисления, перечислим основные из них [1].

1. Алгоритм Евклида нахождения наибольшего общего делителя. Расширенный алгоритм Евклида.

2. Алгоритм разложения числа в цепную дробь.

3. Генерация простых чисел (решето Эратосфена).

4. Алгоритмы разложения числа на множители (факторизация числа), то есть представление целого числа n>0 в виде произведения ab=n, где a и b – целые числа. Эта задача включает в себя две различные задачи: тест на простоту, то есть проверка существования таких чисел a и b, и разложение, то есть нахождение этих чисел.

5. Алгоритм возведения числа в степень (используется при нахождении мультипликативного обратного элемента, то есть при решении уравнения $ax \equiv 1 \pmod m$).

6. Алгоритм решения линейных диофантовых уравнений $ax \equiv b \pmod m$.

С появлением и развитием вычислительной техники теоретико-числовые алгоритмы получили широкое применение. Приведем основные приложения теории чисел в информатике.

1. Свойства целых чисел используются в современном исследовании алгоритмов.

2. Одно из важнейших приложений теории сравнений – система счисления остаточных классов [1, 3, 5], в которой целое число x представлено набором его остатков (или вычетов) по взаимно простым модулям:

$Res(x)=(x \bmod m_1, \dots, x \bmod m_r)$, где $НОД(m_i, m_j)=1$ для $\forall$ i и j$\in$ [1, r].

Знакомые нам традиционные системы счисления являются линейными, позиционными, весовыми. Это означает, что всем позициям соответствуют веса, зависящие от одного основания – основания системы счисления. Вместо этого система счисления остаточных классов использует взаимно простые позиционные основания. Задать систему остаточных классов – значит указать количество и значение взаимно простых модулей. Например, основания (попарно простые модули) – β =[3, 5, 7]. Тогда, например, число 29 в данной системе счисления будет представлено следующим образом:

29=[29 mod β]=[29 mod 3, 29 mod 5, 29 mod 7]=[2, 4, 1].

Главное преимущество системы счисления остаточных классов состоит в отсутствии переносов при выполнении операций сложения, вычитания, умножения. Арифметика замкнута в каждой позиции, то есть согласно свойствам сравнений арифметические действия выполняются полностью и независимо в разных позициях. Например,

Пусть задана система остаточных классов β=[3, 5, 7].

Тогда 5=[2, 0, 5];

13=[1, 3, 6];

5+13=[(2+1) mod 3, (0+3) mod 5, (5+6) mod 7]=[0, 3, 4].

Действительно, 18 mod 3=0, 18 mod 5=3, 18 mod 7=4.

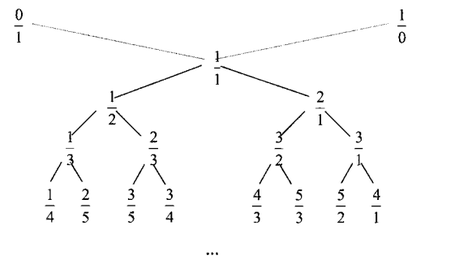
Приведем процедуру, реализующую умножение двух чисел в системе остаточных классов.

```
{n – количество модулей основания системы
   остаточных классов}
{m – массив, хранящий значение модулей основания
   системы остаточных классов}
Type number = Array [1..n] Of Byte; {запись числа в
                          системе остаточных классов}
Procedure Mult_in_soc(x, y : number; Var z : number);
Var i : Byte;
Begin
  For i:=1 To n Do
    z[i]:=(x[i]*y[i]) Mod (m_soc[i])
End;
```

Таким образом, время выполнения операции умножения двух чисел равно O(n), поэтому можно выполнять сложение и умножение длинных чисел так же быстро, как и обычных (коротких) чисел, что весьма полезно при вычислениях на компьютере, в том числе и на параллельных компьютерах.

3. Кроме представления целых чисел и организации эффективного выполнения арифметических операций над большими числами теория чисел используется при исследовании свойств рациональных чисел (последовательность Фарея – последовательность правильных дробей). Существует замечательный способ построения множества всех неотрицательных дробей m/n, где НОД(m, n)=1, носящий название дерева Штерна-Броко [3]. Суть этого способа заключается в том, чтобы начать с двух дробей $\frac{m}{n} = \frac{0}{1}$ и $\frac{m'}{n'} = \frac{1}{0}$, а затем повторить необходимое количество раз следующую операцию: вставить $\frac{m+m'}{n+n'}$ между двумя соседними дробями $\frac{m}{n}$ и $\frac{m'}{n'}$

Новая дробь называется медиантой дробей $\frac{m}{n}$ и $\frac{m'}{n'}$.



Ниже приведена программа формирования n уровней дерева Штерна-Броко.

```
Type pt=^tree;
   tree=Record
        data_c, data_z : Byte; {числитель и
                                знаменатель дроби}
        left, right : pt; {левый и правый потомок}
        End;
Var tsb : pt;
{процедура формирования дерева Штерна-Броко}
Procedure Init_tree(var t : pt; n, lc, lz, rc, rz : Byte);
{n - уровень дерева}
{lc,lz,rc,rz - числитель и знаменатель левого и правого предков}
Begin
  New(t);
 t^.data_c:=lc+rc;
 t^.data_z:=lz+rz;
 t^.left:=Nil;
 t^.right:=Nil;
  if n>1 Then Begin
        Init_tree(t^.left,n-1,lc,lz,t^.data_c,t^.data_z);
        Init_tree(t^.right,n-1,t^.data_c,t^.data_z,rc,rz)
End
End;
Begin
   Readln(n);
   New(tsb);
  Init_tree(tsb,n,0,1,1,0);
End.
```

Дерево Штерна-Броко можно рассматривать как систему счисления для представления рациональных чисел, поскольку каждая положительная несократимая дробь встречается в нем лишь один раз. Воспользуемся символами L и R для идентификации левой и правой ветви при прохождении вниз по дереву от корня к дроби. Например, для дроби $\frac{5}{3}$ необходимо проделать следующий путь: RLR – эта запись и есть запись числа $\frac{5}{3}$ в системе счисления Штерна-Броко.

4. Теоретико-числовые алгоритмы широко используются в различных криптографических схемах [1, 5], где нужны большие простые числа. Так, например, наиболее популярная – криптосистема RSA, разработанная в 1977 г. и получившая название в честь ее создателей: Рона Ривеста, Ади Шамира и Леонарда Эйдельмана, – основана на том факте, что нахождение больших простых чисел в вычислительном отношении осуществляется легко, но разложение на множители произведения двух таких чисел практически невыполнимо. Доказано (теорема Рабина), что раскрытие шифра RSA эквивалентно такому разложению. Поэтому для любой длины ключа можно дать нижнюю оценку числа операций для раскрытия шифра, а с учетом производительности современных компьютеров оценить и необходимое на это время.

Возможность гарантированно оценить защищенность алгоритма RSA стала одной из причин популярности этой системы на фоне десятков других схем. Поэтому алгоритм RSA используется в банковских компьютерных сетях, особенно для работы с удаленными клиентами (обслуживание кредитных карточек).

Рассмотрим теоретико-числовые результаты, положенные в основу этого алгоритма.

Теорема 1. (Малая теорема Ферма.)

Если p – простое число, то

$$x^{p-1} \equiv 1 \pmod{p} \qquad (1)$$

для любого x, не делящегося на p, и

$$x^{p} \equiv x \pmod{p} \qquad (2)$$

для любого x.

Определение. Функцией Эйлера $\varphi(n)$ называется число положительных целых, меньших n и простых относительно n.

N	2	3	4	5	6	7	8	9	10	11	12
$\varphi(n)$	1	2	2	3	2	6	4	6	4	10	4

Теорема 2. Если $n=p\cdot q$ (p и q – отличные друг от друга простые числа), то $\varphi(n)=(p-1)\cdot(q-1)$.

Теорема 3. Если $n=p\cdot q$ (p и q – отличные друг от друга простые числа) и x – простое относительно p и q, то $x^{\varphi(n)} \equiv 1 \pmod{n}$.

Следствие. Если $n=p\cdot q$ (p и q – отличные друг от друга простые числа) и e – простое относительно $\varphi(n)$, то отображение

$$E_{e,n}: x \rightarrow x^{e} \pmod{n}$$

является взаимно однозначным на Z_n.

Очевиден и тот факт, что если e – простое относительно $\varphi(n)$, то существует целое d, такое, что

$$e\cdot d \equiv 1 \pmod{\varphi(n)}. \qquad (3)$$

Пусть $n=p\cdot q$, где p и q – различные простые числа. Если e и d удовлетворяют уравнению (3), то отображения $E_{e,n}$ и $E_{d,n}$ являются инверсиями на Z_n. Как $E_{e,n}$, так и $E_{d,n}$ легко рассчитываются, когда известны e, d, p, q. Если известны e и n, но p и q неизвестны, то $E_{e,n}$ представляет собой одностороннюю функцию; нахождение $E_{d,n}$ по заданному n равносильно разложению n. Если p и q – достаточно большие простые числа, то разложение n практически не осуществимо. Это и заложено в основу системы шифрования RSA. Именно отсутствие на сегодняшний день эффективного алгоритма разложения чисел на множители позволяет использовать систему RSA. Если такой алгоритм найдется, вся эта система рухнет в одночасье.

Таким образом, курс «Теория чисел и ее приложения» концентрирует в себе достаточно большой теоретико-числовой материал и методы информатики, применяемые для эффективного решения задач теории чисел и ее приложений. Он несет в себе огромный образовательный потенциал, материал курса позволит повысить математическую и алгоритмическую культуру, привить устойчивый интерес и к математике, и к информатике через показ красоты, как самой математической теории, так и эффектное решение с помощью компьютера конкретных проблем теории чисел и ее приложений.

Определенную роль в обучении информатике играет программирование, представляющее собой деятельность, которая в узком смысле сводится к кодированию рассматриваемого алгоритма, а в широком – является методологией информатики, то есть вычислительным экспериментом. Использование программирования при изучении курса «Теория чисел и ее приложения» выступает именно как средство организации обучения этому предмету, оно открывает большие возможности расширения круга задач, поскольку позволяет использовать в обучении, во-первых, задачи, требующие значительных вычислительных затрат для их решения, во-вторых, задачи, требующие использования комбинаторных методов решения. На практике часто приходится иметь дело с очень большими числами, и это также становится возможным благодаря использованию компьютеров. Но поскольку целые числа в различных языках программирования представлены ограниченными диапазонами, то возникает необходимость организации представления, хранения и обработки таких чисел. Эти вопросы и соответствующие алгоритмы должны быть рассмотрены в рамках данного курса на практических занятиях.

Программирование как средство организации обучения при изучении курса должно использоваться при проведении лабораторных работ. Оно стимулирует активность и самостоятельность обучаемых, способствует индивидуализации их деятельности на занятии, развитию обучаемого через преодоление своих ошибок благодаря постоянной обратной связи, поддерживаемой компьютером. Оно способствует большей осознанности изучаемого теоретического материала, поскольку требует от обучаемых четкости в формализации используемого материала, точности в выражении своих мыслей и т.п. Кроме этого программирование позволяет сформировать начальные навыки для будущей профессиональной деятельности.

Успех изучения курса во многом зависит от целесообразно подобранной системы задач. Использование специально разработанной системы упражнений наилучшим образом позволит усвоить и закрепить полученные теоретические знания, на основе которых построен любой алгоритм решения задач из области теории чисел и ее приложений.

Разработка эффективных методик проведения занятий позволит в достаточной мере реализовать принципы проблемного обучения, когда большинство теоретических положений естественно вытекает из анализа поставленной практической задачи и может быть проверено учеником при разработке соответствующей программы, даст возможность активизировать познавательные процессы, что в конечном итоге будет способствовать развитию творческого мышления.

Построенный таким образом курс «Теория чисел и ее приложения»

- обеспечит единство научного начала с прикладным и учебным;
- будет способствовать усилению межпредметной связи изучаемого курса с другими науками и иметь высокую прикладную направленность;
- позволит выработать у школьников правильное представление о роли и месте математики и информатики, показать применение компьютерных технологий в различных сферах человеческой деятельности;
- будет способствовать повышению математической культуры учащихся, формированию исследовательских умений;
- позволит закрепить и углубить имеющиеся знания по информатике, привить практические навыки использования ЭВМ как инструмента учебной и практической деятельности.

Литература

1. *Акритас А.* Основы компьютерной алгебры с приложениями / Пер. с англ. М.: Мир, 1994.
2. *Виноградов И.М.* Основы теории чисел. М.: Наука, 1982.
3. *Грэхем Р., Кнут Д., Паташник О.* Конкретная математика. Основание информатики / Пер. с англ. М.: Мир, 1998.
4. Концепция содержания обучения информатике в 12-летней школе // Информатика и образование. 2000. №2. С. 17-22.
5. *Кормен Т., Лейзерсон Ч., Ривест Р.* Алгоритмы: построение и анализ. М.: МЦНМО, 2000.
6. Проект концепции структуры и содержания общего среднего образования (в 12-летней школе) // Информатика и образование. 2000. №1. С. 19-26.
7. Проект федерального компонента Государственного образовательного стандарта начального общего, основного общего и среднего (полного) образования. Образовательная область «Информатика» // Информатика и образование. 1997. №1. С. 3-11.

Т.В. Ашихмина

МЕТОДИКА ПРЕПОДАВАНИЯ ТЕМЫ «ДИНАМИЧЕСКИЕ СТРУКТУРЫ ДАННЫХ»

Изучение темы «Динамические структуры данных» удобно начать с обобщения ранее изученного материала (статические структуры данных) и построения классификации всех существующих в языке программирования Pascal структур данных с рассмотрением примеров каждой из структур. Такой подход дает возможность обсудить необходимость изучения данной темы и ее место в курсе в целом.

Первый раздел данной темы посвящен изучению ссылочного типа данных, множеством значений которого являются адреса ячеек. Для предотвращения часто допускаемых типичных ошибок, связанных с использованием качественно нового для студентов типа, предлагается на первом этапе обучения схематично прорисовывать каждое действие программы. Отсюда возникают два вида заданий для закрепления и контроля знаний. Во-первых, определить, какая структура будет создана после выполнения предложенных команд. Во-вторых, описать тип переменной и записать последовательность команд, создающую изображенную на рисунке конструкцию.

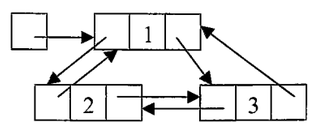
Пример задания первого типа.

Нарисовать структуру значения переменной p после выполнения следующих операторов:

```
type c=^char;
  a=^r;
  r=record
    f1:c;
    f2:a;
   end;
var p,q:a;x:c;
  .....
new(q); new(p); new(x);x^:='a'; p^.f1:=x;p^.f2:=q;
new(x);x^:='b'; q^.f1:=x;q^.f2:=nil;
```

Пример задания второго типа.

Описать переменную p (и, если понадобится, вспомогательные переменные) и выписать операторы, создающие изображенную на рисунке структуру.



Кроме «прорисовки» всех действий над переменными вновь изученного типа для полного понимания нового материала необходимо научить студентов (в особенности педагогических специальностей) грамотно словесно описывать эти действия. Привить этот навык можно в процессе закрепления и контроля полученных знаний. Во-первых, вызывать по возможности каждого студента к доске для выполнения за-

даний первого и второго типов. Во-вторых, в период сдачи студентом отчета по индивидуальным задачам с объяснением описанных в программе действий. Каждый студент должен уметь различать динамические переменные и переменные-указатели на них, а также правильно описывать результат команды присваивания (например, фразой «сейчас переменная... указывает туда же, куда и переменная...»).

Кроме заданий, которые выполняются в тетради или на доске, в первом разделе данной темы могут быть предложены и задачи для решения на компьютере. Задачи в своем решении должны использовать как известные статические типы данных (массивы, записи), так и соответствующие им динамические. Подобрать задачи нужно так, чтобы их решение указывало студентам на различия между такими структурами, например, как массив из указателей на целые числа и указатель на массив из целых чисел, где динамической переменной является либо целое число, либо массив.

Следующие разделы данной темы посвящены изучению непосредственно динамических структур данных.

Это линейные однонаправленные списки, очереди, стеки, деки, деревья и графы.

При изучении каждой структуры предлагается сначала дать общее понятие каждого термина, т.е. описать как последовательность элементов или как иерархическую конструкцию, обладающую некоторыми характеризующими их свойствами, не ссылаясь при этом на язык программирования. Затем только пояснить, что такая конструкция может быть реализована в программе с помощью различных структур данных (в том числе и статических, например массива). После этого, перечислив все преимущества выбора, останавливаемся на одной из динамических структур. Для изучения деков можно использовать двунаправленные линейные или кольцевые списки, для изучения стеков и очередей – линейные однонаправленные, с пояснениями, где (в начале или в конце списка) расположены вершина у стека или начало и конец у очереди.

В практических занятиях по каждому из этих разделов можно выделить две части. В первой части рассматриваются и решаются стандартные для этой структуры данных задачи, решение которых разобрано в большинстве учебников и методических пособий. Эти задачи могут быть общими для всех студентов. Основная цель на первом этапе – научить писать программу по уже известному алгоритму и получить представление о том, для решения какого типа задач может быть использована изучаемая структура.

На второй части практических занятий студентам предлагаются индивидуальные задачи, решаемые с использованием изученной динамической структуры данных. Это задание может быть предложено для самостоятельного выполнения полностью или с предварительным обсуждением между студентом и преподавателем алгоритма решения задачи без применения компьютера до составления программы на языке программирования. Тот или иной подход выбирается в зависимости от успеваемости студентов. При этом можно рекомендовать индивидуальный подход и для каждого студента подобрать свой уровень сложности задач и степень самостоятельности их решения.

В качестве зачетной проверочной работы по данной теме может быть проведена контрольная работа в безмашинном варианте, содержащая как теоретические вопросы, так и задания написать процедуру или функцию, решающую задачу не высокого уровня сложности.

Пример варианта такой контрольной работы.

1. Какие структуры данных называются динамическими? Приведите примеры.

2. Описать процедуру, объединяющую два линейных списка следующим образом: в полученном итоговом списке элементы первого и второго списков чередуются. Данные списки одинаковой длины.

3. Дан стек, элементами которого являются слова, представленный линейным списком, и очередь, элементами которой являются буквы. Написать процедуру вывода слов из стека, начинающихся на буквы, содержащиеся в очереди.

4. Написать процедуру, которая выводит на экран самый правый элемент левого поддерева и самый левый элемент правого поддерева.

5. Описать процедуру, которая выводит значения двух элементов дека, стоящего после первого вхождения заданного элемента и стоящего перед последним вхождением того же заданного элемента.

Хотелось бы отметить, что решение задач по этой теме очень успешно позволяет повторить и закрепить знание статических структур данных (например, записей) и понятие рекурсивного алгоритма (например, при изучении деревьев). Также при подборе задач возможно и необходимо как можно чаще ссылаться на изученные ранее типы данных (строки, массивы, файлы). Например, в большинстве задач возможно предъявление требования того, чтобы входные данные содержались в одном файле, а результат сохранялся в другом.

При такой методике преподавания темы «Динамические структуры данных» для студентов снижается уровень сложности при первом знакомстве с качественно новым типом данных и появляется уверенность при его использовании в задачах.

В.А. Онегов, М.В. Овсянникова

ИССЛЕДОВАНИЕ ОПЕРАЦИЙ. ПРОБЛЕМЫ И ОСОБЕННОСТИ ЕГО СОВРЕМЕННОГО ПРЕПОДАВАНИЯ

Исследование операций – научный метод выработки количественно обоснованных рекомендаций по принятию решений (см. [1], [2]). С принятием решений тесно связано понятие операции, включенное в

название дисциплины. Операция – целенаправленное действие, которое приводит к некоторому результату. Выбор операции и исполнение ее (или предписание на исполнение какому-то исполнителю) осуществляет субъект управления. Ясно, что в распоряжении субъекта управления находится определенный набор операций (допустимые операции), которые он может реализовать. Допустимость операции определяется двумя моментами: 1) количеством и типом параметров, которыми описывается математическая модель системы управления; 2) возможностью распоряжаться значениями параметров системы. Выбор операции (управляющего действия) осуществляется не произвольно, а так, чтобы в результате ее исполнения была достигнута некоторая цель, то есть система пришла в состояние, которое в определенном смысле для управляющего наиболее предпочтительно. Цель обычно ставится в форме оптимизации некоторой функции (функции цели). Задача нахождения оптимума целевой функции – это задача на экстремум (как правило, условный).

Круг задач, относящихся к исследованию операций, чрезвычайно широк. Сюда традиционно относятся: различного типа экономические задачи, сетевое планирование, задачи управления войсками, задачи социально-гуманитарные характера, задачи управления динамическими системами и многие другие.

Сфера практических приложений теории и методов исследования операций постоянно расширяется. Одновременно развивается и сама наука, появляются новые ее направления, которые претендуют на роль самостоятельных наук. Чтобы понять всю масштабность ситуации, достаточно назвать некоторые из них: выпуклое программирование (включая линейное и квадратичное), многокритериальные задачи, дискретное программирование, динамическое программирование, теория игр, теория принятия решений в вероятностной или плохо определенной системе. Список ученых, внесших свой вклад в эту науку и, часто, являющихся основателями целых ее направлений, достаточно впечатляющ. Это выдающиеся математики и прикладники XX века: А.А. Марков, Джон фон Нейман, Р. Беллман, Л.С. Понтрягин, лауреаты Нобелевской премии Дж. Данциг и Л.В. Канторович и многие другие.

Если рассматривать «Исследование операций» в широком смысле, то ее содержание составляют: математическое моделирование в его оптимизационном варианте (в отличие от дескриптивного), математические методы решения задач (связанных с математической моделью) на экстремум, реализация математических методов на ЭВМ, *принятие решений на исполнение оптимальных операций*. Потребность решения оптимизационных задач, содержащих огромное количество числовых данных, явилась одним из стимулов создания первых ЭВМ.

Таким образом, «Исследование операций» представляет собой яркий пример науки интегрального характера. В этом отношении исследование операций естественным образом согласуется с методологическими приемами информатики. Можно даже считать решение задач, возникающих в теории и практике исследования операций, полем приложения методологии информатики, заключающейся в реализации цепочки: математическая модель, вычислительная модель (метод), алгоритм, ЭВМ.

На фоне сказанного совершенно естественным явилось введение дисциплины «Исследование операций» в число дисциплин специальной подготовки студентов (специальность – учитель информатики). Министерским стандартом предусмотрено 72 часа аудиторных занятий на эту дисциплину. Принятая программа-стандарт по исследованию операций содержит традиционный набор разделов: линейное программирование, транспортная задача, многокритериальные задачи, нелинейное программирование, динамическое программирование, элементы теории игр и теория массового обслуживания. Одним из авторов настоящей статьи была предложена параллельная программа, которая отличается от стандарта только двумя пунктами: введен небольшой раздел – задачи переборного характера, и исключен – теория массового обслуживания. Мы считаем, что теорию массового обслуживания следует рассматривать в рамках дисциплины «Компьютерное моделирование» в разделе «Стохастическое моделирование», тем более что стандарт по компьютерному моделированию это предусматривает. В основном, мы следуем принятому стандарту.

На заседании совета УМО по информатике педагогических вузов вызвала дискуссию методика проведения практических занятий по исследованию операций (как и другим математическим дисциплинам прикладного характера). Математики, которые в основном разрабатывали программы по прикладным дисциплинам математики, настаивали на традиционных практических занятиях (так, как они проводятся по дисциплинам чистой математики). Нам кажется, что для всех дисциплин прикладной математики (исследование операций, теория вероятностей и математическая статистика, численные методы, уравнения математической физики) практические занятия должны проводиться в форме лабораторных занятий в компьютерных классах. Этот подход был поддержан профессором М.П. Лапчиком и доцентом С.А. Ждановым. Мы твердо намерены придерживаться этого подхода.

Ниже воспроизводится фрагмент примерной программы по дисциплине «Исследование операций», предложенной на заседании совета учебно-методического совета по информатике педагогических вузов.

1. Пояснительная записка

Цель курса – познакомить с понятием «операция» и методами оптимизационного моделирования. В структуру курса входят традиционные разделы исследования операций: линейное программирование, це-

лочисленное линейное программирование, оптимизационные задачи переборного характера, многокритериальные задачи, нелинейное программирование (наиболее полно – квадратичное программирование), динамическое программирование и элементы теории игр. Для поддержки лабораторных работ предлагается использовать пакет программ, составленных на Pascal. Для самостоятельной работы рекомендуется использовать индивидуальные задания с обязательным письменным отчетом по основным разделам.

2. Объем дисциплины в виде учебной работы

№	Раздел дисциплины	Лекции	П	ЛР
1	Введение. Предмет, история, операция, оптимизационное моделирование, примеры задач	2		
2	Линейное программирование	8		10
3	Метод Монте-Карло для целочисленной злп	2		2
4	Транспортная задача. Метод потенциалов	2		2
5	Оптимизационные задачи переборного характера	2		2
6	Многокритериальные задачи	2		2
7	Нелинейное программирование	6		6
8	Динамическое программирование	6		6
9	Элементы теории игр	6		6

3. Содержание дисциплины

Вид занятий	Всего часов
Общая трудоемкость (по ГОС ВПО)	144
Аудиторные занятия	72
Лекции	36
Практические занятия (семинары)	36
Самостоятельная работа	72

Литература

1. Математический энциклопедический словарь. М.: Сов. энцикл., 1988.

2. *Онегов В.А.* Исследование операций: задачи, методы, алгоритмы: Учебное пособие для студентов высших учебных заведений, обучающихся по спец. 030100 – Информатика. Киров: Изд-во ВГПУ, 2001.

М.В. Петухова

ПРЕПОДАВАНИЕ ИНФОРМАТИКИ НА ГУМАНИТАРНЫХ ФАКУЛЬТЕТАХ ПЕДВУЗА

Сейчас никто уже не сомневается в необходимости обучения специалиста любого профиля основам информатики. Причем не столько основам работы на компьютере, сколько основам информатики именно как науки. Одна из важнейших задач современной системы образования – подготовить людей к жизни и профессиональной деятельности в информационной, компьютеризированной среде общества, научить их эффективно использовать ее возможности и защищаться от негативных воздействий. Для решения этой задачи необходима информационная ориентация системы образования. Необходимо внедрение современных информационных технологий (ИТ) во все сферы учебно-воспитательного процесса, и не только на уроках информатики. Многие современные исследователи показали, что применение средств информационно-компьютерных технологий при изучении основ наук повышает эффективность учебного процесса в плане овладения умением самостоятельного извлечения и представления знаний, овладения общими методами познания и стратегией усвоения учебного материала, самостоятельного выбора режима учебной деятельности, организационных форм и методов обучения [8].

Но большинство педагогов не готовы работать в условиях информатизации общества. И не только потому, что техническое оснащение школ оставляет желать лучшего – со временем ситуация меняется (уже сейчас осуществляется программа обеспечения сельских школ компьютерной техникой). Главные же причины – неподготовленность преподавателей и отсутствие обобщенных подходов к реализации возможностей информационных технологий. Задача курса информатики в педвузе в этой ситуации – обеспечить соответствующую подготовку студентов всех специальностей.

Курс информатики в вузе призван решать две группы задач – **мировоззренческие** и **практические**.

Посмотрим на преподавание информатики с **мировоззренческих** позиций.

Любая дисциплина – не самоцель, а прежде всего средство развития личности. Информационная культура – неотъемлемая часть профессиональной и общей культуры человека. Именно с таких позиций необходимо подходить к системе образования в современном обществе. Исходя из этого выделяют следующие основные черты формирующейся новой системы образования [4]:

1) переход от «обучения» (дающего «знания-описания») к «образованию» (дающему «знания-инструменты»);

2) фундаментализация образования и развитие творческих способностей личности;

3) применение новых информационных технологий в процессе отбора, накопления, систематизации и передачи знаний.

Первостепенную роль здесь должны играть дисциплинарные и междисциплинарные курсы, содержащие наиболее фундаментальные знания, являющиеся базой для общей и профессиональной культуры, т.е. такие знания, которые способны формировать широкий, целостный взгляд на мир и позволяют преодолеть предметную разобщенность.

Курс информатики, как никакой другой, носит ярко выраженный междисциплинарный характер. Он имеет громадный резерв для формирования общеучебных и интеллектуальных умений, творческого и операционального мышления. Вместе с этим в нем находят отражение и гуманитарные знания, нравственные и правовые проблемы, возникающие в информационном обществе. В то же время неразрывна связь информатики с математикой, которая обеспечивает аппаратом многие науки (не только информатику, а, например, и физику).

Возможно, именно из-за такой двойственности (собственно предмет изучения и междисциплинарность) информатику в школе пытались включить то в математическую область знания, то в образовательную область «Технология». Сейчас аналогичная ситуация в вузе – по новым стандартам введен единый курс «Математика и информатика». Но, несмотря на это, информатика – самостоятельная образовательная область, в смысле фундаментальности ничуть не уступающая математике.

Нельзя не посмотреть на проблему подготовки учителей любой специальности по информатике и с **практических** позиций.

Информатика тесно связана с практикой, но, исходя именно из этого, необходимо как можно скорее уйти от преобладания «кнопочно-технологической идеологии» [1].

Многие специалисты в области образовательной информатики говорят сейчас о смене парадигм в системе общего и профессионального образования. Если раньше был заказ общества на компьютерную грамотность специалиста, то сейчас – на освоение информационных ресурсов, системное видение информационных процессов, их автоматизацию и управление [1], [6].

Процесс информатизации общества требует и информатизации образования, школа должна стать информационной, т.е. в основе имеющей «информационно-технологическую среду с развивающейся архитектурой учебно-познавательного пространства» [9], учитывающую взаимодействие учащихся, преподавателей и администрации между собой и с информационным обществом.

В связи с этим можно выделить практические **задачи**, которые должны уметь решать члены информационного, компьютеризированного общества, ныне студенты, в дальнейшем – учителя различных предметов, администраторы образовательных учреждений.

Во-первых, это применение современных информационных технологий в своей учебной и научной деятельности. Речь идет, например, о подготовке рефератов, курсовых и квалификационных работ, ведении научных и практических исследований, причем не только об оформлении материалов, но и о поиске нужной информации с использованием современных баз данных, сетей, обработке различных данных, представленных в табличном виде и т.п.

Во-вторых, это использование ИТ в будущей педагогической деятельности. Здесь можно выделить следующие направления:

a) процесс обучения, т.е. применение обучающих программ по своему предмету, использование информационных технологий общего назначения для представления и обработки данных;

b) предметная подготовка учителя к урокам, кружковым и факультативным занятиям. Это может быть поиск информации, обработка и представление данных, оформление дидактического материала и т.п.;

c) применение информационных технологий для ведения документации, необходимой учителю-предметнику и классному руководителю (электронный журнал, конспекты уроков, отчеты, представление и хранение данных об учениках, родителях и т.п.).

В-третьих, применение ИТ в управленческой, административной деятельности (использование программ общего и специального назначения, средств коммуникации и т.д.).

Понятно, что для квалифицированного решения такого рода задач явно недостаточно обучить студентов работе с некоторыми программными средствами, реализующими основные ИТ. Здесь необходимы навыки информационного моделирования, формализации задач, знания общих принципов реализации информационных процессов современными ИТ.

Таким образом, основной практической задачей курса информатики в педагогическом, как, собственно говоря, и в любом, вузе является подготовка студентов к практической деятельности и дальнейшему самообразованию с использованием современных средств и методов работы с информацией.

Чтобы обеспечить полноценное обучение информатике будущих учителей различных специальностей, необходимо сформулировать общие идеи, цели, содержание, принципы организации учебного процесса.

Общие идеи системы обучения информатике для гуманитарных специальностей можно сформулировать так:

1) информатика – фундаментальная дисциплина. Но специфика этой дисциплины для гуманитарных специальностей такова, что, не предоставляя в полной мере фундаментального знания в своей области, она должна в первую очередь способствовать формированию у студентов научного способа мышления, создания общенаучной картины мира. Что предполагает формирование знаний по теоретическим вопросам, связанным с информационным моделированием, системным видением информационных процессов, их автоматизации и управления;

2) в обучении информатике должна быть предметная и педагогическая ориентация, т.е. прослеживаться связь со специальностью студента;

3) при обучении информатике должны быть сформированы общие навыки решения практических (учебных, специальных, педагогических, научных) задач с использованием современных ИТ.

Соответственно можно выделить следующие **цели** обучения информатике студента-гуманитария (основываясь на [7]):

• подготовить полноценного члена информационного общества, т.е. подготовить студента к жизни в информационной, компьютеризированной среде общества, научить эффективно использовать ее возможности и защищаться от негативных воздействий;

• подготовить специалиста, т.е. обучить методологии познания, основанной на современных методах информационных процессов, предполагающих использование ИТ;

• подготовить учителя, который будет работать с детьми нового поколения, не мыслящими свою жизнь без компьютера. А значит, и педагогические технологии должны соответствовать уровню детей.

Что касается **содержания**, то, поскольку в нынешних учебных планах введен единый курс «Математика и информатика», важно рассмотреть не просто сумму двух предметов (как сейчас чаще всего делается), а их синтез. Математика, без сомнения, обеспечивает информатику аппаратом, но обе эти дисциплины носят фундаментальный характер. Они имеют достаточно большую область пересечения, которая и должна стать основой содержания.

Можно предложить, например, такой вариант содержания и распределения часов дисциплины «Математика и информатика» (опираясь на Государственный стандарт):

Примерная программа

1. Введение. Математические методы в целенаправленной деятельности. Информация, ее свойства. Информационные процессы.

2. Представление информации. Язык, алфавит, кодирование. Системы счисления. Множества и операции над ними. Объекты и отношения. Структуры и графы.

3. Моделирование. Понятие модели. Виды моделей. Информационная модель. Математическая модель. Алгебра высказываний как интерпретация модели отношений.

4. Измерение информации. Семантический подход к измерению информации. Вероятностный подход к измерению информации. Единицы измерения информации.

5. Компьютер как средство хранения, обработки и передачи информации. Архитектура и общие принципы работы компьютера. Программное обеспечение ЭВМ. Этапы решения задач на ЭВМ. Алгоритмизация.

6. Обработка информации. Обработка текстовой информации. Обработка графической информации. Обработка табличной информации. Базы данных.

7. Передача информации. Общие принципы передачи информации с помощью современных средств. Компьютерные сети и телекоммуникации.

Примерное распределение аудиторных часов по темам

№	Тема	Лекции	Практические	Всего
1	Введение	2	0	2
2	Представление информации	6	4	10
3	Моделирование	4	4	8
4	Измерение информации	2	2	4
5	Компьютер как средство хранения, обработки и передачи информации	8	6	14
6	Обработка информации	4	18	22
7	Передача информации	2	2	4
	ВСЕГО	**28**	**36**	**64**

Важным моментом при формировании содержания дисциплины должна стать преемственность со школьным курсом. Здесь одна из главных проблем – различный уровень знаний и умений абитуриентов по информатике. Ситуация, конечно, будет меняться, но процесс этот, особенно для провинции, длительный. Поэтому в зависимости от уровня подготовки студентов по информатике (с математикой ситуация более стабильна) акценты в содержании можно смещать. Например, если ориентироваться на будущее (когда в полной мере будут выполняться требования к подготовке выпускника средней школы по информатике), то в качестве основного направления информатики для гуманитариев в вузе можно выбрать информационное моделирование и базы данных.

Основными **принципами организации учебного процесса** по информатике возможно назвать следующие:

1. Лекционные занятия – синтез математики, информатики и предмета (по специальности студента).

2. Практические занятия не разделять на семинарские и лабораторные, по возможности все проводить в компьютерном классе.

3. Индивидуальный подход – разные степени сложности задач, обеспечение студентов дифференцированным дидактическим материалом для изучения в индивидуальном темпе (с учетом разного начального уровня подготовки по информатике).

4. Задачи, связанные со специальностью студента, в том числе с педагогической деятельностью, по возможности «сквозные».

При обучении информатике будущих учителей гуманитарных специальностей необходимо преодолеть сложившийся стереотип, важно осознать, что изучать в основном «нажатие на клавиши», т.е. сужать задачу только до взаимодействия с компьютером, – путь совершенно не перспективный. Будущее информатизации школы не только за учителями информатики, но в значительной мере за учителями всех специальностей, способными обучать в условиях информационной среды.

Литература

1. *Бешенков С.А.* Новые составляющие нашего мировоззрения // ИНФО. 1999. №10.
2. *Дейнеко С.В.* Методика обучения информатике учащихся вузов // ИНФО. 2000. №4.
3. *Ефимов В.И.* Информатизация образования: шаг вперед – два шага назад // ИНФО. 2000. №6.
4. *Кинелев В.Г.* Контуры системы образования XXI века // ИНФО. 2000. №5.
5. *Колин К.К.* О структуре и содержании образовательной области «Информатика» // ИНФО. 2000. №10.
6. *Кузнецов А.А.* О концепции содержания образовательной области «Информатика» в 12-летней школе // ИНФО. 2000. №7.
7. *Морозова И.Ю.* Курс информатики на филологическом факультете педвуза / ИНФО. 1999. №9.
8. *Роберт И.В.* Распределенное изучение информационных и коммуникационных технологий в общеобразовательных предметах // ИНФО. 2001. №5.
9. *Румянцев И.А., Степанов С.А.* Концепция среднего общеобразовательного учреждения постиндустриального общества // ИНФО. 2001..№8.
10. *Семенов А.Л.* Роль информационных технологий в общем среднем образовании // ИНФО. 2001. №2.
11. *Скибицкий Э.Г., Шкабура О.В.* Стиль мышления как стратегия решения задач с использованием компьютера // ИНФО. 2000. №10.

С.Г. Волченков

ДВЕ ЗАДАЧИ ПО ИНФОРМАТИКЕ ИЗ ТЕОРИИ ЧИСЕЛ

В этой статье приведены две задачи, предлагавшиеся автором на олимпиадах по информатике. Обе эти задачи объединяет не только их принадлежность к классической теории чисел, но и общая проблематика, связанная с соотношением «чистая математика – информатика». В первые годы становления информатики в ней, в основном, решались математические задачи, так как первыми информатиками, естественно, стали математики. Затем в этой области появилась масса красивых идей и задач и из других разделов науки, техники и просто здравого смысла. Сейчас сложилась ситуация, когда на различных соревнованиях по информатике стараются даже не давать задач, опирающихся на слишком специальные знания из других областей, в частности математики. Возникает угроза потери для информатики ряда красивых идей и методов, разработанных в смежных науках. Приводимые ниже задачи и комментарии к ним должны показать, что далеко не всегда специальные математические знания должны рассматриваться как препятствие дать данную задачу на обычных школьных или студенческих соревнованиях по информатике.

Задача 1. Разбиение на пары с простой суммой

Для заданного натурального числа N требуется разбить числа 1, 2, 3, … , N на максимально возможное число пар с простой суммой.

Сначала небольшой математический комментарий. В теории чисел известен факт, называемый постулатом Бертрана, согласно которому для любого натурального N между N и 2N всегда существует простое число. Указанный факт долгое время оставался недоказанным, отсюда и закрепившееся за ним название постулата. Благодаря работам русского математика П.Л. Чебышева постулат Бертрана был доказан. Сейчас он известен не только математикам, но и многим школьникам, однако считается достаточно сложным, и даже на математических олимпиадах не принято давать задач, опирающихся в своих решениях на него.

С помощью постулата Бертрана наша задача решается легко. Отметим сначала, что если N нечетно, то одному числу пары заведомо не будет и одно число можно отбросить. Так вот, отбросить можно само N, так как для четного N все числа от 1 до N разбиваются на нужные пары. В самом деле, рассмотрим то самое простое число p, которое должно быть между N и 2N. Образуем первую пару, дающую простую сумму: N и (p-N). Заметим, что такую же сумму дают и другие пары: N-1 и (p-N)+1, N-2 и (p-N)+2 и т.д. Множество чисел от p-N до N оказалось разбитым на нужные пары. Поэтому далее процесс можно рекурсивно продолжить для оставшихся чисел от 1 до (p-N-1). Отметим, что число p-N-1 опять-таки четно, что требуется для продолжения процесса.

Казалось бы, предложенное решение непосредственно опирается на постулат Бертрана, и давать данную задачу школьникам нельзя. Но так ли это? А что изменилось бы, если бы постулат Бертрана не был бы верен? Ответ прост. Число N не сможет ни с одним из имеющихся чисел дать простую сумму и, значит, заведомо должно быть исключено из рассмотрения, а процесс надо продолжать для оставшихся чисел. Алгоритм же поиска нужного простого числа и алгоритм установления, что такого числа нет, очевидно, один и тот же. Итак, единственно, зачем оказался нужным сам постулат – это приятное удивление оттого, что задача всегда решается до последнего числа!

Задача 2. Разбиение на группы с простой суммой

Для заданного натурального числа N>1 требуется разбить множество чисел 1, 2, 3, …, N на минимально возможное число групп с простой суммой.

Математический комментарий к этой задаче, по форме очень похожей на предыдущую, опирается на гораздо более серьезный математический факт. Более того, на недоказанную гипотезу Х. Гольдбаха, дополненную Л. Эйлером, утверждающую, что любое четное число представимо в виде суммы двух простых, а также на теорему И.М. Виноградова о представлении достаточно большого нечетного числа в виде суммы трех простых.

Итак, небольшой анализ задачи. Сумма первых N натуральных чисел равна S=N*(N-1)/2, что хорошо известно любому школьнику, начавшему изучать арифметические прогрессии. Из этой формулы легко заметить, что для всех N>2 сумма всех чисел до N включительно не является простым числом и, значит, количество искомых групп более одной. А вот далее, вступают в действие вышеупомянутые математические факты. Если S – четное число, то, по гипотезе Гольдбаха, количество групп равно двум. Если же S – нечетно, то, по теореме Виноградова, групп должно быть не более трех. Остается заметить, что их может быть и две, если S-2 оказывается простым. В другую группу при этом попадает одно простое число 2.

Опять же зададим себе вопрос, как решать задачу, если мы не знаем ни гипотезы Гольдбаха (которая, может быть, еще и не верна), ни теоремы Виноградова. Оказывается, что решать задачу все равно надо будет так же.

Во-первых, находим S по формуле. Кстати, если школьник забыл эту формулу или не знал ее, он может успешно вычислить S, просуммировав N чисел.

Далее, определяем четность S. В случае нечетного S вычитаем из S двойку и проверяем S-2 на простоту. Если результат является простым числом, то задача решена. Если нет, то вычитаем тройку, и получаем заведомо четное число S-3, которое, как и в случае четного S, надо разделять на две группы. Почему на две, а не больше? Да потому, что требуется найти наименьшее число групп и поиск надо начинать с меньших значений. То, что поиск на этом и закончится (а он закончится, так как гипотеза Гольдбаха проверена для всех чисел, которые могут быть заданы в тестах, и даже в миллионы раз больших!), явится для школьника приятным подарком. Остается порекомендовать не забыть в программе отдельно рассмотреть случай N=2 – единственный случай простоты числа S.

Итак, рассмотрены две задачи, которые, безусловно, придуманы на основании сложных математических фактов. Для решения же этих задач, по большому счету, не нужны ни знания этих фактов, ни сами эти факты.

Может быть, приведенные задачи являются уникальными в этом смысле? Отнюдь, нет. Мы уже давно привыкли искать совершенные паросочетания в двудольном графе, не заботясь о выполнении условий теоремы Холла (по ходу дела проверится) или строить максимальные потоки, не оценивая минимальные разрезы, производить перебор вариантов, не только не оценив заранее объем этого перебора, но даже зная о теоретической необозримости этого перебора (игра в шахматы, и не только). И здесь следует заявить – информатика должна решать массу задач. Хорошо, если эти задачи решаются с помощью математических или иных точных методов. Но если это не так, то задачи все равно надо решать, и решения задач далеко не всегда могут опираться на исследованные другими науками кочки. Ни одна наука не может в абсолютной мере претендовать на роль первопроходца и истины в последней инстанции. Наша жизнь намного богаче любой науки.

О.Н. Корчемкина

ОРГАНИЗАЦИЯ УЧЕБНОЙ ПРАКТИКИ ПО ИНФОРМАТИКЕ СТУДЕНТОВ ЭКОНОМИЧЕСКИХ СПЕЦИАЛЬНОСТЕЙ

Благодатной почвой для внедрения новых информационных технологий является экономическая деятельность. Почти все задачи, которые возникают в процессе деятельности хозяйствующих субъектов, поддаются автоматизации. Быстрая и бесперебойная обработка значительных потоков информации является одной из главных задач любого предприятия.

Создание индустрии информатики и превращение информационного продукта в товар приводит к глубинным социальным изменениям в обществе, трансформируя его из индустриального в информационное. В связи с этим рыночные отношения предъявляют повышенные требования к уровню подготовки экономических кадров. Наряду с фундаментальными экономическими и иными знаниями студент экономической специальности должен приобрести знания, умения и навыки оптимального применения ПК в своей дальнейшей профессиональной деятельности [1].

При подготовке будущего специалиста важное место занимает учебная практика по информатике, во время которой студент реализует знания, умения и навыки, полученные при изучении курса информатики в целом.

В 2000-2001 учебном году впервые согласно учебному плану проводилась учебная практика по информатике для студентов института экономики. Изучив опыт проведения такого рода практик, а также учитывая специфику нашего учебного заведения, был разработан следующий вариант ее организации. Далее сформулированы цели, содержание и этапы практики.

Цели и задачи учебной практики по информатике

1. Приобретение студентами практических навыков комплексного использования информационных технологий для решения реальных задач.

2. Развитие умения формализовать реальную задачу.

3. Развитие умения выбрать оптимальный программный инструментарий для решения конкретной практической задачи.

4. Развитие самостоятельности при исследовании студентами дополнительных возможностей программных средств, изученных в курсе информатики.

5. Развитие научно-исследовательского потенциала у студентов как будущих специалистов.

6. Самостоятельное приобретение необходимых сведений из различных экономических областей знаний, таких, как маркетинг, менеджмент, анализ финансово-хозяйственной деятельности предприятия, бухгалтерский учет, финансы, экономика и т.д.

Содержание практики

Содержанием учебной практики по информатике является разработка и создание некоторого конечного продукта в электронном виде с использованием изученных программных средств (MSWord, MSExcel, MSPaint, MSPowerPoint), их дополнительных возможностей, не рассмотренных на занятиях, и осуществлением связей между ними [3, 11].

Таким продуктом может быть пакет шаблонов и форм, модель работы некоторой системы, инструкция по использованию дополнительных возможностей программных средств для некоторого класса задач на конкретном примере и т.п. (примеры заданий см. ниже).

Результатом практики, помимо конечного продукта в электронном виде, должен быть пакет отчетной документации по выполнению задания практики.

Структура практики

Учебная практика по информатике рассчитана на 72 часа и состоит из нескольких этапов.

Вводная конференция (2 часа)

Руководитель практики знакомит с программой и правилами прохождения практики, предлагает студентам возможные варианты заданий. Студенческой группе предоставляется список с примерными заданиями в следующей форме:

№ п/п.	Задание	Рекомендуемая литература	ФИО студента

К началу третьего этапа практики каждый студент должен внести свою фамилию и выбрать тему задания из предлагаемого перечня или предложить свою (для этого в списке рекомендуется оставить пустые строки).

Темы заданий должны удовлетворять следующим критериям:

- практическая значимость;
- актуальность;
- возможность реализации имеющимися средствами;
- возможность применения всех рассмотренных в курсе информатики программных средств, их интеграции.

Примеры заданий по практике

1. Подготовить пакет шаблонов и форм для статистических исследований [13].

2. Подготовить пакет шаблонов и форм бухгалтерской документации [2,5].

3. Подготовить пакет шаблонов и форм для анализа финансово-хозяйственной деятельности предприятия по основным экономическим показателям и сравнения их значений с нормативами [4, 12].

4. Исследовать возможности анализа данных в табличном процессоре на примере конкретной задачи (например, анализ деятельности торговой организации за некоторый период) и подготовить рекомендации по их использованию [9, 10, 11, 14].

5. Исследовать возможности применения математических функций в табличном процессоре на примере конкретных задач и подготовить рекомендации по их использованию [9, 10, 11, 14].

6. Исследовать возможности применения финансовых функций в табличном процессоре на примере конкретных задач и подготовить рекомендации по их использованию [10, 11, 12, 15].

7. Исследовать возможности применения статистических функций в табличном процессоре на примере конкретных задач и подготовить рекомендации по их использованию [10, 11, 12, 14, 15].

8. Исследовать возможности табличного процессора для решения алгебраических задач (на конкретных примерах) и подготовить рекомендации по их использованию [9, 10, 11, 14].

9. Исследовать возможности табличного процессора для решения геометрических задач (на конкретных примерах) и подготовить рекомендации по их использованию [9, 10, 11, 14].

10. Построить модель работы узлов компьютера (с помощью табличного процессора) [7, 9, 10, 11, 14].

Подготовительная работа (4 часа)

Студент выбирает задание, подбирает соответствующую литературу, готовит примерный план действий.

Планирование (6 часов)

Руководитель практики проводит индивидуальные консультации с каждым студентом, в ходе которых:

- задание конкретизируется и разбивается на подзадачи;
- составляется развернутый план выполнения задания;
- определяется примерный график выполнения пунктов плана;
- определяются этапы промежуточного контроля;
- выбираются основные и вспомогательные программные инструменты для решения каждой подзадачи;
- уточняется список литературы.

Выполнение задания (48 часов)

Студенты в компьютерном классе выполняют задание, консультируясь по мере необходимости с руководителем практики. На этапах промежуточного контроля, определенных ранее, руководителю практики рекомендуется производить промежуточную оценку выполнения соответствующей части задания.

Подготовка отчета (6 часов)

К этому этапу студент может приступать, когда создание конечного продукта завершено. Кроме того, на этом этапе студенту необходимо подготовить свое выступление на заключительной конференции.

Отчетная документация

1. Титульный лист.
2. Формулировка задания.
3. Краткое обоснование актуальности, назначение и возможности практического использования результатов выполнения задания (конечного продукта).
4. План выполнения задания.
5. Формализация задания (модель) с указанием входных, выходных данных, связей между ними.
6. Описание созданного продукта.
7. Инструкция по применению результатов (конечного продукта).
8. Перечисление инструментария, использованного при выполнении задания.
9. Список использованной литературы.

Заключительная конференция (6 часов)

Защита и обсуждение выполненных заданий. На заключительной конференции участвуют студенты всей группы (или подгруппы). Студенту отводится для выступления 10 минут, в течение которых он должен:

- сформулировать тему задания;

• кратко описать созданный по заданию продукт;
• определить область и возможности его применения;
• продемонстрировать продукт на демонстрационном мониторе;
• ответить на вопросы руководителя практики и студентов.

В конце конференции руководитель практики объявляет оценки и подводит общие итоги.

Критерии оценки за практику:
• точность и полнота выполнения задания;
• возможность реального применения созданного продукта;
• степень самостоятельности при выполнении задания;
• разнообразие и использование дополнительных возможностей инструментария;
• оценки за промежуточные этапы выполнения задания;
• полнота и четкость отчетной документации;
• выступление на заключительной конференции.

В процессе организации и проведения учебной практики по информатике для студентов института экономики возник ряд проблем, обусловленных как объективными, так и субъективными причинами. Основная из них связана с противоречием между стремлением дать студентам задания экономического характера (связанные с их основной специальностью) и недостаточным уровнем экономической подготовки как самих студентов (после I-II курсов), так и преподавателей информатики (не экономистов).

В целом такая практика способствует приобретению студентами практических навыков комплексного использования информационных технологий для решения реальных задач, развитию научно-исследовательского потенциала у будущих специалистов, а также стимулирует к самостоятельному приобретению сведений из различных областей знаний.

Литература

1. Автоматизированные информационные технологии в экономике / Под ред. Г.А. Титоренко. М.: Компьютер, ЮНИТИ, 1999.
2. Альбом унифицированных форм первичной учетной документации по учету кассовых операций. М/, 1998.
3. *Бабушкина И.А., Овсянникова М.В., Пятышева Е.А.* Практикум по программному обеспечению ЭВМ (в 2-х ч.). Киров, 2000.
4. *Бердникова Т.Б.* Анализ и диагностика финансово-хозяйственной деятельности предприятия. М.: ИНФРА-М, 2001.
5. *Глушков И.Е.* Бухгалтерский учет на современном предприятии. Новосибирск, 2001.
6. *Ильина М.* Работа в Word 7 на примерах. М.: БИНОМ, 1996.
7. Информатика / Под ред. Н.В. Макаровой. М.: Финансы и статистика, 1997.
8. *Кост Р., Шпаенер И., Валентин Р.* Word 7 для Windows 95. М.: Бином, 1997.
9. *Кэмбелл М.* Excel. Ответы. М.: Бином, 1996.
10. *Новиков Ф., Яценко А.* Microsoft Office 97 в целом. СПб.: BHV, 1997.
11. *Симонович С.В.* Информатика. Базовый курс. СПб: Питер, 2000.
12. *Шеремет А.Д., Сайфулин Р.С.* Финансы предприятия. М., ИНФРА-М, 1997.
13. Экономическая статистика. М, 1998.
14. Microsoft Excel 97: практическое пособие. М.: ЭКОМ, 1997.
15. Microsoft Word 97: практическое пособие. М.: ЭКОМ, 1997.

С.Ю. Ямбарышева

ИСПОЛЬЗОВАНИЕ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ ПРИ ИЗУЧЕНИИ КУРСА «ЧИСЛЕННЫЕ МЕТОДЫ»

В самых разнообразных областях современной науки и техники все чаще и чаще приходится встречаться с такими математическими задачами, для которых невозможно получить точное решение классическими методами или же решение может быть получено в таком сложном виде, что оно становится неприменимым для практического использования. Например, очень часто приходится встречаться с решением систем линейных алгебраических уравнений, в которых имеется несколько десятков или сотен неизвестных, с задачей отыскания корней алгебраического уравнения высокой степени и корней трансцендентных уравнений, с необходимостью решения систем дифференциальных уравнений, которые не интегрируются в элементарных функциях и т.д.

В настоящее время наблюдается период бурного развития вычислительной техники. Поэтому расширяется круг задач, решаемых численными методами и с помощью ЭВМ. Также происходит изменение взглядов на весь комплекс вопросов, связанных с применением компьютера, и требований к численным методам решения задач.

Первоначально элементы математики появились вместе с необходимостью решения практических задач: измерение на местности, навигация и т.д. Поэтому математика была численной – ее целью являлось получение решения в виде числа.

Численное решение прикладных задач всегда интересовало математиков. Крупнейшие ученые, такие, как Ньютон, Эйлер, Лобачевский, Гаусс, Чебышев, занимались разработкой численных методов решения задач.

Настоящее время характерно резким расширением приложений математики. Это во многом связано с созданием и развитием вычислительной техники. Необходимо заметить, что с момента создания первой ЭВМ скорость вычисления арифметических операций возросла от 0,1 операции в секунду при ручном счете до нескольких миллиардов операций в секунду у современных компьютеров.

Распространенное мнение о всемогуществе современных ЭВМ часто порождает обманчивое впечатление, что математики избавились почти от всех хлопот, связанных с численным решением задач, а поэтому разработка новых методов решения задач уже не столь существенна. В действительности это не так. Общий закон развития науки состоит в том, что потребности развития общества часто ставят перед ней задачи, превышающие ее возможности.

Расширение возможностей приложения математики привело к математизации других разделов наук: химии, экономики, биологии, геологии, географии, психологии, медицины и разделов техники. Этот процесс состоит в построении математических моделей различных процессов и явлений, а также в разработке методов их исследования.

Применение ЭВМ и расширение математического образования увеличивает возможности в направлении построения и исследования математических моделей.

Современные успехи в решении проблем общества, связанных с атомными, космическими, экономическими и исследованиями в других областях, вряд ли были бы возможны без применения компьютеров и численных методов.

Из всего вышеизложенного следует, что многим специалистам необходимы знания для теоретического исследования с помощью численных методов типичных математических задач. В настоящее время сюда принято относить задачи математического анализа (вычисление погрешностей, дифференцирование и интегрирование), задачи алгебры (решение дифференциальных и интегральных уравнений, задачи оптимизации, обратные задачи). Эти теоретические исследования оказывают большую помощь при решении конкретных практических задач из разных областей науки.

Современные информационные технологии позволяют упростить трудоемкие (рутинные) вычисления, а также наглядно представить результаты. Под информационными технологиями понимается совокупность методов и технических средств, применяемых для сбора, хранения, обработки, передачи, представления данных, то есть для выполнения всех информационных процессов.

Рассмотрим решения нелинейных уравнений с одним неизвестным вида $f(x)=0$. Для этого необходимо выполнить следующее:

1) отделение корней;

2) уточнение отделенного корня одним из известных методов (метод половинного деления, или бинарный метод, метод хорд, метод касательных, комбинированный метод, или метод хорд и касательных, метод простой итерации). Для этого необходимо знать точность уточнения, а затем по соответствующим формулам произвести вычисления. При выборе метода необходимо доказать, что все условия его применения выполняются, особенно это касается метода простой итерации.

Современные компьютеры позволяют использовать для решения таких задач самые разнообразные информационные технологии: языки программирования (Turbo Pascal, Delphi, Visual Basic и другие), табличные процессоры или электронные таблицы (чаще всего Excel), математических систем (MathCad) и многое другое.

Рассмотрим их применение для отделения корней на конкретном примере.

Задача. Дано нелинейное уравнение с одной переменной: $\sqrt{4x+8} - 3\cos 2x = 0$. Найти все его решения с точностью $E=10^{-5}$.

Выполним только отделение корней этого уравнения.

ОТДЕЛЕНИЕ КОРНЕЙ

Пусть дано нелинейное уравнение $f(x)=0$. Отделить корни этого уравнения – значит найти такие отрезки, в каждом из которых лежит только один корень данного уравнения.

Графическое отделение. Суть этого способа заключается в построении графика функции $f(x)$ на некотором отрезке [a,b], после чего выделяются отрезки, каждому из которых принадлежит только один корень.

а) Построить график функции $y=f(x)$ и найти отрезки, которым принадлежат точки пересечения графика этой функции и оси абсцисс. В приведенном примере (рис. 1) можно выделить три отрезка, в каждом из которых лежит только одна точка пересечения. В данном случае выделены три таких отрезка, каждому принадлежит только один корень.

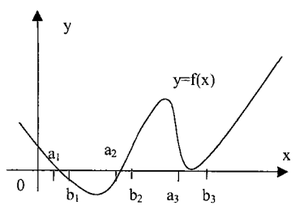


Рис. 1

б) Иногда трудно построить график исходной функции, поэтому можно преобразовать исходное уравнение $f(x)=0$ к равносильному уравнению вида $f_1(x)=f_2(x)$, построить графики функций $y=f_1(x)$ и $y=f_2(x)$, найти отрезки, которым принадлежат абсциссы точек пересечения графиков. В приведенном примере (рис. 2) выделены три отрезка, каждому из которых принадлежит только одна абсцисса точек пересечения графиков двух функций.

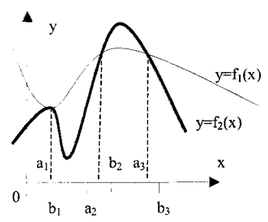


Рис. 2

Графический способ отделения корней используется очень часто. При построении графика очень важно выбрать такой отрезок области определения функции, чтобы ему принадлежали все корни уравнения, что иногда трудно выполнить. Кроме того, если корни располагаются очень близко, то возникает проблема правильного установления границ отрезков изоляции. Поэтому графическое отделение корней необходимо использовать вместе с аналитическим способом.

Аналитическое отделение. В основе этого способа отделения корней лежит следующая теорема Больцано-Коши (условие существования корня на отрезке):

Теорема 1. Если непрерывная функция f (x) принимает значения разных знаков на концах отрезка [a, b], то внутри этого отрезка содержится хотя бы один корень.

Эту теорему можно усилить, получим условие существования единственного корня на отрезке:

Теорема 2. Если непрерывная функция f (x) принимает значения разных знаков на концах отрезка [a, b], а внутри этого отрезка имеет конечную производную первого порядка, сохраняющую свой знак на данном отрезке, то внутри этого отрезка содержится только один корень.

Существование первой производной и сохранение ее знака на отрезке говорит о том, что на данном отрезке непрерывная функция является строго монотонной (возрастающей или убывающей), поэтому если на концах данного отрезка она имеет значения разного знака, то она пересекает ось Ox только в одной точке.

Разобьем исходный отрезок [a, b] на n равных частей (чем больше n, тем лучше, так как тогда на полученном отрезке функцию можно будет считать монотонной в силу ее малой изменяемости). Шаг разбиения h можно найти по формуле: $h=\frac{b-a}{n}$. Получим n+1 точку разбиения:

$x_0=a, x_1=a+h=x_0+h, x_2=a+2h=x_1+h, ...,$

$x_{n-1}=a+(n-1)h=x_{n-2}+h, x_n=a+nh=x_{n-1}+h=b.$

Сейчас последовательно определяем знаки функции на концах каждого отрезка $[x_i, x_{i+1}]$, где i=0, n–1. Если знаки разные, то есть $f(x_i)*f(x_{i+1})<0$, то на данном отрезке существует изолированный корень. Если это произведение равно 0, то один из концов отрезка является корнем.

Таким образом, выполнение этого этапа сводится к построению графика функции и использованию аналитического способа.

Как можно реализовать такое отделение корней, используя информационные технологии?

I способ – составим две программы для отделения корней графическим и аналитическим методами.

Для построения графика необходимо заметить, что областью определения функции является интервал [-2; +∞]. Но так как функция y=3cos2x принимает свои значения на промежутке [-3; 3], то и функция $y=\sqrt{4x+8}$ должна принимать свои значения на этом же отрезке. Таким образом, необходимо отделять корни уравнения на отрезке [-2; 1].

Строим график функции. Для этого, в самом простом случае, необходимо знать не только отрезок, которому принадлежат все корни, но и наименьшее и наибольшее значения функции *y=-f(x)* на этом отрезке. Тогда можно будет произвести масштабирование и нормирование по осям, которые необходимы для построения графика.

```
Program Example1; {Графическое отделение корней
уравнения на отрезке}
Uses Graph, Crt;
Var  a, b, m1, m2, d, m : Integer;
Function F(x : Real) : Real; {функция}
Begin
   F:=Sqrt(4*x+8)-3*cos(2*x);
End;
Procedure Grahpic(a, b, min, max : Real);
Var  k1, k2, x, y : Real;
     x1, y1, x2, y2, k : Integer;
      st : String;
Begin
       {определяем коэффициенты для масштаба
       по осям}
       k1:=640/(b-a); k2:=480/(max-min);
       {координаты начала координат на экране}
       x2:=round(k1*abs(a)); y2:=round(k2*abs(m2));
       {построение осей, делений и надписей по оси Ox}
       If (y2>=0) and (y2<=480) Then Line(0,y2,640,y2);
       If (x2>=0) and (x2<=640) Then Line(x2,0,x2,480);
       x1:=x2-round(k1);k:=-1;
       While x1>=0 Do
       Begin   str(k,st);OutTextXY(x1-3,y2+10,st);
             Line(x1,y2-5,x1,y2+5);
             x1:=x1-round(k1); dec(k);
       End;
       x1:=x2+round(k1);k:=1;
       While x1<=640 Do
       Begin   str(k,st);OutTextXY(x1-3,y2+10,st);
             Line(x1,y2-5,x1,y2+5);
             x1:=x1+round(k1); inc(k);
       End;
       {построение графика по точкам}
       x:=a;
       While x<=b Do
       Begin   If4*x+8>=0 Then
                  Begin  y:=F(x);
                     x1:=Round(x2+k1*x);
                      y1:=Round(y2-k2*y);
                      PutPixel(x1,y1,15);
                  End;
                x:=x+0.01;
       End;
End;
```



```
Begin
   Writeln(«Введите a и b»); Readln(a,b);
   Writeln(«Введите наименьшее и наибольшее
          значения функции на данном отрезке»);
    Readln(m1,m2);
    d:=detect; InitGraph(d,m,'c:\bp\bgi');
    Grahpic(a,b,m1,m2);
    Readln;
    CloseGraph;
    End.
```

Результат выполнения этой программы для отрезка [-3; 2] при наименьшем значении функции -1, а наибольшем -5, приведен на рис. 3.

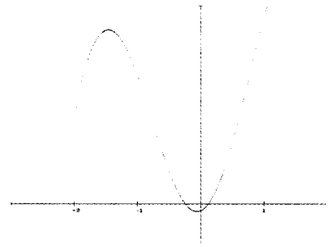


Рис. 3

Таким образом, данное уравнение имеет два корня: первый принадлежит отрезку [-0,5; 0], а второй - [0; 0,5]. Для выполнения этой части можно было графически изображать только ту часть графика, которая располагается вблизи оси Ox (например, если значения функции лежат в интервале [-0,2; 0,2], эта часть выделена на рис. 3) Тогда масштабирование по оси Oy не требуется, так как коэффициент будет постоянным, не зависящим от наименьшего и наибольшего значения функции на отрезке. Результат работы такой программы приведен на рис. 4.

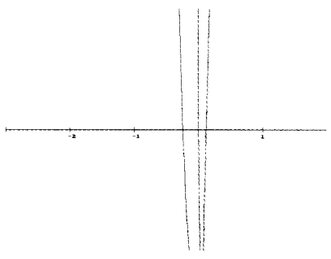


Рис. 4

В этом случае будет потеряна наглядность графика функции, но перед нами такая задача не ставится.

Сейчас можно воспользоваться второй программой, которая отделяет корни аналитическим методом.

```
Program Example2; {Аналитическое отделение корней
                   уравнения на отрезке}
Uses Crt;
Var a, b, h, x1, x2 : Real;     {a и b -это концы отрезка}
n, i : Integer;                 {n - количество отрезков
                                 разбиения}
Function F(x : Integer) : Real;  {исходное уравнение}
Begin
   F:=sqrt(4*x+8)-3*cos(2*x);
End;
Begin  Clrscr;
   Writeln(«Введите a и b»);
    Readln(a,b);
   Writeln(«Введите n»);
    Readln(n);
   h:=(b-a)/n; {шаг деления}
   For i:=0 To n Do
  Begin  x1:=a+i*h;x2:=x1+h;
        If F(x1)*F(x2)<0 Then Writeln
                        ("[",x1:5:3,",",x2:5:3,"]")
         Else If F(x1)*F(x2)=0 Then
            If F(x1)=0 Then Writeln(x1:5:3,
                                «- корень»)
         Else Writeln(x2:5:3, «- корень»);
   End;
   Readln;
End.
```

По этой программе для отрезка [-2; 1] и n=30 получаем тоже два отрезка: [-0.3; -0.2] и [0.1; 0.2]. Но в этом случае отрезки можно получить достаточно маленькие.

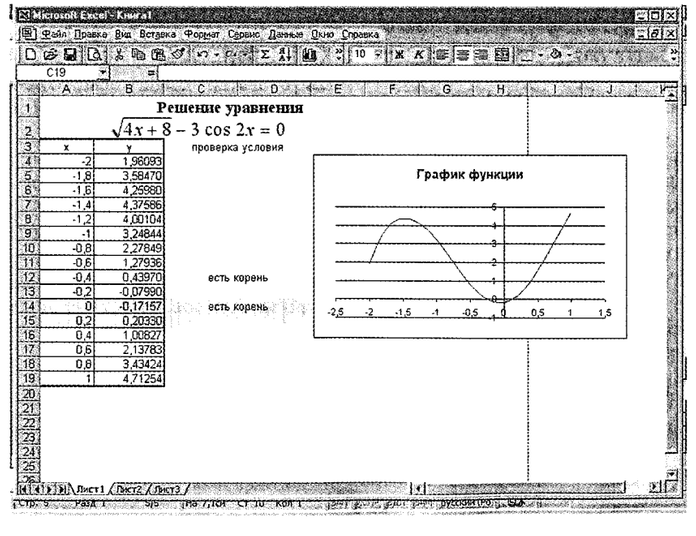


Рис. 5

II способ – использование табличного процессора для каждого метода отделения. Для этого можно составить расчетные таблицы и построить точечную диаграмму. Так же можно написать макрос на VBA, с помощью которого отделяются корни на некотором отрезке.

Сначала составим расчетную таблицу и построим график, а также выделим отрезки, для которых выполняется условие существования корня (см. рис. 5).

Составление таблицы можно описать следующими шагами:

1) Создадим заголовок.

2) Заполним шапку таблицы (названия столбцов A, B и C).

3) Заносим в ячейку A4 начальное значение аргумента – -2 , а в ячейку A5 – -1,8. После чего используем для заполнения значений аргумента x автозаполнение по закономерности до ячейки A19 (до x=1).

4) Заносим в ячейку B4 формулу для вычисления значения функции при x=-2 – =КОРЕНЬ(4*A4+8)-3*COS(2*A4). Копируем эту формулу для ячеек с B5 до B19. Так как в формуле используется относительная ссылка на ячейку A4, то она будет автоматически изменяться.

5) По полученной таблице значений аргумента и значений функции строим точечную диаграмму с помощью мастера диаграмм.

6) В ячейку C4 заносим проверку условия существования корня на отрезке – =ЕСЛИ(B4*B5<=0;»есть корень»;» «). Таким образом, будут отмечены только те отрезки, которые необходимо было найти.

К достоинству табличного процессора нужно отнести удобное построение графика функции, не требующее дополнительных расчетов (масштабирование, нормирование и т. д.).

Но следует заметить, что данный способ имеет один существенный недостаток – при изменении границ отрезка или шага разбиения приходится таблицу переформировывать, причем придется также изменять и параметры диаграммы, так как она может не охватить все новые значения. Этого можно избежать, если создать макрос, который будет по исходным данным производить табулирование и выводить полученную таблицу значений на лист, причем также позволит построить соответствующий график.

III способ – использование математической системы MathCad.

Здесь можно составить расчетную таблицу и построить график.

При составлении расчетной таблицы используется табулирование значений. Для этого необходимо:

1) Задать формулу, описывающую исходную функцию (y:=...).

2) Ввести дискретный аргумент: его начальное и конечное значения, а также шаг (x:=-2, -1.8 ..1 – здесь: -2 и 1 – это начальное и конечное значения аргумента, -2 и -1.8 – задают шаг его изменения).

3) Определить место вывода таблицы значений (x=, y(x)=).

Для построения графика необходимо:

1) Задать формулу, описывающую исходную функцию (если проводили табулирование, то это делать не надо).

2) Щелкнуть мышью в свободном месте рабочего поля, где нужно разместить график (ниже формулы).

3) Выбрать в главном меню Insert – Graph – X-Y-Plot.

4) Ввести:

- под осью Ox – начальное значение аргумента (-2), имя аргумента (x) и конечное значение (1);

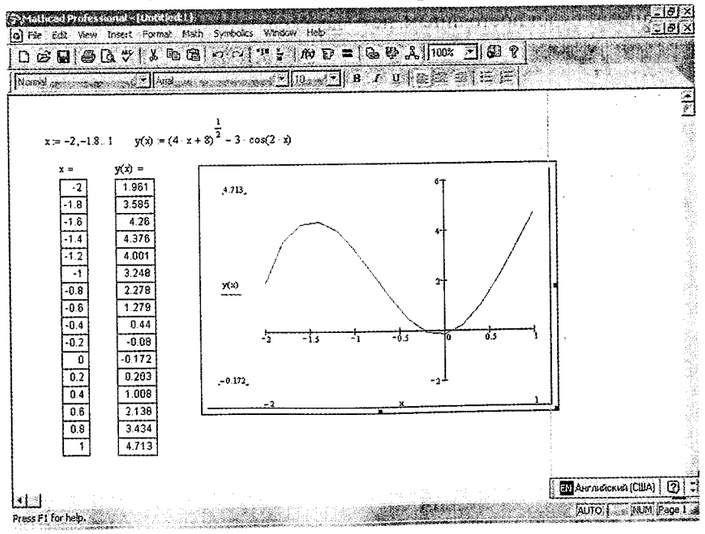


Рис. 6

- напротив центра оси Oy – имя функции из формулы (y(x)).

5) Щелкнуть вне графика.

В итоге получим следующее (см. рис. 6).

Таким образом, графическое отделение корней удобнее всего выполнить с помощью табличного процессора Excel или математической системы MathCad, так как в языке программирования приходится выполнять масштабирование и нормирование. Для аналитического отделения можно применить любую из предложенных технологий.

Т.Н. Андреева

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ ОБРАБОТКИ ТАБЛИЧНЫХ ДАННЫХ

Автоматизация расчетных задач не всегда требует программирования и использования систем управления базами данных. Для решения многих задач идеально подходят программные продукты, называемые электронными таблицами. Идея, на которой основывается работа электронных таблиц, до гениального проста: клетчатая бумага, а в клетках равноправно «живут» числа, текст, формулы...

Идея создания электронных таблиц принадлежит Дениелу Бриклину, который, однажды занимаясь утомительными вычислениями, связанными с определением изменения суммы налога в результате роста процентной ставки на издержки и прибыль предприятия, вдруг понял, что есть иной, более удобный способ выполнения этих расчетов. Основная проблема здесь заключалась в том, что при изменении какой-то одной цифры приходилось пересчитывать все производные от нее величины. Электронный калькулятор, конечно, несколько облегчал эту задачу, но все равно такие расчеты требовали слишком много времени. Малейшая описка могла испортить всю работу, которая оформлялась на большом листе тщательно разлинованной бумаги под названием spreadsheet* (развернутый лист).

Программистский опыт натолкнул Бриклина на мысль, что все эти нудные жонглирования числами неплохо бы поручить компьютеру. Свою идею он обсудил с внештатным инженером-программистом Робертом Фрэнкстоном. Тот заинтересовался идеей и в конце 1978 г. начал писать программу, а уже к весне следующего года закончил ее. Свое детище Фрэнкстон и Бриклин назвали VisiCalc (Visible Calculator). VisiCalc создавала на экране персонального компьютера столбцы чисел, которые мгновенно изменялись, если изменялась какая-либо позиция электронной таблицы.

Продавать свою программу Бриклин и Фрэнкстон начали осенью 1979 г. Изначально программа была написана для машины Apple-2, и по существу, именно она сыграла главную роль в огромном успехе этого компьютера, потому как VisiCalc была первой программой, которая уже сама по себе оправдывала приобретение микрокомпьютерной системы. Буквально в течение первого года после продажи VisiCalc сбыт персональных компьютеров резко возрос, а сама программа разошлась тиражом около 100 тыс. экземпляров по цене около 200 долл. за программу.

VisiCalc оставалась бестселлером целый год, что, естественно, привело к созданию десятков программ-подобий, творцы которых стремились нажить капитал на столь прибыльном деле. Так, вскоре появилась программа SuperCalc, в которой основные идеи VisiCalc были многократно усовершенствованы.

Спустя три года появилась программа Lotus 1-2-3, сразу же продемонстрировавшая свои неоспоримые преимущества перед VisiCalc. Она завоевала широкое признание, поскольку соединяла в себе лучшие качества системы VisiCalc с графическими возможностями и средствами информационного поиска.

Митчел Кэпор, создатель Lotus 1-2-3, при разработке своей программы ориентировался на 16-разрядный процессор фирмы IBM, а не на 8-разрядные процессоры, которые в основном использовались в микрокомпьютерах того времени. Он был убежден, что огромное превосходство фирмы IBM скоро сделает этот процессор общепризнанным стандартом для всей индустрии персональных компьютеров, а это даст его программе ощутимые преимущества в конкурентной борьбе. Расчет Кэпора полностью оправдался. Система Lotus 1-2-3 была объявлена в конце 1982 г. и вскоре заняла первую строку в списке самых популярных программ, а компания Lotus стала самой большой независимой компанией-производителем программных средств.

Lotus 1-2-3 сделала для фирмы IBM то же, что VisiCalc сделала в свое время для фирмы Apple. Успех компании Lotus привел к ужесточению конкуренции, вызванной появлением на рынке новых электронных таблиц, таких, как VP Planner компании Paperback Software и Quattro Pro компании Borland International, которые предложили пользователю практически тот же набор инструментария, но по значительно более низким ценам. Программу Lotus 1-2-3 во многом превзошла система Microsoft MultiPlan для машин Apple, IBM PC и других. Но у программы Lotus 1-2-3 было одно существенное преимущество: она действовала в несколько раз быстрее.

Табличный процессор Microsoft Excel, вышедший рынок в августе 1985 г., предлагал более простой гра-

*Позднее так стали называть электронные таблицы.

Современный пользователь употребляет термин «электронные таблицы» для обозначения двух понятий:

- формы представления данных на экране монитора в виде таблицы практически неограниченного размера;
- программы (или пакета программ) для обработки табличных данных.

Для обозначения последнего понятия иногда применяют термин «табличный процессор».

фический интерфейс в комбинации с ниспадающими меню. Функциональные возможности пакета при этом значительно расширены и улучшено качество выходной информации.

Последние версии Excel сочетают простоту использования «интеллектуальных» приложений с широчайшими возможностями Web. Появились трехмерные документы, установление связей между файлами и таблицами значительно упростилось по сравнению с прежними версиями. Контекстные меню расширены, а дополнительные программные инструменты облегчают решение сложных прикладных задач. Помощники (ассистенты) помогают пользователю задавать функции и выдают рекомендации, если существует более простой метод решения текущей задачи. В программу Excel встроена удобная подсистема помощи, которая в любой момент готова выдать необходимую справку. Так как MS Excel является программой пакета MS Office, то интерфейс похож на другие программные средства MS Office, и это также повышает удобство работы и упрощает импорт и экспорт данных с других приложений данного пакета.

Описанные до сих пор новшества касаются в основном комфорта в работе и быстрого освоения программы. Одним из важнейших функциональных расширений программы, предназначенным для профессионалов, является встроенная в Excel среда программирования Visual Basic Application (VBA) для решения прикладных задач. Благодаря VBA фирме Microsoft удалось не только расширить возможности языка макрокоманд Excel 5.0, но и ввести новый уровень прикладного программирования, поскольку VBA позволяет создавать полноценные прикладные пакеты, которые по своим функциям выходят далеко за рамки обработки электронных таблиц. Классическим образцом расчетных задач являются вычисления с обозримым числом исходных данных. Возможности Excel значительно шире. Обработка текста, управление базами данных – программа настолько мощна, что во многих случаях превосходит специализированные программы-редакторы или программы баз данных.

Подобное расширение спектра функциональных возможностей электронной таблицы, как правило, ведет к усложнению работы с программой, но разработчикам Excel удалось найти золотую середину, максимально облегчив пользователю освоение программы и работу с ней. Благодаря этому Excel быстро завоевала популярность среди широкого круга пользователей. Создатели программы сделали акцент на разработке организации и графического представления данных. И как результат – Excel на протяжении многих лет представляет собой фактический стандарт с точки зрения функциональных возможностей и удобства работы.

Так табличные процессоры прошли долгий путь развития от утилитарных («программ для себя») до программных продуктов, предназначенных для широкого распространения и продажи. Этот путь был связан с изменениями технической и программной среды разработки и эксплуатации программ, с появлением и развитием самостоятельной отрасли – информационного бизнеса, для которой характерны разделение труда фирм – разработчиков программ, их дальнейшая специализация, формирование рынка программных средств и информационных услуг.

Имеющиеся сегодня на рынке табличные процессоры способны работать в широком круге экономических приложений и могут удовлетворить практически любого пользователя. Но можно предположить, что глобальная информатизация общества, являющаяся одной из доминирующих тенденций цивилизации XXI века, послужит причиной того, что у пользователей возникнут новые потребности, касающиеся табличных процессоров. Очевидно, что новые потребности в совокупности с новыми возможностями, которые смогут предоставить аппаратные средства нового поколения на базе квантовых процессоров, положат начало новому направлению в развитии программного обеспечения.

Литература

1. Информатика: Учебник / Под ред. Н.В. Макаровой. М.: Финансы и статистика, 1997.
2. *Кузнецов А.А., Семенов А.Л., Уваров А.Ю.* О проекте концепции образовательной области «Информатика и информационные технологии» // Информатика. Приложение к газете «Первое сентября» 2001. №17. 1-7 мая.
3. *Николь Н., Альбрехт Р.* Электронные таблицы Excel 5.0. Практическое пособие / Пер с нем. М.: ЭКОМ, 1994.
4. *Рогов И.П.* Office 97 (Microsoft Office 97). М.: БИНОМ, 1997.
5. *Столяров А.М., Столярова Е.С.* Excel 2000 для себя. М.:ДМК Пресс, 2002.
6. Язык компьютера / Пер. с англ.; Под ред. и с предисл. В.М. Курочкина. М.: Мир, 1989.

ИНФОРМАТИКА В ШКОЛЕ

В.В. Юферев, Г.К. Корякина

КОМПЬЮТЕРНАЯ СРЕДА: НОВЫЕ ГОРИЗОНТЫ И НОВЫЕ ИЗМЕРЕНИЯ (образовательная информатика в физико-математическом лицее г. Кирова)

Компьютерная среда и виртуальный мир, окружив нас, внесли в процесс осознания реального мира свои поправки, информатика в образовательной среде диктует свои условия. Гибкая и пластичная, способная хранить, быстро обрабатывать, транспортировать огромные массивы информации, наученная создавать виртуальные образы и модели реального мира, открыла новые горизонты и новые измерения сознания,* привела нас к необходимости привлечения в нашу интеллектуальную деятельность новых форм организации мышления.

Тогда, когда образование в средней школе меняет свой вектор в сторону воспитания личности, способной развиваться самостоятельно, в его структуру вносятся новые содержательные линии. Предметная форма обучения остается основной и, являясь средством для осознания реальной действительности, призвана создавать личность, способную развивать себя. Каждая учебная дисциплина должна интегрироваться в общее знание и новое состояние. Этому содействует компьютерная среда, в которой личностно-ориентированная педагогика представляет собой наиболее эффективное условие для реализации задач образовательной системы.

К проблеме несоответствия ресурсного потенциала компьютерных систем и материальных средств муниципальных образовательных учреждений г. Кирова

Ресурсный потенциал компьютерных систем настолько высок и настолько привлекателен, насколько низким является уровень финансовых и материальных средств муниципальных образовательных учреждений. В 1998 г., по данным управления образования города, в средних муниципальных образовательных учреждениях насчитывалось 663 компьютера, среди которых IBM-совместимых было 277. Прошедшие с того времени годы не принесли заметного улучшения. Лишь небольшое количество образовательных учреждений за счет собственных средств укрепили свою компьютерную базу.

Городская программа информатизации на 1999-2005 гг., забуксовав, по некоторым позициям уже устарела. За эти годы изменился класс компьютеров, класс коммуникационных сетей. Определилась тенденция развития информатики в ряде школ.

Уже в 2000 г. в физико-математическом лицее было около 36 компьютеров, 4 принтера, 2 сканера, 2 копира на 400 учеников. Работу в сети Internet обеспечивал модем. В сети Internet мы ведем электронную переписку, открыли свои странички. 2 инженера и 2 лаборанта обеспечивали работоспособность компьютерной системы. Все годы лицей работает в очень тесном сотрудничестве с Вятским педагогическим университетом.

В системе муниципального образования выделилось лишь несколько школ, в которых есть компьютерные классы. Большое количество школ либо не имеет компьютеров, либо наличие их не влияет на образование детей. В условиях отсутствия средств и перспективы на них программа информатизации системы образования может рассматриваться лишь для решения ограниченного круга проблем.

Наши предложения состоят в том, чтобы направить централизованные усилия на обеспечение структуры образования по уровням следующего характера:

1) обеспечение потока писем и документов из управления образования в школы и назад; деловая переписка;

2) обеспечение потока педагогической информации дидактическими материалами контрольных и тестирующих работ; создание банка педагогической информации;

3) обеспечение конференций, семинаров, дистанционного образования;

4) создание электронного варианта образовательных программ в каждой школе и по каждому предмету, обеспечение их доступности;

5) обеспечение образовательной деятельности в режиме реального времени. Олимпиады и конкурсы.

• Одновременно и, вероятно, независимо отдельные школы будут развивать свои образовательные программы в соответствии со своими собственными ресурсами: материальные и финансовые средства, цели и задачи, характер и концепция, кадры и психология.

• Помимо этого возможны межрегиональные интеграции образовательных программ отдельных школ с использованием коммуникационных и информационных систем.

Мы можем реально представить себе, что информатизация образования происходит тремя блоками:

Попытаемся проанализировать реальные возможности лицея.

• *Лицей: доля информатики в учебном плане*

В учебном плане лицея информатика представлена в модуле «Информационные технологии» образовательной области «Технология».

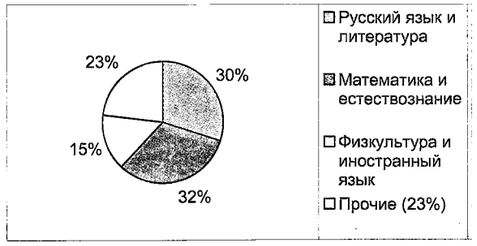
Она изучается с 5-го по 11-й класс по 2 часа в каждом, всего 14 часов в неделю. В начальной школе информатики нет. Общее количество часов в неделю по учебному плану в полной вертикали с 1-го по 11-й классы – 311. Легко считается доля информатики. Рассмотрим таблицу долей (или таблицу ценностей) изучаемых по учебному плану дисциплин.

Учебный план лицея				Базисный план	
Ранг	*Наименование предметов*	*Кол-во часов в нед.*	*Доля в общем кол-ве (%)*	*Кол-во часов в неделю*	*Доля в общем кол-ве (%)*
1.	Русский язык и литература	85	27,3	78	30,2
2.	Математика	61	19,6	46	17,77
3.	Обществознание, включая граждановедение, историю, географию	40	12,9	35	13,9
4.	Физика	23	7,4	12	4,6
5.	Иностранный язык	22	7	19	7,3
6.	Физкультура	20	6,4	20	7,3
7.	Информатика и информационная технология	14	4,5	2-4	0,8-1,5
8.	Химия	8	2,57	8	3,1
	Другие предметы	16	5,1		

Информатика и информационные технологии занимают в этой таблице 7-е место между физкультурой и химией. Таблица показывает, что русским языком и литературой мы занимаемся по учебному плану в лицее почти столько же, сколько математикой и физикой вместе, при этом надо заметить, что математика и физика для лицея являются профильными.

Во второй половине таблицы представлены доли учебных дисциплин в базисном плане. Базисный план – это минимальный обязательный образовательный стандарт. В нем позиции русского языка и литературы, математики и обществознания сохраняются, а позиции других дисциплин меняются. Место физики резко снижается, информатика опускается в самые низкие слои.

И в учебном плане лицея и в базисном плане распределение учебного времени оказывается следующим:



Русский язык и литература – одна треть всего времени;

математика и естествознание – еще одна треть времени;

все остальное – последняя треть.

В учебном плане лицея доля информатики увеличена в 4-6 раз, физики – почти в 2 раза, математики в старших классах – в 2 раза. Это обязательный для всех учеников лицея учебный план. Кроме обязательной части в лицее формируется дополнительная образовательная часть в виде факультативов и спецкурсов в общем объеме 156 часов в неделю. Из них на долю информатики и компьютерных технологий приходится до 10%. Это спецкурсы по олимпиадным задачам, издательские системы в школе, компьютерный монтаж видеофильмов в лицее.

Образовательные программы по информатике и информационным технологиям, включающие учебный план и спецкурсы, выводят учеников на выполнение многочисленных проектов, защищаемых на итоговых экзаменах по информатике: «Моделирование реакций в области ядерной физики», «Редактирование и обработка результатов интеллектуального марафона», «Анализ цен товаров различных фирм» и т.д.

Программы выводят на круг новых задач, на новые ступени в новых областях, горизонты которых трудно охватить своим сознанием. Компьютерные средства становятся необходимым элементом современной культуры.

Принимая во внимание, что ученическая среда в лицее в значительной своей части мотивирована на образование, на достижение реального результата, на стремление к самореализации, образовательная деятельность не ограничивается ни учебным планом, ни факультативами и спецкурсами.

• *Лицей: олимпиады и конкурсы*

Ученики лицея в течение года принимают участие более чем в двух десятках конкурсов и олимпиад городского уровня и выше. Значительная часть из них носит массовый характер. По итогам городских предметных олимпиад в прошедшем году ученикам лицея вручено 116 дипломов; областных – 62; в международном математическом конкурсе «Кенгуру» – 162 диплома и грамот; в турнире городов по математике – 51 диплом; в международной дистанционной олимпиаде «Третье тысячелетие» – 25 дипломов и т.д.

Среди всего этого многообразия олимпиад, в которых ученики лицея принимают участие, количество конкурсов по программированию – 7. Они начинаются в ноябре и заканчиваются в апреле. Все эти олимпиады, конкурсы, турниры, конференции несут огромную образовательную нагрузку и образуют специальную образовательную линию.

Для них характерно огромное количество нестандартных задач, которые предполагают более глубокое проникновение в разделы учебных дисциплин. Участие в олимпиадах требует от учеников высоких скоростей вычислительных процессов, большого объема памяти, точного определения логических связей и многих других интеллектуальных способностей. Требуются особенные черты характера: настойчивость, трудолюбие, упрямство и тому подобное…

Для ученика участие в олимпиадах – это возможность получить более объективную оценку его личностных качеств, возможность соотнести себя в среде своих сверстников, реализовать себя, расширить свой кругозор и круг общения, это надежда достичь высоких результатов, удовлетворить свои амбициозные потребности. Поэтому участие в олимпиадах раскрывает перед учеником многочисленные возможности, каких не может предложить учебный план, в особенности по информатике и информационным технологиям. Сочетание огромных волевых, физических, моральных и интеллектуальных качеств представляет собой некоторую неординарную личность, личность, успешно продолжающую учиться в высших учебных заведениях, с той же настойчивостью и постоянством стремящуюся к новым знаниям и новым горизонтам, способную нести высокие образовательные нагрузки.

• *Лицей: летняя школа*

Все, что было описано, заканчивалось с окончанием учебного года, с переводными экзаменами.

Но лицей с переводными экзаменами не заканчивается. Лето, которое начинается вместе с каникулами, несет в себе совершенно другие возможности организовать образование. Уже 6 лет во время летних каникул дети организуются для совместного отдыха и учебы. Сначала это был летний лицей, где учебными дисциплинами были математика, физика, история, иностранный язык, психология. Последние 3 года это летний компьютерный лагерь, в котором было 2 ядра:

• одно из них – школа олимпиадных задач;

• в другой школе литература реализовывалась через проект «Журналист–мастер» с рубриками: «репортаж с места событий», «интервью», «реклама», «проблема» и др. Ученики издавали газету, сами выполнялись все технические процедуры.

Осуществлялись проекты:

• иностранный язык – через проект «Компьютерное обеспечение разговорных тем»;

• история – через раздел «Введение в античную философию» с помощью проекта «Рабочая тетрадь» с конкурсными заданиями;

• мировая художественная культура – через раздел «Великие творения человека», где детьми изготовлялись сборники вопросов КВН.

Все эти проекты требовали освоения своих компьютерных сред. Спецкурсы по видеомонтажу, web-дизайну развивали практическую направленность образовательной информатики.

Таким образом, лето насыщалось интересной учебой в природных условиях вне рамок учебного года, когда решались задачи, не вкладывающиеся в учебные программы и планы. Ученики из школы олимпиадных задач вернулись домой с полной образовательной программой «в кармане». Каждый получил CD-диск с записью всех задач, рассматриваемых в школе.

Все это вместе взятое и представляет собой нашу школу, в которой образовательная информатика играет одну из ведущих ролей. Итогами работы лицея в этом направлении явились победы учеников на всероссийских и международных олимпиадах по информатике, успешная учеба в высших учебных заведениях, работа в качестве программистов и аналитиков, изданные в соавторстве учениками и выпускниками учебные пособия по информатике.

Е.В. Ведерникова

ПРЕПОДАВАНИЕ ИНФОРМАТИКИ В КФМЛ

Я работаю преподавателем информатики в КФМЛ – Кировском физико-математическом лицее с углубленным изучением математики и физики.

Как живется учителю информатики в таком специализированном учреждении, какие вопросы приходится решать, какие проблемы возникают?

Первая проблема – это выбор программы.

В настоящее время практика преподавания информатики в школе характеризуется следующими факторами:

• в обучении информатике сегодня используется несколько различных программ и учебных пособий, значительно отличающихся друг от друга по содержанию и направленности изложения материала, глубине изучения отдельных вопросов;

• складывается новая структура обучения информатике в среднем звене, в рамках которой базовый курс, обеспечивающий обязательный общеобразовательный минимум подготовки школьников по информатике, должен быть перенесен в среднюю школу;

• появилась тенденция размежевания задач формирования компьютерной грамотности и задач изучения основ информатики.

Перед курсом основ информатики стоит комплекс учебно-воспитательных задач, выходящих за рамки прикладных задач формирования компьютерной грамотности. Такое изменение направленности и целей обучения информатике несомненно ведет к коррек-

тировке его содержания. *В школах с углубленным изучением отдельных предметов курс информатики должен соответствовать высокому образовательному уровню этих учреждений.*

При выборе программы обучения возникает вопрос: какие цели должна реализовывать программа? Цели зависят от многих факторов (от профилирующего предмета, от наличия компьютерной техники и т. д), но в большей степени от того, с каким уровнем подготовки учащихся предстоит работать.

Уровень развития учащихся нашего лицея характеризуется повышенной подготовкой школьников в области математики и физики, высокими мыслительными способностями и интеллектуальными умениями. Отсюда требование повышенной подготовки в области информатики, тесной взаимосвязи ее с курсами математики (алгеброй, геометрией), и физики.

На преподавание информатики в КФМЛ отводится 2 часа в неделю. Дополнительное время занятий – это спецкурсы по информатике.

Основа программы – изучение программирования.

Цели программы:

• развить интерес к информатике (проблем с развитием интереса у школьников нет, всем нравится информатика, в данном случае речь идет об интересе именно к программированию);

• развить алгоритмическое, а следовательно, и структурное мышление;

• выстроить межпредметные связи, особенно с профильными предметами: математикой и физикой;

• создать условия на уроках информатики для развития творческих способностей учащихся.

На уровне 5-6-х классов изучение программирования ведется на примере языка Лого, в дальнейшем для изучения используется язык Паскаль.

Обучение программированию подразумевает выработку определенных **умений**:

• составлять алгоритмы для решения задачи и «видеть» алгоритмы в той или иной задаче;

• выбирать структуры данных, писать структурированные программы;

• писать, отлаживать и «читать »программы;

• составлять систему отладочных тестов.

В курс информатики КФМЛ входят **темы**:

• понятие операционной системы, ее составные части, свойства;

• понятие алгоритма, виды алгоритмов, способы их записей;

• алгоритмический язык Лого;

• основы булевской алгебры логики;

• системы счисления;

• структуры данных;

• алгоритмический язык Турбо-Паскаль;

• некоторые алгоритмы обработки данных (методы сортировки массивов, методы поиска).

• операционная система MS-DOS, основные команды;

• представление чисел в памяти компьютера;

• обзор технологий программирования.

Программа лицея по информатике для среднего звена является авторской и полностью реализует цели, означенные выше. Она разработана под руководством С.М. Окулова, декана факультета информатики Вятского государственного педагогического университета.

До недавнего времени мы не могли найти программу, которая бы реализовывала эти же цели, но на уровне 10-11-х классов, и являлась бы логическим продолжением программы 5-9-х классов.

В предлагаемых Министерством образования России программно-методических материалах есть множество программ и среди них программа «Информатика и основы вычислительной техники. 8-11 классы» А.Г. Гейна, А.И. Сенокосова для классов с углубленным изучением информатики. Но в ней математика присутствует только в изучении темы «Обзор вычислительных методов решения задач», а на углубленное изучение выбранного языка программирования отводится 20 часов. Выбор этой программы нас не устраивал.

И вот в последнем издании программно-методических материалов появился курс, основной задачей которого является знакомство учащихся с применением методов информатики для решения математических задач, а также с математическими методами, используемыми в информатике. Курс называется «Информатика и программирование» (авторы Н.Л. Беленькая, А.Г. Гейн, С.Л. Островский).

Изучение содержания обучения и тематического планирования позволило, изменяя структуру курса, исключая некоторые темы и изменяя количество часов для изучения отдельных тем, сформировать приемлемый вариант скорректированной программы (тем более что предлагаемый вариант - программа-максимум, и она предполагает исключение ряда тем).

Программа для 10-11-х классов также отвечает основным требованиям целеполагания и дополняет изучение темами:

• компьютерная графика на плоскости и в пространстве;

• точные и приближенные вычисления;

• приближенное решение уравнений;

• решение систем линейных уравнений;

• методы численного интегрирования.

Одной из особенностей преподавания информатики в 10-11-х классах лицея является наличие параллели – классов с экономическим уклоном. Включение в программу обучения информатике в этих классах тем, связанных с экономикой, изучение возможностей прикладных программ для моделирования экономических ситуаций, поиска решений – все это позволило выстроить взаимосвязь информатики с экономикой.

Особенно интересно применение «метода погружения». Ученикам предлагается «окунуться» в определенные экономические отношения между работниками вновь созданной компьютерной фирмы. Изучая определенные программы и продвигаясь по служебной лестнице (оператор, секретарь, кладовщик, экономист, начальник рекламного отдела, начальник планового отдела, бухгалтер), ученики выполняют функции данного работника и решают предложенные им задачи с обязательным применением компьютера.

Можно сказать, что на сегодняшний день в лицее сформирован программно-методический комплекс, позволяющий реализовать углубленное изучение информатики с акцентом на программирование и взаимосвязь с математикой и экономикой.

Вторая проблема – как учить, какие формы и методы применять.

Если в обучении ставятся цели формирования мыслительных способностей и интеллектуальных умений, то в процесс обучения вводятся упражнения по развитию общелогических умений и внелогических элементов деятельности (догадка, фантазия, воображение), творческих способностей, которые характеризуются:

• беглостью мысли (количество идей, возникающих в единицу времени);

• гибкостью ума (способность переключаться с одной мысли на другую);

• оригинальностью (способность находить решения, отличные от общепринятых);

• любознательностью (чувствительность к проблемам окружающего мира);

• умением выдвигать и разрабатывать гипотезы.

Ученики ждут новых форм знакомства с учебным материалом, в которых могла бы воплотиться их активность, деятельный характер мышления, стремление к самостоятельности.

Важным условием организации процесса усвоения знаний является выбор системы развивающего обучения.

Развитие личности, познавательной самостоятельности, умения видеть и формулировать проблему, анализировать факты, умение выдвигать гипотезы и проверять их – вот конкретные цели системы развивающего обучения.

Реализация основных принципов этой системы:

• проблемность обучения;

• индивидуализация и дифференциация;

• формирование приемов умственной деятельности творческого типа на уроках информатики – позволяет решить проблемы методики обучения.

Третья проблема – реализация творческих способностей школьников.

Творческой личности нужен выход за рамки своей среды общения, нужны условия конкурентности, соревновательности. Поэтому наши ученики участвуют в различных творческих конкурсах, проектах. Это и так называемый немецкий проект или конкурс, проводимый фирмой Сименс, конкурс по созданию мультимедийного шоу; конкурсы видеофильмов (в лицее есть соответствующее оборудование, программное обеспечение, обеспечивающее компьютерный монтаж видеофильмов); конкурсы Web-страничек, поддержка лицейского сайта.

Все это позволяет ученикам найти применение своим способностям, выразить свое отношение, свое видение той или иной проблемы и ситуации.

Е.В. Андреева, И.Н. Фалина

ПРЕПОДАВАНИЕ ИНФОРМАТИКИ В СУНЦ МГУ

Специализированный учебно-научный центр МГУ им. М.В. Ломоносова (школа-интернат им. А.Н. Колмогорова) был создан для обучения в старших классах средней школы ребят из различных регионов России (ранее – Советского Союза), одаренных в области математики и физики. В настоящее время в школе существует также и химический класс. Более 75% выпускников интерната стали студентами различных факультетов Московского университета.

Профильные классы являются неотъемлемой ступенью в системе непрерывного образования. Специфика подготовки выпускников таких классов (школ) позволяет им легко продолжать образование в высших учебных заведениях соответствующего профиля.

Цели создания специализированных классов:

• удовлетворение интересов школьника к конкретному предмету за счет более глубокого изложения материала, в том числе знакомство с нетрадиционными для школьной программы разделами и современными направлениями конкретной дисциплины;

• процесс образования в таких классах, за счет отбора детей, как правило, личностно-ориентированный, что позволяет школьнику наиболее полно удовлетворить свои интересы к конкретному предмету;

• школьникам предоставляется возможность заниматься научно-исследовательской работой под руководством специалистов;

• подготовка учеников в таких классах направлена на поступление в соответствующие вузы для продолжения образования.

При работе со школьниками специализированных классов необходимо учитывать проблемы преемственности преподавания профильных дисциплин в школе и в вузе. Со стороны школы в цепочке непрерывного образования «школа-вуз» рассмотрим опыт преподавания информатики в физико-математических классах СУНЦ МГУ.

Специфика набора в такие классы старшей школы такова, что дети с достаточно высоким уровнем знаний по физике и математике имеют разный уровень подготовки по информатике (отметим, что с аналогичной проблемой сталкиваются преподаватели

курса информатики в вузах). Эта особенность только усиливается, если в подобных классах обучаются ребята из различных регионов. У школьников, поступивших в СУНЦ, разброс уровня знаний по информатике, в силу всем известных причин, огромен: одни из них являются победителями областных олимпиад по информатике, а у других отсутствуют минимально приемлемые знания и умение работать с компьютером. Как правило, школьники, поступившие в 10-11-й физико-математический класс, уже изучали информатику в течение одного-двух лет. Они знакомы с основами алгоритмизации, имеют представление о системах счисления, работали с простейшими текстовыми редакторами, имеют опыт написания несложных программ.

Таким образом, перед сотрудниками кафедры информатики СУНЦ МГУ стояла задача разработки программы и методики преподавания информатики в классах с различным входным уровнем знаний учащихся, соответствующей высокому образовательному уровню школы. Для этого был проведен анализ требований, предъявляемых к знаниям и умениям студентов младших курсов, необходимым для успешного усвоения университетских курсов информатики на естественных факультетах МГУ имени М.В. Ломоносова (ВМиК, механико-математическом, физическом). На основе проведенного анализа были сформулированы достаточно высокие цели обучения информатике в физико-математических классах (10-11-е классы):

- подготовить школьников к успешному изучению университетских курсов, связанных с информатикой;
- выработать представление о целях, задачах и методах информатики как науки и показать ее связь с курсами математики и физики;
- развивать творческие способности учащихся;
- обеспечить овладение учащимися основами знаний о процессах преобразования, передачи и использования информации; раскрыть значения информационных процессов в формировании современной картины мира; показать роль информационных технологий и вычислительной техники в развитии современного общества;
- привить учащимся навыки сознательного и рационального использования компьютеров в учебной, а затем и профессиональной деятельности.

Содержательной основой курса информатики в СУНЦ МГУ является связка: «структура данных – действия над этой структурой». На протяжении всего курса идет движение по усложняющимся цепочкам: *абстрактная структура данных – реализация в ЭВМ этой структуры – операции над структурой как единым объектом – алгоритмы обработки данных этой структуры.*

Например, такой абстрактный объект, как целое число, существует независимо от того, как такая структура реализована в ЭВМ. В рамках курса рассматривается компьютерное представление целых объектов, типы целочисленных объектов, реализованные в Турбо-Паскале, основные операции, допустимые над целыми объектами, ошибки, возникающие при работе на компьютере с целочисленной арифметикой.

Или такой объект, как очередь. Рассматриваются основные алгоритмы обработки этой структуры, никак не привязываясь к реализации этой структуры в ЭВМ. Рассматриваются различные способы компьютерной реализации этой структуры: через массивы или через динамические переменные. Рассматриваются классы задач, эффективное решение которых возможно при представлении данных в виде очереди.

Кроме этого в курсе изучаются основы теории информации, алгебра логики, системы счисления, технологии программирования, включая объектно-ориентированное и визуальное программирование.

Таким образом ученикам прививается умение формализовать задачу, алгоритмическое мышление, т.е. умение составлять алгоритмы для уже формализованных задач, умение структурировать данные и реализовывать основные структуры на компьютере, решать задачи средней сложности на одном из языков программирования (обычно – Турбо-Паскале), воспитывается программистская культура.

Система специальных курсов (современные операционные системы, программирование на языке ассемблера, решение олимпиадных задач по информатике, базы данных, теоретическое программирование, основы вычислительной математики и т.д.) позволяет большинству школьников удовлетворить свои интересы в области изучения информатики и современных компьютерных технологий.

Такие информационные технологии, как текстовые процессоры, электронные таблицы, графические редакторы и т.п., в основном курсе не изучаются, но ряд заданий курса построен так, что требует их использования. Например, некоторые свои программные разработки учащиеся обязаны сопровождать грамотно оформленным руководством пользователя, результаты работы программы иногда требуется представить в виде таблиц и графиков и т.п. Вместе с полученными в рамках курса знаниями об основных принципах организации того или иного программного обеспечения этот подход обеспечивает достаточно высокий уровень осознанной компьютерной грамотности учащихся.

В качестве основных методических задач программы обучения информатике можно выделить выравнивание знаний по базовому курсу и подготовку школьников к успешному изучению курсов, связанных с информатикой, в вузах. Методику преподавания информатики для физико-математических классов с различными стартовыми знаниями мы назвали «выравнивающая и развивающая методика». Методика базируется на принципах развивающего обучения и когнитивной психологии; относится к технологиям личностно-ориентированного обучения: в центр внимания педагога ставится личность ученика, основной

акцент в деятельности педагога переносится с преподавания на учение. Девиз данной методики можно сформулировать так: «Информатика – это так просто, информатика – это так сложно». Учащимся, фактически начинающим обучение информатике на серьезном уровне «с нуля» или находящимся под влиянием такого феномена, как компьютерная тревожность (боязнь испортить или сломать компьютер, ощущение незнания и неумения, боязнь нового и незнакомого программного обеспечения), мы стараемся показать, что эта дисциплина им вполне доступна. Тем же, чей уровень компьютерной грамотности и программирования изначально относительно высок, мы в рамках того же самого курса стараемся показать, что информатика – гораздо более глубокая наука, чем кажется на первый взгляд, и им еще есть чему поучиться.

На преподавание информатики в СУНЦ МГУ отводится 3 часа в неделю: 1 час – лекция и 2 часа – семинарские занятия по подгруппам, состоящим из 10-13 человек. Практическая часть занятий проходит в компьютерных классах, оснащенных IBM-совместимыми персональными компьютерами, соединенными в локальную сеть. Кроме того, каждый школьник располагает как минимум двумя академическими часами в неделю для самостоятельных занятий в компьютерных классах во внеурочное время. Эти часы в основном используются для выполнения домашних заданий, но кроме этого ребята имеют возможность самостоятельно изучать различные информационные технологии и системы программирования, а также готовиться к творческим конкурсам по различным предметам.

В основе методики лежат такие *принципы развивающего обучения,* как:

1) принцип обучения на высоком уровне трудности;

2) принцип ведущей роли теоретических знаний;

3) принцип концентрированности организации учебного процесса и учебного материала;

4) принцип группового или коллективного взаимодействия;

5) принцип полифункциональности учебных заданий;

6) принцип взаимосвязи интенсификации умственного развития и содержания учебного материала и др.

Из основных *положений когнитивной психологии,* на которые опирается методика, можно выделить следующие:

1) в процессе обучения возникают не знания, умения и навыки, а их психологический эквивалент – когнитивные структуры, т.е. схемы, сквозь которые ученик смотрит на мир, видит и воспринимает его;

2) ведущей детерминантой поведения человека является не стимул как таковой, а знание окружающей человека действительности, усвоение которого происходит в процессе психического отражения;

3) из всех способностей человека функция мышления является руководящей, интегрирующей деятельность восприятия, внимания и памяти;

4) для всестороннего развития мышления в содержание обучения кроме материалов, непосредственно усваиваемых учащимися, необходимо включать задачи и проблемы теоретического и практического характера, решение которых требует самостоятельного мышления и воображения, многочисленных интеллектуальных операций, творческого подхода и настойчивых поисков;

5) для эффективного развития мышления когнитивная психология рекомендует использовать эффект «напряженной потребности».

Разработанная методика обладает двумя целевыми функциями: выравнивающей и развивающей.

Задачи развивающей функции: научить школьников воспринимать процесс обучения в качестве исследовательской работы; воспитывать стремление к самообучению; формировать систему адекватной самооценки; постоянно поддерживать высокий уровень мотивации к учению.

Задачи выравнивающей функции: определить входной уровень учащихся по информатике; ликвидировать пробелы в знаниях и умениях учащихся, причем эта задача должна решаться за счет специальной организации учебного процесса параллельно с изучением нового материала, а часто и благодаря ему; на протяжении всего учебного процесса вести мониторинг соответствия знаний и умений учащихся требованиям обязательного базового уровня.

Термин «выравнивающая» не является синонимом термина «уравнивающая»: методика позволяет организовать индивидуальную работу с каждым учащимся. Развивающая функция является ведущей по отношению к выравнивающей, так как процесс ликвидации пробелов выполняется в основном учащимися самостоятельно, учитель, используя методику, направляет деятельность ученика.

Методика подразумевает отказ от дифференциации по уровню знаний и умений, тем самым сглаживаются отрицательные стороны применения дифференцированного обучения. В отличие от традиционных технологий данная методика предполагает иной характер оценки учебной деятельности. Качество и объем выполненной учеником работы оценивается не с точки зрения ее соответствия субъективному представлению учителя о посильности, доступности знания абстрактному, усредненному ученику, а с точки зрения субъективной возможности ученика. В каждый конкретный момент оценка отражает персональное развитие ученика, качество его учебной деятельности.

Практическая реализация методики состоит в выполнении следующих требований к организации учебного процесса:

- все учащиеся обучаются по одной и той же программе (отказ от дифференциации по уровню знаний и умений);

• теоретический программный материал излагается на высоком уровне трудности в концентрированной форме; практические задания, одинаковые для всего класса, имеют специальную *концентрическую структуру*;

• используется работа в командах с оцениванием каждого участника по ответу одного;

• каждому ученику предоставляется свой темп обучения, задача учителя – вывести личность каждого ученика в режим развития;

• оценка, выставляемая ученику, отражает качество его учебной деятельности – когда ученик работает на пределе своих возможностей, он обязательно заслуживает высшей оценки, даже если с точки зрения возможностей другого ученика это посредственный результат;

• весь учебный курс разбивается на блоки, программный материал каждого блока закончен по смыслу (содержательная целостность) и невелик по объему (6–9 часов);

• изучение каждого блока строится с использованием следующих элементов учебного процесса: лекция, практическое занятие, самостоятельная работа учащихся, проверка знаний, обязательная работа в компьютерных классах во внеурочное время; обязательное требование к организации процесса обучения – создание условий для активизации развивающего потенциала каждого элемента учебного процесса;

• к каждому учебному блоку разрабатывается *практикум*, в который включены задачи концентрической структуры: ядро (базовый уровень) – 1-й уровень развития – 2-й уровень развития;

• в каждом практикуме есть задачи, которые не требуют хороших навыков работы с компилятором, текстовым редактором; для решения таких задач требуется время, ручка, бумага; когда задача решена, текст программы на Паскале занимает всего несколько строк; на первых порах решение таких задач выравнивает общий уровень учащихся и повышает их самооценку, так как отсутствие навыков работы на ЭВМ и, в частности, знание конкретного языка программирования не является в данном случае определяющим;

• к каждому заданию разрабатывается система проверочных тестов (практическая реализация концентрической структуры задачи), которая позволяет учитывать уровень знаний и способностей школьника;

• к каждому блоку разрабатывается система контроля и проверки;

• в случае вариативности заданий ученику предоставляется право выбора задания в соответствии с его интересами.

Как было сказано, практические задания, в некотором смысле одинаковые для всего класса, имеют специальную *концентрическую структуру*. Ядро (базовый уровень) – часть задания, которую школьник должен выполнить обязательно. Если школьник находится в «условно слабой» группе, то для получения отметки «отлично» ему достаточно выполнить только ядро задания. Школьник из «условно средней» группы для получения отметки «отлично» должен выполнить в рамках задания ядро и 1-й уровень развития. Соответственно школьнику из «условно сильной» для получения отметки «отлично»группы необходимо выполнить задание полностью.

Приведем пример задачи концентрической структуры.

Задача. Дана последовательность целых чисел. Написать программу сортировки этой последовательности по возрастанию.

Базовый уровень: программа, написанная учащимся, должна правильно сортировать любую входную последовательность.

1-й уровень развития: к требованиям базового уровня добавляется требование реализовать один из известных алгоритмов сортировки (в задании говорится – какой именно), подсчитать сложность алгоритма для входных последовательностей разного вида и сравнить полученные результаты с известной теоретической оценкой сложности.

2-й уровень развития: к требованиям 1-го уровня развития добавляется требование реализовать алгоритм сортировки по его описанию (обычно рекурсивный), теоретически оценить и проверить сложность реализованного алгоритма.

Концентрическая структура задачи проявляется и при проверке (например, при повышении уровня развития осмысленно расширяется набор тестов, на которых проверяется программа ученика). Отметим, что информатика, как ни одна другая дисциплина, позволяет разрабатывать задания с концентрической структурой. Использование таких заданий позволяет решить следующие проблемы:

• устраняется негуманное деление по уровню развития (школьник видит, что он решает ту же задачу, что и остальные, ему всегда открыт путь для выполнения задания в повышенном объеме);

• повышается мотивация учебной деятельности (школьник из слабой группы говорит: «Я тоже могу выполнить это задание», школьник из сильной группы не хочет понизить свой статус в глазах одноклассников);

• учителю не приходится тратить дополнительных сил и времени на составление и проверку разноуровневых заданий.

Обучение на высоком уровне трудности сопровождается соблюдением меры трудности, которая выражена в контроле качества усвоения. Главное в этом контроле не оценка знаний и навыков посредством отметок, а дифференцированное и возможно более точное определение качества усвоения, его особенностей у разных учеников данного класса. Используемая в СУНЦ МГУ система контроля основана на принципе развивающего обучения «в изучении программного материала идти вперед быстрым темпом». Быстрый темп изучения – это отказ от топтания на месте, от однообразного повторения пройденного. Практическая реализация принципа изучения в быстром темпе подразумевает постоянный контроль за знаниями и умениями учащихся, так как без убежденности в полном усвоении материала всеми учениками нет смысла двигаться вперед. В таком случае этот принцип выродится в спешку.

Для каждого блока выбирается свой вид контроля знаний. Но при любом способе контроля необходимо выполнять следующие условия: незамедлительное сообщение результатов оценки, любое задание должно быть проверено и оценено, максимальное включение ученика в процесс проверки. Разработанная система контроля состоит из следующих компонентов.

1. *Автоматическое тестирование программ учащихся на компьютерах*. В СУНЦ МГУ был создан сетевой программный комплекс для проверки учебных и олимпиадных задач в автоматическом режиме и тестирования знаний учащихся по теоретическим основам информатики. Система успешно применяется в учебном процессе с 1999-2000 учебного года. На нее возложены функции автоматической проверки заданий практикумов по программированию, проведение олимпиад в режиме *on-line* и тестирование знаний учащихся по любым предметам.

2. В контрольные работы включаются *«развивающие» вопросы и задачи*, т.е. такие, о которых непосредственно не рассказывалось на лекциях и семинарах, но для решения которых изложенных сведений, быть может с привлечением знаний из других школьных дисциплин, достаточно. Оценка «развивающего» вопроса не выносится за рамки оценки всей контрольной работы.

3. Используется такая форма контроля знаний, как *доклады с тремя участниками* – докладчик, содокладчик и оппонент. Система докладов с тремя участниками позволяет не только оценить знания учеников, но и формирует культуру спора. Каждый из участников оценивается по нескольким критериям: фактическое изложение материала, умение работать с литературой, оформление выступления.

4. При проверке домашних заданий преподаватели придерживаются правила *«одна неделя задержки – минус один балл»*. Выполнение каждого задания рассчитано на определенное время, как правило 1-2 недели. Если ученик сдает задание в срок, то он получает ту отметку, которую заслуживает. Если ученик по каким-либо причинам (кроме болезни) не сдает свою работу в срок, то отрицательная отметка не выставляется. Но ученик знает, что если через неделю он сдает работу на «отлично», то получает только «хорошо», через две недели отметку выше тройки он не получит. Такой подход вместе с принципом «*любое задание будет оценено*» стимулирует выполнение домашних заданий и предупреждает накапливание несданных работ.

5. В течение учебного года проводятся четвертные и семестровые *зачеты или коллоквиумы* с привлечением выпускников, на которые выносятся основные теоретические вопросы.

6. В конце года все классы сдают экзамен по информатике.

Методическая поддержка курса в первую очередь обеспечивается учебно-методическими пособиями, подготовленными сотрудниками кафедры информатики СУНЦ МГУ: «Информатика» (издательство факультета ВМиК МГУ), «Системы счисления и компьютерная арифметика» (издательство «Лаборатория базовых знаний»), «Сборник задач по программированию на языке Турбо-Паскаль» (издательство Бочкаревой). В школьной сети в электронном виде находятся тексты лекций. Используются обучающие программы по основам операционных систем (разработка факультета ВМиК МГУ) и языку HTML. Разработаны и используются программы для автоматической проверки практикумов и для компьютерного тестирования.

Методика выравнивающего и развивающего обучения информатике для физико-математических классов применяется в СУНЦ МГУ с 1996 г. Педагогический эксперимент показал, что предложенная методика позволяет организовать эффективный процесс обучения информатике, учитывая при этом специфические проблемы и особенности обучения информатике. В качестве *основных результатов* применения методики можно выделить следующие:

• методика позволяет за один учебный год параллельно с освоением нового программного материала ликвидировать пробелы в знаниях и умениях;

• достигается достаточно высокий уровень усвоения учащимися материала в пределах требований к обязательным результатам обучения;

• при освоении обязательного базового курса у всех учащихся постоянно поддерживается интерес к изучаемому материалу;

• формируется познавательная самостоятельность ученика и развиваются его творческие способности;

• у школьников формируется система адекватной самооценки, которая повышается в процессе обучения;

• преподаватель имеет возможность работать в режиме «индивидуального подхода» практически на одном и том же методическом материале;

• применение методики высвобождает преподавателю время для творческой работы над содержанием и способами обучения.

Л.Г. Белиовская

ЭВРИСТИЧЕСКИЕ ТЕХНОЛОГИИ В РАБОТЕ С ИНТЕЛЛЕКТУАЛЬНО ОДАРЕННЫМИ ДЕТЬМИ ЛИЦЕЯ № 1557 г. МОСКВЫ*

Знания – это сокровище,
которое всегда и всюду с тем, кто им обладает.

(Китайская поговорка)

В последние годы в России наблюдается переход к новому типу образования: от образования традиционного, с ассоциативной моделью знаний, к развивающему образованию с динамической моделью, от ориентации на среднего ученика к индивидуальному подходу [1]. При этом изменяется и сам процесс познания: от обучения как функции запоминания переходим к процессу умственного развития.

Технологией реализации задач развивающего образования являются дидактические процессы. Они состоят из познавательной деятельности, ее мотивации и управления познавательной деятельностью. Отличительной особенностью одаренных детей является тот факт, что эти учащиеся уже обладают достаточно высокой степенью мотивации. По этой причине наибольшие усилия по организации дидактических процессов с одаренными детьми направлены непосредственно на познавательную деятельность учащихся и управление этой деятельностью. Для этого в работе с интеллектуально одаренными детьми в первую очередь используются проблемно развивающие и эвристические методы индивидуальной работы.

Кратко остановимся на методах эвристической технологии. Достаточно высокий уровень развития знаний учащихся позволяет использовать в обучении более динамичные и творческие методы: исследовательский и эвристический. Монологический, диалогический, показательный, программированный и алгоритмический методы с их высокой степенью зависимости от преподавателя и довольно низкой скоростью освоения материала несостоятельны в работе с одаренными детьми. Исследовательский метод познавательной деятельности предлагает учащимся самим, согласно своим интересам и наклонностям, сформулировать задачу и исследовать ее. При эвристическом методе объясняется часть знаний, преднамеренно создаются проблемные ситуации, и ребенок получает право на поиск ответов, решений на поставленные задачи. Эвристические технологии базируются на таком специфическом виде работ, как коллективные самостоятельные работы [2]. Все учащиеся разбиваются на мини-группы, каждая из них самостоятельно изучает какую-либо часть целого процесса, а затем объединяем исследования мини групп в построении наиболее оптимального варианта целого процесса.

Перечень основных направлений и конкретных образовательных задач при работе с одаренными детьми

Компьютерное конструирование и моделирование учебных и реальных ситуаций, объектов и процессов

Имеется множество причин, в силу которых использование компьютеров в качестве инструментов познания является эффективной альтернативой компьютерным обучающим системам. Людьми, которые получают максимальные знания из обучающей системы, являются разработчики этой системы, а не учащиеся, для которых эта система предназначена. Разработчики получают эти знания в процессе создания системы. Другими словами, простейшим способом выучить что-либо является обучение этому других. Процесс разработки и создания образовательных материалов заставляет разработчика более глубоко изучить предмет. Это приводит к тому, что разработчик лучше понимает предмет, чем обучаемые, мышление которых ограничивается и контролируется обучающей системой. Отсюда следует простой вывод: необходимо расширить возможности учащихся, обеспечив их широкими возможностями компьютера в плане представления информации и самостоятельной разработки обучающих систем.

Инструменты познания активно вовлекают учащихся в процесс формирования знаний, что способствует их пониманию и усвоению, а не только воспроизведению в памяти того, что получено от преподавателя. Необходимо отметить, что инструменты познания не проектируются для того, чтобы снизить объем обработки информации с целью сделать процесс обучения более легким и эффективным, что является целью обучающих систем и большинства обучающих технологий. Инструменты познания скорее обеспечивают среду и средство, заставляющие обучаемых более интенсивно размышлять об изучаемом предмете и генерировать при этом идеи, что невозможно без этих инструментов.

Создание программы – это обязательный атрибут информатики. Эвристические технологии позволяют создавать программные решения проблемных задач, причем сами программы обладают развивающим потенциалом. Эту работу можно проводить на базе двух сред: интерактивной симуляционной или мультимедийной.

Интерактивная симуляционная среда «Искусственная жизнь» SIM LIFE [3] является своеобразной биологической лабораторией для глобальных экспериментов, позволяющей моделировать собственные миры, изменять их физические свойства; исследовать эволюционные и генетические процессы в популяциях и экосистемах. На первом этапе ученики осваивают возможности базовой программы, затем создают на ее основе свой подобный проект, например «Моя школа», «Мой город» и т.п. Целью же работы может стать создание прикладной прогностической программы, реально отражающей развитие школы, управы, города. Этот процесс полностью соответствует главному принципу образования: от простого к сложному. Учащиеся переходят от игры к новой собственной игре и затем к прикладной программе по мониторингу, экологии, управлению на уровне управы, города.

* По опыту работы в коллективе Ю.Н. Белехова,
при интеллектуальном влиянии Е.Г. Кабакова,
при административной поддержке Т.Н. Грабарник

Открыть для себя мультимедиа, работая в группе

Наша эра – это эра информации и коммуникации. Мультимедиа оказывает особое влияние на то, каким образом мы будем получать информацию, делать покупки и проводить досуг. Поэтому для нас тем более важно как можно раньше ознакомиться с новыми технологиями. «Только тот, кто знает, насколько разнообразны возможности мультимедиа, и умеет пользоваться ими, сможет утвердиться на рынке труда XXI столетия».

Мультимедийные среды (например, среда ЛогоМиры [4]), Flash или гипермедиа (составление Web-страниц) позволяют ученику, освободившись от рутинной работы, уделить достаточно внимания наиболее полному отображению решения исследовательских задач, обсуждению ее результатов, проверке и уточнению гипотез.

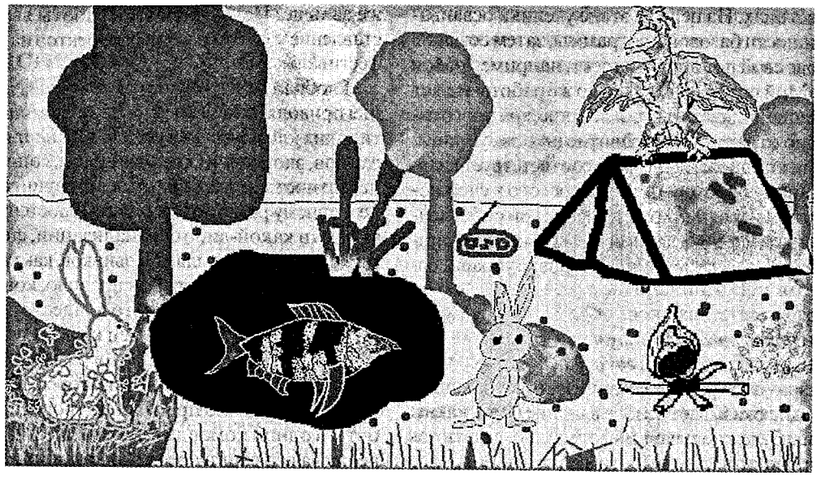
Представления с использованием средств мультимедиа и гипермедиа являются захватывающими, так как они многомодальны, т.е. одновременно воздействуют на несколько органов чувств и поэтому вызывают повышенный интерес и внимание у аудитории. Это очень важно при работе с новым «видеопоколением».

Учащимся дается краткая характеристика исследовательских задач и возможностей той или иной мультимедийной среды. Проводится групповое обсуждение постановок индивидуальных исследовательских задач учеников. Обсуждается представление информации по кадрам, размещение текстовой и графической информации в проекте с точки зрения эргономики. Детально прорабатывается авторский текст, включенный в проект, и заключительные выводы. На этом этапе активно используется высокий потенциал научных знаний преподавателей. При выборе темы исследования учащиеся руководствуются в первую очередь своими симпатиями и интересом к предмету исследования. Никаких ограничений и требований к выбору тем не предъявляется, поэтому исследовательские задачи охватывают все области знаний: и естественные науки, и гуманитарные. Преподаватели выступают в качестве экспертов и консультантов. Затем результаты исследования оформляются в анимационный проект на компьютере. Итак, проект готов. Что же дальше? Целью всей этой работы является представление мультимедийных проектов на конкурсах и фестивалях.

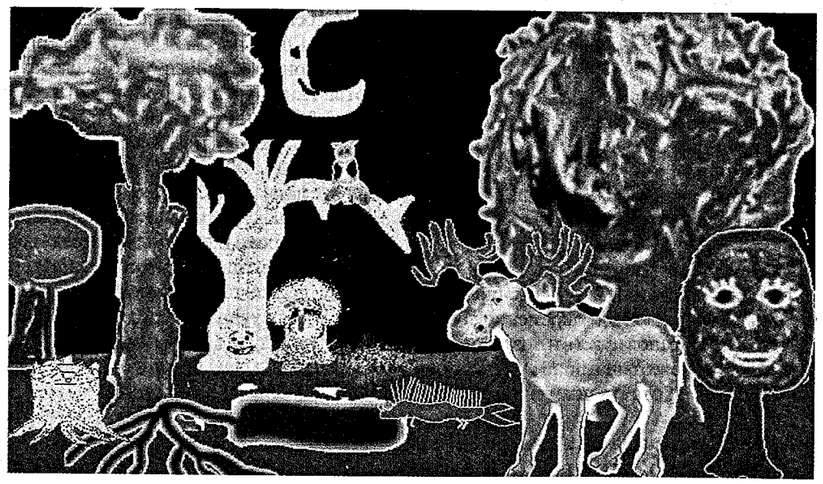
Глобализация рынков и технические преобразования приводят к неуклонному возрастанию значимости таких ключевых элементов, как, например, знание языков, экономическое мышление, социальная компетентность и способность к коммуникации в группе. Раннему развитию этих способностей, выходящих за рамки какой-либо специализации, способствуют такие мероприятия для школьников, как конкурс мультимедийных проектов. Как правило, конкурс проводится не только для тех, кто владеет техническими премудростями, он обращен ко всем тем, кто умеет сочинять, музицировать, фотографировать, снимать фильмы или обладает организаторскими способностями. В нем важно проявление творчества, духа работы в одной команде и увлеченности. В принципе не так важен ранг и престижность конкурса. Для конкурсантов главное, что их работу оценило компетентное жюри, увидели их сверстники, важно, что им пришлось выступать перед незнакомой аудиторией, отвечать на вопросы и выслушать критику. Все это – первый опыт ведения научно-исследовательской работы. Наши ученики принимают участие в работе секции «Новые информационные технологии» научно-практической конференции «Поиск», которая проводится Московским домом научно-технического творчества молодежи [5,6], в конкурсе шоу мультимедиа в рамках программы «Молодежь и знания», который проводит фирма «Сименс» совместно с Немецким культурным центром им. Гете в Москве, в Выставке научно-технического творчества молодежи на ВВЦ [7], в научно-практических конференциях, проводимых ведущими вузами России. Многие наши работы награждены призовыми дипломами и медалями.

Хотелось бы, пусть не столь подробно, как это он заслуживает, остановиться на интересном мультимедийном групповом проекте. Автор идеи его создания – талантливый педагог, большой друг детей, методист Института новых технологий образования Е.Г. Кабаков [8]. Каждому ребенку в группе предлагалось нарисовать один из элементов ЛЕСА, самый главный на его взгляд, и дать словесное пояснение. Затем из этих отдельных элементов составлялась общая картина ЛЕСА. Дети вместе с педагогом сами решали, где на общей картине ЛЕСА расположить тот или иной рисунок товарища, каким должен быть его масштаб. Через воспроизведение ЛЕСА, словесное или графическое, выявляется отношение тестируемого к окружающему миру, осознание его места в нем. Это один из многочисленных тестов Фрейда. Такие проекты были предложены ученикам 6, 7, 8, 10-х классов.

И по тому, насколько интересной получилась картина, можно судить о духовном богатстве ребят. А удачная композиция рисунка – показатель психологической комфортности ребят в данной группе.



Картина группы учеников 10-го класса



Картина группы учеников 7-го класса

Сенсорное развитие интеллекта

Одно из фундаментальных требований к современной образовательной среде – сенсорное развитие интеллекта учащихся, насквозь пронизанное информатикой. Наиболее естественно оно реализуется в телесно-двигательных играх, побуждающих учащихся решать самые различные познавательно-продуктивные, логические, эвристические и манипулятивно-конструкторские проблемы. Наиболее целостными и продвинутыми в этом смысле являются среды ЛЕГО-педагогики Control Lab [9] и RoboLab, включающие использование сенсоров, микропроцессоров и других аппаратных и программных цифровых устройств, вплоть до программных языков высокого уровня типа Лого [10]. Подобный подход задает исходные элементы некоторой материально-вещественной среды и общие структурные схемы учебной активности в ней, но оставляет большой простор для личного творчества учащегося. От работы руками учащиеся беспрепятственно переходят к наблюдению и анализу совершаемых ими процедур на экране компьютера. Изучение современных технологических роботизированных процессов путем их конструирования и программного моделирования является практической реализацией программы по приобретению учащимися профессиональных навыков.

В последнее время начинает меняться тематическая направленность проектов. Если раньше тематика работ имела ярко выраженную фантазийную направленность, то сейчас становится наиболее важным выполнение социально значимых проектов, которые имеют практическое применение на предприятиях Зеленоградского района г. Москвы. Мы готовим несколько таких работ, результаты которых будут внедрены в производство, строительство, решение транспортных проблем города.

Литература

1. Современные образовательные технологии. Пособие для вузов и ИПК / Под ред. Г.К. Селевко. М.: Народное образование, 1998.
2. *Белехов Ю.Н.* Методологические принципы работы с интеллектуально одаренными детьми в Зеленограде // Образование и наука. Вып. 11. Опыт и перспективы работы с интеллектуально одаренными детьми в ЗУО. М., 1999.
3. Искусственная Жизнь: Метод. пособие для учителя. Примеры компьютерных экспериментов / Сост. М.Е. Рыжликова. Русская версия. М.: Институт Новых Технологий Образования (ИНТ), 1998.
4. *Рубашкин Д.Д.* Мультимедиа в школе. М.: Институт Новых Технологий Образования (ИНТ), 1996.
5. Научно-практическая конференция «ПОИСК-2000»: Тез. докл. М., 2000; http://www.dnttm.ru
6. Научно-практическая конференция «ПОИСК-2001». Тез. докл. М., 2001; http://www.dnttm.ru
7. Фестиваль научно-технического творчества молодежи Москвы и Московской области: Каталог. М.: ВВЦ, 2001.
8. *Кабаков Е.Г.* Радость сотворчества // Компьютер в школе. 1999. №10.
9. LEGO – Лаборатория (Control Lab). Проекты: Учеб.-метод. пособие. М.: Институт Новых Технологий Образования (ИНТ), 1998.
10. *Martha N.* RoboLab. Getting started 2. Teacher's Guide for Robolab 2.0 Software. Massachusetts: US, 1999.

Е.А. Васенина

ЦЕЛИ ИЗУЧЕНИЯ ИНФОРМАТИКИ В ШКОЛЕ И УЧЕБНИКИ «НОВОЙ ВОЛНЫ»

Последние годы (начиная примерно с 1998 г.) ознаменовались появлением значительного количества учебников и учебных пособий по информатике, которые дополнили группу учебников, ставших уже традиционными [6, 10, 11, 12, 13, 14]. В это же время усиливается дискуссия о содержательном наполнении курса школьной информатики, происходит пересмотр и качественное изменение целей обучения в силу того, что основная цель, которая декларировалась при введении курса информатики в систему общеобразовательных учебных дисциплин средней школы – достижение компьютерной грамотности, – утрачивает актуальность. С одной стороны, в условиях нарастающей компьютеризации общества она может быть достигнута другими средствами (самостоятельное освоение компьютера на бытовом уровне, использование компьютера в преподавании других дисциплин, узконаправленная профессиональная подготовка и т.д.), а с другой – столь утилитарная цель, как подготовка компьютерно грамотного пользователя, даже в малой степени не исчерпывает общеобразовательного потенциала такой перспективной науки, как информатика.

На первый план, и это закреплено в Проекте федерального компонента Государственного образовательного стандарта начального общего, основного общего и среднего (полного) образования образовательной области «Информатика» (ниже везде «Проект образовательного стандарта») [16], выдвигаются цели развития мышления школьников и формирования у них информационной картины мира как непременной составляющей научного мировоззрения. Не забыта и цель подготовки учащихся к труду и продолжению образования в условиях информационного общества (по сути, та же компьютерная грамотность).

В соответствии с поставленными целями в Проекте образовательного стандарта определены шесть основных содержательных линий школьного курса информатики. Рассмотрим, как указанные содержательные лини были представлены в традиционных учебниках информатики, и как изменилась ситуация в учебниках «новой волны».

Линия информационных процессов. В традиционных учебниках понятия «информация» и «информационные процессы» рассматриваются в ознакомительном ключе. От бытового представления об информации как сведениях об окружающем мире сразу происходит переход к техническому пониманию информации как последовательности сигналов. Измерение информации сводится к измерению объема данных.

Новые учебники уделяют данной тематике больше внимания. Говорится о многогранности понятия «информация», о связи фундаментальных понятий «информация» и «модель», о роли информации в управлении, функциях обратной связи, рассматриваются различные подходы к измерению информации. [1, 2-5, 20, 21].

Наиболее удачно тема представлена в учебнике [1], где достаточная полнота освещения вопроса соединена с простотой и ясностью изложения. Это качество характерно для учебника [1] в целом и является одним из основных достоинств данной книги, весьма интересной как с содержательных, так и с методических позиций.

Еще более подробно тема рассмотрена в пособии [20], где объем содержания и стиль изложения ориентирован на старшеклассников. Однако все учебники объединяет теоретичность изложения. Неясно, каким образом использовать компьютер на уроках, посвященных данной тематике.

Линия представления информации. Для новых учебников характерно внимание к языку как способу представления информации [20]. Новое развитие получила тема представления различных видов инфор-

мации, которая имела место и в традиционных учебниках [13, 14]. С новой силой зазвучала тема «Системы счисления» [1, 20] , которая и раньше не была забыта [12, 14], но занимала более скромное место.

Все эти факты весьма отрадны, но есть два существенных замечания. Во-первых, изложение темы вновь абсолютно теоретично. Как применить в обучении компьютер – неясно. Это порождает тем большие проблемы, чем больший объем учебника посвящен данной тематике.

Во-вторых, акцент смещен в сторону представления данных именно в памяти ЭВМ, скажем так, «на микроуровне». А вот вопросам представления информации и организации данных «на макроуровне», в виде величин и структур данных, уделено значительно меньше внимания, нежели в традиционных учебниках. Например, понятие массива (единственной структуры данных, предложенной к рассмотрению) изучается только в учебниках [7] и [20].

А ведь практическая работа по организации и структурированию информации, представлению ее в виде, удобном для компьютерной обработки, немало способствует развитию мыслительных способностей учащихся. Здесь представляется интересным подход к обработке данных как параметров некоторого объекта, их представление в соответствующем виде при переходе от информационной модели к компьютерной [2-5].

Линия исполнителя (компьютера). Когда речь идет об устройстве и основных принципах работы компьютера, наиболее заметно изменение позиции, угла зрения авторов новых учебников в изложении данной темы. Если в традиционных учебниках главенствовал функциональный подход, рассмотрение основ архитектуры, то для большинства новых учебников характерен подход конструктивный и, более того, пользовательский. В ряде учебников [8, 17] основным устройством компьютера назван системный блок, а в нем материнская плата. Интересно, что скажут авторы, если конструкторы вычислительной техники сумеют обойтись без материнской платы или, паче чаяния, без системного блока – ведь аппаратные средства так быстро эволюционируют.

А вот в традиционных учебниках основные блоки компьютера выделяются с точки зрения происходящих в них информационных процессов, и это вряд ли устареет (как не устарело еще со времен Ч. Бэббиджа). Безусловно, ученику также важно знать и основы конструкции современных персональных компьютеров, но это не отменяет важности фундаментальных основ. Здесь следует отметить попытку соединения обоих подходов, представленную в учебнике [1].

Кроме того, практически во всех традиционных учебниках рассматриваются принципы функционирования компьютера, даются ответы на вопросы, как компьютер обрабатывает последовательности двоичных разрядов (работа логических элементов), как достигается автоматизация обработки данных (основной алгоритм работы процессора, по терминологии А.П. Ершова).

Надо сказать, что в учебнике [20] рассматриваются логические основы устройства компьютера и магистрально-модульный принцип его построения, но сделано это так, что требуется усилие для понимания важности роли сумматора, триггера и даже процессора в устройстве и функционировании компьютера.

Важным представляется и тот факт, что в отличие от новых учебников, где изложение темы насквозь теоретично, авторы большинства традиционных учебников [13,14] предлагали учебные программные средства, с помощью которых ученик мог составить программу «в машинных кодах», промоделировать работу электронных элементов и схем. Все это описывалось в учебнике, и предлагались задания для выполнения практической работы.

Несомненным достоинством современных учебников является соединение изучения вопросов аппаратного обеспечения компьютера с вопросами базового программного обеспечения ([1], [7], [20] и др.), что продолжает направление, начатое еще учебником [13], и формирует представление о компьютере как комплексе, органично соединяющем аппаратную и программную составляющие.

Алгоритмическая линия. Если составление алгоритма и фиксация его средствами некоторого языка, изучение способов организации действий и данных (базовых алгоритмических структур и структур данных) составляло основу практически всех традиционных учебников информатики, то в новых учебниках алгоритмизация в значительной степени утратила свои позиции. Хорошо это или плохо, постараемся обсудить ниже, а пока сравним некоторые факты (см. табл.).

В качестве комментария к таблице следует отметить:

1. В учебниках [11, 13] изложение разделов «Устройство ЭВМ» и «Применение ЭВМ» ведется с алгоритмических позиций, ибо основная идея учебников – идея алгоритмизации. Даже системы обработки текстов и информационно-поисковые системы рассматриваются не с пользовательских позиций, а с позиции алгоритмов работы с текстами или алгоритмов поиска. Это в еще большей степени усиливает значимость идеи алгоритмизации.

2. Хотя в учебниках [2-5] объем материала, посвященного вопросам алгоритмизации, очень мал, эти вопросы весьма органично соединяются с изучением информационных технологий и информационного моделирования, что в определенной степени усиливает внимание к алгоритмизации.

3. Особо подчеркнем, что учебник [20], где алгоритмизации посвящено 10% объема учебного материала, это *углубленный* курс информатики для *старших* классов.

4. Несмотря на значительную разницу между традиционными и новыми учебниками в объеме учебного материала, посвященного алгоритмизации, в содержании (перечне рассматриваемых вопросов) ее почти не наблюдается – рассматриваются основные алгоритмические конструкции, полпрограммы, организация данных в форме простых величин. Это происходит оттого, что в новых учебниках очень мало задач, как разбираемых, так и для самостоятельного решения, в то время как в традиционных учебниках практическая деятельность учащихся рассматривалась как основа обучения.

Таблица

Учебник	Примерное отношение объема материала, посвященного вопросам алгоритмизации в % к общему объему учебника	Инструмент реализации алгоритмических конструкций	Изучаемые способы организации действий	Изучаемые структуры данных
[10]	70%	Школьный алгоритмический язык, Basic	Базовые алгоритмические конструкции, подпрограммы	Простые величины, массивы, строки
[13]	60%	Школьный алгоритмический язык	Базовые алгоритмические конструкции, подпрограммы	Простые величины, массивы, строки
[14]	70%	Язык исполнителей +, Basic	Базовые алгоритмические конструкции, подпрограммы	Простые величины, массивы, строки
[12]	40%	Запись на псевдокоде + Basic	Базовые алгоритмические конструкции, подпрограммы, графические команды.	Простые величины, массивы
[1]	20%	Язык обобщенного графического исполнителя + Pascal	Базовые алгоритмические конструкции, подпрограммы	Простые величины
[20]	20%	Visual Basic	Базовые алгоритмические конструкции, подпрограммы	Простые величины, массивы
[2-5]	15%	LOGO	Базовые алгоритмические конструкции, подпрограммы	Простые величины
[7]		Pascal	Базовые алгоритмические конструкции, подпрограммы	Простые величины, массивы

Линия формализации и моделирования. В традиционных учебниках формализации и моделированию уделялось немало внимания: в учебниках [10], [12], [14] понятие информационной модели рассматривалось в рамках темы «Этапы решения задач», в учебниках [11, 13] предлагалось «информационное моделирование исполнителя», практически во всех учебниках рассматривался вычислительный эксперимент. Однако в новых учебниках интерес к этой тематике значительно возрастает.

Моделирование изучается как метод познания ([2], [20]), даются разнообразные классификации моделей ([2-5], [20]), построение информационной модели рассматривается как основной этап решения задачи с помощью компьютера не только в случае составления программы на языке программирования, но и в случае реализации алгоритма с помощью информационных технологий общего назначения ([1], [2-5]). Более того, например, в учебниках [2-5] информационные технологии рассматриваются как инструмент реализации информационных моделей. Такой подход делает изложение темы моделирования в этом учеб-

нике особенно интересным, ибо дает широкие возможности для практической деятельности учащихся.

Линия информационных технологий. В новых учебниках, по сравнению с традиционными, эта тема получила наибольшее развитие. Тому есть много причин, но главная – появление большого числа удобных, мощных пользовательских программных средств, реализующих информационные технологии общего назначения. Именно этим программным средствам, изучению их интерфейса, техники управления и взаимодействия с ними посвящено наибольшее число страниц современных учебников. Особенно ярко это проявляется в учебниках [17-20] и [21, 22]. Даже в тяготеющем к фундаментальности учебнике [20] раздел «Информационные технологии» изложен сугубо с позиций пользователя.

Тем отраднее работать с учебником [1], где не рассматриваются конкретные программные средства, а значит, нет конкретных инструкций о нажатии конкретных кнопок, зато информационные технологии общего назначения показаны с позиций их функциональных возможностей.

Такой представляется, на взгляд автора, эволюция содержания школьных учебников информатики. Если попытаться проанализировать это содержание с точки зрения его соответствия провозглашаемым ныне целям изучения информатики – открывается парадоксальная вещь. Именно первые учебники, будь то учебники линии А.П. Ершова–А.Г. Кушниренко или В.Г. Житомирского–А.Г. Гейна, или даже в определенной степени учебник В.А. Каймина, в значительно большей мере ориентированы на развитие свойств ума и формирование информационной картины мира, нежели многие из учебников «новой волны».

Интересно вспомнить, какие цели считали главными авторы традиционных учебников, имея в виду достижение компьютерной грамотности. Главной идеей учебников линии А.П. Ершова–А.Г. Кушниренко была идея алгоритмизации, цель состояла в формировании умения составлять алгоритм как план некоторой деятельности и умения фиксировать этот план с помощью некоторого языка. Группа авторов, руководимая В.Г. Житомирским, объявляла главным в обучении информатике научить решать задачи с помощью ЭВМ.

Решение задачи с помощью компьютера предполагает:

- умение отобрать нужную для решения задачи информацию;
- представить ее в удобном для обработки виде, выбрав подходящие структуры данных;
- определить метод получения результирующей информации;
- в соответствии с выбранным методом составить план действий для получения результата;
- детально его проработать и записать таким образом, чтобы составленный план (алгоритм) можно было выполнить формально, не задумываясь о цели работы (любая прикладная программа и компьютер в целом является именно формальным исполнителем);
- заранее просчитать различные варианты действий исполнителя, спрогнозировать его деятельность в различных ситуациях;
- проанализировать полученные результаты, найти ошибки и внести соответствующие коррективы.

Согласитесь, что обучение перечисленным умениям помогает формировать весьма полезные мыслительные навыки и что в школьном курсе затруднительно найти учебный предмет, в рамках которого это можно было бы сделать лучше, чем на уроках информатики, а точнее, в рамках обучения программированию. Особенно это касается привычки к непрерывному тщательному планированию любой деятельности и способности к прогнозированию ее результатов, просчету вариантов хотя бы на несколько шагов вперед.

Авторы учебников «новой волны» также уделяют внимание изучению алгоритмизации и программирования (с помощью того или иного инструментария), однако этой тематике уделяется, как правило, не более четверти учебного времени, и овладевать искусством составления плана деятельности ученики начинают отнюдь не с первых шагов изучения информатики.

Так, в учебнике И. Семакина и др. [1], ориентированном на изучение базового курса в среднем школьном звене, понятие алгоритма изучается в 10-й главе (из 12). Рассматриваются только базовые алгоритмические конструкции, которые предполагается реализовывать с помощью обобщенного графического исполнителя (ГРИС), а в рамках углубленного курса – с помощью языка программирования Паскаль. При этом организация данных рассматривается лишь в виде простых переменных. Интересно, что в этом учебнике понятие алгоритма связывается с управлением и использованием информации для организации обратной связи – хороший пример того, как изучение алгоритмизации и программирования работает на формирование информационной картины мира.

Учебник Н. Угриновича [20] предназначен для углубленного изучения информатики в 10-11-х классах, однако и здесь понятие алгоритма рассматривается уже после того, как изучены основы логики, представление и кодирование информации, даны начальные знания об устройстве и программном обеспечении компьютера. Причем рассматривается будто впервые, и раньше ученики не слышали такого слова. (Заметим в скобках, что такая ситуация характерна для данного учебника: непонятно, изучен ли уже базовый курс или информатика для учеников начинается в

10-м классе. Если верно последнее, то для начинающих учебник весьма сложен как по содержанию, так по методам и стилю изложения. Но если базовый курс уже позади, то можно бы опереться на уже имеющиеся знания, а этого в учебнике не чувствуется.)

К изучению предлагается объектно-ориентированное программирование на языке Visual Basic. И вновь этой тематике посвящены две главы из шестнадцати (5-я и 14-я). И даже в этом небольшом объеме основная доля принадлежит изложению конструкций языка в объяснительно-иллюстративном ключе, хотя такой подход в наименьшей степени способствует научению методам составления алгоритмов и самостоятельности в планировании действий.

Интересный подход к изучению алгоритмических конструкций предложен в серии учебников под редакцией Н. Макаровой [2, 3, 4, 5], охватывающих обучение информатике с 6-го по 11-й класс. Понятие алгоритма и базовые алгоритмические структуры первоначально не связаны с их реализацией в каком-либо языке. Например, с понятием цикла ученики впервые встречаются при изучении графического редактора, когда графический объект может быть построен как последовательность повторяющихся фрагментов. Для данных учебников характерно взаимопроникновение изучения алгоритмов и информационных технологий (разделы, которые обычно противопоставляются друг другу). Далее ученикам 6-8-х классов в качестве инструмента реализации алгоритмических конструкций предлагается язык ЛОГО – и это именно инструмент, с помощью которого ученики наилучшим образом отображают ход своих мыслей, фиксируют план решения задачи и сообщают его компьютеру.

Логично было бы предположить, что в старших классах изучение алгоритмизации и программирования будет продолжено именно в таком ключе – применение к решению разнообразных задач, тем более что ранее в учебниках очень много внимания было уделено информационному моделированию. Однако старшеклассникам авторы предлагают в течение двух лет совершенствоваться в области компьютерного делопроизводства, издательского дела, получать иные пользовательские навыки.

Освоению информационных технологий общего назначения в учебниках отдается не менее половины объема. А такие книги, как [17, 18, 19] и [21, 22], целиком посвящены подготовке пользователей, пусть и достаточно грамотных. Причин тому много. Это и реакция на преобладание обучения программированию в первые годы, и желание использовать новые замечательные возможности, предоставляемые новыми замечательными программами, это и ответ на запросы общества, желающего как можно быстрее научиться работать «на компьютере».

Трудно спорить с тем, что необходимо учить работе с текстами и графикой, пользоваться электронными таблицами, базами данных, телекоммуникациями. Весь вопрос, как это делать. Большинство учебников (исключение составляет, похоже, лишь учебник И. Семакина и др.) предлагают конкретные инструкции по управлению конкретными программными средствами. Как только программные средства поменяются (или в распоряжение ученика попадет другое средство), так сразу предлагаемые знания потеряют актуальность. Но базовые технологии общего назначения останутся, просто иначе будут реализованы. Следовательно, стоит показывать, какие основные функциональные возможности они предоставляют и как эти возможности можно применить к решению задач.

Что касается технических приемов взаимодействия с конкретными программными средствами, то здесь ученикам следует предоставить максимально возможную самостоятельность. Ведь тот, кто умеет мыслить, технику управления осваивает без труда. Может быть, стоит удалить из основного текста учебника конкретные указания, собрав их в приложении в виде *кратких* инструкций по выполнению наиболее общих действий, характерных для изучаемых программных средств. Это было бы неплохим подспорьем учителю для подготовки дидактических раздаточных материалов.

Практически для всех новых учебников характерно декларативное изложение больших объемов теоретического материала. Замечательно, что появились новые разделы, посвященные информации, ее представлению и измерению, информационным процессам, изучению арифметических и логических основ компьютера. Это, безусловно, нужно и важно, особенно с позиций формирования информационной картины мира, но если преподавать, руководствуясь схемой, предлагаемой учебниками, то изучение информатики практически не будет отличаться от изучения биологии, истории, химии и пр. Это выслушивание учителя, выучивание учебника, выполнение упражнений, например по переводу чисел из одной системы счисления в другую, и переход к следующему параграфу.

Знание дается ученику свыше, а ведь были надежды, что применение компьютеров позволит изменить традиционные методы в преподавании не только информатики, но и других школьных предметов, сделать познавательную работу учеников более самостоятельной, осознанной, творческой, а значит, и более эффективной. Информатика, которая должна была изменить методологию обучения, сама оказалась «переработанной» репродуктивной методикой, характерной для традиционной школы.

Вновь обратимся к учебникам. Если строго следовать данному в них порядку изучения материала, то при рассмотрении большого числа глав и параграфов компьютер можно вообще не включать – незачем. Авторы возразят – на то и учитель, чтобы орга-

низовать обучение оптимальным путем. Однако в учебнике должен присутствовать не только содержательный блок, но и процессуальный, или блок средств. Иными словами, читая хороший учебник, учитель видит, как он будет вести занятия из урока в урок, как подведет учеников к пониманию материала и какие средства (в нашем случае, аппаратно программные) сможет для этого применить.

В этом отношении вновь более выигрышно выглядят традиционные учебники информатики. Хотя их авторы знали, что большая доля учеников будет изучать информатику в безмашинном варианте, они ставили задачу применения компьютера на каждом уроке и показывали учителю, как он может это сделать при наличии соответствующих средств. Было принято не просто писать учебник, но создавать программно-методический комплекс, поддерживающий этот учебник.

На это могут возразить, что сейчас имеется достаточно большое количество учебных программ. Учителю есть из чего выбрать и применить на уроках, скажем, обучающие программы по устройству ЭВМ или тренажеры, моделирующие работу логических элементов (и множество других). Но все это будут средства, с помощью которых, по выражению С. Пейперта, «компьютер программирует ребенка», а надо, чтобы «ребенок программировал компьютер»

Да, учебники «новой волны», обобщая 15-летний опыт преподавания, показывают, что информатика – это не только программирование, как это было вначале. Да, в них «под одной крышей» собраны вопросы, обязательные для полноценного изучения основ науки информатики:

- информация, ее представление и измерение;
- арифметические и логические основы компьютера;
- алгоритм и базовые алгоритмические конструкции;
- моделирование и формализация;
- устройство, основные принципы, аппаратная реализация работы компьютера;
- программное обеспечение компьютера: системное и прикладное.

Однако наряду с этим проявилась и недостаточность простого механического включения всех этих «параграфов» в учебник. Теперь требуется думать, как объединить достаточно разрозненные даже по стилю изложения темы в единое целое. И сделать это так, чтобы ученик мог с максимальной пользой для развития своего ума использовать компьютер на каждом уроке.

Можно предположить, что цементирующей идеей вновь станет идея алгоритмизации, соединенная с моделированием. Среда программирования, равно как и технологии обработки текстовой, графической, табличной информации, будут рассматриваться как инструменты для реализации алгоритмов, в которых, в свою очередь, воплощаются информационные модели окружающей нас действительности – неисчерпаемого источника всевозможных задач, для решения которых совершенно необходимо применить компьютер.

В частности, предметными областями, дающими множество интереснейших с точки зрения составления алгоритма задач, являются те самые параграфы, изучение которых так теоретизировано сейчас – двоичное кодирование, алгебра логики, системы счисления и пр. Действительно, пусть ученик вместо механического заучивания двоичной таблицы умножения (сложения) или правил (алгоритмов!) перевода десятичных чисел в двоичные или изучения таблиц ASCII и UNICODE, задумается, как все многообразие окружающей его информации представить всего двумя знаками и, главное, как научить этому компьютер. И почему двумя? Может быть, три – выгоднее? Или четыре? И как можно этот алгоритм (например, представления чисел в двоичной системе счисления) реализовать на компьютере? Воспользоваться электронными таблицами или составить программу на Паскале?

Именно в процессе поиска ответов на подобные вопросы и будет формироваться информационная картина мира, которую трудно нарисовать словами – она рождается в процессе самостоятельной практической познавательной работы учеников под руководством учителя.

Движение в этом направлении уже идет, его элементы можно увидеть и в рассмотренных выше учебниках, да и в традиционных «старых» учебниках тоже. Важно только не абсолютизировать ту или иную точку зрения, но найти гармоничное решение сложных проблем.

Литература

1. Информатика. Базовый курс для 7-9 классов /И. Семакин и др. М.: Лаборатория Базовых Знаний, 1999.
2. Информатика. 6-7 класс / Под ред. Н.В. Макаровой. СПб.: Питер Ком, 1998.
3. Информатика. 7-8 класс / Под ред. Н.В. Макаровой. СПб.: Питер Ком, 1999.
4. Информатика. 9 класс / Под ред. Н.В. Макаровой. СПб.: Питер Ком, 1999.
5. Информатика. 10-11 класс / Под ред. Н.В. Макаровой. СПб.: Питер Ком, 1999.
6. Информатика: учебник для 8-9 кл.средней школы / А.Г. Гейн, Е.В. Линецкий и др. М.: Просвещение, 1994.
7. *Кузнецов А.А., Апатова Н.В.* Основы информатики. 8-9 класс: Учебник для общеобраз. учеб. заведений. М.: Дрофа, 1999.
8. *Ляхович В.Ф.* Информатика: Пособие для учащихся 10-11 классов. М.: Просвещение, 1998.
9. Обязательный минимум содержания образования по информатике // Информатика и образование. 1999. №7. С. 2.
10. Основы информатики и вычислительной техники. Учеб. пособие для учащихся. В 2 ч. / Под ред. А.П. Ершова, В.М. Монахова. М.: Просвещение, 1985-1986.

11. Основы информатики и вычислительной техники. Учеб. пособие для учащихся / Под ред. А.П. Ершова. М.: Просвещение, 1988.
12. Основы информатики и вычислительной техники. Учеб. пособие для учащихся / В.А. Каймин и др. М.: Просвещение, 1988.
13. Основы информатики и вычислительной техники. Учеб. пособие для учащихся / А.Г. Кушниренко, Г.В. Лебедев и др. М.: Просвещение, 1990.
14. Основы информатики и вычислительной техники: Пробный учебник для 10-11 кл. ср. шк. / А.Г. Гейн, В.Г. Житомирский и др. М.: Просвещение, 1992.
15. *Пейперт С.* Переворот в сознании: Дети, компьютеры и плодотворные идеи / Пер. с англ. М.: Педагогика, 1990.
16. Проект Федерального компонента Государственного образовательного стандарта начального общего, основного общего и среднего (полного) образования. Образовательная область «Информатика» // Информатика и образование. 1997. №1.
17. *Симонович С.А., Евсеев Г.А., Алексеев А.Г.* Общая информатика. 5-9 класс: Учеб. пособие для средней школы. М.: АСТ Пресс, 1998.
18. *Симонович С.А. и др.* Специальная информатика. 10-11 класс: Учеб. пособие для средней школы. М.: АСТ Пресс, 1998.
19. *Симонович С.А. и др.* Практическая информатика. 5-11 класс: Учеб. пособие для средней школы. М.: АСТ Пресс, 1998.
20. *Угринович Н.Д.* Инфрматика и информационные технологии. Учеб. пособие для 10-11 классов. Углубленный курс. – М.: Лаборатория Базовых Знаний, 2000.
21. *Шафрин Ю.А.* Информационные технологии: В 2 ч. Ч. 1: Основы информатики и информационных технологий. М.: Лаборатория Базовых Знаний, 1999.
22. *Шафрин Ю.А.* Информационные технологии: В 2 ч. Ч. 2: Офисные технологии и информационные системы. М.: Лаборатория Базовых Знаний, 1999.

В.А. Матюхин

АВТОМАТИЗАЦИЯ ПРОВЕРКИ РЕШЕНИЙ ШКОЛЬНИКОВ ПРИ ИЗУЧЕНИИ ПРОГРАММИРОВАНИЯ

Обычно при изучении программирования используются два способа проверки решений школьников. На первых порах хватает «пристального взгляда» – когда учитель, внимательно посмотрев текст программы, выносит вердикт о ее правильности или ошибочности. По мере того как программы становятся все сложнее, этот метод оказывается практически непригодным. Бывает очень сложно заметить какую-нибудь небольшую ошибку (а такие ошибки школьники допускают очень часто). Порой, наоборот, в тексте программы вообще ничего не понятно, но программа работает.

Единственное, что остается – это тестировать программу. Программа запускается учителем 2-3 раза и, если она отработала правильно, считается правильной. Такое поверхностное тестирование, конечно, не позволяет учесть все возможные случаи, и очень часто оказывается, что школьнику было зачтено решение, не имеющее практически ничего общего с правильным.

Однако процесс тестирования можно автоматизировать. Заметим, что речь идет не об очередной «обучающей программе», которая призвана заменить собой учителя, речь идет лишь о перекладывании на компьютер сугубо механической работы учителя по проверке решения. При этом учитель высвобождает достаточно много времени и может уделять внимание тем ученикам, которым необходима его помощь.

Процесс проверки решения становится гораздо более качественным: если вручную мы можем прогнать программу лишь на 2-3 тестах, то при использовании автоматической проверки число тестов может быть на порядок больше. Как правило, 10-15 тестов хватает, чтобы «покрыть» практически все варианты работы программы, и, особенно, частные случаи. При этом прогон такого количества тестов автоматической системой занимает обычно не более 1-2 минут.

Когда школьник считает, что решил задачу, он с помощью специального программного обеспечения (системы автоматической проверки) отсылает решение на проверку. Решение автоматически компилируется, запускается на наборе заранее подготовленных тестов, которые подаются программе школьника в качестве входных данных, после чего анализируются выданные программой результаты. Если все ответы правильные, то программа считается правильной, о чем сообщается школьнику. Если на каком-то из тестов программа отработала не правильно, то об этом также сообщается школьнику. При этом сообщается причина (ошибка во время исполнения, неверный ответ, превышение предела времени и т.д.). Сообщать ли школьнику тест, на котором произошла ошибка, решает учитель.

Итак, предположим, что на каком-то из тестов программа школьника работает неправильно. В данном случае мы получаем достаточно уникальную учебную ситуацию. Мы знаем о том, что в решении есть ошибка, но совершенно не делаем никаких подсказок по поводу того, где она. Здесь возможны два варианта дальнейших действий учителя.

Первый вариант: ученику сообщается и сам неправильный тест. В этом случае, используя отладчик, школьник может достаточно легко выяснить, из-за чего его программа работает неправильно.

Научить школьников пользоваться отладчиком очень важно. Отладчик для школьника играет роль своего рода «зеркала». Можно сказать, что программа – это изложение мыслей школьника о том, как должна решаться задача. Отладчик позволяет сравнить, что школьник хотел сказать, что он реально сказал, и к чему это все приводит, то есть в некотором смысле

позволяет увидеть слабые места и ошибки в решении. Устраняя ошибки в программе (еще раз подчеркнем, что программа – это изложение мыслей), школьник наводит порядок и в своих мыслях. Таким образом, достигается одна из важнейших целей изучения программирования: школьник учится мыслить строго логически, четко излагать свои мысли.

Второй вариант: неправильный тест школьнику не сообщается. Искать ошибку в этом случае гораздо сложнее. Прежде чем устранять ошибку, школьнику надо найти тест, на котором его программа работает неправильно. Ему приходится анализировать условие задачи на предмет того, не осталось ли неучтенных случаев. Ему приходится придумывать самые разнообразные тесты, запускать свою программу, анализировать выданные ей результаты. В некотором роде происходит перемена ролей: если до этого от школьника требовалось, чтобы его программа заработала правильно, то теперь перед ним стоит задача «подловить» эту программу, так как пока случай неправильной работы не найден, говорить об устранении ошибки невозможно.

При таком подходе школьник учится критически относиться к продуктам собственного труда, анализировать все возможные случаи. Надо отметить, что это очень важный навык не только для программирования, а практически для любой сферы человеческой деятельности.

Одно из технических требований при использовании системы автоматической проверки – задачи должны формулироваться очень строго, с указанием всех возможных ограничений. Также при использовании системы от школьников требуется очень жесткое соблюдение форматов входных и выходных данных (иначе автоматическая проверка невозможна). Впрочем, это скорее одно из достоинств использования системы: школьники привыкают точно следовать требованиям технического задания (это также очень ценный навык, причем не только для программиста).

Однако, чтобы описанная схема работала, система должна пользоваться у школьника неким «авторитетом». Иначе, получив ответ об ошибке от проверяющей системы, школьник может вместо того, чтобы искать ошибку в своей программе, начать говорить, что «у вас ошибка в тестах», «ваша система вообще неправильная» и т.д.

Для того чтобы у школьников сформировался «авторитет» системы, на первых порах ее использования следует в случае ошибки показывать школьнику неверный тест. Особое внимание нужно обратить на правильность тестов – очень обидно бывает искать ошибку в абсолютно правильной программе, которая не проходит все тесты из-за ошибки в тестах. Если такая ситуация повторяется два-три раза, то после этого заставить школьника критически относиться к своей программе и искать в ней ошибку, когда ему приходит «неверный ответ», практически невозможно – школьник будет требовать, чтобы учитель прежде всего проверил тесты.

Однако если у школьников «авторитет» системы сформировался, то момент, когда школьнику приходит сообщение «задача зачтена», становится очень ценным для школьника. Очень часто при ручной проверке школьник долго пишет программу, лишь только ради того, чтобы учитель запустил ее на каких-то простейших случаях, после чего оказывается, что программа уже никому не нужна. При автоматической проверке, во-первых, школьник знает, что его решение действительно качественно проверено, и раз оно зачтено, то задача решена верно. Во-вторых, теперь программа пишется ради того, чтобы отправить ее на проверку (разочарования, что «вот я написал, а оно никому не нужно», не возникает – программа ушла на проверку и там где-то хранится).

Таким образом, использование системы автоматической проверки существенно изменяет весь учебный процесс. Фактически, в учебном процессе помимо учителя и ученика появляется третий субъект – «проверяющая система».

Использование систем автоматической проверки при проведении учебно-тренировочных сборов по информатике, при проведении занятий в Кировской летней компьютерной школе и в СУНЦ МГУ дало однозначно положительные результаты. Можно сказать, что при использовании такой системы преподавание идет на качественно ином уровне.

Однако выявились и некоторые проблемы, которые существуют на сегодняшний день.

Подготовка тестов – достаточно сложный процесс, требующий очень высокой квалификации, особенно если вспомнить требование, что в тестах ни в коем случае не должно быть ошибок.

Впрочем, однажды подготовленные тесты могут использоваться сколько угодно раз. Поэтому учитель вообще может не уметь делать тесты, а использовать чужие тесты. Однако для этого должна быть накоплена достаточно большая база задач с тестами. Фактически, накопление такой базы задач эквивалентно разработке курса информатики и является очень непростой задачей.

К сожалению, на сегодняшний день автоматические системы используются в основном для проведения олимпиад, что накладывает на них свои особенности. Необходима специальная система, которая была бы ориентирована именно на использование в учебном процессе.

Задачи, к которым уже сейчас существуют подготовленные тесты, также в основном олимпиадного характера. Однако основная проблема заключается даже не в этом.

Сегодня существует достаточно много различных систем, не совместимых между собой. Тесты, подготовленные для одной из систем, не могут быть использованы в другой. Таким образом, есть огромная необходимость в разработке единого формата обмена задачами, который поддерживался бы всеми системами (или легко конвертировался бы специальными конверторами в специфический формат системы). Это позволит очень сильно упростить процесс обмена задачами и ускорит накопление базы задач.

Требования, которым должна удовлетворять система автоматической проверки, ориентированная на использование в учебном процессе, следующие. Она должна легко устанавливаться и настраиваться, быть достаточно устойчивой, в том числе и к «взломам» со стороны школьников. Процесс обмена задачами должен быть максимально простым (приносится один файл, содержащий все тесты к задаче и другую необходимую информацию, и системе дается команда импортировать его – все технические детали этого процесса должны быть скрыты от пользователя). Система должна поддерживать разные режимы работы в случае неправильного решения – сообщать или не сообщать неверный тест, сообщать его после определенного числа неправильных попыток и т.д.

К сожалению, ни одна из существующих на сегодняшний день систем не удовлетворяет всем этим требованиям одновременно.

Поэтому была создана группа, которая ставит своей целью собрать и проанализировать существующие на сегодняшний день форматы представления тестов в системах автоматической проверки и на их основе выработать стандарт обмена задачами, который бы был максимально универсальным, при этом достаточно удобным в использовании. Следующим шагом работы станет разработка системы автоматической проверки, которая бы удовлетворяла всем описанным выше требованиям. Параллельно будет вестись создание базы задач.

Дополнительную информацию можно найти на contest.cmc.msu.ru/standard.

ИНФОРМАТИКА И ЯЗЫК

Е.Н. Колодкина

ЭКСПЕРИМЕНТАЛЬНОЕ ПАРАМЕТРИЧЕСКОЕ ИССЛЕДОВАНИЕ КОНКРЕТНОСТИ И ОБРАЗНОСТИ В СТРУКТУРЕ ЗНАЧЕНИЯ СЛОВА

На рубеже XX и XXI вв. в центре современных лингвистических исследований вновь находится категория значения. Несмотря на многовековую историю вопроса (начиная с античности до современного пропозиционального подхода), не существует единой концепции значения, способной однозначно и непротиворечиво объяснить все факты языка. Более того, лингвисты, принадлежащие к различным лингвистическим школам традиционного логико-рационалистического подхода, употребляя в своих работах термин «значение», часто вкладывают в него различный, иногда взаимоисключающий смысл.

Психолингвистика, возникшая как реакция на недостаточность системно-структурного подхода, ставящая одной из своих основных задач создание интегративной теории значения, находится лишь на подступах к решению этой сложной задачи. Тем не менее представляется возможным, вслед за А.А. Залевской [1], выделить пять подходов к значению, разрабатываемых в психолингвистике и когнитивной науке в целом: ассоциативный, параметрический, признаковый, прототипный и ситуационный.

Параметрический подход выделяется среди других подходов к значению на основании того, что значение слова может быть разложено на ряд составляющих, степень выраженности которых поддается количественному измерению. Впервые значение «измерил» Ч. Осгуд, разработавший со своими сотрудниками методику семантического дифференциала [2]. Ч.Осгуд пришел к выводу, что метод семантического дифференциала измеряет коннотативное значение слова, которое может быть описано через локацию в рамках некоторого пространства, характеризуемого тремя независимыми параметрами: оценки, силы и активности. За работой Ч. Осгуда последовал целый ряд исследований различных параметров значения и их взаимодействия (см.: 1, 106-109).

Одним из направлений параметрического подхода к значению является исследование психологической структуры значения через шкалирование носителями языка тех или иных параметров. Методика шкалирования широко применяется в психологии, а результатами являются так называемые «семантические нормы» (см., напр.: 3, 4).

Среди отечественных психологов шкалирование впервые применил В.Ф. Петренко [5]. Метод семантического дифференциала использовался В.Ф. Петренко в разработке психосемантического подхода к исследованию сознания и личности, в рамках которого рассматриваются различные формы существования значений в индивидуальном сознании [6, 7].

Вслед за В.Ф. Петренко семантическое шкалирование было использовано в работах Е.Ю. Мягковой и Е.Н. Колодкиной, изучавших психологическую структуру слова в рамках психолингвистической концепции значения, разрабатываемой школой А.А. Залевской. Е.Ю. Мягкова осуществила анализ параметра эмоциональности значения слова в лексиконе индивида [8], который позднее был положен в основу психолингвистической концепции эмоциональной нагрузки слова [9].

Автор данной статьи описала параметры конкретности, образности и эмоциональности и их взаимодействие в психологической структуре 215 русских существительных [10]. Представляется интересным возвратиться к результатам данного исследования, а именно к анализу количественных показателей конкретности и образности и их взаимодействия в связи с новым поворотом дискуссии взаимодействия слова и образа и учетом достижений современных когнитивных исследований.

Так, С.А. Чугунова, осуществляя исследование ментального образа как медиатора понимания текста, предпринимает попытку объяснить механизмы взаимодействия слова и образа, увязывая решение этого вопроса с проблемой образности слова и его конкретности [11]. С.А. Чугунова, проанализировав существующие в современной когнитивной психологии концепции, так или иначе затрагивающие феномен конкретности (теорию двойного кодирования, гипотезу контекстуальной вариативности, гипотезу связанности, модель переработки эмплицитных и эксплицитных репрезентаций, гипотезу перцептивных ассоциатов), пришла к выводу, что все они нуждаются в тщательной проверке. Тем не менее, опираясь на концепцию единой интегративной сущности единиц памяти и интерпретируя результаты своего собственного исследования, С.А. Чугунова понимает образность как способность слова вызывать в сознании и подсознании тот или иной мысленный образ с доминированием различных его составляющих, в отличие от конкретности как одного из проявлений образности, когда слово инициирует в сознании реципиента мысленный образ предмета, за которым в социуме закреплен тот или иной знак.

Мы не можем полностью согласиться с трактовкой конкретности как проявления образности, предложенной С.А. Чугуновой. Мы считаем, что конкретность и образность в структуре значения слова в индивидуальном лексиконе взаимосвязанные, но все же различные параметры.

В психологии и лингвистике имеется многочисленная литература как по конкретности, так и по абстрактности. В нашем экспериментальном исследовании мы исходили из традиционной трактовки конкретного как чувственного, наглядного в противовес абстрактному как понятийному, мысленному. Как известно, категории конкретного и абстрактного входят в число основных лингвистических понятийных координат, в системе которых описывается устройство языка. Первую попытку составить четкую классификацию имен связывают с выделенными еще Аристотелем [12] двумя видами субстанции – телесной и бестелесной. В лингвистике подход, согласно которому имена, обозначающие предметы непосредственно воспринимаемой действительности, определяются как конкретные, а имена, обозначающие названия действий, состояний, качеств в отрыве от носителей и производителей, как абстрактные, принято приписывать Г. Паулю [13]. Такой подход осуществляется, в частности, в разграничении конкретных и абстрактных существительных как слов, обозначающих качественно различные объекты: слова, обозначающие чувственно воспринимаемые объекты, конкретны, недоступные чувственному восприятию – абстрактны. Несмотря на то, что в лингвистике и психологии существует обширная литература, где указывается, какое слово считать конкретным, а какое абстрактным, при отнесении многих лексико-семантических группировок слов к конкретной или абстрактной лексике возникают серьезные проблемы. В ходе нашего экспериментального исследования мы получили показатели конкретности 215 существительных, которые разделили на группы по степени выраженности конкретности.

Образность мы понимали как способность слова вызывать у индивида яркий или смутный ментальный образ. Такая трактовка образности полностью соответствует междисциплинарному пониманию образа как предмета в отраженном виде. В современной психологии образ принято рассматривать как сложный многоуровневый феномен концептуально-процедуральной природы (см.: 11). В лингвистике принято выделять образный компонент в значении слова. При этом не существует единства мнений о месте образного компонента в структуре значения слова (см.: 14). Проведя анализ немногочисленных существующих в лингвистике классификаций образности, мы пришли к выводу, что принципы выделения образных существительных, разработанные в лингвистике, в большинстве случаев не совпадают с принципом организации внутреннего лексикона человека по параметру образности.

Отметим, что нами, вслед за A. Paivio [3], была выявлена высокая корреляционная зависимость между параметрами конкретности и образности. Вычисленный нами коэффициент ранговой корреляции по Спирмену r=91 показывает, что большинству исследуемых слов были приписаны одни и те же ранги. В то же время для некоторых групп слов показатели образности значительно превысили показатели конкретности. В первую группу вошли названия эмоций: ВЕРНОСТЬ, ИЗУМЛЕНИЕ, ПЕЧАЛЬ, СТРАСТЬ, УДИВЛЕНИЕ. Вторая группа представлена названиями вымышленных существ: ДЕМОН, ПРИВИДЕНИЕ. В третью группу вошли слова, вызывающие сильные эмоции: АВАРИЯ, ЗИМА, КРИК, ПАНИКА, ТЕМНОТА. Для ряда существительных, а именно слов-терминов, напротив, показатели конкретности оказались выше образности: ИЕРАРХИЯ. Таким образом, взаимосвязь между конкретностью и образностью существует далеко не для всех групп слов в лексиконе человека. Анализ количественных показателей свидетельствует, что конкретность и образность отражают различные психолингвистические феномены, при этом каждый занимает свое особое место в лексиконе индивида. Сошлемся также на выводы В.Ф. Петренко о том, что за понятиями «конкретность», «абстрактность» и «образность» скрываются разные для субъекта психические реальности [7]. Отметим, что характер взаимосвязи конкретности и образности во внутреннем лексиконе и механизмы, лежащие в основе этого взаимодействия, еще предстоит изучить.

Литература

1. *Залевская А.А.* Введение в психолингвистику. М., 1991.
2. *Osgood E.E., Suci G.J., Tannenbaum P.H.* The Measurement of meaning. Urbane: University of Illinois Press, 1957.
3. *Paivio A., Guille J.C., Madigan S.A.* Concretness, imagery and meaningfulness values for 935 nouns // Journal of Experimental Psychology. Monograph Supplement, 1968. Vol. 76, No.1. Part 2. P. 1-25.
4. *Toglia M.P., Battig N.F.* Handbook of semantic wordnorms. Hillsdale, New York, Lawrence Erlbaum, 1978.
5. *Петренко В.Ф., Нистратов А.А.* Коэффициенты образности, конкретности и ассоциативной значимости для 8 русских существительных // Общение, текст, высказывание. М.,1981. С. 5-17.
6. *Петренко В.Ф.* Психосемантика сознания. М.,1988.
7. *Петренко В.Ф.* Основы психосемантики. Смоленск, 1997.
8. *Мягкова Е.Ю.* Психолингвистическое исследование эмоциональной нагрузки слова: Автореф. дис. ... канд. филол.наук. М.,1986.
9. *Мягкова Е.Ю.* Эмоциональная нагрузка слова: опыт психолингвистического исследования. Воронеж, 1990.
10. *Колодкина Е.Н.* Конкретность, образность и эмоциональность 215 русских существительных // История русского литературного языка и стилистика. Калинин, 1985. С. 40-57.
11. *Чугунова С.А.* Образ ситуации как медиатор процессов понимания художественного текста. Автореф. дис. ... канд.филол.наук. Тверь, 2001.
12. *Аристотель.* Категории. М.,1939.
13. *Пауль Г.* Принципы истории языка. М.,1960.
14. *Колодкина Е.Н.* О специфике психолингвистической трактовки параметров конкретности, образности и эмоциональности значения существительных// Психолингвистические проблемы семантики и понимания текста. Калинин, 1986. С. 70-81.

А.С. Вахрушев

ОСОБЕННОСТИ ПРЕПОДАВАНИЯ ИНТЕГРИРОВАННЫХ ДИСЦИПЛИН ПО ИНФОРМАТИКЕ НА АНГЛИЙСКОМ ЯЗЫКЕ

Одной из специальностей, по которым ведется подготовка специалистов в Вятском государственном педагогическом университете, является «учитель информатики и учитель английского языка». К старшим курсам студенты имеют фундаментальную подготовку по информатике и достаточно свободно владеют английским языком. В целях интеграции информационных и языковых дисциплин на факультете информатики предусмотрен ряд спецкурсов на английском языке. Рассмотрим особенности преподавания на примере курса «Углубленное изучение программного обеспечения Microsoft для операционной системы Windows». Помимо овладения техническими терминами одной из дополнительных возможностей курса была пропедевтика подготовки к сдаче экзамена на степень MCP/MCSE/MCT(Microsoft Certified Professional/Microsoft Certified System Engineer/ Microsoft Certified Treiner). Так, один из центров приема экзаменов находится в Московском техническом университете им. Баумана. Курс состоял из 7 лекций и такого же количества практических занятий (лекции и практика проходили через неделю).

Первой особенностью курса был выбор разделов с целью перекрытия огромного фактологического материала. Мы остановились на следующих темах:

1. Fundamentals of Windows 98.
2. Installation and Configuration.
3. Network Components.
4. Access permissions in Microsoft Enviroments.
5. The Registry, User Enviroments.
6. Installing and Configuring Devices.
7. Monitoring, Tuning and Troubleshooting.

Большое внимание было уделено подбору вопросов внутри каждой темы (Приложение № 1).

Каждое практическое занятие строилось по одному и тому же плану: устный опрос по теме и решение задач (Приложение № 2) как на компьютере, так и в устной. Как правило, все задачи содержали множественный ответ (Multi-Choice Answer). Если задачи решались не на компьютере, то студенты были обязаны давать развернутые пояснения.

Отличные и хорошие ответы студентов на зачете и результаты компьютерного теста (80-85%) говорят об овладении материалом курса и возможности сдачи сертификационных экзаменов фирмы Microsoft.

Литература

1. *Колесников А., Панько В.* Microsoft office для Windows. Киев: BHV, 1996.

2. *Шапошников И.* Web-сайт своими руками. Киев: BHV, 2000.

3.*Birnes W.J.* Microcomputer Application Handbook. New York: McGraw Hill Publishing Company, 1999.

4. Glenn, Weadock MCSE Windows 98 IDG Books, 2000

5.*Mueller S.* Upgrading and Repairing PCs QUE Corporation, 1993.

Приложение № 1. Список вопросов по темам

Members of the Windows family tree (design, goals, features)

Architecture of Windows 95/98

What sets of Windows 98 differ from Windows 95

Installing Windows 98

Automating setup process

Performing a new install

Performing an upgrade

Dual booting Windows

Uninstalling Windows 98

Installing and configuring network adaptors

Installing and configuring network protocols

Configuring Client for Microsoft Networks

Configuring browse mastering

Installing and configuring Service for NDS

Installing and configuring modems, including multilink setups

Installing and configuring printers

Installing and configuring displays

Installing and configuring Microsoft Backup

Configuring hard disks with partitioning, compression and FAT32 conversion

Installing and configuring File and Printer Sharing

Creating and sharing folder resources

Creating and sharing network printer resources

Assinging access permissions for shared folders with passwords

Assinging access permissions for shared folders with user permissions

The Registry's files and major branches

Backing up and restoring the Registry

Checking for corrupt Registrly files by using SCANREG and SCANREGW

Configuring remote Registry administration on a network

Configuring Personal Web Server

Monitoring system performance using Net Watcher, System Monitor and Resource Meter

Tuning and optimizing a Windows 98 PC

Optimizing the hard disk by using Disk Defragmenter And Scandisk

Checking for corrupt files by using System File Checker

Build-in Troubleshooters

Приложение № 2. Задачи

The FAT32 file system allows partitions up to 2_____ in size.

Large cluster sizes in FAT16 can mean wasted hard disk space.

Long file names are unavailable in_____mode and _____mode.

In user-level access control, Windows 98 gets the list of users from a_____.

The FAT32 file system is readable by the following operating systems: A Windows NT4.0

B Windows 95 OSR1

C Windows 95 OSR2

D Windows 98

E MS-DOS 5.0 and above.

You create two files in the directory C:\My Documents, the first one named «Note to Nina.doc» and the second one named «Note to Alex.doc». What`s the DOS equivalent file name of the second file?

A NOTETOAL.DOC

B NOTE~TO~.DOC

C NOTETO~1.DOC

D NOTETO~2.DOC

You are performing an upgrade of a Windows 95 system, and you want to keep all of your existing settings, You start Windows 95. Insert the Windows 98 upgrade CD, and begin the installation process. What do you have to specify in order to ensure that Windows 98 keeps all your settings intact? A The installation directory must be C:/WINDOWS

B The installation directory must be the same

C Click the «I want to keep existing settings and components» radio button in the Upgrade Windows dialog box.

D Do nothing; Windows 98 will migrate settings Automatically.

You're preparing to upgrade a Windows 95 to Windows 98. Name three things that happen during an upgrade from Windows 95's MS-DOS Mode that do not happen during an upgrade from Windows 95's graphical user interface.

A: Setup performs hardware detection for Plug and Play hardware

B: Setup performs hardware detection for legacy hardware

C: Setup prompts you to choose a Typical, Portable, Custom or Compact installation type

D: Setup prompts you to specify an installation directory.

Е.В. Кипрская

ЭКСПЕРИМЕНТАЛЬНОЕ ИССЛЕДОВАНИЕ СТРАТЕГИЙ ИДЕНТИФИКАЦИИ РУССКОЯЗЫЧНЫХ ЭВФЕМИЗМОВ

Эвфемизмы (далее – Э) – это «эмоционально-нейтральные слова или выражения, употребляемые вместо синонимичных им слов или выражений, представляющихся говорящему неприличными, грубыми или нетактичными. Например: "пожилой" вместо "старый", "уклониться от истины" вместо "соврать"» [3, 121].

Э характеризуются высокой степенью подвижности. Ими могут заменяться также табуизированные названия, архаичные («хозяин тайги» вместо «медведь», «шут с ним» вместо «черт с ним»).

Под Э понимаются также «окказиональные индивидуально-контекстные замены одних слов другими с целью искажения или маскировки подлинной сущности обозначаемого» [3,121].

Э противопоставляется дисфемизм (далее – Д) – это «замена эмоционального и стилистически нейтрального слова более грубым, пренебрежительным и т. п., например "загреметь" вместо "упасть", "рассопливиться" вместо "заплакать", "сыграть в ящик" вместо "умереть", "осточертеть" вместо "надоесть"» [3, 121].

Процессы эвфеминизации изучаются в лексикологии и психолингвистике. Проблемы, связанные с восприятием и переработкой Э и Д, являются актуальными в наши дни и должны быть изучены в рамках психолингвистической концепции внутреннего лексикона, разрабатываемой А.А. Залевской и ее школой. По публикациям последних двух десятилетий, среди наиболее актуальных направлений исследований лексики А.А. Залевская выделяет «психологическую структуру значения слова, т.е. попытки теоретически и экспериментально установить, что лежит за словом в индивидуальном сознании и подсознании, какие параметры значения актуальны для пользователя словом и т.д. и т.п.» [5, 6].

В связи с этим нам представляется интересным выявление различных стратегий при переработке русскоязычных Э и Д. Для этой цели был проведен ассоциативный эксперимент, который является «наиболее разработанной техникой психолингвистического анализа семантики» [1, 76].

Данный эксперимент проводился на факультете информатики ВГПУ. Предметом исследования являлись 26 русскоязычных Э и Д, отбор которых был основан на их высокой степени распространенности как в книжно-письменной, так и в устной речи. Главным преимуществом ассоциативного эксперимента является, по мнению В.П. Белянина [1, 81], «его простота, удобство применения, так как он может проводиться с большой группой испытуемых одновременно. Испытуемые работают со значением в "режиме употребления", что позволяет выделять и некоторые неосознанные компоненты значения».

Участниками эксперимента были 24 студента 3-го курса. Всем испытуемым было предложено написать на листочке рядом с каждым Э и Д первое, что придет им в голову. Ограничений по времени не было, но рекомендовалось ответы не обдумывать и не возвращаться к уже пройденному. Всего было затрачено около 25 минут на 26 Э и Д.

Первый обзор показал, что наибольшую трудность для ассоциирования вызвали следующие Э и Д: «канцер» (15 отказов), «находиться в объятиях Морфея» (13), «синий чулок» (5), «оператор очистных соору-

жений» (4), «деструктивный» (3). По одному отказу получили следующие словосочетания: «считать ворон», «второй хлеб», «корабль пустыни», «хозяин тайги». Примечательно, что такие распространенные Э, как «королева полей», «миротворческая операция» и «лицо кавказской национальности», получили по 2 отказа.

Поскольку ассоциативный эксперимент, по мнению В.П. Белянина [1, 80], «показывает наличие в значении слова (а также предмета, обозначаемого словом) психологического компонента, то он дает возможность построить семантическую структуру слова и служит ценным материалом для изучения психологических эквивалентов семантических полей и вскрывает объективно существующие в психике носителя языка семантические связи слов». Так, в Э «педикулез» в сознании испытуемых присутствует и такой психологический момент этого слова, как «ножницы», «дети», «плохо». В лингвистических же словарях он отсутствует.

По данным того же автора [1, 81], «ассоциативное поле у каждого человека свое и по составу наименований, и по силе связей между ними». Однако «центр» ассоциативного поля в целом является достаточно стабильным, и он определяется зависимостью ассоциаций от культурно-исторических традиций народа. Очень интересные данные приводит А.А. Залевская: самые частые ассоциации-существительные к стимулу «хлеб» для русских – «соль», для узбеков – «чай», для французов – «вино», для немцев – «масло».

В ходе эксперимента было выявлено, что актуализация той или иной связи в ответе не носила случайный характер, а зависела от ситуации. Например, на стимул «миротворческая операция» были даны ответы «Палестина», «Югославия», «Косово».

При более детальном анализе данных были построены ассоциативные поля некоторых Э и Д:

• Царь зверей (24 человека): лев – 23, тигр – 1.

• Кончина (24 человека): смерть – 19, похороны – 2, покойник – 2, гроб – 1.

• Хозяин тайги (24 человека): медведь – 15, тигр –3, чукча – 2, волк – 1, олень – 1, Высоцкий – 1, егерь –1.

• Черное золото (24 человека): нефть – 21, богатство – 1, деньги – 1, драгоценности – 1.

• Слабослышащий (24 человека): глухой – 21, тугоухий – 1, слуховой аппарат – 1, глухонемой – 1.

Как уже отмечалось выше, задачей проведенного нами экспериментального исследования было выявление стратегий идентификации русскоязычных Э и Д. После количественной обработки бланков были составлены списки ассоциаций по каждому Э и Д. Анализ полученных ассоциаций позволил выявить определенные идентификационные стратегии, которые были реализованы ими. Следует отметить, что в работах [5; 6] выделяются следующие идентификационные стратегии: мотивирующая, прямая дефиниция предъявленного слова, стратегия категоризации, конкретизация через синоним/симиляр, иллюстрация примером, идентификация через метафоризацию, опознание значения слова через объект обозначаемого словом действия, опознание значения слова через субъекта обозначаемого словом действия, приписывание некоторого признака действию, обозначаемому предъявленным словом, словообразовательная стратегия, опознание по сходству звукобуквенного состава, стратегия отказа от дефиниции.

1. *Прямая дефиниция* оказалась наиболее актуальной стратегией для нашего материала, она реализуется в большинстве рассмотренных Э и Д: «вооруженный конфликт» – война (14); «второй хлеб» – картофель (19); «царь зверей» – лев (23), «корабль пустыни» – верблюд (19); «королева полей» – кукуруза (14).

2. *Стратегия иллюстрации примером* оказалась не слишком актуальной. Фактически выявлено только два случая ее реализации: «миротворческая операция» – операция войск ООН в Косово; «шут с ним» – это когда все по фигу.

3. Как считает Н.С. Шумилина[2; 124], «словообразовательная стратегия применительно к фразеологии определена как *структурная* (структурно-грамматическая). При реализации этой стратегии дефиниция строится по модели исходной фразеологической единицы». Мы не отходим в своем эксперименте от данного положения и выявили следующие Э и Д, ассоциация которых строится по исходной грамматической модели: «миротворческая операция» (прил. + сущ.) – голубые береты (2); «считать ворон» (гл. + сущ.) – думать о своем (2); «шут с ним» (сущ. + пр. с мест.) – черт с ним (80); «сыграть в ящик» (гл. + сущ. с пр.) – сыграть в футбол (1).

4. *Стратегия опознания по сходству звукобуквенного состава* у Э и Д наблюдается достаточно редко. В большинстве случаев она реализовывалась для тех Э и Д, значение которых было непонятным для испытуемых: «синий чулок» – Пеппи Длинныйчулок (5); «деструктивный» – конструктивный (1), «загреметь» – гром (2).

5. *Стратегию отказа от дефиниции* можно считать очень актуальной для Э и Д. Общее число отказов уже указывалось.

Анализ данных ассоциативного эксперимента дает нам основания для следующих выводов:

1. Как и следовало ожидать, опознание Э и Д в подавляющем большинстве случаев осуществлялось через прямую дефиницию. Подобная стратегия выявлена в 59 % ответов.

2. На структурном уровне было выявлено влияние частеречной отнесенности слова-стимула на частеречную отнесенность слова-реакции. Во многих реакциях принадлежность Э или его ключевого слова (в словосочетании) определяла, будет ли реакция относиться к классу существительных, глаголов, прилагательных и т.д. Например: «кончина» – гибель, смерть, похороны, гроб, покойник; «рассопливиться» – заболеть, реветь; «шут с ним» – хрен с ним.

3. В ходе анализа было выявлено, что стратегия отказа от дефиниции использовалась при реакции на малоизвестные и малоупотребительные Э и Д. Такая тенденция наблюдалась в 37 % случаев.

4. Стратегия метафоризации не была реализована в ходе данного эксперимента, что можно объяснить сложностью «производства» метафор для русскоязычных Э и Д.

Литература

1. *Белянин В.П.* Введение в психолингвистику. М., 1999.

2. *Шумилина О.С.* Экспериментальное исследование стратегий идентификации иноязычных фразеологизмов // Психолингвистические исследования слова и текста. Тверь, 1997. С. 121-128.

3. Лингвистический энциклопедический словарь. М. ,1990.

4. *Залевская А.А.* Некоторые актуальные направления психолингвистического исследования лексики // Проблемы семантики: психолингвистические исследования. Тверь, 1991. С. 5-16.

5. *Родионова Т.Г.* Стратегии идентификации неологизмов глаголов: Дис. ... канд. филол. наук. Тверь, 1994.

6. *Тогоева С.И.* Психолингвистическое исследование стратегий идентификации значения словесного новообразования: Дисс. ... канд. филол. наук. Калинин,1989.

Е.Н. Колодкина, Н.С. Христолюбова

ПСИХОЛИНГВИСТИЧЕСКОЕ ИССЛЕДОВАНИЕ ПАРОНИМИИ

Феномен паронимии интересовал исследователей с древнейших времен. Еще древнегреческие риторики, а позднее римские философы занимались изучением и разработкой словесных фигур, образуемых созвучием, и их влиянием на слушателей. Упоминание о паронимии находим в трудах многих отечественных и зарубежных лингвистов (В.В. Виноградов, А.А. Реформатский, Д.Э. Розенталь, H.W. Fowler). Тем не менее специальные работы по паронимии стали появляться лишь в середине 60-х гг. XX в. (О.В. Вишнякова, С.М. Грабчиков, В.П. Григорьев, Л.Н. Федотова).

Самым известным и широко цитируемым является определение О.С. Ахмановой, согласно которому паронимы – это слова, которые вследствие сходства в звучании и частичного совпадения морфемного состава могут либо ошибочно, либо каламбурно использоваться в речи [1]. Однако современные исследователи, определяя паронимы, указывают на недостаточность критерия звукового сходства и на необходимость его сочетания с семантическим критерием. Так, Л.Н. Федотова определяет паронимы как слова, настолько сходные по звучанию, что при определенных условиях они могут сближаться и по смыслу [2]. Серьезной проблемой является определение места паронимии среди смежных лексико-семантических группировок. В последние два десятилетия появилось много работ, в которых предпринимаются попытки отграничить паронимию от таких смежных явлений как омонимия, полисемия, синонимия и парономазия и тем самым определить ее специфику (см. 2).

В данной публикации мы попытаемся подойти к явлению паронимии с точки зрения носителя языка и определить ее специфику во внутреннем лексиконе индивида. Трактуя идеолексикон как многоярусную систему многократно пересекающихся линейных и иерархических связей, с помощью которых упорядочивается и хранится разносторонняя вербальная и невербальная информация, мы, вслед за А.А. Залевской, определяем идентификацию как процесс соотнесения слова с той информацией, которая стоит за словом в индивидуальном сознании [3, 4]. При этом осуществляется соотнесение слова с другими словами или иными единицами лексикона в рамках тех или иных идентификационных моделей, которые и определяют специфику данного слова или лексико-семантической группировки в лексиконе индивида. Нам представляется, что идентификационные стратегии паронимов смогут определить их место в лексиконе и критерии отграничения от смежных лингвистических явлений.

Анализ существующих типологических классификаций паронимов, в основу которых положены различные принципы, позволил приложить их для выбора и описания экспериментального материала. Отметим, что если в английском языке существуют многочисленные классификации паронимов, построенных на различных основаниях, то в немецком языке паронимы выделяются лишь на основании двух – фонетического и словообразовательного – критериев. В наш экспериментальный список вошли 20 английских паронимических рядов и 20 паронимических пар на немецком языке.

Паронимические ряды были составлены таким образом, чтобы исследуемые слова отражали максимальное количество оснований, положенных в различные классификации паронимов.

В ходе свободного ассоциативного эксперимента студентам V курса факультета иностранных языков, изучающим английский и немецкий языки, были предъявлены для ассоциирования списки паронимических рядов в начале по отдельности, а затем – в паронимических рядах. В ходе двух экспериментов было получено 2144 реакции и 176 отказов. Укажем, что количество отказов было значительно больше (в 1,3 раза на материале английского и в 2 раза на материале немецкого – второго иностранного языка) во втором эксперименте, когда паронимические ряды предъявлялись испытуемым попарно.

На первом этапе анализа полученных ассоциативных полей паронимических рядов слов-стимулов мы выделили две большие группы реакций: автоматизированные и смысловые. Принципом их выделения служит отнесение полученных реакций к глубинному (смысловые стратегии) или поверхностному (ав-

томатизированные стратегии) уровням внутреннего лексикона. Стратегии глубинного уровня отражают процессы всестороннего осмысления значения слова-стимула, определение его места в лексиконе по множеству параметров, в том числе, по большому количеству семантических признаков. Стратегии поверхностного уровня актуализируются не через процесс понимания, т.е. обращение к глубинному ярусу лексикона, а через автоматизированные реакции, не затрагивающие сам процесс понимания, а реализующиеся в готовом виде на поверхностном ярусе лексикона.

При анализе ассоциативных полей паронимических рядов слов-стимулов на материале английского и немецкого языков прослеживается увеличение автоматизированных (формализованных) реакций. Приведем примеры автоматизированных словообразовательных реакций; WOMAN – women, men; PERSONALITY – person; TO DO – doing; CORN – cornflakes (английский язык); ADRESSANT – Adresse; LIEBEN – liebe; ERKENNBUR – unrekennbar (немецкий язык); и фонетических реакций: ALLEY – valley, BREATH – death, ALLUSION – confusion; MITTE – Miete, EXAMINATOR – Terminator. Очевидно, что процесс осмысления паронимической пары идет через констатацию формального различия, подтверждая фонетический и словообразовательный критерии выделения паронимов. Отметим, что в эксперименте на английском языке прослеживается весьма значительное увеличение автоматизированных реакций (более чем на 12 %), тогда как на немецком материале увеличение автоматизированных реакций было менее значительным (на 4 %). Мы объясняем это тем, что паронимические ряды на немецком языке были образованы по словообразовательному принципу, тогда как в английском языке этот принцип был лишь одним из критериев подбора паронимов. Таким образом, можно предположить, что уже при первом предъявлении слова с аффиксом шел его словообразовательный анализ. Такой же анализ шел и при повторном предъявлении слов-стимулов попарно.

Одновременно с некоторым увеличением числа автоматизированных реакций во время второго эксперимента, трактуемого нами как обращение испытуемых к верхнему ярусу лексикона при понимании слова, мы отмечаем и противоположную тенденцию. Полагаем, что увеличение количества отказов во втором эксперименте, когда стимулы предъявлялись в паронимических рядах, можно объяснить тем, что при сопоставлении слов-паронимов происходит их осмысление на глубинном уровне, а это, в свою очередь, приводит к уменьшению количества ассоциатов за счет уменьшения легко получаемых реакций верхнего уровня. Возможно, постулат о двух уровнях идентификации слова нуждается в пересмотре, а внутренний лексикон является еще более сложной системой, чем мы представляли, механизмы устройства и функционирования которой требуют дальнейшего исследования.

Среди многочисленных стратегий идентификации на глубинном уровне особое место занимают идентификационные модели, которые мы назвали паронимическими: через синоним; через антоним; через другое значение многозначного слова; посредством парономазии. Приведем примеры реакций по каждой из стратегий на английском и немецком языках. Следующие примеры могут проиллюстрировать идентификацию

• через синоним: ALLY – friend, DOER – maker, WOMEN – female; SPITZHEIT – Witz, FORSHE – Mut, SPUTZE – Gipfel;

• через антоним: TO CLOSE – to open, WOMEN – man, ALLY – enemy, KÜRZE – Länge, ADRESSANT – Empfänger;

• через другое значение полисемантичного слова: TO AFFECT – (воздействовать) influence, love, to pretend, ALLUSION – (намек) hint, sly, slight, (ссылка) Bible; WONDER – (удивление) an open mouth, surprise, (чудо) magician; SPITZE – (кончик, вершина) Geburge, Berg, Broken, (глава) der König, der Häupling; посредством парономазии: EFFECT – defect, perfect, ILLUSION – collision, SCORN – mourn, horn, born, BREATH – death; WÜRZE – Schürze, WELT – Geld.

Таким образом, опознание слова-паронима через синоним, антоним, другое значение полисемантичного слова или через явление парономазии доказывает существование явления паронимии во внутреннем лексиконе, т.е. ее психолингвистическую реальность и особое, специфическое место среди других лексико-семантических группировок языка, а наиболее частотные модели идентификации паронимов через фонетическую и словообразовательную стратегии подтверждают лингвистические критерии выделения слов-паронимов.

Литература

1. *Ахманова О.С.* Словарь лингвистических терминов. М., 1966.

2. *Федотова Л.Н.* Паронимия в английском языке. Тверь, 1994.

3. *Залевская А.А.* Психолингвистические проблемы семантики слова. Калинин,1982.

4. *Залевская А.А.* Введение в психолингвистику. М., 1999.

А.А. Свицова

ЭПИТЕТ КАК ЭМОЦИОНАЛЬНО-ЭКСПРЕССИВНОЕ СРЕДСТВО В РОМАНЕ В.С. МОЭМА «ЛУНА И ГРОШ»

Творчество писателя – это его способ общения с читателем, средство передачи эстетической информации и его видения мира.

Всякое произведение художественной литературы призвано воздействовать на интеллект и эмоции читателя. Этого эффекта можно добиться употреблением самых разнообразных выразительных средств и стилистических приемов.

Как известно, одним из самых широкоупотребительных стилистических средств является эпитет. В данной статье нам хотелось рассмотреть эпитет с точки зрения его семантики и его роли в создании художественных образов в произведении С. Моэма «Луна и грош».

Эпитет не только называет объективный признак явления, предмета, персонажа, но и выражает субъективное отношение к нему автора, привнося этим в характеристику оценочно-аффективные момент, позволяющий выявить идейно эстетическую позицию автора, создать определенный образ.

Задача эпитета – выделить характерную черту, сосредоточить внимание на определенном признаке, выдвинув его на первый план (3, 36).

Анализ романа С. Моэма «Луна и грош» позволил сделать вывод, что эпитет является основным стилистическим приемом, используемым автором в данном произведении.

Портретная характеристика изобилует субъективно-оценочными прилагательными и наречиями. Описание внешности, социальной среды, внутреннего мира героев, их морально-эстетических вкусов в значительной степени строится с помощью определений и эпитетов. Если сжать характеристику до логического предела, то ее можно представить как сводку-перечень эпитетов, в результате чего возникают яркие образы. Мы свободно представляем персонаж даже при неполном выборочном перечислении некоторых эпитетов.

Важно отметить, что индивидуальность, неповторимость портрета или ощущения создается с помощью обычных, почти постоянных и часто повторяемых слов. Пристрастие С. Моэма к неоднократному повторению одних и тех же эпитетов имеет определенную цель – создать представление о том или другом персонаже, о его физическом и духовном облике.

Отдельные эпитеты приобретают у С. Моэма лейтмотивный характер обозначая собой константные параметры личности.

Так, например, в портрете Дирка Стрев наиболее часто встречаются эпитеты ***round*** (в различных комбинациях до 9 раз), ***fat, absurd, ridiculous***, характеризующие его внешний вид:

He was a fat little man, with short legs, young still... [4, 80] – *Он был маленький, толстый, с короткими ножками* [5, 50];

He remainded you of those fat merchants that Rubens painted [4, 80] – *Он напоминал жизнерадостных толстых торговцев, которых любил рисовать Рубенс* [5, 50].

В описании этого персонажа часто употребляется эпитет ***foolish***, который перекликается с вышеупомянутыми.

...And you despised him because he was foolish [4, 79] – *...И его презирали за дурость* [5, 49];

... and he kept kind foolish eyes fixed on me [4, 111] – *...и он смотрел на меня своими добрыми, глуповатыми глазами* [5, 75].

Возникает образ, более похожий на шута, чем на художника, способного создать серьезные картины.

Если, создавая портрет Дирка, автор дает более полную характеристику его внешнего вида (только по отдельным деталям мы можем судить об ограниченности его внутреннего мира), то, описывая Бланш Стрев, он фокусирует внимание не на внешности, а на тех чертах, которые позднее позволяют раскрыть глубину и сложность ее характера. По тому, как часто автор прибегает к эпитетам ***quiet***, ***silent***, ***calm*** и повтору однокоренных с ними слов, можно нарисовать портрет женщины очень спокойной, молчаливой, кроткой.

As a rule she was so calm [4, 106] – *Обычно она была спокойна и сдержанна* [5, 77];

She was silent, but she had always been silent [4, 127] – *Она молчала, но она и всегда была молчалива* [5, 95].

Главного героя, Чарльза Стрикленда, автор описывает на протяжении длительного отрезка его жизни, отсюда и очень широкий спектр используемых эпитетов: ***good, kind, dull, broad, heavy, large, hefty***.

...in point of fact he was broad & heavy with large hands and feet and he wore his evening clothes clumsily [4, 38] – *...на деле он был широкоплеч, грузен, руки и ноги были больше, и вечерний костюм сидел на нем мешковато* [5, 17];

He was clean shaven and his large face looked uncomfortably naked [4, 38] – *Он был чисто выбрит, и его большое лицо казалось неприятно обнаженным* [5, 17].

Позже мы встречаем в тексте эпитеты, которые свидетельствуют о резкой перемене во внешнем облике героя: ***untidy, ill-kempt, old Nerfolk, unbrushed, unshaved, ragged, torn, deep***.

*He wore an old Nerfolk jacke*t [4, 56] – *На нем был старый поношенный пиджак* [5, 31];

... and his face with the red stubble of the unshaven chin [4, 62] – *...лицо его, с небритой рыжей щетиной на подбородке* [5, 36].

Эти эпитеты и их синонимы повторяются настолько часто, что создается яркий образ этого персонажа. Пристрастие к эпитету, его обильному использованию – это одна из черт индивидуальности манеры письма С. Моэма. Любое логическое определение, даже самое безобразное, может выступать в роли эпитета в определенном контексте.

Каждый эпитет в произведениях С. Моэма выполняет определенную стилистическую функцию и всегда эстетически мотивирован.

Особенность С. Моэма в использовании эпитета заключается в том, что писатель тяготеет к неоднократному повтору одних и тех же эпитетов, и это указывает на то, что цель Моэма – добиться сильного эмоционального эффекта.

В использовании эпитета проявляется непревзойденное достоинство языкового стиля С. Моэма, которое заключается в умении выбрать наиболее точное слово для эмоционально-экспрессивной характеристики героя.

Литература

1. *Арнольд И.В.* Стилистика современного английского языка. Л., 1981.

2. *Гальперин И.П.* Стилистика. М., 1971.

3. *Никитина С.Е., Васильева С.Е.* Экспериментальный системный толковый словарь стилистических терминов. М., 1996.

4. *Maugham S.* The Moon and Sixpence. M., 1969.

5. *Моэм С.* Луна и грош. М., 1983.

А.И. Бардовская

ПСИХОЛИНГВИСТИЧЕСКОЕ ИССЛЕДОВАНИЕ СИНЕСТЕТИЧЕСКИХ ПРИЛАГАТЕЛЬНЫХ СОВРЕМЕННОГО АНГЛИЙСКОГО ЯЗЫКА

Одной из актуальных проблем современного семантического исследования является синестезия, играющая значительную, но малоизученную роль в системе развития значений слова. Синестетический перенос встречается еще в работах Гомера. Современные индоевропейские языки изобилуют примерами синестетического употребления слова, но особенно характерно это для прилагательного [2, 11, 12].

Несмотря на то, что четкого, непротиворечивого определения синестезии до сих пор не существует, в основе всех определений синестезии положено принятое в психологии понимание синестезии как переноса качества одного ощущения на другое, переноса качества одной сенсорной модальности на другое.

В лингвистике предпринимаются первые попытки семантического исследования синестетических переносов, рассматривается тематическое своеобразие прилагательных, обнаруживающих способность к разным типам переноса, например: осязание – слух, зрение – вкус, слух – обоняние и т. д.

В дипломной работе М. Межбурд приводятся результаты исследования семантических полей 30 синестетических прилагательных современного английского языка, в ходе которого были установлены наиболее характерные модели синестетического переноса, определена наполненность каждой из выделенных групп прилагательных.

Нам представилось интересным, функционируют ли модели синестетического переноса, выявленные на основании лингвистического анализа, в индивидуальном сознании носителя языка, а именно носителя русского языка, изучающего английский язык, и выделить специфику их функционирования.

Мы считаем подобный подход актуальным и перспективным, поскольку предмет психолингвистики определяется в сопоставлении, а иногда противопоставлении лингвистического и психолингвистического подходов.

Таким образом, актуальность проблемы состоит в совмещении лингвистического и психолингвистического подходов к проблеме синестезии.

Объектом нашего исследования явилось 21 прилагательное современного английского языка, которые наиболее ярко демонстрируют характерные виды синестетического переноса:

осязание – слух: *soft, smooth, hard, harsh, rough, rude, sharp, warm, cold*;

осязание – зрение: *soft, harsh, sharp, warm, cool, cold, hot*;

осязание – вкус: *soft, harsh, smooth, sharp*;

осязание – обоняние: *soft, warm, cool, harsh, sharp, cold, cool, hot*;

зрение – слух: *bright, clear, dark*;

слух – зрение: *loud, noisy, quiet, still*;

слух – запах: *loud*;

вкус – обоняние: *sweet, bitter, acid*;

вкус – слух: *sweet*;

вкус – зрение: *sweet, acid*.

В ходе ассоциативного эксперимента, проведенного по методике свободного ассоциирования с ограниченным (до 5) количеством реакций, были получены ассоциативные поля каждого из 21 прилагательного экспериментального списка. Таким образом, основным методом исследования является психолингвистический эксперимент. Испытуемыми выступали 11 школьников 11-го класса Вятской гуманитарной гимназии. В ходе эксперимента слова предъявлялись испытуемым устно. Согласно инструкции испытуемые в ходе ассоциирования должны были записать несколько пришедших на ум вербальных реакций. Время ассоциирования ограничивалось.

Результаты исследования представляют ассоциативные поля слов-стимулов. В ходе анализа полученных результатов были использованы количественный метод с элементами статистического анализа и метод внутриязыкового сопоставления результатов. Под ассоциативным полем мы понимаем всю совокупность вербальных реакций, полученных на одно слово-стимул. Ассоциативное поле слова отражает внутреннюю структуру индивидуального лексикона – сложной иерархической структуры многократно пересекающихся полей, выделяемых по множеству признаков разного уровня обобщения и структурной оформленности (от признака понятия до концепта, от фонемы до слова) [3, 4, 5, 6, 7, 8, 9].

Средством доступа к внутреннему лексикону является слово, тот элемент, в котором перекрещиваются все многомерные поля внутреннего лексикона. Слово позволяет определить, что стоит за ним в индивидуальном сознании и подсознании.

Для того, чтобы осознать, понять, т.е. идентифицировать слово, необходимо найти его место во внутреннем лексиконе. Типичные стратегии идентификации, т.е. стратегии, по которым слово наиболее часто осознается, и позволяет нам это осуществить [1, 10, 13, 14, 15, 16, 17].

Общее количество полученных реакций – 668. Количество реакций в ассоциативных полях варьируется от 43 (*sweet*) до 20 (*still*). Наиболее часто повторяющиеся (частотные), стереотипные реакции принадлежат стимулам *warm – weather* (11), *cool girl* (10), *acid – lemon* (11).

Проанализировав модели синестетического переноса, существующие в индивидуальном сознании, мы обнаружили следующие тенденции:

1. Большинство полученных реакций идентифицируют номинативное значение прилагательных. Например, в ассоциативных полях тактильных прилагательных наиболее стереотипными оказались реакции *grass* 6 (*soft*), *surface* 8 (*smooth*), *teeth* 7 (*sharp*), *weather* 11 (*warm*), *sun* 5 (*hot*), у прилагательных, описывающих в своем номинативном значении зрительные ощущения – *light* 6 (*bright*), *sky* 4 (*clear*), *room* 4 (*dark*), слуховые ощущения – *voice* 8 (*loud*), *street* 3 (*noisy*), вкусовые ощущения – *ice-cream* 6 (*sweet*), *chocolate* 8 (*bitter*), *lemon* 11 (*acid*). Исключение составляют стимулы *rude* и *still*, в ассоциативных полях которых подобная модель идентификации не представлена, и стимулы *hard* и *quiet*, в ассоциативных полях которых подобные реакции единичны.

2. Все исследуемые прилагательные, за исключением *smooth*, имеют в своих ассоциативных полях реакции, идентифицирующие собственно метафорический перенос, связанный либо с номинативным значением, либо с каким-либо типом синестетического переноса. Наиболее стереотипными здесь явились реакции *character* 3 (*soft*), *man* 6 (*harsh*), *joke* 3 (*rude*), *pain* 2 (*sharp*), *heart* 2 (*warm*), *temperament* 2 (*hot*), *mind* 2 (*bright*), *soul* 2 (*clear*), *fear* 4 (*dark*), *scorn* 3 (*bitter*), *mood* 2 (*acid*).

3. Часто происходит переход различных ощущений, описанных прилагательными, в сферу слухового восприятия, на что указывает реакция *voice* (*soft, harsh, rough, rude, clear, sweet*). Причем наиболее частотна она на стимулы *harsh, soft* – 5, менее частотна на *rough, rude* – 3, *sweet* – 2, единична на стимул *clear*.

Реже происходит перенос в сферу зрительного восприятия, на что указывают реакции *eyes* (*soft*), *face* (*sweet*), *face* 5 (*acid*), *work* (*rude*), и вкусового восприятия: *bread, meal* (*harsh*), *meat* (*rough*), *pepper* (*hot*). Наиболее редки реакции, идентифицирующие перенос в сферу обоняния: *breathing, air* (*hard*).

4. Можно предположить, что единичные реакции идентифицируют совокупность различных восприятий, указывающих на то, что ведущим в индивидуальном сознании может оказаться тип восприятия, отличный от отраженного в традиционной лингвистической классификации. Например, реакция *cream* на стимул *cold* может идентифицировать как тактильные (температурные), так и зрительные и вкусовые ощущения.

В самом общем виде исследованные нами прилагательные можно разделить на 3 группы по потенциалу к образованию переносных значений, в том числе синестетических и собственно метафорических:

1. Прилагательные с наибольшим потенциалом к образованию переносных значений:

soft (осязание – слух, осязание – зрение, собственно метафорический перенос);

hard (осязание – слух, осязание – обоняние, собственно метафорический перенос);

harsh, rough (осязание – вкус, осязание – слух, собственно метафорический перенос);

sweet (вкус – слух, вкус – зрение, собственно метафорический перенос);

rude (осязание – слух, осязание – зрение, собственно метафорический перенос).

2. Прилагательные с небольшим потенциалом к образованию переносных значений:

hot (осязание – вкус);

bright, clear (зрение – слух, собственно метафорический перенос);

acid (вкус – зрение, собственно метафорический перенос);

sharp (осязание – слух, собственно метафорический перенос).

3. Прилагательные с наименьшим потенциалом к образованию переносных значений:

warm, cold, cool, dark, loud, noisy, quiet, bitter (собственно метафорический перенос), *smooth* (номинативное значение).

При этом следует оговориться, что синестетические прилагательные, проявляющие тенденцию лишь к собственно метафорическому переносу, имеют в своих ассоциативных полях реакции, разнообразно идентифицирующие их значение, т.е. их нельзя назвать «ущербными» по сравнению с прилагательными, имеющими потенциал к синестетическому переносу.

Литература

1. *Барсук Л.В.* Психолингвистическое исследование особенностей идентификации значений широкозначных слов (на материале существительных). Автореф. ... канд. филол. наук. Саратов, 1991.

2. *Горелов И.Н., Седов Н.Ф.* Основы психолингвистики. М., 1997.

3. *Залевская А.А.* Проблемы организации внутреннего лексикона человека. Калинин, 1977.

4. *Залевская А.А.* Проблемы психолингвистики. Калинин, 1983.

5. *Залевская А.А.* Слово в лексиконе человека: психолингвистическое исследование. Воронеж, 1990.

6. *Залевская А.А.* Индивидуальное значение: специфика и принципы функционирования. Тверь, 1992.

7. *Зимняя И.А.* Психолингвистические аспекты обучения говорению на иностранном языке. М., 1985.

8. *Зимняя И.А.* Психология обучения неродному языку. М., 1989.

9. *Зимняя И.А.* Психология обучения иностранному языку в школе. М., 1991.

10. *Лачина И.С.* Особенности идентификации прилагательных: Автореф. дис. ... канд. филол. наук. Тверь, 1993.
11. *Лурия А.Р.* Основные проблемы нейролингвистики. М., 1975.
12. *Лурия А.Р.* Язык и сознание. М., 1979.
13. *Медведева И.Я.* Опыт психолингвистического исследования антонимии // Психолингвистические исследования в области лексики и фонетики. Калинин, 1981.
14. *Мягкова Е.Ю.* Структурные опоры при обучении пониманию и переводу иностранного текста // Психолингвистические исследования: слово и текст. Тверь, 1995.
15. *Рафикова Н.В.* Динамика ядра и периферии семантического поля текста: Автореф. дис. ... канд. филол. наук. Тверь, 1994.
16. *Родионова. Т.Г.* Стратегии идентификации неологизмов-глаголов: Автореф. дис. ... канд. филол. наук. Тверь, 1994.
17. *Тогоева С.И.* Психолингвистическое исследование стратегии идентификации значения словесного новообразования: Автореф. дис. ... канд. филол. наук. Саратов, 1991.

Г.К. Шейдуллина

СТИМУЛИРОВАНИЕ РЕЧЕВОЙ ДЕЯТЕЛЬНОСТИ СТУДЕНТОВ, РАЗВИТИЕ ПРОФЕССИОНАЛЬНО-ПЕДАГОГИЧЕСКИХ НАВЫКОВ И УМЕНИЙ (на примере темы «Traits of character. Difficult children»)

Тема «Traits of character. Difficult Children», изучаемая на IV курсе, является продолжением тематики, начатой на II и III курсах, представленной в темах «Воспитание в семье и школе», «Чувства и эмоции», «Обсуждение характеров людей». Необходимо отметить, что студенты IV курса уже отличаются высоким уровнем владения языком, имеют богатый словарный запас, владеют активными формами речевой деятельности, такими, как ролевая игра, диспут, дискуссия, подходят к такому важному виду деятельности, как стилистическая интерпретация текста, выявление его художественных особенностей, оценка главных персонажей.

Достоинством темы «Traits of character. Difficult Children» является то, что она позволяет нам органично использовать знания, получаемые по педагогике и психологии, для РД, предоставляя богатый материал для обсуждения проблемы воспитания детей, проблемы активного участия взрослых в проблемах детей и не только. Очень важно у студента педагогического вуза формировать представление о самом себе как о субъекте, развить способность к рефлексии и саморефлексии, способствовать его взаимодействию с окружающим миром. На наш взгляд, именно эта тема, рассмотренная в необходимом психолого-педагогическом ключе, позволяет «сформировать гуманитарный способ мышления, способствует позитивной интеграции личности в окружающий мир, ее социализации».

Для реализации этой задачи автор данной статьи предлагает использовать целый ряд дополнительных заданий, тестов и текстов, которые также отличает информативность и эмоциональная привлекательность.

Задание 1

После ознакомления с лексикой по теме предлагается обдумать следующий вопрос: «What sort of person are you?» и результаты размышлений занести в таблицу: «Fill in the table. Try to describe yourself honestly».

My Personality

Positive traits of	Neutral characteristics	Negative characteristics
Positive traits of	Neutral characteristics	Negative characteristics

В качестве заключительного вопроса к этому заданию предлагается ответить на вопрос «What should you do (or what steps should you take) to eliminate your negative traits of character?», можете выразить это в следующей форме: «I wish I were less... talkative, I wish I were more careful».

Выполнение подобного задания вызывает искренний интерес у студентов и показывает, что, как правило, студенты стремятся адекватно оценить себя и в данной учебной ситуации делают это уже на английском языке.

Задание 2

«The words and expressions on the list describe 2 different people. How many of the m apply to you? If you think a description applies to you put a tick in the box. If the description does not fit you, put a cross in the box. Be honest with yourself» [3, 26].

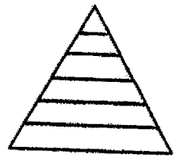
1	hardly	18	tolerant
2	meticulous	19	weak-willed
3	discreet	20	submissive
4	vain	21	taciturn
5	demanding	22	humble
6	indulgent	23	conscientious
7	ill-balanced	24	devoted
8.	shrewd	25	sentimental
9	calculating	26	genial
10	touchy	27	timid
11	impulsive	28	sensitive
12	industious	29	sensible
13	obstinate	30	meek
14	unpredictable	31	romantic
15	pragmatic	32	irresolute
16	independent	33	unpretentious
17	straightforward	34	lenient

Задание 3 [3, 27]

Put in the order of priority the qualities you would expect to find in people, e.g. a friend, a husband/wife, a tourist guide, a prime minister, a judge, a farmer, a parent, a street-sweeper, a businessman.

Задание 4 [3, 27]

A: Look at A. Maslov's triangle and place the needs of people in what you think to be the order of importance (preference)



The needs of people:
- self-actualization (personal fulfillment)
- physiological needs: food, shelter, clothes
- social needs: friendship, membership, companionship
- social security: safety in the workplace: insurance, pensions authority, leadership, power.

1. How do people progress from one level of need to another?
2. Is it possible to rearrange the order of needs?
3. Identify your personal needs at the present moment.

B: Look at the following values and associate them with different kinds of people:
- self-respect
- security
- warm relations with others
- a sense of accomplishment
- a sense of being well-respected
- a sense of belonging

Make a list of what values you think are most important.

C: Here are some values that exclude each other:
- self-reliance – government reliance
  - hard word – «easy life»
  - respect for individual – dislike of individual
  - independence – dependable security
  - parental values – peer group values
  - self-oriented – people-oriented
  - parent-centred household – child-centred household
  - saving – spending

husband-dominated home – wife-dominated home.

Be ready to discuss the consequences of having too much of any of the mutually exclusive traits. Use the adjectives you have learned.

Как видно, задание позволяет выстроить определенную иерархию ценностей и приоритетов в своей собственной жизни и жизни других людей и дать объяснение этому.

Задание 5

A person impresses you favourably if he/she is...

Задание 6. «Psychological Portrait»

Think of a person. It can be your groupmate, a teacher, any student, any famous person, etc... Give his/her psychological sketch so that others can guess who this person is. It is not a description of appearance! Dwell on the following: (Please, don't offend a person)
- general traits of character;
- temperament;
- intellect;
- attitude towards circumstances, events, occurrences;
- attitude towards people;
- attitude towards duties;
- manner of behaviour;
- making decisions.

Задание 7

Answer the questions of the test «How ambitious are you» and count your scores [1, 36].

Задание 8

Let's study a World Guide to good manners (4, p.39-41)

1. Look at the cartoons. What nationalities are the people? What makes it easy for you to identify them?
2. What is the stereotype English man or woman? What do you think is the stereotype for your nationality? Do you believe in stereotypes?
3. Which adjectives in the box do you think go with the nationalities below?

hard-working	easy-going	punctual	casual
reserved	emotional	lazy	talkative
hospitable	sociable	formal	fun-loving
enthusiastic	quiet	tolerant	friendly
sophisticated	well-dressed	nationalistic	respectful
humorous	serious	outgoing	romantic

American	Japanese	German
French	Italian	British

Is your nationality one of those above? If so, which adjectives did you choose? If not, choose some adjectives which you think go with your nationality. Here are some quotations from the text.

«We live in a global village but how well do we know and understand each other? Here is a sample test. Imagine you have arranged a meeting at 4 o'clock. What time should you expect your foreign colleagues to arrive? If they're German, they'll be bang on time. If they're

American, they'll probably be fifteen minutes early. If they're British, they'll be 15 minutes late, and you should allow up to an hour for the Italians».

«In Pakistan you mustn't wink. It is offensive».

«In Thailand you should clasp your hands together and lower your head and your eyes when you greet someone...»

Possible tasks

«Think of one or two examples of bad manners, For example, in Britain, it is considered impolite to ask people how much they earn»

«Do you agree with the saying «When in Rome, do as the Romans do?»

Задание 9 [Based on 2, 74]

A. «What is your horoscope sign?

Study the list:

Aries= Ram – Овен
Taurus=Bull – Телец
Gemini =Twins – Близнецы
Cancer – Рак
Leo – Лев
Virgo – Дева
Libra – Весы
Scorpio – Скорпион
Sagittarius=Archer – Стрелец
Capricorn – Козерог
Aquarius=Water-carrier – Водолей
Pisces=Fishes – Рыбы

Let's get acquainted with every sign, for example; «Some Virgos seem rather emotionless and cool. They know how to arrange and organize things. Observant and outspoken with no trouble in expressing themselves. Avoids excess of work. Talented in Arts and Literature».

B. What is your horoscope for this week (month)? Take any current newspaper or magazine and render the information from Russian into English.

Задание 10 [3, 38]

Test «How inhibited are you?»

We all need some inhibitions to function as social creatures - if you dosed off whenever you were bored, assaulted anyone who roused your temper, or voiced each unkind thought, you'd soon have no one to socialize with. Too many inhibitions, make us frustrated and ineffectual. How uptight are you? Take this liberating quiz and find out. Remember, though, to be honest in your responses. Read each statement below and mark it with a «T» for true and an «F» for false. If you are unsure of the anwser, just jot down (?)

1. I like to look at people when I talk to them.
2. 1 have a lot of resentments.
3. I like sports.
4. People frequently disappoint me.
5. I'm fairy easy to understand.
6. When someone hurts me, it rankles for a long time.
7. I like to talk about myself.
8. When I'm angry with someone, 1 tend to avoid him/her.
9. I'm not always entirely truthful or honest.
10. 1 hate to upset or disappoint anyone.
11. 1 have a lot of prejudices.
12. I sometimes feel I've unwittingly committed some wrong.
13. 1 care strongly about a number of people.
14. When things go amiss, I become angry with myself.
15. When I am upset or discouraged, 1 say so
16 1 worry about losing friends.
17. When something's wrong with me, 1 can't feel much concern for others.
18. I grow tense and wary when people praise me too much.
19. 1 enjoy looking at myself in the mirror.
20. People often take advantage of me.
21. Life would be easier if people would say what they feel.
22. I worry about making a good impression
23. I enjoy looking at a handsome man/woman.
24. Sometimes I feel utterly worthless.
25. 1 have strong feelings about a lot of things.
26. I often feel I'm being silently criticized.
27. At times, I have to fight for my rights.
28. I don't like to let people know I'm angry with them.
29. When a relationship sours, 1 try to determine what's wrong and fix it.
30. It is difficult for me to express love or tenderness

SCORING

Give yourself 1 point for each «?», 3 for every «F» answer to odd-numbered statements (1, 3, 5, etc.) and 3 for every «T» response to even-numbered statements (2, 4, 6, etc.). Add up your total.

18 points or fewer. Wild thing. You may have some specific inhibitions (fear of crowds, policemen..., strangers), but your general level of inhibition is low. Often your emotions take over and you act first-sometimes rashly- and think later. You may blurt out unkind remarks, latch on to people who later prove undesirable, make impulsive, expensive purchases – all out of lack of control. Many people are charmed by your unaffected manner. The more staid sort, however, sees you as Just a touch scary. Spontaneity isn't enough – you need thus plus control.

19-34 points. Comfortably relaxed. Warm and responsive, you're highly attuned to yourself and the world. Your emotional reactions are swift and strong, yet tempered with discipline. You communicate well and seldom leave any doubt as to your opinion. You're a loyal, open-hearted friend. You're easily hurt And that pain can turn to an explosive anger. Your main faults – if they can be called that – are over-generosity and a tendency to be blind to the defects of those you love.

35-50 points. Careful, but not too. Emotionallyyou're somewhat reserved and cautious, opening up only to those you know and trust. Your strength and control make you dynamic in social and business spheres, for you face crises stoically. You do pay a price for all that restraint, though suffering from an inability both to be warm and giving toward people you like and to vent your hostility toward those you don't.

More than 50 points. Why so uptight? Alas, you're an unhappy captive of your inhibitions. Frequently incapable of expressing what you feel, you often don't even know your own emotions. The overinhibited person may internalize emotions he/she can't express – he may turn his anger, say, against himself, with guilt, shame or despair – the unhappy by-product. If you score falls in this range, loosen up! Gravitate toward friendly folks with whom you feel comfortable and expansive, avoid those who rough you up emotionally. Enroll in an encounter group, anger workshop, or self-assertiveness class – and do it now!

Задание 11

Prepare small reports from the book «How to win friends and influence people» by Dale Carnegie, for example.

«When dealing with people, let us remember we are not dealing with creatures of logic. We are dealing with creatures of emotion, creatures bristling with prejudices and motivated by pride and vanity». Tell your report to the group.

Приведенные задания и упражнения имеют своей целью привнести новую информацию по теме, решая проблему информативности и новизны материала. Их использование позволяет также синтезировать все знания, полученные по педагогике и психологии в РД, способствуя более активному формированию межпредметных связей. И наконец, они развают «гностические психолого-педгогические умения, т.е. знание о самом себе и умение корректировать свое поведение». В условиях современного общества очень важным является умение человека правильно представить себя, и если студент педагогического вуза понимает сильные и слабые стороны своей личности, свой внутренний потенциал, то это дает ему возможность более успешной реализации себя в педагогической деятельности.

Литература

1. Headway, Pre-intermediate, Student's book, Oxford University Press.
2. *Bogorotiskaya V.N., Khrustalyova L.V.* English VIII, Москва: «Версия»
3. Хорошо ли вы знаете себя?: Учеб.-метод. м-лы для студ. V курса ф-та англ. яз. Н. Новгород, 1993.
4. Headway, Intermediate Oxford University Press.

Т.Н. Федоровская, Н.В. Павлова

ЛЕКСИЧЕСКИЕ АРХАИЗМЫ И ИХ СТИЛИСТИЧЕСКИЕ ФУНКЦИИ (НА ПРИМЕРЕ ПРОИЗВЕДЕНИЙ Э.А. ПО)

Э.А. По (1809-1849) – один из выдающихся писателей и поэтов-романтиков XIX века. Его художественному наследию посвящено множество литературоведческих исследований. Вместе с тем представляется необходимым подходить к творчеству такого интересного и трудного стилиста, как Э. По, с лингвистической точки зрения. В данной статье затронута проблема архаизированности языка произведений Э. По, в частности использование лексических архаизмов, что является, по мнению авторов, отличительной чертой стиля американского поэта и новеллиста.

В лингвистической литературе достаточно подробно освещены проблемы определения, классификации и стилистической направленности архаизмов [1, 3, 4, 5]. Под архаизмами понимаются «слова или выражения, вышедшие из повседневного употребления и потому воспринимающиеся как устарелые» или «тропы, состоящие в употреблении старого (старинного) слова или выражения в целях исторической стилизации, придания речи возвышенной стилистической окраски, достижения комического эффекта и т.п.» [2, 56]. Таким образом, в языке художественных произведений потенциальные возможности архаизмов обычно интерпретируются как фактор стиля. Традиционно архаизмы принято рассматривать как разновидность поэтизмов, ввиду того что основной сферой использования архаизмов является поэзия, а также благодаря сходству стилистических функций этих групп слов. Такое объединение едва ли правомерно. Во-первых, сфера использования архаизмов не ограничивается поэзией, в качестве экспрессивного изобразительного средства архаизмы употребляются также и в других стилях речи. Во-вторых, круг стилистических функций архаизмов неизмеримо шире, нежели только придание тексту оттенка патетичности, возвышенности. Обзор форм, типов и функций лексических архаизмов в стихах и новеллах Э. По может служить подтверждением данного положения.

Представляется необходимым более подробно остановиться на языковых особенностях произведений Э. По, в частности на специфике лексикона поэта. К особенностям словаря Э. По следует прежде всего отнести преобладание слов, относящихся к разряду лексики ограниченного употребления, т.н. «высоких» слов, над нейтральной, общеупотребительной лексикой. К словам «высокой стилистической тональности» принадлежат в первую очередь книжно-литературные слова, обладающие возвышенной стилистической окраской. Появление их в новеллах и стихах Э. По вполне закономерно: лексика в его произведениях является возвышенной в силу возвышенности основополагающих в творчестве По тем. Можно привести немало примеров синонимических пар, где нейтральному в эмоциональном и стилистическом отношении слову германского происхождения Э. По предпочитает характеризующееся торжественной стилистической окраской слово романского происхождения: *phantasm – ghost, awe – fear, infancy – childhood, demeanor – behavior, cease – stop, aid – help* и т.д. Из приведенных пар этимологических дублетов Э. По выбирает слова романского происхождения, что отражает, по всей видимости, стремление писателя к литературной отшлифованности и изяществу языка.

К лексике «высокого стилистического тона» относится и традиционный поэтический словарь, лексические единицы которого также находят широкое применение в произведениях Э. По. К ним относятся:

– архаизмы: *surcease – an end or cessation, affright – sudden fear or terror* и др.: *«vainly I had sought to borrow/From my books surcease of sorrow/sorrow for the lost Lenore...»* («The Raven»);

– архаические формы слов (морфологические варианты общеупотребительных слов): *morrow, morn* –

morning; eve, even и т.д. («Eagerly I wished the morrow...»);

– поэтизмы: *unto (archaic or poetic), brow – forehead, alarum – alarm: «The winds that bathed my brow came unto me laden with soft sighs...»* («Eleonora»); *«...the noise... alarumed and reverberated throughout the forest»* («The Fall of the House of Usher»);

– «чистые» поэтизмы, неупотребительные нигде, кроме поэзии: *o'er, ne'er, spurn'd, thro', 'tis* и т.п.

Важно отметить, что использование архаизмов в целях придания тексту оттенка торжественности, возвышенности не является типичным для литературных произведении прозаического жанра. Лирический или патетический тон рассказа создается, как правило, с помощью иных стилистических средств. В то же время способность архаизмов придавать тексту определенный «исторический колорит» находит широкое применение в прозе, где архаизмы выступают как средство исторической стилизации (архаизации). Особенно четко данная функция архаизмов прослеживается в исторических новеллах и романах. В ряде рассказов Э. По также прибегает к употреблению архаизмов в целях речевой характеристики персонажей, живших в одну из предшествовавших в развитии общества исторических эпох. Так, рассказ «Король Чума» может рассматриваться как образец мастерски исполненной стилизации речи героев – представителей лондонских трущоб XIV века. В рассказе «Падение дома Ашеров» в ткань повествования включаются отрывки, написанные более архаизированным языком, чем остальной текст рассказа:

«... Who entereth herein, a conqueror hath bin;
Who slayeth the dragon, the shield he shall win.
And Ethelred uplifted his mace, and struck upon the head of the dragon, which fell before him, and gave up his pesty breath, with a shriek so horrid and harsh, and withal so piercing, that Ethelred had fain to close his ears with his hands against the dreadful noise of it, the lice whereof was never before heard» («The Fall of the House of Usher»).

Архаичные явления прослеживаются в этом отрывке на всех языковых уровнях – фонетическом (*hath bin*), лексическом (*withal, fain*), грамматическом (*slayeth, entereth, hath bin*), синтаксическом (*the lice whereof was never before heard* – инвертированный порядок слов). Слово *slayeth* является не только грамматическим архаизмом, это также семантический архаизм, синонимичный современному *to kill*. Роль архаичных явлений в цитируемом отрывке – стилизация под старину.

Рассуждая о специфике стиля Э. По в целом, важно подчеркнуть, что применение поэтизмов отличается у Э. По большим своеобразием. Характерной чертой стиля Э. По является использование слов поэтического словаря, в том числе и архаизмов, в прозе, причем архаичные языковые явления у По-новеллиста и По-поэта несут сходную стилистическую нагрузку. Типичным для этого автора является использование архаизмов в новеллах как средства создания архаизирующе-возвышенного стиля, что сближает их с функциями архаизмов в поэтических произведениях. Это связано с тем, что рассказы Э. По редко могут быть отнесены к какой-либо конкретной исторической эпохе, действие новелл происходит как бы вне времени, что обусловлено, в свою очередь, отстраненностью тем и героев от обыденности, повседневности. В произведениях Э. По создана целая галерея образов прекрасных, ангелоподобных героинь, которые своими странными стремлениями, познаниями, неизлечимой грустью сильно напоминают личность их творца и являются, по сути, выразителями авторских мыслей и идей. Архаичность и опоэтизированность языка этих «авторских героинь» призвана, с одной стороны, показать высокий строй их внутренней жизни, придать персонажам черты утонченности, аристократизма, и в то же время подчеркнуть возвышенный характер темы.

Интересно отметить, что многие образы и темы, зарождаясь в поэзии, проходят как лейтмотив через все творчество Э. По, появляясь в более поздних прозаических произведениях: *«the same images are found again, in every impressive variation of phrase...»* («Eleonora»). Новеллы «Элеонора», «Лигейя», «Морелла» и ряд лирических произведений объединяет тема смерти молодой прекрасной женщины и горя ее возлюбленного. Этот мотив вообще характерен для американской литературы начала XIX века, а у Э. По связан и с личной жизненной драмой.

Общность идей и образов предопределяет выбор сходных экспрессивных средств – архаизмов. Таким образом, происходит перенос стилистических функций архаизмов из поэзии в прозу. Данное утверждение может быть также верифицировано тем фактом, что архаизмы, придающие речи персонажей торжественный тон, появляются в рассказах в самые драматичные, кульминационные моменты повествования, обычно связанные с сильными душевными переживаниями персонажей – в моменты, в которых раскрывается основная идея произведения. Таким образом, само содержание текста требует использования торжественно-патетической лексики.

Принято считать, что возвышенная стилистическая окраска возникает в архаизмах благодаря принадлежности их к традиционному поэтическому словарю. Действительно, бывает трудно провести строгую разграничительную линию между поэтизмами и архаизмами. По мнению языковедов, это связано с тем, что многие слова поэтической лексики в настоящее время осознаются уже как архаизмы. Таким образом, архаизмы частью примыкают, а частью входят в традиционный поэтический словарь. В словарях такие лексические единицы снабжены обычно двойной пометой – «archaic or poetic». Принимая во внимание эту особенность архаичной лексики и учитывая то, что некоторые лексические единицы могут разными авторами пониматься и как архаизмы, и как поэтизмы, на материале стихов и новелл Э. По выявлены следующие типы лексических архаизмов:

1) архаизмы лексико-фонетические, в которых устаревшим является звуковой облик слова: *«Now and then, alas, the conscience of man takes up a burthen so heavy in horror that it can be thrown down only into the grave»* («The Man of the Crowd»);

2) архаизмы лексико-словообразовательные – слова, устаревшие только в какой-то морфологической части, чаще всего в суффиксе: *«This – all this – was in the olden/ Time long ago...»* («The Haunted Palace»);

3) семантические архаизмы: *fain* в устаревшем значении «gladly, with eagerness»: *«I would fain have them believe that I have been... the slave of circumstances beyond human control»* («William Wilson»);

4) собственно лексические архаизмы – *whither «where, in what direction»; anon «soon, shortly»: «Oh! Whither shall I fly? Will she not be here anon?»* («The Fall of the House of Usher»);

5) фразеологические архаизмы – *in sooth «in truth, truly»: «And Ethelred... waited no longer to hold parley with the hermit, who, in sooth, was of an obstinate and maliceful turn...»* («The Fall of the House of Usher»).

Архаизмы в вышеприведенных примерах выступают в качестве одного из компонентов языкового материала, наряду с особенностями синтаксиса и морфологии, способствующего передаче основной идеи текста.

В заключение хотелось бы отметить, что обращение к архаичной лексике отнюдь не делает язык произведений Э. По тяжелым – он музыкален и строен по звучанию, рассказы Э. По как бы стремятся перерасти в стихотворение.

Впервые появившись в поэзии, образы и темы возникают в перефразированном виде в прозаических произведениях. В новеллах и стихах вновь и вновь возникает излюбленная тема Э. По – тема соприкосновения мира живых и мира мертвых, звучит идея трагической судьбы поэта в мире бескрылых существ и бессмертия души.

Общность тематики обусловливает сходство стилистической задачи архаизмов в поэзии и прозе Э. По: включаясь в общую стилистическую тональность текста, архаизмы вносят в него экспрессивную коннотацию, придавая произведению патетический характер высокого стиля. С помощью архаизмов создается особый, неповторимый колорит рассказов Э. По, не соотносимый с какой-либо исторической эпохой, переносящий читателя в сферу трансцендентного, в увлекательный мир эфемерных фантазий и неземных мечтаний.

Литература

1. *Арнольд И.В.* Стилистика современного английского языка (Стилистика декодирования). Л.: Просвещение, 1973.

2. *Ахманова О.С.* Словарь лингвистических терминов. М.: Сов. энцикл., 1966.

3. *Кузнец М.Д., Скребнев Ю.М.* Стилистика английского языка / Под ред. Н.Н. Амосовой. Л.: Учпедгиз, 1960.

4. *Пелевина Н.Ф.* Стилистический анализ художественного текста. Л.: Просвещение, 1980.

5. *По Э.* Избранное. Сборник. На англ. яз. / Сост. Е.К. Нестерова. М.: Радуга,1983.

6. Стилистика английского языка / А.Н. Мороховский, О.П. Воробьева, З.В. Тимошенко. Киев: Вища школа, 1991.

СТУДЕНЧЕСКАЯ НАУКА

С.Ю. Иванов

КОМПЬЮТЕРНАЯ АРИФМЕТИКА: ИЗВЛЕЧЕНИЕ КВАДРАТНОГО КОРНЯ ИЗ МНОГОРАЗРЯДНОГО ЧИСЛА

В статье освещается достаточно популярная тема в программировании – работа с многоразрядными числами, для которых не хватает диапазона стандартных типов данных. Впервые в образовательной информатике эта тема рассмотрена в учебном пособии [1]. Наиболее полное изложение можно найти в работах [2, 3]. Однако, даже в этих публикациях задача извлечения квадратного корня из многоразрядного числа не затрагивается. Основой для рассматриваемого в статье алгоритма послужил справочный материал [4, с. 45-46].

Квадратный корень извлекается из натурального числа, и пусть мы находим целую часть результата. Воспользуемся следующей схемой решения задачи.

1. Многоразрядное число разбиваем на несколько пар (граней) чисел. Если число содержит четное количество разрядов, то первая грань содержит две цифры, если нечетное, то одну. Так, в случае с числом 236488 имеем три грани, каждая из которых содержит две цифры: 23 64 88, а в случае с числом 15385 первая грань содержит всего одну цифру: 1 53 85. Количество таких граней определяет количество разрядов в ответе. Для примера возьмем число 574564 и разобьем его на грани: 57 45 64. Таким образом, ответом является трехразрядное число (обозначим через y).

2. Далее берем первую грань 57 и подбираем число x, такое, что x^2 было как можно ближе к 57, но не превосходило его. Получаем x=7, так как 7*7=49<57, а 8*8=64>57. Первая цифра ответа y равна 7.

3. Найдем разность между первой гранью 57 и x^2 =49: 57-49=8. Эту разность припишем к следующей грани, получаем 845.

4. Подбираем следующую цифру. Находим новое значение x, такое, чтобы $(2*y*10+x)*x$ не превосходило 845 и было наиболее близким к 845. Это число 5, так как (14*10+5)*5=725<845, а (7*10+6)*6=876>845. Итак, вторая цифра ответа 5 и y=75.

5. Повторим шаг №3. Найдем разность: 845-725=120, к следующей грани «подклеим» эту разность: 12064.

6. Выполняем действия шага №4. $(2*y*10+x)*x<=12064$. Находим x, оно равно 8, так как (150*10+8)*8=12064. Итак, y=758.

Примечание. Если корень не извлекается в результате повторения шагов 3, 4, то после последней цифры заданного числа ставят запятую и образуют дальнейшие грани, каждая из которых имеет вид 00. В этом случае процесс извлечения корня прекращается, когда достигается требуемая точность.

Пояснений требует шаг 4: непонятно, почему используется формула для подбора очередной цифры ответа. Пусть $10*y+x=\sqrt{z}$. Возведем в квадрат число $10*y+x$, где y, x – цифры. Получим $100*y^2+10*y*x+10*y*x+x^2=z$. Значение y подобрано на шаге 2, оно дает вклад $100*y^2$ в число z. Подбор цифры x обязан дать максимальный вклад в число z-$100*y^2$.

Прежде всего необходимо определить структуры данных, константы и переменные, которые потребуются для работы.

Константа NRaz определяет максимальное количество разрядов в записи длинного числа.

TArray – массив для хранения разрядов длинного числа

TarrayLA – запись, хранящая знак (Znak) длинного числа, количество разрядов в нем (NumRaz) и сами разряды числа (Digit).

Переменная a – число, из которого необходимо извлечь квадратный корень.

```
Const NRaz=100;
Type Chifra=0..9;
  TArray=Array[1..NRaz] of Chifra;
  TArrayLA=Record
    Znak:0..1;
   NumRaz:Integer;
   Digit:TArray;
   end;
Var a:TArrayLA;
```

Далее нам потребуется набор следующих операций (рассмотрение их можно найти в учебном пособии [3]):

Plus (a,b) – сложение двух многоразрядных чисел a и b, результат возвращается в a.

Minus (a,b) – вычитание многоразрядного числа b из a, результат возвращается в a.

Multiplication (a,b) – умножение двух многоразрядных чисел a и b, результат возвращается в a.

IsBMore (a,b) – логическая функция, принимает значение True, если b больше, чем a, в противном случае результат функции False.

IsABEqual (a,b) – логическая функция, возвращает True, если a и b равны, в противном случае – False.

Процедура SqrtMy будет отвечать за извлечение квадратного корня. Для ее реализации нам также понадобятся процедуры:

• сдвига влево MoveLeft (a,1), где в параметре указываем, в каком числе (a) производим сдвиг и на сколько разрядов (1);

• перевода обычного короткого числа в формат длинного Trans (a, 2), цифра 2 будет представлена в формате длинного числа, и результат возвращается в a;

• процедура Zero (a), которая обнуляет значение длинного числа (a).

В переменную Rez формируем результат, переменная y представляет собой следующее число 2*Rez*10, т.е. фактически это то же самое, что и y в разобранном примере, а переменная x – для подбора очередной цифры. P2 – переменная вида $(2*y*10+x)*x$ для подбора очередной цифры (x), P1 – число, к которому стремится P2 при подборе очередного x. One – хранит число «один» в формате длинного числа, эта переменная потребуется для увеличения многоразрядного числа на единицу, Two хранит 2 в формате многоразрядного числа, она необходима для умножения числа на 2.

```
Procedure SqrtMy(Var a:TArrayLA);
 Var Rez,p1,p2,x,y,one,two:TArrayLA;
Begin
 Zero(Rez); Zero(p1); Zero(p2); Zero(y); {обнуляем
                                        переменные}
 Trans(one,1);
 Trans(two,2);
 p1.NumRaz:=1;
if (a.NumRaz mod 2)<>0 then p1. Digit [1]: =
= a.Digit[a.NumRaz]      {Смотрим, сколько цифр
                          будет в самой первой
                          (или левой) грани и
                          записываем их в число p1}
 else begin
  p1.Digit[2]:=a.Digit[a.NumRaz];
  p1.Digit[1]:=a.Digit[a.NumRaz-1];
  p1.NumRaz:=2;
 Dec(a.NumRaz);
 end;
Dec(a.NumRaz)            {Перебираем грани числа a,
                          пока в нем все разряды
                          не будут просмотрены,
                          т.е. пока a.NumRaz>=0}
While a.NumRaz>=0 do begin
Zero(x);
 While IsBMore(p2,p1) or IsABEqual(p2,p1) do begin
  Plus(x,one); {Берем следующее
                значение x}
  p2:=y;       {Запоминаем
                значение y в p1}
  Plus(p2,x);  {Прибавляем к
                этому значению x}
  Multiplication(p2,x);
               {И умножаем на x}
end;
 Minus(x,one); {Откат x на одно значение, так как
                получено число, которое превосходит
                то, к которому мы стремились.
  Plus(p2,x);  Для нового x вновь считаем число
               вида (2*y*10+x)*x}
  Multiplication (p2,x);
  MoveLeft(Rez,1);
  Plus(Rez,x);

  Minus(p1,p2);          {В p1 запоминаем разницу
                          между p1 и p2
  MoveLeft(p1,2);        Замена операции умножения
                          на 100, сдвигаем число p2
                          на 2 разряда}
 Zero(p2);
 p2.Digit[2]:=a.Digit[a.NumRaz];   {Берем следующую
 p2.Digit[1]:=a.Digit[a.NumRaz-1]; грань и записываем
                                   ее в p2}
 p2.NumRaz:=2;
  Plus(p1,p2);           {К найденной нами разнице p1
                          прибавляем новую грань p2}
 Zero(p2);
 y:=Rez;
  Multiplication(y,two);  | Переменная y
  MoveLeft(y,1);          | принимает вид
  Dec(a.NumRaz,2);        | (2*y*10)
end;
a:=rez;                  {В переменную a возвращаем
                          найденный ответ}
End;
```

По алгоритму эти 4 строчки считают число вида $(2*y*10+x)*x$, где $2*y*10$ – это p2

Литература

1. *Касаткин В.Н.* Информация. Алгоритм. ЭВМ. М.: Просвещение, 1991.

2. *Андреева Е.В., Фалина И.Н.* Информатика: системы счисления и компьютерная арифметика. М.: Лаборатория Базовых Знаний, 1999.

3. *Окулов С.М., Пестов А.А.* 100 задач по информатике. Киров: Изд-во ВГПУ, 2000.

4. *Гусев В.А., Мордкович А.Г.* Математика: Справочные материалы. М.: Просвещение, 1990.

А.Н. Ронжин

МЕТОДЫ ПОИСКА ОШИБОК В ПРОГРАММНОМ КОДЕ С ИСПОЛЬЗОВАНИЕМ ДИРЕКТИВ КОМПИЛЯТОРА В ПРОЦЕССЕ ОТЛАДКИ ПРОГРАММ

Отладка – это процесс поиска и исправления ошибок в программе, препятствующих ее корректной работе. Она представляет собой набор процедур и действий, начинающихся с выявления самого факта ошибки и заканчивающихся установлением точного места и характера этой ошибки.

Отладка является наиболее сложной и продолжительной частью в процессе разработки современного программного обеспечения. По мнению специалис-

тов, более 70% рабочего времени программиста затрачивается на отладку программы и только 30% – на ее разработку и написание!

В данной статье рассматривается возможность использования директив компилятора для упрощения процесса отладки программ и написания более «гибкого» программного кода.

Использование директив компилятора

Директивы компилятора вставляются в программу в виде особым образом оформленных комментариев и модифицируют те или иные возможности компилятора в процессе компиляции.

Существует три вида директив: переключающие, условные и параметрические.

Переключающая директива содержит букву, обозначающую опцию и знак «+» или «–». Знак «+» означает установку опции в активное состояние, а «–» – в пассивное.

Процесс отладки программ

Собственно, для отладки программы важны директивы {$D+}, {$L+}. С их помощью в исполняемый код программы вставляется отладочная информация, поэтому программу из среды программирования можно выполнять «по шагам», просматривая значения тех или иных переменных.

Для обсуждения методов поиска ошибок в программом коде с помощью директив компилятора рассмотрим следующую программу, написанную на языке программирования Турбо Паскаль:

```
Program Test;
Const n=10;
Type arr=array[1..n] of integer;

Procedure init(var a:arr);
var i:integer;
begin
 i:=1;
 while i<n+1 do begin
 readln(a[i]);
 inc(i);
 end;
end;

Procedure solve(var a:arr;i:integer);
begin
 if a[i]<0 then a[i]:=-a[i];
 if a[i]>=0 then solve(a,i+1);
end;

Procedure print(a:arr);
var i:integer;
begin
 for i:=1 to n do
  write(a[i],' ');
end;

var a:arr;
Begin
 init(a);
 solve(a,1);
 print(a);
End.
```

В данной программе пользователь вводит с клавиатуры целые числа, а программа меняет первое отрицательное число на положительное. На первый взгляд программа должна работать корректно, но в ней есть несколько ошибок, которые возможно найти, используя при отладке директивы компилятора.

• Контроль ошибок ввода/вывода {$I±}

Использование директивы {$I+} приводит к тому, что при работе программы строго контролируется соответствие поступающих констант в программу типам переменных, которым значения этих констант будут присвоены. Если типы констант и переменных не совпадают, например при вводе символа вместо числа, выполнение программного кода будет прервано.

Предположим, что мы выполняем вышеприведенную программу и вводим следующие числа: 35, 12, 0, 27, –15, 32x, –34, 50, –73, 100.

Система выдает ошибку времени выполнения (106), курсор будет позиционирован на операторе readln (a[i]);

В чем же здесь дело? Программа ожидала ввода целочисленного значения, а было введено нечисловое значение «32x», что и привело к ошибке выполнения. В небольшой программе такая ошибка не доставит много хлопот, можно ввести и заново, но если необходимо набрать 20 или 30 чисел и снова будет допущена ошибка? Пришлось бы начать ввод чисел сначала.

Установка директивы {$I–} позволит устранить автоматический контроль ошибок ввода/вывода и позволит самим тестировать такие ошибки внутри программы.

Тестирование ошибок ввода/вывода внутри программы осуществляется посредством функции IOResult, которая возвращает 0, если операция выполнена удачно; и номер ошибки – в противном случае.

Если преобразовать программу следующим образом, то любое неправильно введенное значение можно ввести еще раз, поскольку увеличение счетчика введенных чисел происходит только после проверки правильности ввода:

```
{$I-}
.....
Procedure init(var a:arr);
var i:integer;
begin
 i:=1;
while i<n+1 do begin
 readln(a[i]);
 if IOresult=0 then inc(i); {Проверка на правильность ввода}
 end;
end;
```

Такая проверка на правильность ввода чисел применима и к вводу из файла.

• Контроль за переполнением стека {$S±}

При использовании ключа {S+} в программу вставляется код проверки стека на переполнение. Прерывание работы программы с ошибкой Stack overflow чаще всего означает, что в программе есть подпрограмма, использующая рекурсивные вызовы, работа которой из-за ошибки завершиться не может.

Попробуем ввести следующие числа: 35, 12, 0, 27, –15, 32, –34 50, –73, 100.

Какой должен быть ответ? Он очевиден: мы встречаем первое отрицательное число, меняем его на положительное и прекращаем работу. Результат должен быть: 35, 12, 0, 27, 15, 32, –34 50, –73, 100. Вместо этого программа зацикливается.

Включив в программный код директиву компилятора {$S+}, мы увидим, что процедура Solve зацикливается и программа выдает сообщение о переполнении стека (202), курсор устанавливается на начало этой процедуры. Все дело в том, что, встретив число –15, программа заменяет его на 15, проверка if a[i]>=0 then solve(a,i+1) уже не работает, так как число a[i] изменилось и стало положительным. Соответственно программа заменяет не первое, а все отрицательные числа.

Изменив логическую конструкцию внутри процедуры, мы получим желаемый результат:

```
{$I-}
{$S+}
.....
Procedure solve(var a:arr;i:integer);
begin
 if a[i]<0 then a[i]:=-a[i]
 else solve(a,i+1);
end;
```

• Контроль на принадлежность допустимому диапазону {$R±}

Использование директивы {R+} позволяет контролировать выход за границу массивов и выход за границу допустимого диапазона значений при операциях над целочисленными переменными. Таким образом, при попытке обращения к несуществующему элементу массива или во время выполнения операции над целыми числами результат не является допустимым для соответствующего типа, тогда выполнение программного кода прерывается.

Вернемся к программе и введем следующие значения: 35, 12, 0, 27, 15, 32, 34 50, 73, 100. Среди них отрицательных чисел нет, и программа снова начинает зацикливаться в процедуре Solve.

Чтобы понять причину ошибки, необходимо проанализировать программный код: рекурсия выполняется бесконечное количество раз вместо 10, поскольку после просмотра последнего числа вызов рекурсии происходит снова и, не находя отрицательного числа, программа снова и снова производит рекурсивный вызов.

Каким же образом можно выявлять подобные ошибки? Можно включить директиву {$R+} (проверку диапазона) в начало программы. Запустив программу еще раз, мы увидим сообщение об ошибке времени выполнения – выход за допустимый диапазон (201), поскольку индекс массива выходит за допустимые границы, как только программа начинает выполнять оператор: if a[i]<0 then a[i]:=-a[i] при значении i=11. Чтобы устранить данную ошибку, необходимо ввести оператор, который бы проверял выход значения индекса массива за допустимые границы:

```
{$I–}
{$S+}
{$R+}
.....
Procedure solve(var a:arr;i:integer);
Begin
If i<n+1 then begin   {Проверка выхода значения
                       индекса массива
                       за допустимые границы}
 if a[i]<0 then a[i]:=-a[i]
 else solve(a,i+1);
 end;
end;
```

Существуют такие ситуации, когда, вероятно, потребуется нарушить границы, например, при работе с функциями Succ и Pred (возращающими следующие и предыдущие значения переменной) для перечисляемых типов данных.

Можно установить контроль принадлежности допустимому диапазону выборочно, размещая в начале программы {$R–} директиву, а для каждой части программы, которая требует контроля, {$R+} в начале этой части и директиву {$R–} в конце. Тогда проверку нашей рекурсивной процедуры можно было бы написать следующим образом:

```
{$I–}
{$S+}
{$R–}
.....
Procedure solve(var a:arr;i:integer);
Begin
 If i<n+1 then begin
 {$R+}
 if a[i]<0 then a[i]:=-a[i]
 else solve(a,i+1);
 {$R–}
 end;
end;
```

Контроль принадлежности допустимому диапазону будет осуществляться только внутри оператора: if i<n+1 then begin ..... end; и нигде больше.

Стиль написания программ

Лучшим средством облегчить неизбежную отладку является профилактика ошибок еще на этапе разработки программы. Применение следующих правил при написании программного кода сократит количество ошибок и значительно облегчит их поиск в Ваших программах:

1) Писать текст программы и производить ее отладку следует небольшими частями. Перед тем, как использовать результаты работы одной части программы в другой, следует добиться правильной работы первого блока.

2) Информацию в процедуры следует передавать через параметры, избегая ссылок на глобальные переменные.

3) Текст программы должен быть читаемым, не следует писать операторы вплотную друг к другу. Желательно, чтобы в строке было не более одного оператора, поскольку отладчик Турбо-Паскаля работает построчно, такой подход облегчит локализацию ошибки.

4) Желательно разбивать текст программы на процедуры и функции. Не следует писать большие процедуры (содержащие более 20-30 строк).

5) Выполняемые в программе действия надо как можно чаще сопровождать комментариями {}.

И главное, не забывать два закона поиска ошибок в программе:

Закон № 1. В каждой новой программе есть ошибка.

Закон № 2. Исправив ошибку в программе, вы получили новую программу (смотри Закон № 1).

В.В. Шихов

СИСТЕМЫ УПРАВЛЕНИЯ СЕТЬЮ

1980 год отмечается резким ростом в области применения корпоративных информационных сетей. Как только компании поняли, что корпоративная среда информационного обмена может обеспечить им сокращение расходов и повышение производительности, они начали устанавливать новые и расширять существующие сети почти с такой же скоростью, с какой появлялись новые технологии. В скором времени стали очевидны проблемы, число которых все больше увеличивалось по мере расширения круга решаемых задач.

Рост временных затрат и численности персонала, необходимых для решения как каждодневных задач, как и задач стратегического планирования, оказались в нелинейной зависимости от числа объектов управления.

Компании оказались настолько зависимы от устойчивости и производительности работы корпоративной сети, что простои, вызванные сбоями как на уровне аппаратной, так и на уровне программной части, приводят к реальным потерям в бизнесе, а у многих организаций просто не хватало ресурсов для содержания соответствующего штата и поддержания центра управления сетью. Увеличение разнообразия информационных задач приводит даже к практике отдачи на сторону задач управления сетью, такая технология называется аутсортинг.

В первую очередь следует четко обозначить круг проблем, подлежащих решению. Для этого необходимо в первую очередь определить сам термин «управление сетью». Мы будем понимать это как решение круга задач, связанных с поддержанием сети в работоспособном состоянии (как физических компонентов, так и программных). В соответствии с трактовкой ISO (ISO/IEC 7498-4:1989 Information processing systems – Open Systems Interconnection – Basic Reference Model – Part 4: Management framework, ISO/IEC 10040:1998 Information technology – Open Systems Interconnection – Systems management overview), можно выделить *пять задач системы управления сетью.*

• **Управление конфигурацией сети** (*configuration management*) включает в себя регистрацию устройств, работающих в сети, их местоположение, сетевые адреса и идентификаторы, управление параметрами сетевых операционных систем, поддержание схемы сети. Задача состоит в конфигурировании параметров элементов сети и сети в целом. Для элементов сети, таких, как маршрутизаторы, коммутаторы и т.п., определяется их конкретная конфигурация. Для сети в целом управление конфигурацией обычно начинается с построения карты сети, т.е. с отображения реальных связей между элементами сети и их изменений – образования новых физических или логических каналов, изменения таблиц коммутации и маршрутизации.

• **Управление ошибками** (*fault management*) – эта группа задач включает выявление, определение и устранение последствий сбоев и отказов в работе сети. Программное обеспечение может выполнять не только регистрацию сообщений об ошибках, но и их фильтрацию, уведомление обслуживающего персонала, а также анализ на основе некоторой корреляционной модели. В то же время в большинстве случаев размеры сети и количество объектов в ней таковы, что требуется только своевременное уведомление администратора о факте возникновения проблемы, все остальное он сделает самостоятельно.

• **Анализ эффективности** (*performance management*). Задачи этой группы связаны с оценкой на основе накопленной статистической информации таких параметров, как время реакции системы, пропускная способность реального или виртуального

канала связи между двумя конечными абонентами сети, интенсивность трафика в отдельных сегментах и каналах сети, вероятность искажения данных при их передаче через сеть, а также коэффициент готовности сети или ее определенной транспортной службы. Задачи анализа производительности и надежности сети нужны как для оперативного управления сетью, так и для планирования развития сети и могут требовать для своего решения использование таких методов исследования работы сети, как анализаторы протоколов, которые позволяют захватывать циркулирующие в сети пакеты, изучать их содержимое. Основываясь на результатах анализа, можно вносить обоснованное и взвешенное решение об изменении конфигурации сети, оптимизации ее производительности, выявлении причин неполадок. Хотя проведение такого рода исследования и помогает ответить на многие вопросы, связанные с функционированием информационной системы, но для полного анализа иногда требуется до нескольких дней. Для облегчения работы системного администратора в пакеты анализа сетевого трафика включаются такие функции, как настройка правил фильтрации, активизация функций захвата по различным условиям и событиям, элементы экспертных систем с базой данных сигнатур пакетов.

• **Управление безопасностью** (*security management*) включает в себя контроль доступа к ресурсам сети (данным и оборудованию) и сохранение целостности данных при их хранении и передаче. В его функции входит процедура аутентификации, проверка привилегий, поддержка ключей шифрования, управление полномочиями. К этой же группе можно отнести такие наиболее ответственные (именно из-за их недостатков происходит большинство вторжений в систему) механизмы, как управление паролями, внешним доступом, коммуникации с внешними сетевыми ресурсами.

• **Учет работы сети** (*accounting management*). Задачи этой группы включают регистрацию времени использования различных ресурсов сети – устройств, каналов и транспортных служб. Обычно они имеют дело с такими понятиями, как время использования службы и оплата арендуемых ресурсов (billing). Довольно часто нужно владеть информацией о том, как долго был недоступен тот или иной ресурс, например арендуемый канал.

Первые шаги по созданию систем управления сетью были направлены в основном на решение задач по конфигурированию сетевого оборудования и разработке протоколов обеспечивающих удаленное управление. На этом этапе необходимо было с одной стороны обеспечить возможность управления оборудованием различных производителей, а с другой – предоставить единый интерфейс для решения одних и тех же задач.

Результатом таких разработок стала схема взаимодействия агента с менеджером. На основе этой схемы, в принципе, могут быть построены системы практически любой сложности с большим количеством агентов и менеджеров разного типа.

Основополагающим моментом в такой схеме управления является тот факт, что менеджер и агент располагают одной и той же моделью управляемого ресурса, которая отражает только те характеристики ресурса, которые нужны для его контроля и управления. Агент наполняет структуры данных, соответствующие модели управляемого ресурса текущими значениями характеристик данного ресурса. Менеджер использует модель, чтобы иметь информацию о том, чем характеризуется ресурс, какие характеристики он может запросить у агента и какими параметрами можно управлять.

Менеджер и агент взаимодействуют по стандартному протоколу. Этот протокол позволяет менеджеру запрашивать значения параметров, хранящихся у агента, а также передавать агенту управляющую информацию, на основе которой тот должен управлять устройством.

Схема взаимодействия менеджер – агент лежит в основе таких популярных стандартов управления, как стандарты Internet на основе протокола SNMP и стандарты управления ISO/OSI на основе протокола CMIP.

Системы управления, построенные на протоколах SNMP и CMIP, хорошо зарекомендовали себя в решении как задачи по управлению оборудованием и удаленного наблюдения (RMON), так и в задачах управления сетевыми службами. Но переход от простых локальных сетей к сложным гетерогенным сетям, которые на основе различных аппаратных платформ и операционных систем предоставляют услуги электронной почты, сервера Web, баз данных и многих других приложений и сервисов, вывел на первый план проблемы разграничения и контроля доступа к корпоративным ресурсам. Программное обеспечение различных производителей хранит информацию о настройках, пользователях и рабочей среде пользователей в своих собственных конфигурационных файлах. Это приводит к дублированию информации в различных приложениях, усложняет администрирование и делает невозможным использование конфигурационной информации одного приложения другими. В то же время конечным пользователям становится все сложнее работать с сетевыми ресурсами (например, при смене пароля необходимо изменить его во всех приложениях). Все это, естественно, увеличивает вероятность ошибок и стоимость владения системой в целом. Следовательно, для комфортной работы необходима единая служба каталогов с аутентификацией доступа ко всем ресурсам сети. Такая служба должна централизованным образом хранить не только имена пользователей и их пароли, но и множество данных различных типов (например, контактную информацию, конфигурацию приложений,

пользовательские настройки приложений и рабочих сред). Кроме того, она должна предоставлять пользователям возможность доступа к собственным данным с любого компьютера как в локальной сети, так и в любой точке глобальной сети.

Для создания единой учетной записи пользователя и получения доступа к разным сервисам необходим мостик между службой каталогов и различными приложениями.

Исследователями из Мичиганского университета для упрощения доступа к сервисам каталогов X.500 был разработан протокол DAP. Стандарт CCITT X.500 описывает базовую структуру и функциональную модель службы каталогов и протокол доступа к каталогу (Directory Access Protocol, DAP)

В соответствии с X.500 служба каталогов состоит из *пяти основных элементов*:

1. Служба каталогов включает модель базовой структуры хранимой информации. Она состоит из записей (entries), атрибутов (attributes) и значений (values). Записи представляют элементы службы каталогов, соответствующие сетевым пользователям, разделяемым ресурсам, сетевым принтерам, приложениям и т.д. Каждая запись содержит атрибуты. Атрибуты представляют свойства, в соответствии с которыми служба каталогов классифицирует записи. Например, пользователь может иметь такие атрибуты, как фамилия, имя, отчество, адрес электронной почты, пользовательское имя и пароль. Каждый атрибут, в свою очередь, имеет конкретное значение. К примеру, атрибут «Фамилия» может иметь значение «Иванов».

Таким образом, служба каталогов представляет собой объектную ассоциативную базу данных, куда входят объекты – записи, характеризуемые парами атрибут-значение.

2. Информация службы каталогов иерархически организована в виде дерева, которое носит название информационного дерева каталога (Directory Information Tree, DIT). DIT хранит записи каталога и использует пространство имен для однозначной идентификации точного местоположения каждой записи в дереве. В качестве вершины дерева обычно используется двухбуквенный код страны. Конструкции, располагающиеся ниже, предоставляют все более конкретные сведения: например, название компании, далее названия отделов и т.д. Подобно пути к файлу, в файловой системе отличительное имя записи (Distinguished Name, DN) указывает путь к ней и однозначно определяет как саму запись, так и ее местоположение в DIT.

3. Служба каталогов содержит набор команд, которые можно использовать для управления записями. Команды выполняют такие операции, как: Чтение, Запись, Поиск, Сравнение, Удаление, Модификация.

4. Служба каталогов содержит систему для аутентификации пользователей, без регистрации в которой доступ к сервису каталогов не предоставляется. Для своей аутентификации пользователи могут задействовать различные методы, такие, как простая комбинация – имя, пароль.

5. Сервис каталогов может строиться по распределенной модели, чтобы клиенты (пользовательские агенты каталога) могли видеть данные, расположенные на различных серверах (системных агентах каталога), сведенными воедино.

Однако протокол DAP требует от клиента использования стека протоколов OSI, а его реализация отнимает много системных ресурсов. И в 1989 г. в Мичиганском университете был разработан протокол LDAP, подмножество протокола DAP. Из DAP были исключены редко используемые возможности и осуществлен перенос со стека протоколов OSI на стек TCP/IP. Текущая версия LDAP (версия 2) определенна IETF в документах RFC 1777 и RFC 1778. Другое новое приложение LDAP – для упрощения использования LDAP в Internet, RFC 1959 – описывает формат URL для прямого доступа к серверам LDAP посредством браузера Web. Этот формат вводит новый префикс протокола – ldap. И Netscape Communicator и Internet Explorer поддерживают LDAP URL. Netscape Communicator отображает результат запроса в той же манере, как и обычную страницу HTML, а Internet Explorer добавляет результат запроса в адресную книгу. Спецификация RFC 2307 описывает использование LDAP в качестве сетевого информационного сервиса, в котором операционные системы могут хранить информацию о пользователях, протоколах и проч.

Таким образом, LDAP предоставляет услуги сервиса каталогов и хранилища всей идентификационной и конфигурационной информации, а также средства контроля доступа к ней со стороны пользователей и приложений.

На современном этапе развития сетевых технологий вопросы управления сетевым оборудованием, сбором статистики его функционирования, а также вопросы определения и разделения доступа к различным информационным службам уже не являются отдельными задачами управления сетью. В данный момент наиболее важна возможность управления корпоративной информационной средой, определения присущей ей информационных потоков и их разделения в соответствии с выполняемыми ролями сотрудниками предприятия. Для решения такого рода задач необходима тесная интеграция систем управления, функционирующих на различных уровнях, в единую систему.

А.В. Козвонина

ОБ ИСПОЛЬЗОВАНИИ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ В ПРЕПОДАВАНИИ КУРСА «ТЕОРИЯ ВЕРОЯТНОСТЕЙ И МАТЕМАТИЧЕСКАЯ СТАТИСТИКА»

Теория вероятностей – это математическая наука, изучающая закономерности случайных явлений.

Первые работы, в которых зарождалась теория вероятностей, относятся еще к XVII веку и связаны с именами Д. Кардано, Х. Гюйенса, Б. Паскаля, П. Ферма, Я. Бернулли и др. Эти выдающиеся ученые занимались задачами, связанными с азартными играми, при этом они предвидели фундаментальную роль науки, изучающей случайные явления. Понятно, что в те давние времена задачи теории вероятностей решали с помощью элементарных арифметических и комбинаторных методов.

Следующие шаги науки стали возможными благодаря исследованиям А. Муавра, П. Лапласа, К. Гаусса, С. Пуассона и др. и повлекли за собой развитие аналитических методов теории вероятностей.

Дальнейшим успехам теория вероятностей обязана В.Я. Буняковскому, П.Л. Чебышеву, А.А. Маркову, А.М. Ляпунову и т.д.

Методы теории вероятностей широко применяются в различных областях естествознания и техники: в теории надежности, теории массового обслуживания, теории ошибок наблюдений, общей теории связи, теории автоматического управления, в теоретической физике, астрономии, геодезии и во многих других теоретических и прикладных науках. Теория вероятностей является основой математической и прикладной статистики.

В последние годы методы теории вероятностей все глубже проникают в различные области науки и техники, способствуя их развитию. В то же время все мощнее становится аппарат, посредством которого решаются задачи теории вероятностей. Еще в прошлом веке дополнительным средством решения различных задач стал компьютер. Но совсем недавно компьютер стали использовать при обучении теории вероятностей. В этой работе рассматривается возможность использования информационных технологий при преподавании дисциплины «Теория вероятностей и математическая статистика», доказывается рациональность и эффективность этого использования.

Курс теории вероятностей и математической статистики входит в программу большинства высших учебных заведений. Для преподавания данной дисциплины будущим учителям информатики возможно и необходимо не просто использовать новейшие информационные технологии в качестве программного обеспечения лабораторных практикумов, но и программировать в них. Таким образом, у студентов появится возможность взглянуть на информационные технологии «изнутри», в интеллектуальном и творческом ключе.

При любых особенностях проведения курса неоспорим тот факт, что в ходе изучения предмета учащиеся должны решить достаточное количество примеров и задач, поняв алгоритм и смысл их решения. Кроме того, студенты должны научиться видеть в конкретных научных, технических, житейских проблемах вопросы, задачи, допускающие решения методами теории вероятностей. Желательно также, чтобы студенты после изучения данного предмета смогли, например, правильно обрабатывать данные проведенного эксперимента (физического, технического, педагогического, психологического), интерпретировать результаты.

Курс теории вероятностей и математической статистики разделен на ряд тем, практически в каждой из которых можно выделить несколько классов задач. В большинстве случаев это числовые задачи, решаемые с помощью математических и статистических функций. При этом можно заметить, что для каждого конкретного типа задач достаточно легко составить четкий алгоритм их решения.

Исходя из этого, можно сделать вывод, что для упрощения работы с задачами можно прибегнуть к знаниям, умениям и навыкам, относящимся к разделу информатики «Алгоритмизация», а также к помощи ЭВМ. В то же время не рационально было бы использовать компьютер просто в качестве калькулятора и не воспользоваться другими его возможностями.

Учитывая то, что данные в задачах математической статистики представляют собой числовые ряды (и чаще всего многомерные), вполне логично, решая задачи теории математической статистики, работать с использованием табличного процессора Excel.

Microsoft Excel поставляется вместе с Microsoft Office и используется миллионами людей. По некоторым оценкам, примерно 90% от общего числа пользователей работают только со средствами рабочего листа Excel без применения языка программирования VBA (Visual Basic for Application), а значит, они используют лишь не более 10% от реальных возможностей Excel. Таким образом, большая часть средств Excel остается невостребованной просто потому, что многие не знают о наличии VBA или не умеют им пользоваться.

На самом же деле, VBA – относительно легкий язык программирования. Он прост в освоении и позволяет быстро получать ощутимые результаты - конструировать профессиональные приложения, решающие практически все задачи в среде Windows. При этом создание многих приложений с помощью VBA проще и быстрее, чем с другими языками программирования.

Visual Basic for Application – развитая система визуального программирования для создания прикладных программ, макросов. Макрос – это файл, в котором хранится программа последовательности действий, заданная пользователем. С помощью макросов можно автоматизировать многие типовые технологические этапы при работе с документами.

Так как студенты, обучающиеся по специальности «информатика», в учебном курсе «Технологии программирования» изучают среду программирования Visual Basic, а в курсе «Программное обеспечение ЭВМ» – электронные таблицы Excel, то в курсе «Теория вероятностей и математическая статистика» предоставляется возможность использования макропрограммирования в Excel путем создания макросов в виде программных модулей на языке Visual Basic for Applications.

В нашу задачу сейчас не входит подробное рассмотрение тем курса теории вероятностей и математической статистики. Рассматривается лишь то, при преподавании каких тем рационально воспользоваться возможностями Excel и VBA.

Требования к знаниям и умениям студентов (на момент начала изучения курса):

• хорошая подготовка по базовым математическим дисциплинам (главным образом по математическому анализу и линейной алгебре);

• умение работать в Excel, в частности владение такими приемами работы, как вставка формул и функций, в первую очередь математических и статистических:

- наличие понятия об относительных и абсолютных адресах;
- знание правил автоматического изменения относительных адресов, действующих при копировании и перемещении формул;
- знание назначения основных типовых команд табличного процессора;
- знание типов данных, хранимых в ячейках электронной таблицы, форматирования числовых данных в ячейках;
- знание графических возможностей табличных процессоров для представления данных (различные виды диаграмм) и умение их использовать;

• владение основами программирования на языке VBA в Excel:

- знание о существовании двух типов макросов: клавишных и языковых (программный модуль на языке VBA), умение их создавать;
- знание основных типов данных, операторов языка VBA.

Рассмотрим ряд примеров из курса теории вероятностей и математической статистики.

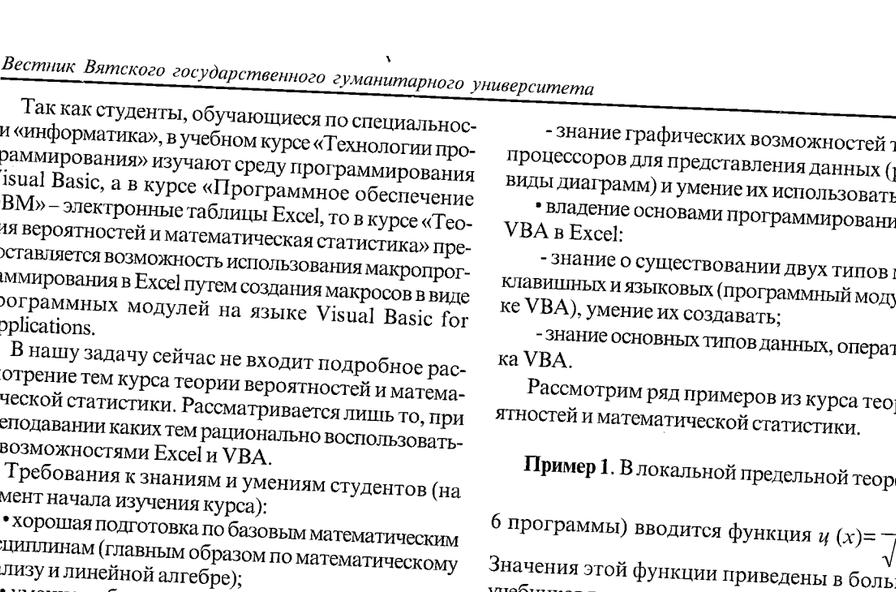
Пример 1. В локальной предельной теореме (тема 6 программы) вводится функция $\mu(x)=\frac{1}{\sqrt{2\pi}}e^{-\frac{x^2}{2}}$.

Значения этой функции приведены в большинстве учебников по теории вероятностей, но не во всех. Она используется при решении большого количества задач. Студент может сам составить такую таблицу значений. Очевидно, что высчитывать такое выражение вручную затруднительно. На помощь могут прийти электронные таблицы Excel. Нужно лишь уметь вставлять формулы в ячейку таблицы.

Например, B2=(EXP(–СТЕПЕНЬ($A2+B$1*0,01;2)/2))/КОРЕНЬ(2*ПИ())

($A2+B$1*0,01) – это x, $A2 – целая часть; B$1*0,01 – дробная часть.

Эта формула копируется на все необходимые значения (см. рис. 1).



	A	B	C	D	E	F	G	H	I
1		0	1	2	3	4	5	6	7
2	0	0,39894228	0,398922	0,398862	0,398763	0,398623	0,398444	0,398225	0,397966
3	0,1	0,396952547	0,396536	0,39608	0,395585	0,395052	0,394479	0,393868	0,393219
4	0,2	0,391042694	0,390242	0,389404	0,388529	0,387617	0,386668	0,385683	0,384663
5	0,3	0,381387815	0,380226	0,379031	0,377801	0,376537	0,37524	0,373911	0,372548
6	0,4	0,36827014	0,366782	0,365263	0,363714	0,362135	0,360527	0,35889	0,357225
7	0,5	0,352065327	0,350292	0,348493	0,346668	0,344818	0,342944	0,341046	0,339124
8	0,6	0,333224603	0,331215	0,329184	0,327133	0,325062	0,322972	0,320864	0,318737
9	0,7	0,312253933	0,31006	0,307851	0,305627	0,303389	0,301137	0,298872	0,296595
10	0,8	0,289691553	0,287369	0,285036	0,282694	0,280344	0,277985	0,275618	0,273244
11	0,9	0,26608525	0,263688	0,261286	0,258881	0,256471	0,254059	0,251644	0,249228
12	1	0,241970725	0,239551	0,237132	0,234714	0,232297	0,229882	0,22747	0,22506
13	1,1	0,217852177	0,215458	0,213069	0,210686	0,208308	0,205936	0,203571	0,201214
14	1,2	0,194186055	0,19186	0,189543	0,187235	0,184937	0,182649	0,180371	0,178104
15	1,3	0,171368592	0,169147	0,166937	0,16474	0,162555	0,160383	0,158225	0,15608

Рис. 1

Плюс в том, что, даже если под рукой нет учебника, всегда можно подсчитать значение функции, причем при любых x.

Еще одной возможностью Excel, которой можно воспользоваться при изучении теории вероятностей, является построение диаграмм. Графики делают различные функции наглядными.

Продолжим пример с функцией $\mu(x)=\frac{1}{\sqrt{2\pi}}e^{-\frac{x^2}{2}}$

X	Y
-5	1,49E-06
-4,5	1,6E-05
-4	0,000134
-3,5	0,000873
-3	0,004432
-2,5	0,017528
-2	0,053991
-1,5	0,129518
-1	0,241971
-0,5	0,352065
0	0,398942

X	Y
0,5	0,351361
1	0,241005
1,5	0,128744
2	0,053561
2,5	0,017354
3	0,004379
3,5	0,000861
4	0,000132
4,5	1,57E-05
5	1,46E-06

График функции

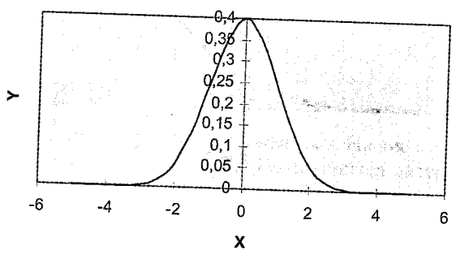


Рис. 2

По виду графика (см. рис. 2) можно сделать вывод, что эта функция распределена по нормальному закону распределения.

Пример 2 (тема «Параметрическое оценивание закона распределения»). Дано следующее распределение успеваемости 100 студентов-заочников, сдававших 4 экзамена:

Таблица 1

Число сданных экзаменов	0	1	2	3	4
Число студентов	1	1	3	35	60

Случайной величиной является число сданных экзаменов из четырех сдаваемых. Обозначим ее X. Установим закон распределения этой величины.

Построим сначала его непараметрическую оценку (оценивание закона распределения, не требующее предварительного выбора его модели и оценивания входящих в нее параметров).

Величина X – дискретная. Дискретный вариационный ряд, заданный столбцами 2 и 4 табл. 2, дает непараметрическую оценку закона распределения числа сданных экзаменов среди четырех сдаваемых.

Таблица 2

i	Число сданных экзаменов x_i	Число студентов m_i	Частость $\hat{p}_i=\frac{m_i}{n}$
1	2	3	4
1	0	1	0,01
2	1	1	0,01
3	2	3	0,03
4	3	35	0,35
5	4	60	0,60
Итого		n=100	1,00

$$\hat{p}_1=\frac{1}{100}=0,01$$

$$\hat{p}_2=\frac{1}{100}=0,01$$

$$\hat{p}_3=\frac{3}{100}=0,03$$

$$\hat{p}_4=\frac{35}{100}=0,35$$

$$\hat{p}_5=\frac{60}{100}=0,60$$

Далее необходимо сформулировать гипотезу о модели закона распределения случайной величины X – числе сданных экзаменов среди четырех сдаваемых. Процесс сдачи четырех экзаменов представим как четыре испытания, относительно которых сделаем следующие допущения:

• эти испытания независимы, т.е. вероятность сдачи любым студентом любого экзамена не зависит от того, будет или нет сдано любое количество других экзаменов;

• вероятность сдачи студентом любого отдельно взятого экзамена одна и та же и равна p, а вероятность «несдачи» равна 1–p.

Конечно, эти допущения могут вызвать некоторые сомнения, но они не будут противоречить результатам наблюдений. При этих допущениях экзамен можно считать испытанием Бернулли и число сданных экзаменов среди четырех сдаваемых будет иметь биномиальный закон распределения, т.е. вероятность того, что студент сдаст x экзаменов, равна.

Модель 1

$$P(X=x)=C_4^x p^x (1-p)^{4-x},\ x=0,1,2,3,4.$$

Найдем оценку параметра p, входящего в модель 1. Вспомним, что в условных испытаниях Бернулли состоятельной несмещенной и эффективной оценкой вероятности является частость. В рассматриваемом

примере p – вероятность того, что студент сдаст экзамен, поэтому частость этого события, учитывая, что имеются сведения об успеваемости 100 студентов, вычисляем следующим образом:

$$\hat{p} = \frac{\textit{число экзаменов, сданных 100 студентами}}{\textit{число экзаменов, сдаваемых 100 студентами}} = \frac{\sum_{i=1}^{5} x_i m_i}{4 \cdot 100} = \frac{0 \cdot 1 + 1 \cdot 1 + 2 \cdot 3 + 3 \cdot 35 + 4 \cdot 60}{100 \cdot 4} = 0{,}88.$$

Так как $\frac{\sum_{i=1}^{5} x_i m_i}{100} = \overline{X}$ – это среднее число экзаменов, сданных одним студентом, то $\hat{p}$ можно было бы определить и так:

$$\hat{p} = \frac{\textit{среднее число экзаменов, сданных одним студентом}}{\textit{число экзаменов, сдаваемых одним студентом}} = \frac{\overline{X}}{4} = 0{,}88.$$

Подставив в модель 1 вместо параметра p его оценку , получим параметрическую оценку неизвестного закона распределения числа сданных экзаменов, построенную в предположении, что допустима биномиальная модель

Модель 2

$$P(X=x) = C_4^x 0{,}88^x 0{,}12^{4-x};\ x = 0,\ 1,\ 2,\ 3,\ 4.$$

Теоретические вероятности $p_i^{теор}$ и частоты $m_i^{теор}$, вычисленные в предположении, что имеет место модель 2, содержатся в столбцах 5 и 6 табл. 3.

Нужно отметить, что составление табл. 3 занимает достаточно много усилий и времени. Сделать такую же таблицу с помощью Excel гораздо быстрее и проще.

Для составления табл. 4 использованы, в частности, следующие формулы:

B8=СУММ(C3:C7)

D3=$C3/$C$8

E3=ЧИСЛКОМБ(4;B3)*СТЕПЕНЬ(0,88;B3)*СТЕПЕНЬ(0,12;4B3)

F3=$C$8*E3

Поскольку различия между соответствующими числами столбцов 4 и 5 или между числами столбцов 3 и 6 небольшие, можно сделать предварительное заключение о приемлемости биномиальной модели. Это заключение подтверждается графиком, составленным в Excel (см. рис. 3), на котором кривая вероятностей $p_i^{теор}$ близка к кривой частостей $\hat{p}_i$.

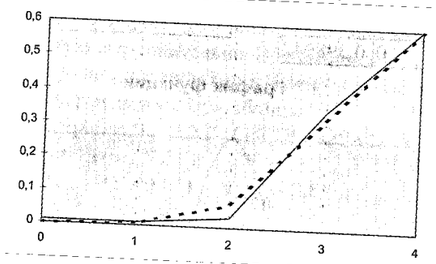


Рис. 3

Таблица 3

i	Число сданных экзаменов x_i	Число студентов m_i	Частота $\hat{p}_i = \frac{m_i}{n}$	$p_i^{теор} = C_4^{x_i} 0{,}88^{x_i} \cdot 0{,}12^{4-x_i}$	$m_i^{теор} = np_i^{теор}$
1	2	3	4	5	6
1	0	1	0,01	0,00021	0,021
2	1	1	0,01	0,00608	0,608
3	2	3	0,03	0,06691	6,691
4	3	35	0,35	0,32711	32,711
5	4	60	0,60	0,59969	59,969
Итого		n=100	1,00	1,0000	

$$p_1^{теор} = C_4^0 \cdot 0{,}88^0 \cdot 0{,}12^{4-0} = \frac{4!}{0!\cdot 4!} \cdot 1 \cdot 0{,}00021 = 0{,}00021$$

$$p_2^{теор} = C_4^1 \cdot 0{,}88^1 \cdot 0{,}12^{4-1} = \frac{4!}{3!\cdot 1!} \cdot 0{,}88 \cdot 0{,}00173 = 0{,}00608$$

$$p_3^{теор} = C_4^2 \cdot 0{,}88^2 \cdot 0{,}12^{4-2} = \frac{4!}{2!\cdot 2!} \cdot 0{,}7744 \cdot 0{,}0144 = 0{,}06691$$

$$p_4^{теор} = C_4^3 \cdot 0{,}88^3 \cdot 0{,}12^{4-3} = \frac{4!}{3!\cdot 1!} \cdot 0{,}68147 \cdot 0{,}12 = 0{,}32711$$

$$p_5^{теор} = C_4^4 \cdot 0{,}88^4 \cdot 0{,}12^{4-4} = \frac{4!}{0!\cdot 4!} \cdot 0{,}59969 \cdot 1 = 0{,}59969$$

$$m_1^{теор} = 100 \cdot 0{,}00021 = 0{,}021$$

$$m_2^{теор} = 100 \cdot 0{,}00608 = 0{,}608$$

$$m_3^{теор} = 100 \cdot 0{,}06691 = 6{,}691$$

$$m_4^{теор} = 100 \cdot 0{,}32711 = 32{,}711$$

$$m_5^{теор} = 100 \cdot 0{,}59969 = 59{,}969$$

Таблица 4

	A	B	C	D	E	F
1	i	Число сданных экзаменов	Число студентов	Частость	Теоретическая вероятность	Теоретическая частота
2	1	2	3	4	5	6
3	1	0	1	0,01	0,00020736	0,020736
4	2	1	1	0,01	0,00608256	0,608256
5	3	2	3	0,03	0,06690816	6,690816
6	4	3	35	0,35	0,32710656	32,710656
7	5	4	60	0,6	0,59969536	59,969536
8	Итого		100	1	1	100

Пример 3 (Схема Бернулли). Выделим типы задач, решаемых при изучении данной темы. Их три:

1. Нахождение вероятности того, что в результате n независимых испытаний событие A появится k раз. (При этом номера испытаний значения не имеют, в каждом испытании событие A может произойти с вероятностью p и не произойти с вероятностью q=1–p.)

2. Нахождение наивероятнейшего числа появления событий в схеме Бернулли, т.е. числа, при котором вероятность Pn(k) наибольшая.

$np\text{-}q \notin Z$

$k_0 = [np\text{-}q]+1$

$np\text{-}q \in Z$

$$k_0 = \begin{bmatrix} np\text{-}q \\ [np\text{-}q]+1 \end{bmatrix}$$

3. Задача о случайных блужданиях. На числовой прямой в координате с точкой 0 находится материальная точка. Каждую секунду она испытывает толчок вправо (с вероятностью p) или влево (с вероятностью $q=1-p$), делая при этом один шаг. Требуется найти вероятность того, что через n секунд материальная точка окажется в точке с координатой $m \in Z$.

$$k = \left| \frac{m+n}{2} \right|$$

$$P_n(k) = C_n^k \cdot p^k \cdot q^{n-k}$$

Для того, чтобы стало более понятно, рассмотрим несколько примеров задач каждого типа.

1-й тип

• Трижды бросается игральная кость. Какова вероятность того, что при этом ровно два раза выпадет 6 очков?

• Баскетболист делает 3 броска мячом в корзину. Вероятность попадания мяча в корзину при каждом броске равна 0,6. Найти вероятность того, что баскетболист ни разу не промажет.

• Известно, что вероятность рождения девочки равна 0,49, а мальчика 0,51. Найти вероятность того, что в семье, в которой 5 детей, 2 мальчика.

• Монета бросается 10 раз. Какова вероятность того, что герб выпадет при этом ровно 4 раза.

• В урне находятся 6 белых и 9 черных шаров. Из урны извлекают шар, фиксируют его цвет, после чего возвращают шар обратно в урну. Указанный опыт повторяется трижды. Какова вероятность того, что из трех вытащенных при этом шаров ровно два окажутся белыми?

2-й тип

• Найдите наиболее вероятное число выпадения 6 при 46 бросаниях кости.

• Вероятность того, что денежный приемник автомата при опускании монеты сработает неправильно, равна 0,03. Найдите наиболее вероятное число случаев правильной работы автомата, если опущено 150 монет.

• Если считать рождение мальчика и девочки равновероятными событиями, то найти наивероятнейшее число мальчиков в семье из 1) 6 детей; 2) 5 детей.

• Вероятность брака одного изделия на заводе равна 0,01. Чему равно наиболее вероятное количество бракованных изделий в упаковке из 150 штук.

3-й тип

• Два шахматиста условились сыграть 10 партий. Вероятность выигрыша каждой отдельной партии первым шахматистом p=2/3, а вторым q=1/3. Чему равна вероятность ничейного исхода?

• На экране компьютера появляется светящаяся точка, каждую секунду сдвигающаяся вправо или влево (равновероятно). В двух шагах вправо от точки ее появления располагается точка поглощения. Найти вероятность того, что светящаяся точка проживет на экране 1) 2 секунды; 2) 4 секунды.

Эти задачи можно решить на компьютере с помощью приложений Excel на языке VBA.

Разработаем макрос для первого типа задач. Для его создания в составе рабочей книги выполняется команда меню Сервис/Макрос с указанием способа создания макроса – Редактор Visual Basic.

Для нашего программного модуля необходима экранная форма (команда меню Insert/UserForm), где будут расположены все нужные для работы объекты:

текстовые окна, командные кнопки и поля ввода. Для этого конкретного макроса нам потребуются:

- 3 объекта TextBox – для ввода данных: n, k, p;
- 4 объекта Label: 1 – для описания типа задачи; 3 – для пояснения данных, которые должны быть введены в поля ввода;
- 2 объекта CommandButton: 1 – для запуска решения задачи по введенным данным; 1 – для завершения работы макроса.

Следующим этапом разработки является введение значений свойств объектов (свойства Caption объектов Label и CommandButton), а также названия самой формы. В результате форма принимает такой вид (см. рис. 4)

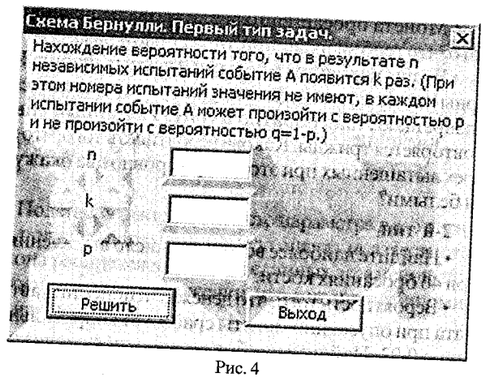


Рис. 4

Далее необходимо запрограммировать работу макроса – создать процедуры (Sub) обработки событий (нажатия на командные кнопки).

При нажатии на кнопку «Решить» (CommandButton1) необходимо:

- проверить корректность ввода (поля ввода непустые, в них введены числа, большие 0, n=k, p=1);
- если данные введены некорректно, то нужно очистить поля ввода и вывести на экран соответствующее сообщение;
- если данные введены правильно, то
  - считываются введенные данные;
  - подсчитывается результат, используя стандартные функции рабочего листа Excel;
  - выводится сообщение с результатом.

При нажатии на кнопку «Выход» (CommandButton2) происходит завершение работы макроса.

Ниже приведен полный окончательный текст программы.

```
Private Sub CommandButton1_Click() {Процедура
                                   обработки события нажатия
                                   на кнопку «Решить»}
f = (TextBox1.Text <> "") And (TextBox2.Text <> «») And (TextBox3.Text <> "") {Проверка корректности ввода}
If f Then f = IsNumeric(TextBox1.Text) And IsNumeric (TextBox2.Text) And IsNumeric(TextBox3.Text)
If f Then f=(TextBox1.Text>0) And (TextBox2.Text>0) And (TextBox3.Text>0)
If f Then f=(TextBox1.Text>=TextBox2.Text) And (TextBox3.Text<=1)
If Not f Then          {В случае некорректного ввода}
MsgBox («Некорректный ввод.») {Сообщение
                               о некорректном вводе}
TextBox1.Text = ""     {Очистка полей формы}
TextBox2.Text = ""
TextBox3.Text = ""
Else                   {В случае корректного ввода}
n=Val(TextBox1.Text)   {Считывание введенных
                        значений }
k=Val(TextBox2.Text)
p=TextBox3.Text
With Application.WorksheetFunction
rez=(.Fact(n)/(.Fact(k)*.Fact(n - k)))*p^k*(1–p)^(n–k)
                       {Решение по формуле}
End With
MsgBox (rez)           {Вывод ответа}
End If
End Sub
Private Sub CommandButton2_Click() {Процедура
                                   обработки события нажатия
                                   на кнопку «Выход»}
End                    {Выход}
End Sub
```

Далее приведены общий вид форм макросов (рис. 5 и рис. 6) и тексты программ для второго и третьего типов задач. Процесс создания этих программных модулей практически не отличается от процесса создания первого макроса и состоит из тех же этапов.

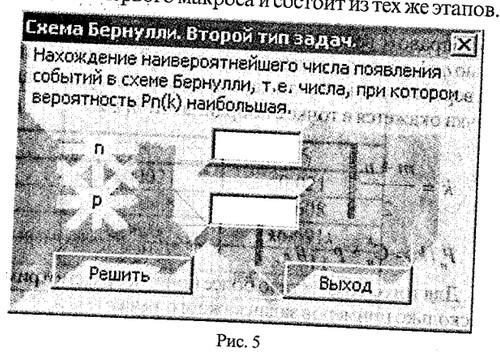


Рис. 5

Текст программного модуля для решения второго типа задач.

```
Private Sub CommandButton1_Click() {Процедура
                                   обработки события нажатия
                                   на кнопку «Решить»}
f=(TextBox1.Text <> "") And (TextBox2.Text <> "")
                       {Проверка корректности ввода}
If f Then f=IsNumeric(TextBox1.Text) And IsNumeric
(TextBox2.Text) And IsNumeric(TextBox3.Text)
If f Then f=(TextBox1.Text>0) And (TextBox2.Text>0)
If f Then f=(TextBox2.Text <=1)
If Not f Then          {В случае некорректного ввода}
MsgBox («Некорректный ввод.») {Сообщение о
                               некорректном вводе}
TextBox1.Text =""      {Очистка полей формы}
TextBox2.Text =""
Else                   {В случае корректного ввода}
n=Val(TextBox1.Text)   {Считывание
                        введенных значений}
p=TextBox2.Text
q=1–p
rez=n*p–q              {Высчитывание выражения rez}
t=Fix(rez)             {Отбрасывание дробной части числа}
If t<rez Then          {Если rez – не целое}
k=t+1
MsgBox (k)             {Вывод ответа}
Else                   {Если rez – целое}
k1=rez
k2=t+1
MsgBox (k1)            {Вывод ответов}
MsgBox (k2)
End If
End If
End Sub
Private Sub CommandButton2_Click() {Процедура
                                   обработки события нажатия
                                   на кнопку «Выход»}
End                    {Выход}
End Sub
```

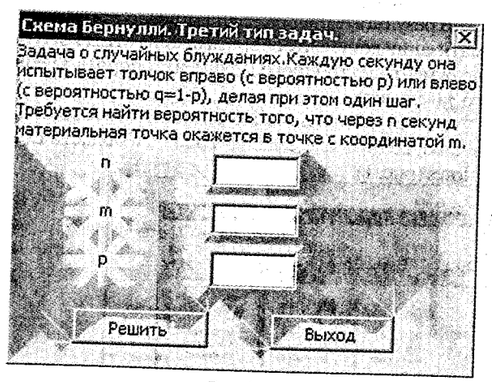


Рис. 6

Текст программного модуля для решения третьего типа задач.

```
Private Sub CommandButton1_Click() {Процедура
                                   обработки события нажатия
                                   на кнопку «Решить»}
f=(TextBox1.Text <> "") And (TextBox2.Text <> "") And (TextBox3.Text <> "")
                       {Проверка
                        корректности ввода}
If f Then f=IsNumeric(TextBox1.Text) And IsNumeric
(TextBox2.Text) And IsNumeric(TextBox3.Text)
If f Then f=(TextBox1.Text>0) And (TextBox2.Text>0)
And (TextBox3.Text>0)
If f Then f=(TextBox1.Text>=TextBox2.Text) And
(TextBox3.Text<=1)
If Not f Then          {В случае некорректного ввода}
MsgBox («Некорректный ввод.») {Сообщение
                               о некорректном вводе}
TextBox1.Text = ""     {Очистка полей формы}
TextBox2.Text = ""
TextBox3.Text = ""
Else                   {В случае корректного ввода}
n=Val(TextBox1.Text)   {Считывание
                        введенных значений }
m=Val(TextBox2.Text)
p=TextBox3.Text
k=(n + m)/2
With Application.WorksheetFunction
rez=(.Fact(n)/(.Fact(k)*.Fact(n–k)))*p^k*(1–p)^(n–k)
                       {Решение по формуле}
End With
MsgBox (rez)           {Вывод ответа}
End If
End Sub
Private Sub CommandButton2_Click() {Процедура
                                   обработки события нажатия
                                   на кнопку «Выход»}
End {Выход}
End Sub
```

Можно заметить, что макросы не сильно отличаются друг от друга, и, продумав один, очень легко составить другие. Имея макрос, достаточно лишь выделить в задаче исходные данные, ввести их на форму и нажать на кнопку «Решить». Все остальное сделает программа.

Составление для решения задачи макроса с полными комментариями весьма эффективно. Во-первых, при написании требуется полностью владеть всеми подробностями решения, формулами, алгоритмом, диапазоном значений, которые могут принимать используемые переменные, а следовательно, прежде чем писать программу нужно тщательно изучить ее. Во-вторых, взглянув на хорошо составленный макрос, легко понять принципы решения задачи. В связи с этим на форме программы целесообразно сделать метку с описанием типа задач, решаемых с помощью данного макроса. В-третьих, имея макрос, не нужно тратить время на элементарные математические вычисления.

Использование Excel для решения конкретных задач курса теории вероятностей и математической статистики – это реальная возможность интеграции математики и информатики с целью повышения эффективности преподавания курса. Следовательно, необходимо дальнейшее изучение возможностей применения информационных технологий при преподавании данного предмета.

Литература

1. *Боровков А.А.* Теория вероятностей. М., Наука, 1986.
2. *Вентцель Е.С., Овчаров Л.А.* Теория вероятностей. М., 1973.
3. *Гнеденко Б.В.* Курс теории вероятностей. М., 1954.
4. *Гмурман В.Е.* Теория вероятностей и математическая статистика: Учеб. пособие для вузов. М.: Высш. шк., 1977.
5. *Колмогоров А.Н., Журбенко И.Г., Прохоров А.В.* Введение в теорию вероятностей. М.: Наука, 1982. (Б-чка «Квант». Вып. 23).
6. *Гнеденко Б.В., Хинчин А.Я.* Элементарное введение в теорию вероятностей. М.: Наука, 1982.
7. Информатика: Учебник / Под ред. Н.В.Макаровой. М.: Финансы и статистика, 1999.
8. *Колемаев В.А., Калинина В.Н.* Теория вероятностей и математическая статистика: Учебник / Под ред. В.А. Колемаева. М.: ИНФРА–М, 1999. (Сер. «Высшее образование»).
9. *Калинина В.Н., Панкин В.Ф.* Математическая статистика: Учеб. для студ. сред. спец. учеб. заведений. М.: Высш. шк., 2001.
10. Программы педагогических институтов. Сборник № 6. М.: Просвещение, 1984.
11. *Солодовников А.С.* Теория вероятностей. М.: Просвещение, 1983.
12. *Чистяков В.П.* Курс теории вероятностей. М., 1982.

Е.Ю. Калганников

СИСТЕМА АВТОМАТИЗАЦИИ ДОКУМЕНТООБОРОТА ШКОЛЫ

В настоящее время российские школы встречаются с большим количеством новых проблем, которые предъявляют все возрастающие требования к системе внутришкольного управления, а следовательно, и к работе с управленческими документами.

XXI век – век информационных технологий - трудно представить без повсеместного использования компьютеров, в том числе и в практической деятельности руководства школы, где важное место занимает работа с документами. Жизненно необходимой стала система автоматизации документооборота школы, которая обеспечила бы всесторонний и оптимальный уровень управления и помогла бы директору и его заместителям уменьшить непроизводительную трату времени, повысить эффективность управленческого труда.

В научно-исследовательской работе автор поставил задачу рассмотреть возможность внедрения в педагогический процесс школы современных средств работы с информацией. Предлагаемый вариант автоматизированной системы базируется на программном продукте Lotus Notes фирмы Lotus Development Corporation, являющейся подразделением корпорации IBM.

Lotus Notes уже активно используется во многих учебных заведениях как за рубежом, так и в России. Среди российских учебных заведений можно назвать Новгородский университет им. Я. Мудрого, Московский международный университет, Московский государственный университет им. Э. Баумана и многие другие. Решение о выборе технологий Lotus является выгодным, безопасным и надежным решением, в том числе по причинам того, что Lotus является частью IBM и в течение многих лет предлагает на рынке проверенные, передовые технологии.

В результате была создана модель системы автоматизации документооборота школы МЕДУЗА (Менеджмент электронного Документооборота Учебного Заведения)

Структурно система состоит из девяти баз данных, значки которых помещены в рабочей области Lotus Notes (рис. 1).

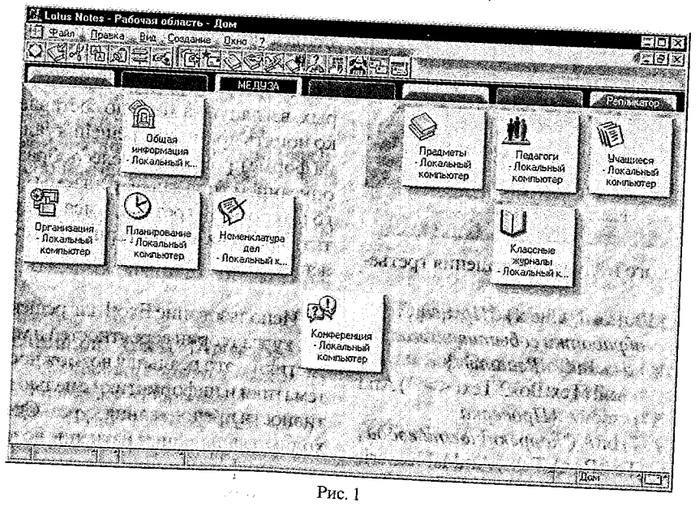


Рис. 1

Обмен информацией между ними производится согласно схеме (рис. 2).

Ниже приводится их краткое описание.

Общая информация. Состоит из одного документа «Общая информация о школе». Содержит разнообразные, редко меняющиеся сведения о школе (рис. 3).

Организация работы. База данных включает в себя комплект основных форм об организации и распорядке работы.

Планирование. Предназначена для хранения документов, необходимых для организации планирования работы на учебный год. Перечень отчетных документов и их вид взяты из «Журнала планирования и учета работы директора» [2] (рис. 4).

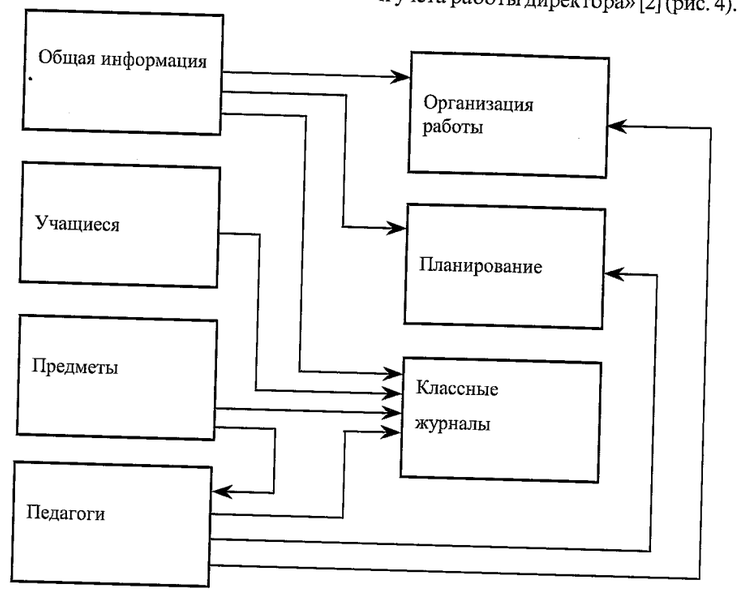


Рис. 2

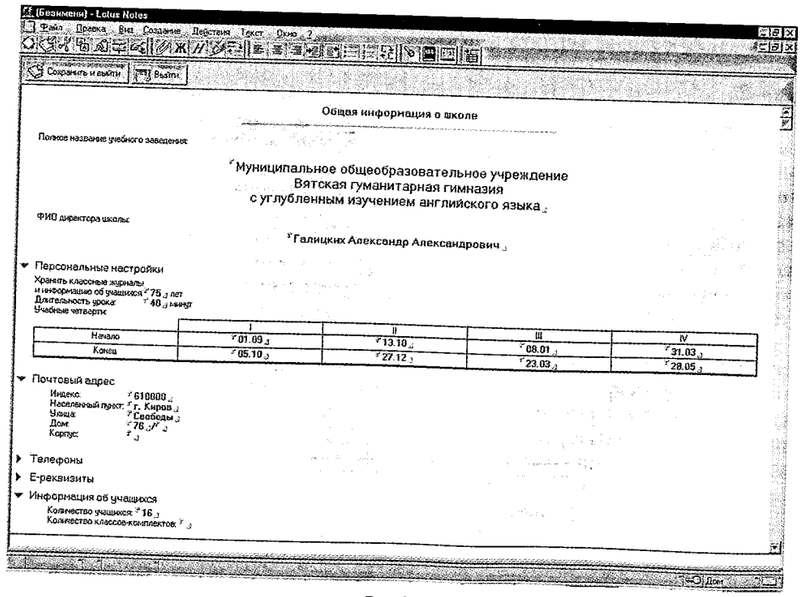


Рис. 3

Номенклатура дел. Перечень типовых документов, задействованных в школьном делопроизводстве. Все дела сгруппированы по тематическим разделам в соответствии с унифицированной классификацией [3, 4].

Предметы. Список предметов, преподаваемых в школе.

Педагоги. Каждый документ содержит сведения об отдельном педагоге.

Учащиеся. Включает одну форму, с помощью которой в базе данных создаются документы с информацией об учащихся (рис. 5).

Классные журналы. База данных представляет собой набор документов, формы которых по виду приближены к страницам обычного школьного журнала.

Разработаны формы для создания отчетов по успеваемости (по каждому ученику по всем предметам, по всем ученикам класса по отдельным предметам), четвертной сводной таблицы посещаемости по отдельным классам и таблицы с итоговыми оценками учащегося.

Конференция. База данных используется для коллективного обсуждения различных проблем, вопро-

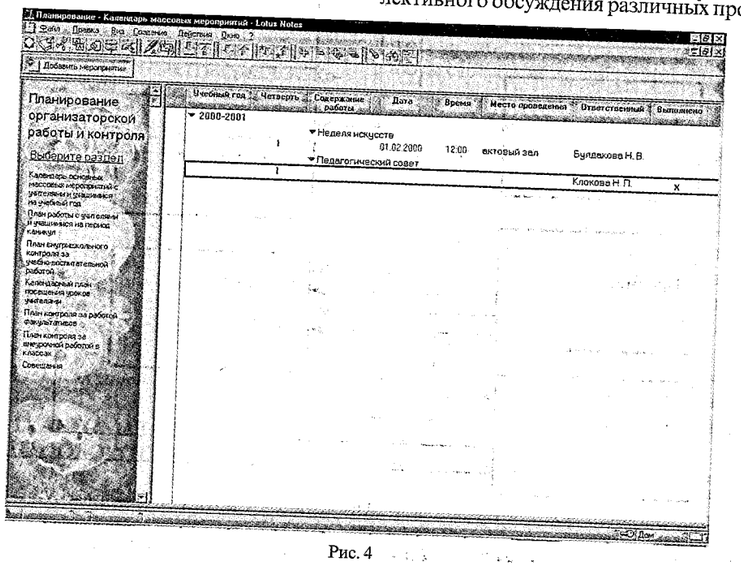


Рис. 4

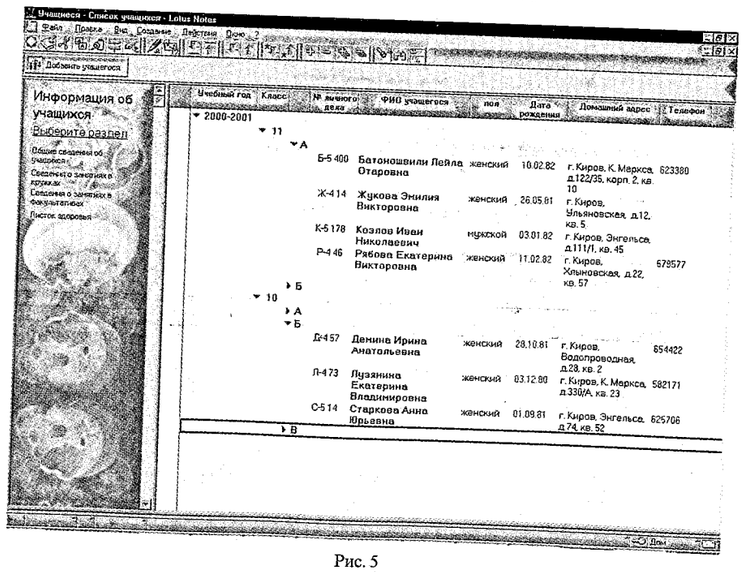


Рис. 5

сов и идей учащимися и педагогами учебного заведения. Базу данных можно представить как неформальное совещание, участники которого могут обмениваться мнениями и замечаниями. Однако участникам нет необходимости собираться в одном месте, пользователи принимают в ней участие в удобное для них время.

Так как вид электронных форм почти копирует стандартные документы, процесс их заполнения облегчается и становится доступным даже неподготовленному пользователю.

Одним из главных преимуществ разработанной модели является возможность работы в сети, которая обеспечивается встроенными функциями Lotus Notes. В сетевом варианте реализуются такие возможности, например, как синхронизация данных в копиях баз данных независимо от их географического местоположения по расписанию на базе технологии клиент-сервер; она может проводиться не только на уровне файлов и документов, но и отдельных полей.

При помощи средств электронной почты возможно обсуждение особо важных проблем, требующее документального подтверждения о прочтении сообщения (письма могут быть заверены электронной подписью, которая точно идентифицирует отправителя, а сами сообщения могут быть зашифрованы). Общие вопросы обсуждаются при помощи системы конференций.

Представленная модель системы обладает широкими возможностями подстройки не только под определенную школу, но и конкретных пользователей. Имеются возможности разграничения прав доступа пользователей к информационным ресурсам как на уровне баз, так и на уровне форм и представлений. Многоуровневый механизм разграничения прав пользователей реализует задачу защиты информации от несанкционированного вмешательства, предоставляет доступ только к той части информации и только на том уровне (создание-чтение-редактирование), который определяется для пользователя администратором системы.

Имеется возможность масштабирования системы – от обслуживания администрации конкретного учебного заведения до создания региональной информационной системы, что позволяет создавать/внедрять систему поэтапно с минимальными капитальными затратами и быстрой окупаемостью вложенных средств.

Кроме того, представленная модель системы автоматизации документооборота может являться частью интегрированной системы управления педагогическим процессом, которая позволит автоматизировать все аспекты: на уровне школы – вплоть до создания компьютерных обучающих курсов по всем предметам и непосредственного взаимодействия между учеником и учителем в виртуальном классе; на уровне города, области, региона – до создания единой автоматизированной информационной системы образования.

Д.В. Бояринцев, А.А. Московкин, С.М. Окулов

О МЕТОДИКЕ ТЕСТИРОВАНИЯ ПРОГРАММНЫХ РЕШЕНИЙ

Программирование – теоретическая и практическая деятельность по обеспечению программного управления обработкой данных, включающая создание программ, а также выбор структуры и кодирования данных. Обучение программированию – сверхсложная задача, методика которой не отработана полностью. Ее важнейшей составляющей является умение тестировать программные решения.

Практические рекомендации по написанию программ основаны на технологии нисходящего проектирования. О строгом математическом доказательстве правильности работы обычно не может быть и речи при достаточно сложной программе. Успешная трансляция не есть признак работающей программы, а правильность работы на некотором наборе тестов еще не означает, что программа всегда будет работать правильно, – следует помнить, что различных комбинаций входных данных бывает, как правило, бесконечно (или «практически» бесконечно) много. Основная идея – не пытаться программировать сразу. Пошаговая детализация автоматически заставляет формировать понятную структуру программы. При этом требуется отслеживать правильность детализации, создавая набор контрольных точек и «срезу» данных в них. Контрольные точки обычно проходятся при любых начальных данных (как точка сочленения в графе). Выбор представления данных – один из фундаментальных аспектов разработки программы. Н. Вирт так определял критерии этого действия: *естественность* – структуры данных обязаны «вытекать» из специфики задачи; *привычность* – структуры данных выбираются так, чтобы программирование приближалось к утверждению «как говорю, так и пишу программу, ничего лишнего»; *оптимальность* – структуры данных выбираются так, чтобы была возможность построения эффективного алгоритма. Например, решая задачу о построении каркаса минимального веса в графе естественно выбрать список ребер, а не матрицу инциденций или матрицу смежности.

Примеры плохого программирования. Приведем ряд примеров, которые не соответствуют технологии структурного написания программ.

1. Следует придерживаться правила о том, что любой фрагмент программного кода (логики) должен иметь одну точку входа и одну точку выхода (пусть даже в ущерб эффективности использования памяти). Ниже приводится пример того, как небольшой фрагмент имеет две точки выхода.

```
While <условие 1> Do Begin
.....
If <условие 2> Then Exit;
.....
End;
```

Использование конструкции бесконечного цикла только усугубляет ситуацию.

```
While True Do Begin
.....
If <условие 2> Then Exit;
.....
End;
```

2. Искусственные приемы, как, например, приведенный ниже случай «насильного» выхода из циклической конструкции типа *For*, приводят, при внесении изменений в программу, к многочисленным и трудно устранимым ошибкам.

```
For i:=1 To N Begin
....
If <условие> Then i:=N+1;
...
End;
```

3. Увлечение параметрами при работе с процедурами и функциями является «болезнью роста» при освоении технологии нисходящего проектирования программ.

```
Type BArray=Array[1..N] Of Byte;

Procedure Swap(Var B:BArray;i,k:Byte);
Var x:Byte;
Begin
 x:=B[i];B[i]:=B[k];B[k]:=x;
End;
```

и вызов *Swap(B,i,k)* вместо следующего просто текста процедуры:

```
Procedure Swap(Var y,z:Byte);
Var x:Byte;
Begin
 x:=y;y:=z;z:=x;
End;
```

и вызова *Swap(B[i],B[k])*.

4. Один профессиональный программист сказал в частной беседе о том, что для него критерием подготовленности выпускников высшего учебного заведения к работе является понимание (глубокое) рекурсии и знание динамических структур данных.

```
Procedure Generate;{*t - глобальная переменная*}
Begin
 ....
 t:=t+1;
 Generate;
 t:=t-1;
 ...
End;
```

Логичнее выглядел бы следующий фрагмент (вопрос об экономии места в стеке адресов возврата не ставится, оно есть и точка).

```
Procedure Rec(t:Byte);
Begin
 ...
 Rec(t+1);
 ...
 End;
```

5. Очень трудно убедить школьников в том, что если переменная используется только в процедуре, то она и должна определяться в этой процедуре, пусть и с дублированием программного кода. Только через большое количество занятий, после многочисленных ошибок приходит понимание факта. Типичный пример такого написания программ приводится ниже по тексту.

```
Program
 Var i,j:Integer;
 Procedure A;
  Begin
   <Работа с переменными i, j>;
  End;
 Procedure B;
  Begin
   <Работа с переменными i, j>;
  End;
....
{**}
Begin
 <Работа с переменными i, j>
End.
```

Временная сложность (t). Обычно на время решения задачи накладываются определенные ограничения. Размерность задачи опишем величиной *N*. Временная сложность алгоритма является линейной при $t \sim O(N)$, полиномиальной при $t \sim O(N^q)$, где q равно обычно 2 или 3, и экспоненциальной при $t \sim O(q^N)$. Эффективными являются алгоритмы первых двух типов. Обычно подсказка о том, какой существует алгоритм решения, содержится в формулировке задачи (имеется в виду олимпиадный вариант). Если дано *N*, большее 50, то следует искать быстрый алгоритм, в противном случае – сосредоточиться на эвристиках, сокращающих традиционный перебор с возвратом.

Ответим на вопрос, как оценить время работы программы в среде Турбо Паскаль? Обычно используют тот факт, что к ячейке с абсолютным адресом *$40:$6C* (назовем ее *Timer*) один раз в 1/18.2 секунды аппаратно прибавляется единица (или один раз в каждые 55 миллисекунд). Допустим, что программа должна работать 5 секунд (зафиксировали время в ячейке с именем *TimeTest*). В начале работы программы значение из *$40:$6C* запомнили в некоторой переменной, например *TimeOld*. После этой подготовительной работы в расчетной части программы осталось проверять условие *Timer-TimeOld>18.2*TimeTest*. Если оно выполняется, то время решения задачи истекло. Так как обычно перед завершением задачи требуется записать в выходной файл результат (хотя бы промежуточный, например, наилучшее из найденных решений в переборной задаче), а для этого требуется время, то условие пишут с учетом этого факта *Timer-TimeOld>18.2*(TimeTest-0.5)*. Ниже приведен текст фрагмента, в котором считается количество прибавления единицы к ячейке в зависимости от времени, выделенного на работу программы.


```
Program TimeCheck;
 Uses Crt;
 Const N=5;
 Var Timer:LongInt Absolute $40:$6C;
  TimeOld,TimeTest,L,i:LongInt;
   pp:Boolean;
Begin
 ClrScr;
 For i:=1 To N Do Begin
 TimeOld:=Timer;{Запоминаем значение таймера.}
 pp:=True;
 TimeTest:=i;{*Время решения задачи.*}
 L:=0;
 While pp Do Begin {*Пока не истекло время, отведенное на решение задачи.*}
 Inc(L);{*Прибавляем единицу к счетчику.*}
 If Timer-TimeOld>18.2*(TimeTest{-0.5}) Then Begin
  pp:=False;
 WriteLn(L);
 End;
 End;
End.
```

Приемы контроля правильности работы программы. Сформулируем ряд приемов, позволяющих избежать простых ошибок в разрабатываемых программах. (1) Обычно формат входных данных задачи строго определен. Если в формулировке требуется контролировать корректность ввода данных, то следует выделить эту часть логики в отдельную процедуру (ы). Однако в последние годы от требования проверки входных данных отказались (видимо, из-за простоты) – они считаются корректными. Несмотря на это, после ввода данных их следует сразу вывести и проверить, что позволит избежать ошибок этого типа (на них обычно не обращают внимания и только после значительных усилий возникает вопрос: «А правильно ли в программе вводятся данные»). (2) Иногда полезно отслеживать (не в пошаговом режиме) трассу работы определенной части программы. Так, при входе в процедуру можно выводить ее имя и значения входных параметров или выводить промежуточные данные в определенных точках программы. К этому же приему относится и введение дополнительных счетчиков, контролирующих прохождение различных ветвей логики. (3) Синтаксические ошибки устраняются обычно моментально. Для поиска, если так можно выразиться, логических ошибок первого типа в среде Турбо Паскаль рекомендуется вести отладку с включенными директивами. Начало программы после нажатия *Ctrl+O+O* выглядит следующим образом:

```
{$A+,B-,D+,E+,F-,G-,I+,L+,N-,O-,P-,Q-,R-,S+,T-,V+,X+}
{$M 16384,0,655360}.
```

Изменение, например, *R*– на *R*+ позволяет контролировать выход индексов за пределы массива. Логические ошибки следующего уровня сложности связаны обычно с тем, что задача была понята неверно, т.е. решена не та задача, которая была поставлена. В этом случае лучше всего «прогнать» решение на примере, просчитанном «вручную» (обычно такой пример приводится в формулировке задачи). Неверным может оказаться замысел алгоритма. Правильные решения получаются не для всех исходных данных. Это, вероятно, самый сложный тип ошибок и его рассмотрению посвящен следующий параграф данной главы.

О тестировании программ. Тест является совокупностью исходных данных для проверки работоспособности программы. Одного теста обычно недостаточно для проверки каждой ветви программы, и говорят о системе тестов. Кроме того, с помощью тестов проверяются количественные ограничения на исходные данные и временные ограничения. Исходя из этого разумно формировать систему тестов, например, следующим образом: первый тест должен быть максимально простым; следующие тесты должны проверять вырожденные случаи (например, при поиске решений квадратного уравнения проверяются вырожденные случаи); обязательно следует проверять граничные случаи (результат описан как тип *Integer*, а значение превосходит 32767); предельные тесты проверяют решение при наибольших значениях входных параметров задачи, и, разумеется, требуется постоянный контроль над временем выполнения программы (при превышении ограничений по времени выбран не эффективный алгоритм решения).

Автоматизация проверки. Обычно в задаче имена входного и выходного файлов определены. Постоянное изменение имени не очень рационально. Рассмотрим, как выходить из данной ситуации. Во-первых, создается директорий для работы с программой и он является текущим в системе программирования Турбо Паскаль. Программа ищет входной файл в текущем директории и записывает в него выходной файл. Пусть есть следующая «варварская» программа – из файла считываются два числа, их сумма записывается в выходной файл. Кроме того, в программу вставлены не очень понятные операторы. Однако они написаны, и с ними проводится отладка.

```
Program add;
 Var a,b : Integer;
 Begin
 Assign(input,'input.txt'); Reset(input);
 ReadLn(a,b);
 Close(input);
 While a=10 Do;
 If a = 20 Then b := Round(a/0);
 Assign(output,'output.txt'); Rewrite(output);
 WriteLn(a+b);
 Close(output);
End.
```

Создаем систему тестов. В текущий директорий записываем файлы с каждым тестом (имена *i.tst*, где *i* – номер теста) и результирующие файлы (имена *i.ans*, где *i* – номер теста).

Имя входного файла	Тест	Имя выходного файла	Результат
1.tst	2 5	*1.ans*	7
2.tst	45 45	*2.ans*	91
3.tst	3 5	*3.ans*	8
4.tst	10 10	*4.ans*	20
5.tst	20 0	*5.ans*	20

Переименование входных файлов в файл с именем *input.txt* и пятикратный запуск программы при очевидных результатах утомительны и не очень удобны. Требуется минимальная автоматизация процесса тестирования программ. Ниже приведен текст программы. Следует набрать и сохранить его в файле *Checker.dpr* (текущий директорий), а затем откомпилировать, используя среду *Delphi*. После этого запустите программу *Checker.exe*. Ее входными параметрами (они задаются в командной строке) являются: имя исполняемого файла (рассматриваемый выше по тексту пример следует откомпилировать в текущий директорий, например, с именем *my.exe*); количество тестов; ограничение по времени (в секундах). Вид командной строки – *Checker.exe my.exe* 5 2 (5 тестов, 2 секунды на тест). Результат очевиден. Итак, имея программу *Checker*, мы значительно упрощаем процесс отладки решений.

```
program Checker;
{$apptype console} – {*Консольное приложение*}
uses Windows,SysUtils,Classes;
var SInf :StartUpInfo; {*Установки для функции
                         CreateProcess*}
  PInf :Process_Information;{*Информация
                         о созданном процессе*}
  TickCount :Longint; {*Количество миллисекунд*}
  YourAnswer :TStringList; {*Ответ тестируемой
                         программы*}
  RightAnswer :TStringList; {*Правильный ответ*}
  FileName :string;{*Имя тестируемой программы*}
  FeedBack : string;{*Результат тестирования.*}
  TestCount : integer;{*Количество тестов*}
  TimeLimit : integer;{*Ограничение по времени*}
  IsTimeLimit : boolean; {*Индикатор окончания
                         лимита времени*}
  ExitCode : cardinal; {*Код завершения
                         тестируемой программы*}
  i :integer;
begin
  FileName := ParamStr(1); {*Первый параметр –
                         имя файла*}
  TestCount:=StrToInt(ParamStr(2));{*Второй–
                         количество тестов*}
  TimeLimit:=StrToInt(ParamStr(3));{*Третий–
                         ограничение по времени*}
  for i:=1 to TestCount do begin
    RenameFile(IntToStr(i)+'.tst','input.txt');{*Переименовать
                         входной тест в input.txt.*}
    FillChar(SInf,SizeOf(SInf),0);
    SInf.cb :=SizeOf(PInf);{*Заполнить структуру*}
    SInf.dwFlags :=STARTF_USESHOWWINDOW;
  StartUpInfo
    SInf.wShowWindow:=sw_Hide;{*Сделать
                         окно процесса невидимым*}
    CreateProcess(PChar(FileName),nil,nil,nil,false,{*Запуск
                         тестируемой программы*}
    CREATE_NEW_CONSOLE or NORMAL_
  PRIORITY_CLASS,nil,
    nil,SInf,PInf);
    TickCount := GetTickCount;{*Запомнить время*}
    IsTimeLimit := true;
    While ((TickCount+TimeLimit*1000)>=GetTickCount)
  and IsTimeLimit do
      if WaitForSingleObject(PInf.hProcess,0) <>
  WAIT_TIMEOUT then{*Ждать, пока программа за-
  кончит работу или истечет тайм-аут*}
      IsTimeLimit := false;
    if IsTimeLimit then begin {*Если истек тайм-аут,
  закончить работу программы*}
    TerminateProcess(PInf.hProcess,0);
    FeedBack := "Time limit exceeded!";{*Превышение
  лимита времени*}
    end
    else begin
    GetExitCodeProcess(PInf.hProcess,ExitCode);{*Получить
  код завершения программы*}
    if ExitCode <> 0 then FeedBack := "Run-time error!"
                         {*Если код завершения не равен нулю,
                         то ошибка времени выполнения*}
    else begin
    YourAnswer := TStringList.Create;
    RightAnswer := TStringList.Create;
    YourAnswer.LoadFromFile("output.txt");
                         {*Загрузить файл с ответами
                         тестируемой программы*}
    RightAnswer.LoadFromFile(IntToStr(i)+'.ans');
                         {*Загрузить файл
                         с правильными ответами*}
    if YourAnswer.Text <> RightAnswer.Text then
  Feedback := "Wrong answer..."
                         {*Если они не совпадают,
                         то тест не пройден*}
    else FeedBack := "Ok."; {*Иначе все в порядке*}
    YourAnswer.Free;
    RightAnswer.Free;
    end;
  end;
  RenameFile("input.txt",IntToStr(i)+'.tst');
                         {*Переименовать входной тест*}
  Writeln("Test"+IntToStr(i)+": "+FeedBack);
                         {*Выдать результат теста*}
  end;
  DeleteFile("output.txt"); {*Удалить
                         ненужный файл*}
  writeln("Completed..."); {*Выдать сообщение
                         об окончании*}
  Readln;
end.
```


Рассмотрим следующую задачу. Даны два квадратных клеточных поля. Размер клетки 1*1. Клетки поля закрашены в белый или черный цвет. Совпадающая часть квадратов – это множество клеток, имеющих одинаковый цвет. Она разбивается на какое-то количество связных областей. Найти площадь наибольшей связной области совпадающей части квадратов.

Пример.

Описание квадратов хранится в текстовых файлах. Символ «1» во входных файлах соответствует черной клетке, символ «0» – белой. В выходном файле должно быть одно целое число, равное площади общей части.

Пример.
Photo1.txt
4
0100
1000
1110
1101
Photo2.txt
4
0111
1011
0010
0001
Result.txt
4

Ограничения. Размер поля *N* в интервале от 2 до 1000. Время работы – не более 5 секунд.

Анализ и решение задачи. В идейном плане задача очень простая. Ее суть сводится к следующему:

If Клетка[i,j] 1-го квадрата=Клетке[i,j] 2 квадрата
Then Клетке[i,j] результата: = 1
Else Клетка[i,j] результата: = 0,

а затем найти связную область с наибольшей площадью.

Проработав этот учебник до данной страницы, программное решение, представленное ниже по тексту, пишется за считанные минуты.

Структуры данных.

```
Const
  Size =100;          {*Размер квадрата*}
  Dx: Array[1..4] Of Integer = (0, -1, 0, 1);
                      {*Вспомогательные массивы Dx, Dy
                      используются в волновом алгоритме
                      обхода связной области*}
  Dy: Array[1..4] Of Integer = (-1, 0, 1, 0);
Type
  PhotoMatch = Array [1..Size, 1..Size] Of Byte;
                      {*Для хранения результирующего квадрата*}
  Helper = Array [1..Size*Size] Of Integer;
                      {*Вспомогательный тип для хранения
                      очереди при обходе в ширину*}
Var
  PhotoM : PhotoMatch;{*Результирующий квадрат*}
  Ox,Oy: Helper;
  Result, TempRes,N: Integer;
```

Напомним логику волнового алгоритма (п. 3.2.).

```
Function Get(Const k,p : Byte): Integer;
                      {*Обход совпавшей части квадратов*}
  Var i, dn, nx, ny, up : Integer;
  Begin
   Ox[1] := k; Oy[1] := p; dn := 1 ; up := 1;
                      {*Заносим координаты первой клетки
                       в очередь*}
   PhotoM [k,p]:=0;
   Repeat
    For i:=1 To 4 Do Begin
     nx := Ox [dn] + dx [i]; ny := Oy [dn] + dy [i];
     If (nx>0) And (ny>0) And (nx<=N) And (ny<=N) And
  (PhotoM[nx,ny]=1)
      Then Begin{*Заносим в очередь*}
       PhotoM[nx, ny] := 0; Inc(up); Ox [up]:= nx;
  Oy [up]:= ny;
      End;
    End;
    Inc(dn);{*Переходим к очередному элементу очере-
  ди*}
    Until dn>up;{*Пока очередь не пуста*}
    Get := up;{*Количество элементов в очереди равно
  площади общей части*}
  End;
```

Процедура ввода данных.

```
Procedure Init;
  Var photo1, photo2: Text;
    char1, char2 : Char;
    x,y : Integer;
  Begin
    Assign (photo1, "photo1.txt"); Reset(photo1);
                      {*Входные файлы*}
    Assign (photo2, "photo2.txt"); Reset(photo2);
    ReadLn(photo1,N);ReadLn(photo2,N);
    For y := 1 To N Do Begin
     For x := 1 To N Do Begin
      Read(photo1, char1); Read(photo2, char2);
      PhotoM [x,y]:=Ord(char1=char2);
                      {*Если «закраска» совпадает, то ...*}
     End;
    ReadLn(photo1); ReadLn(photo2);
   End;
  End;
```

Главная процедура.

```
Procedure Solve;
  Var i, j,t :Integer;
```

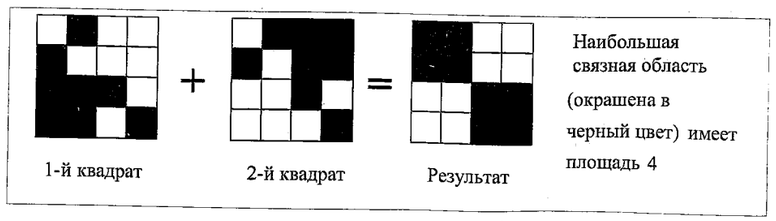


Рис. 1

```
Begin
 Result:=0;
 For i:= 1 To N Do
  For j:= 1 To N Do
  If (PhotoM[i,j]=1) Then Begin
  t:=Get(i,j);
   If t>Result Then Result:=t;{*Площадь наибольшей
связной области*}
  End;
End;
```

Вывод результата.

```
Procedure WriteAnswer;
Var f: Text;
Begin
 Assign (f, "result.txt"); Rewrite(f); Write(f, Result);
Close(f);
End;
```

Основная программа.

```
Begin
 Init;
 Solve;
 WriteAnswer;
End.
```

Итак, часть тестов, вне всякого сомнения, проходит, и есть база для проведения экспериментов. Изменение значения константы *Size* до требований задачи (даже до значения 200) приводит к сообщению *Error 22: Structure too large*. Не хватает оперативной памяти. Напомним, что в среде Турбо Паскаль для задачи выделяется порядка 64 Кбайт*.

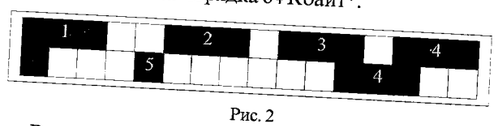


Рис. 2

Размерность задачи такова, что результирующее клеточное поле хранить в оперативной памяти можно только с использованием динамических структур, битовых записей или еще каких-нибудь оригинальным способом. А требуется ли хранить? Возможна построчная обработка поля. Пусть обработана строка. При обработке следующей строки возможны ситуации: область 1 продолжается в следующей строке; область 2 закончилась; области 3 и 4 «сливаются» в одну; область 5 только начинается. Единственная небольшая сложность – правильно отслеживать изменение значений площади при слиянии областей. Основные процедуры этого варианта программы приводятся ниже по тексту.

```
Const Size = 1000;    {*Размер поля*}
Type
 Answer = Array [0..Size] Of LongInt;
        {*Массив ответов (ячейки с площадями)*}
 PhotoLine = Array [0..Size] Of Integer;
                        {*Для обработки строки поля*}
Var
 OldLine, NewLine : PhotoLine; {*oldLine –
                предыдущая, newLine – новая строки*}
 wrInd : Integer;      {*Рабочая ячейка, для поиска
                свободного номера связной области*}
 Result : Integer; {*Результат*}
 TempRes : Answer;
 file1, file2 : Text;
 Procedure Change( DelInd,
NewInd: Integer);
                {*Заменяем номера
устаревших связных областей
                на новое значение*}
 Var i : Integer;
 Begin
  For i := 1 To Size Do Begin   {*Выполняется при
                        слиянии двух связных областей*}
   If NewLine[i] = DelInd Then NewLine[i] := NewInd;
   If OldLine[i] = DelInd Then OldLine[i] := NewInd;
  End;
 End;
 Procedure CompareLines;  {*Сравнения предыдущей
    и новой строк. Находим, какие связные области
    сливаются или дают «потомство»*}
 Var i :Integer;
    char1, char2 :Char;
 Begin
 FillChar( NewLine, SizeOf(NewLine), 0);
 For i := 1 To Size Do Begin
 Read(file1, char1); Read(file2, char2);
 If char1 = char2 Then Begin
        {*Правило одинаковой «закрашенности»*}
  If (NewLine[i-1] = 0) Or (i = 1) Then Begin
                        {*Новая связная область*}
   While TempRes[wrInd]<>0 Do
    If wrInd<Size Then Inc( wrInd ) Else wrInd := 1;
   NewLine[i] := wrInd
   End
  Else NewLine[i] := NewLine[i-1];
        {*Простое дополнение к связной области*}
  If (OldLine[i] <> 0) And (NewLine[i] <> OldLine[i])
Then Begin          {*Слияние или наследование*}
  TempRes[ OldLine[i] ] := TempRes[ OldLine[i] ] +
TempRes[ NewLine[i] ];   {*Сложение значений
                        площадей*}
If TempRes[ NewLine[i] ] <> 0 Then Begin
                {*Слияние*}
  TempRes[ NewLine[i] ] := 0; {*Уничтожение
      устаревшего описания связной области*}
   Change(NewLine[i], OldLine[i]);
  End Else NewLine [i] := OldLine [i];
                                {*Наследование*}
 End;
Inc( TempRes[ NewLine[i] ] );
```

```
Procedure Init;
Begin
 FillChar( OldLine,
SizeOf(OldLine), 0);
 FillChar( TempRes,
SizeOf(TempRes),
0);
 wrInd := 1;
 Result:=0;
```

* В настоящее время происходит отказ от среды Турбо Паскаль на Международных олимпиадах по программированию. В новых средах таких жестких ограничений по оперативной памяти, как правило, нет. Автор считает, что ограничения - один из «двигателей» развития. Вспомним 1-й и 2-й этапы развития технологий программирования. Ограничения по ресурсам ЭВМ являлись мощнейшим рычагом создания эффективных алгоритмов и их исследования.


```
  End
 Else NewLine[i] := 0;
 End;
 ReadLn(file1); ReadLn(file2);
End;
Procedure Solve;
 Var i, j : Integer;
 Begin
 Assign(file1, "photoA.txt"); Reset(file1);
 Assign(file2, "photoB.txt"); Reset(file2);
 For i := 1 To Size Do Begin
  CompareLines;          {*Сравниваем строки*}
   For j := 1 To Size Do
  If (TempRes[j] <> 0) And (TempRes[j] > Result) Then
Result :=TempRes[j];
  OldLine := NewLine;
                {*Новую строку считаем старой*}
 End;
 Close(file1);Close(file2);
End;
```

Продолжим тестирование решения. Вопросы с оперативной памятью решены. Остались временные ограничения – 5 секунд на работу программы с одним тестом. С этой целью необходимо разработать достаточно полную систему тестов. Один тест максимального размера еще не говорит о том, что выполняются требования задачи. Один из файлов (для простоты) можно заполнить единицами (первая вспомогательная программа). А затем? Напрашивается создание файлов максимального размера по принципам, приведенным на рисунке, и, конечно, файла со случайными значениями, а это еще четыре вспомогательные программы (не ручным же способом заполнять файлы). Этим, вероятно, не исчерпываются все возможности, но ограничимся приведенными тестами – их достаточно для нашего изложения. Затем необходимо организовать учет времени решения задачи одним из способов, описанным выше (лучше, естественно, вторым). Что выясняется? Оказывается, только на одном из тестов временное ограничение не выполняется, а именно на тесте со следующей структурой – «на зубьях».

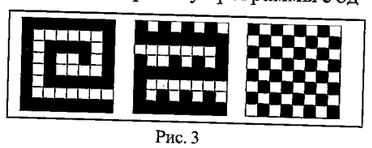


Рис. 3

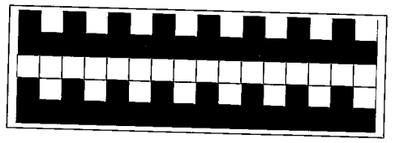


Рис. 4

Почему именно на этом тесте? Вернемся к решению. Ответ просматривается: частота слияния связных областей на тесте значительно возрастает, что вызывает работу процедуры *Change*, т.е. лишний цикл (подсчитайте, сколько раз выполняется процедуры *Change* для области 1000*1000, заполненной по принципу «зубьев»). Можно ли без принципиальных изменений на идейном уровне и в программной реализации добиться выполнения требований задачи? Оказывается, да! Используем идею косвенной адресации. Исключим, точнее, изменим процедуру *Change* из решения. В процедуре просматривался весь массив с целью поиска существующих «родственников». Изменим смысловую нагрузку элемента массива. Она будет содержать:

- данные о площади;
- или ссылку (информацию) на ту ячейку, в которой записано значение площади при слиянии связных областей;
- или ссылку на ссылку при многократном слиянии связных областей.

Как отличить? Стандартный прием – отрицательное значение является признаком косвенной адресации. Исключения многократного просмотра длинных цепочек (при слиянии большого количества связных областей) осуществляется одноразовым прохождением по цепочке, после которого во всех ячейках указывается адрес конечной ячейки со значением площади. Еще один момент рассматриваемой идеи требует упоминания. В предыдущем варианте признаком свободного номера связной области являлось нулевое значение элемента в массиве *TempRes*. В данном варианте этот массив используется для косвенной адресации к ячейкам со значением площади, поэтому требуется введение дополнительного массива с элементами логического типа для этой цели, назовем его *Used*.

```
Type  UsedAn = Array [1..Size] Of Boolean;
      {*Тип для хранения информации
      об использованных номерах связных областей*}
Var Used: UsedAn;
```

При этом предположении поиск свободного номера связной области выглядит следующим образом.

```
Procedure NextWrInd;    {*Массив Used необходим,
    значение Used[i], равное False, говорит о том,
    что значение i – свободный номер связной области*}
Begin
 While Used[wrInd] Do
  If wrInd<Size Then Inc( wrInd ) Else wrInd := 1;
 TempRes [ wrInd ] := 0;
End;
```

Ключевая функция поиска номера ячейки, указывающей на значение площади. При выходе из рекурсии изменяем значения ссылок.

```
Function FindLast( ind : Integer ):Integer;  {*Ищет
    итоговый номер, указывающий на значение площади*}
Begin
 If TempRes [ ind ] > 0 Then FindLast := ind
 Else Begin
  TempRes [ind ] := -FindLast ( Abs ( TempRes [ind] ) );
  FindLast := Abs ( TempRes [ ind ] );
 End;
End;
```

Процедура сравнения соседних строк претерпит незначительные изменения. Приведем ее для сравнения с предыдущей.

```
Procedure CompareLines;
Var i : Integer;
    char1, char2 : Char;
Begin
 FillChar( NewLine, SizeOf(NewLine), 0);
 For i := 1 To Size Do Begin
 Read(file1, char1); Read(file2, char2);
 If charA = charB Then Begin
  If (NewLine[i-1] = 0) Or (i = 1) Then Begin
   NextWrInd;   {*Новый вариант поиска
                 свободного номера связной области*}
   newLine[i] := wrInd;
  End Else NewLine[i] := NewLine[i-1];
 If (OldLine[i]<>0) And (NewLine[i]<>OldLine[i])
Then
   If TempRes [ OldLine[i] ]>0 Then Begin
                              {*Простое слияние*}
   TempRes[OldLine[i]]:=TempRes[OldLine[i]]+
+TempRes[NewLine[i]]; {*Суммируем площади*}
   TempRes[ NewLine [i] ]:= - OldLine [i];
    {*Преобразуем ячейку со значением площади
     в ячейку ссылочного типа*}
  NewLine[i] := OldLine [i]
 End Else NewLine [i] := FindLast ( OldLine [i] );
    {*При слиянии со связной областью,
    которая при данном значении i помечена
    ячейкой ссылочного типа, находим ячейку
    со значением площади этой связной области*}
Inc( TempRes[ NewLine[i] ] );
used [NewLine [i] ] := True ;
End;
End;
 ReadLn(file1);ReadLn(file1);
End;
```

И наконец, новый вариант процедуры *Solve*.

```
Procedure Solve;
Var i:Integer;
Begin
 InitTemp;
Assign(fileA, "photoA.txt"); Reset(fileA);
Assign(fileB, "photoB.txt"); Reset(fileB);
For i := 1 To Size Do Begin
 CompareLines;
 FillChar ( used , SizeOf(used), False);
              {*Считаем все номера свободными*}
 For j := 1 To Size Do Begin
 If TempRes [ NewLine [j] ] < 0 Then
 NewLine [j] := FindLast ( NewLine [j] );
 {*Этим действием мы сокращаем количество
  переадресаций по ссылочным ячейкам при поиске
ячейки с площадью, значительно ускоряя тем самым
обработку следующей строки при слиянии связных
областей*}
   If ( TempRes [ NewLine [j] ] > 0) And Not Used
 [NewLine [j]] Then Begin
   If TempRes [ NewLine [j] ] > Res Then Res := TempRes
 [NewLine [j] ];
    Used [ NewLine [j] ] := True;
   End;
  End;
 OldLine := NewLine;
End;
 Close(fileA);Close(fileB);
End;
```

Для иллюстрации логики, а именно, изменения значений в массивах *NewLine* и *TempRes*, рассмотрим простой пример, первые две строки которого изображены на рис. 5.

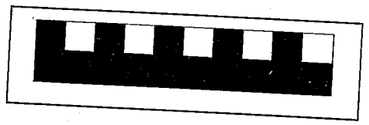


Рис. 5

Этот вариант программы «разбирается» с тестом типа «зубьев» размером 1000*1000 за 3 секунды. Можно ли продолжить совершенствование программы решения задачи? Безусловно, ибо нет предела совершенству. Мы не использовали динамические структуры данных. А может быть, есть вариант не построчного сравнения клеточных полей? Да даже написание программ, генерирующих входные файлы различных типов, интереснейшая задача.

М.В. Перевозчиков

ОБ ОДНОЙ ЗАДАЧЕ ПО ИНФОРМАТИКЕ

На протяжении всего времени существования для Computer Science является актуальной проблема выполнения операции умножения. Дело в том, что операция умножения очень ресурсоемка, а при решении практически любой задачи умножение зачастую неотъемлемо. Причем в данный момент одинаково важна как аппаратная, так и программная реализация умножения, так как аппаратное умножение не всегда оптимально, и, кроме того, в некоторых случаях разработчики аппаратных средств умышленно перекладывают данную задачу на плечи программистов. Примером может служить МикроЭВМ КР1813, не имеющая команды умножения как таковой, хотя создана была для цифровой обработки сигналов в реальном времени, где умножение неотъемлемо.

Известный ученый Яков Трахтенберг предложил метод вычисления произведения двух натуральных

чисел, который заключался в представлении данных чисел в виде полиномов:

$a=a_0x^0+a_1x_1+...+a_mx_m$

$b=b_0x^0+b_1x_1+...+b_nx^n$, где x – основание полинома, в данном случае равное 10, a_i и b_i – i-я цифра в записи числа a и b соответственно. Очевидно, что для нахождения произведения чисел a и b необходимо найти произведение этих полиномов:

$$a \cdot b = a_0b_0x^0 + (a_0b_1 + a_1b_1)x^1 + \ldots + x^i \cdot$$

$$\cdot \sum_{j=\max(i-n,0)}^{\min(i,m)} a_j b_{i-j} + \ldots + a_m b_n x^{m+n} \cdot$$

обозначив коэффициенты при x_i за α_i, получим ряд из чисел $\alpha_1, \alpha_2, \ldots, \alpha_{m+n}$, называемый рядом Трахтенберга[1]. Имея этот ряд, можно легко найти произведение, кроме того, данный ряд, хоть и неоднозначно, задает числа a и b. Для нахождения этих чисел по заданному набору можно проверить все пары чисел, удовлетворяющие условию, что количество цифр в обоих числах на единицу превосходит количество чисел в данном ряде Трахтенберга, далее найти для этих пар чисел ряд Трахтенберга и сравнить его с данным. Но такое решение обладает колоссальными временными затратами, и оно уже неприменимо для трехзначных чисел. Но есть еще один путь: необходимо решить систему уравнений:

$$\begin{cases} a_0b_0 = \alpha_0 \\ a_0b_1 + a_1b_0 = \alpha_1 \\ a_0b_2 + a_1b_1 + a_2b_0 = \alpha_2 \\ \ddots \\ \sum_{j=\max(i-n,0)}^{\min(i,m)} a_j b_{i-j} = \alpha_i \\ \ddots \\ a_m b_n = \alpha_{m+n}, \end{cases} \quad (1)$$

где α_i – i-й член ряда Трахтенберга. Аналитическое решение такой системы для общего случая очень затруднительно. Нетрудно заметить, что решение i-го уравнения, начиная со второго, сводится к решению предыдущего и нахождению a_i и b_i коротким перебором. Решение же первого уравнения заключается в нахождении всех делителей числа , причем таких, что и делитель, и частное не превышали 9, так как a_i и b_i – это цифры.

Итак, совершенствуем первоначальное решение разложением α_0 на множители, не превышающие 9 – получаем первые две цифры искомых чисел, далее рекурсивно подбираем остальные цифры. В результате либо получим пару искомых чисел, либо решение одного из уравнений не будет являться цифрами, то есть либо не будут лежать на отрезке [0; 9], либо не будут целочисленными. Таким образом, если tr – массив с данным рядом Трахтенберга, а и b – массивы для вычисления искомых чисел[2], то решение будет выглядеть следующим образом:

```
for i:=1 to min(tr[0], 9) do
 if( tr[0] mod i = 0) and ( tr[0] div i < 10) then
  for j := 0 to tr[-1] do
  begin
   FillChar(a, sizeof(a), $FF) ;
   FillChar(b, sizeof(b), $FF) ;
    a[-1] := j ;
    b[-1] :=  tr[-1] - j ;
    a[ 0 ] :=  i ;
    b[ 0 ] :=  tr[0] div i ;
  if( a[-1] < b[-1] )or( (a[-1] = b[-1])and( a[0] <= b[0]) )
then Solve(1) ;
  end ;
```

Аргументом процедуры Solve является индекс решаемого уравнения в системе (1). Решив q-е уравнение, процедура вызывает себя же с аргументом q+1, таким образом, превышение q значения m+n будет сигнализировать о нахождении искомой пары чисел.

Решение же любого (пусть i-го) уравнения можно обобщить:

$$a_{\min(m,i)} \cdot b_{i-\max(0,i-n)} + a_{\max(0,i-n)} \cdot b_{i-\min(m,i)} + \sum_{j=\max(0,i-n)+1}^{\min(m,i)-1} a_j b_{i-j} = 0,$$

где неизвестными являются $a_{\min(m,i)}$ и $b_{i-\min(m,i)}$. Если хотя бы одно решение является цифрами, то для каждого решения отдельно продолжаем решать систему, считая что все уравнения до i-го включительно уже решены. Таким образом, если удастся решить систему, найдется и искомая пара чисел. Такое решение имеет **намного** меньшие временные затраты и может применяться для числа порядка нескольких десятков знаков.

Но задача имеет еще одно решение, заключающееся в разложении результата на множители. По основной теореме арифметики, любое число предста-

[1] Ограничение максимального значения j в min(i, m) при вычислении i-го коэффициента объясняется тем, что первый полином, имеет степень m, а i может быть вплоть до m+n. Ограничение же на минимальное значение j в max(i-n, 0) исходит из того, что второй полином имеет степень n.

[2] Здесь массивы имеют следующую структуру: каждый массив заполняется $FF, таким образом все элементы массива равны -1, а в -1-ом элементе массива хранится его «длина», то есть количество заполненных элементов.

вимо в виде $p_1^{\alpha_1} \cdot p_2^{\alpha_2} \cdot p_q^{\alpha_q}$. Разложив результат таким образом, нетрудно увидеть, что решением задачи будут все такие пары чисел $p_1^{\alpha_1'} \cdot p_2^{\alpha_2'} \cdot \ldots \cdot p_q^{\alpha_q'}$ и $p_1^{\alpha_1''} \cdot p_2^{\alpha_2''} \cdot \ldots \cdot p_q^{\alpha_q''}$, причем $\alpha_i' + \alpha_i'' = \alpha_i$. Таким образом, получим $\prod_{i=1}^{q} \frac{\alpha_i}{2}$ различных пар чисел, являющихся искомыми.

Разница этих двух решений заключается в том, что в первом случае набор Трахтенберга, вычисленный по полученным парам чисел, будет в точности совпадать с данным, а во втором случае это условие выполняться не будет.

СВЕДЕНИЯ ОБ АВТОРАХ

АНДРЕЕВА Елена Владимировна – кандидат физико-математических наук, доцент МГУ

АНДРЕЕВА Татьяна Николаевна – ассистент кафедры информационных технологий и ТСО ВятГГУ

АШИХМИНА Татьяна Викторовна – старший преподаватель кафедры информатики и МОИ ВятГГУ

БАБУШКИНА Ирина Анатольевна – старший преподаватель кафедры информатики и МОИ ВятГГУ

БАРДОВСКАЯ Анастасия Игоревна – ассистент кафедры германских языков ВятГГУ

БЕЛИОВСКАЯ Лидия Георгиевна – кандидат физико-математических наук, преподаватель информатики лицея № 1557 г. Москвы

БОЯРИНЦЕВ Дмитрий Владимирович – студент 2-го курса факультета информатики ВятГГУ

БУШМЕЛЕВА Наталья Александровна – кандидат педагогических наук, старший преподаватель кафедры информатики и МОИ ВятГГУ

ВАСЕНИНА Елена Александровна – кандидат педагогических наук, доцент кафедры информатики и МОИ ВятГГУ

ВАХРУШЕВ Алексей Серафимович – кандидат биологических наук, доцент кафедры информационных технологий и ТСО ВятГГУ

ВЕДЕРНИКОВА Елена Витальевна – учитель информатики физико-математического лицея г. Кирова.

ВОЛЧЕНКОВ Сергей Геннадьевич – кандидат физико-математических наук, доцент Ярославского государственного университета.

ИВАНОВ Сергей Юрьевич – студент 3-го курса факультета информатики ВятГГУ

КАЛГАННИКОВ Егор Юрьевич – студент 1-го курса факультета информатики ВятГГУ

КИПРСКАЯ Екатерина Викторовна – старший преподаватель кафедры германских языков ВятГГУ

КОЗВОНИНА Анастасия Валерьевна – студентка 5-го курса факультета информатики ВятГГУ

КОЛОДКИНА Елена Николаевна – кандидат филологических наук, доцент, зав. кафедрой германских языков факультета информатики ВятГГУ

КОРЧЕМКИНА Оксана Николаевна – ассистент кафедры информационных технологий и ТСО

КОРЯКИНА Галина Константиновна – завуч физико-математического лицея г. Кирова

МАТЮХИН Виктор Александрович – аспирант МГУ

МОСКОВКИН Алексей Александрович – студент 4-го курса факультета информатики ВятГГУ

ОВСЯННИКОВА Марина Владимировна – старший преподаватель кафедры информатики и МОИ ВятГГУ

ОКУЛОВ Станислав Михайлович – кандидат технических наук, доцент, декан факультета информатики ВятГГУ

ОНЕГОВ Владислав Алексеевич – кандидат физико-математических наук, доцент, заведующий кафедрой информатики и МОИ ВятГГУ

ПАВЛОВА Наталья Вячеславовна – ассистент кафедры германских языков ВятГГУ

ПЕТУХОВА Мария Владиславовна – кандидат педагогических наук, доцент, зав. кафедрой информационных технологий и ТСО ВятГГУ

ПЕРЕВОЗЧИКОВ Максим Валерьевич – студент 1-го курса факультета информатики ВятГГУ

РАЗОВА Елена Владимировна – старший преподаватель кафедры информатики и МОИ ВятГГУ

РОНЖИН Андрей Николаевич – студент 2-го курса факультета информатики ВятГГУ

СВИЦОВА Анна Альбертовна – старший преподаватель кафедры германских языков ВятГГУ

ФАЛИНА Ирина Николаевна – кандидат педагогических наук, доцент МГУ

ФЕДОРОВСКАЯ Татьяна Николаевна – кандидат филологических наук, доцент кафедры английского языка Института лингвистики ВятГГУ

ХРИСТОЛЮБОВА Наталья Сергеевна – студентка 5-го курса Института лингвистики ВятГГУ

ЧЕКАНОВ Дмитрий Александрович – старший преподаватель кафедры информатики и МОИ ВятГГУ

ШЕЙДУЛЛИНА Гульнара Каюмовна – ассистент кафедры германских языков ВятГГУ

ШИХОВ Владислав Викторович – студент 5-го курса факультета информатики ВятГГУ

ЮФЕРЕВ Владислав Владимирович – директор физико-математического лицея г. Кирова

ЯМБАРЫШЕВА Светлана Юрьевна – старший преподаватель кафедры информатики и МОИ ВятГГУ

Вятский государственный гуманитарный университет
Факультет информатики

Факультет информатики стал самостоятельным подразделением Вятского государственного гуманитарного университета в 1999 году.

Декан факультета – кандидат технических наук, доцент кафедры ИМОИ Окулов Станислав Михайлович.

Зам. декана по учебно-воспитательной работе – кандидат педагогических наук, старший преподаватель кафедры ИМОИ Бушмелева Наталья Александровна.

Зам. декана по заочному отделению факультета – кандидат педагогических наук, доцент кафедры ИМОИ Васенина Елена Александровна.

Обучение ведется по двум специальностям:

«030100 – Информатика с дополнительной специальностью английский язык» с квалификацией специалиста – ***учитель информатики, учитель английского языка.***

Основные направления обучения

1. Развитие мыслительных способностей в процессе столь специфической деятельности, как программирование, включающее в себя профессиональное владение основными технологиями программирования, – процедурное, объектно-ориентированное, визуальное программирование (Pascal, Delphy, Visual Basic и др.).
2. Развитие способности к быстрому самостоятельному освоению новых информационных технологий и программных средств, профессиональное владение основными информационными технологиями общего назначения (обработка текстов и электронных таблиц, системы компьютерной графики и многое другое).
3. Свободное владение английским языком, в частности соединение профессиональных знаний в области информатики с языковой подготовкой, что дает качественно новый уровень подготовки специалиста.
4. Владение основами педагогики, психологии, теории познания, дидактики и методики.

«010200 – Прикладная математика и информатика» с квалификацией специалиста – математик, системный программист.

Основные направления специализации

1. Серьезная математическая подготовка.

В дополнение к традиционной высшей математике изучаются курсы дискретной математики, теория алгоритмов, методы оптимизации, теория игр, исследование операций и многие другие.

2. Глубокая подготовка в области программирования.

Акцент делается на системное программирование, методы трансляции, программное обеспечение компьютерных сетей.

Изучаются курсы:

- архитектура компьютерных сетей;
- сетевое администрирование и управление сетью;
- сетевые операционные системы;
- методы защиты информации;
- распределенные базы данных;
- глобальные сети и многое другое.

3. Серьезная языковая подготовка специалистов в области сетевого и системного программирования, что на порядок повышает их конкурентоспособность на рынке труда.
4. Поскольку данная специальность не является педагогической, здесь нет специальной подготовки в области педагогики и методики. Однако по желанию студент может выбрать для изучения курсы по психологии, социологии, экономике, государственному праву, что также расширяет сферу его профессиональной подготовленности.

Важно и то, что на факультете информатики готовят всесторонне образованных людей, которые знают мировую историю и культуру, философию, концепции современного естествознания.

Сегодня здесь обучается 273 студента, из них 11 – заочно.

В настоящее время в рамках факультета существует 3 кафедры:

Кафедра информатики и методики обучения информатике (ИМОИ): заведующий – к.ф.-м.н., доцент Онегов В.А.;

Кафедра информационных технологий: заведующий – к.п.н., и.о. доцента Петухова М.В.;

Кафедра германских языков: заведующий – к.ф.н., доцент Колодкина Е.Н.

На факультете работают 42 преподавателя (из них 3 профессора, 8 доцентов). Средний возраст преподавателей – 28 лет. Это говорит о высоком научном потенциале специалистов факультета.

Научные направления: на факультете открыта аспирантура по специальности «130002 – Теория и методика обучения и воспитания (информатика)» под руководством доктора педагогических наук, действительного члена РАО, профессора А.А. Кузнецова.

На сегодняшний день аспирантуру закончили и защитили кандидатские диссертации 3 человека, продолжают обучение 4 человека.

Научно-исследовательская работа преподавателей и студентов направлена на решение следующих основных задач: развитие фундаментальных научных исследований, повышение квалификации преподавательских кадров.

На факультете сложился ряд перспективных направлений научных исследований:

- методология информатики (на примере отдельных дисциплин информационного цикла);
- психолого-педагогические вопросы методики преподавания информатики в школе и в вузе;
- внедрение в учебный процесс новых информационных технологий, их использование в преподавании гуманитарных и социально-экономических дисциплин;
- психолингвистические проблемы понимания слова и текста;
- теория средних величин и др.

На кафедре информатики и методики обучения информатике регулярно работает научно-методический семинар (рук. доцент Онегов В.А.), на котором рассматриваются проблемы, связанные с взаимодействием предметов информационного цикла в связи с введением новых образовательных стандартов.

Работа в помощь школе и органам образования проводится по следующим направлениям: проведение педагогической практики в школах и УПК г. Кирова и области; написание статей и учебных пособий по актуальным вопросам информатики и др.

Преподаватели факультета читают лекции для учителей школ и преподавателей других вузов, проводят семинары, выступают с докладами на областных и всероссийских конференциях.

Лучшие работы студентов представляются на университетские конкурсы. Раз в году на факультете проводится декада студенческой науки, в рамках которой организуются олимпиады по информатике. Ежегодно студенты факультета занимают призовые места на республиканских олимпиадах по информатике.

Работа со школьниками: студенты и преподаватели занимаются со школьниками в кружках, в профильных классах, на семинарах и спецкурсах. Для учащихся ежегодно организуются областные олимпиады. Школьники имеют также возможность обучаться в школе программирования «Аналитик», созданной при факультете.

Внеучебные занятия: в организации студенческой жизни во многом помогают традиции: учебно-методический сбор «Первокурсник», КВН между студентами I и II курсов, проводимый к Дню учителя, факультетский День здоровья, «Студенческая весна» с обязательной подготовкой в течение всего года в танцевальном коллективе, вокальном ансамбле и др.

Вступительные экзамены

Прием документов: 15.06.2002 – 15.07.2002 (г. Киров, ул. Ленина, д. 111, каб. 319)
Подготовительные курсы: 24.06.2002 – 13.07.2002
Экзамены: 16.07.2002 – 31.07.2002

Специальность «Информатика и английский язык»

* Информатика (устно)
* Иностранный язык (устно)
* Русский язык и литература (сочинение)

Специальность «Прикладная математика и информатика»

* Математика (письменно)
* Информатика (устно)
* Русский язык и литература (сочинение)

Зачисление – 1.08.2002

На факультете осуществляется целевой внеконкурсный прием по направлениям департамента образования.

На период подготовительных курсов и экзаменов абитуриенты обеспечиваются общежитием.

Справки

по телефону приемной комиссии – 67-87-78
по телефону деканата факультета информатики – 67-06-11

Окулов Станислав Михайлович
Основы программирования
М.: ЮНИМЕДИАСТАЙЛ, 2002. – 424 с.: ил.
ISBN 5-94774-003-6

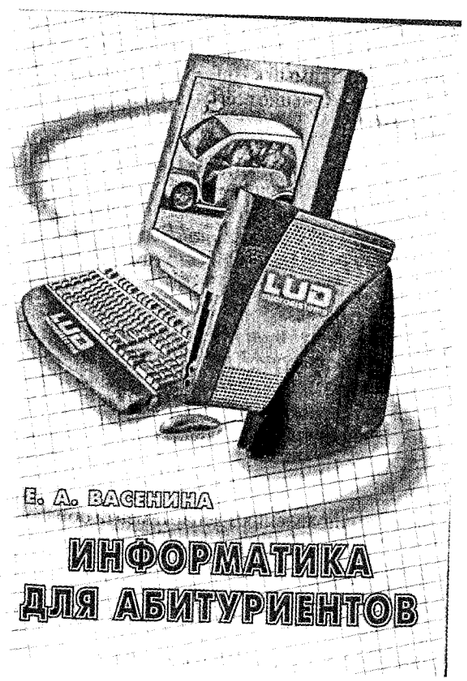
Серия «Технический университет». Учебное издание.

В учебнике рассмотрены основные управляющие конструкции системы программирования Турбо Паскаль, процедуры и функции, строковый, вещественный и файловый типы данных. Приводится материал для изучения массивов, методов сортировки и поиска, а также по динамическим структурам данных. Рассмотрены следующие структуры данных: списки, стеки, очереди, двоичные деревья, АВЛ-деревья и Б-деревья. В материалах для чтения обсуждаются практически все вопросы, входящие в школьный минимум знаний по информатике.

Книга является достаточно полным учебником по программированию, реализующим сложную задачу – формирование у читателя структурного стиля мышления. Учебным материалом является система программирования Турбо Паскаль, а также большое число задач, включая задачи на алгоритмы сортировки и поиска.

Достаточно подробно рассмотрена работа с динамическими структурами данных.

Книга рассчитана на широкий круг читателей от школьника и студента до специалиста, решающего с помощью программирования прикладные задачи.



Васенина Елена Александровна
Информатика для абитуриентов
Учебное пособие
Киров: Изд-во ВГПУ, 2002. – 168 с.
ISBN 5-900185-72-9

Учебное пособие предназначено для учащихся и учителей информатики средней общеобразовательной школы и содержит теоретический материал для подготовки к вступительным экзаменам по информатике в вуз. Книга будет полезна также студентам вузов различных профилей, начинающим изучать информатику.

Онегов Владислав Алексеевич
Исследование операций: задачи, методы, алгоритмы
Учебное пособие
Киров: Изд-во ВГПУ, 2001. – 224 с.
ISBN 5-93825-004-8

В учебном пособии представлены традиционные разделы науки Исследование операций. Основной упор сделан на методы и алгоритмы решения разнообразных задач. Все алгоритмы доведены до отлаженных Pascal-программ. Материал излагается таким образом, что возможна работа с ним в самостоятельном режиме. Пособие предназначено студентам, специализирующимся по информатике и математическим методам в экономике и всем желающим познакомиться с идеями и задачами, решаемыми методами исследования операций. Пособие рекомендовано УМО Министерства образования РФ по специальностям педагогического образования в качестве учебного пособия для студентов высших учебных заведений, обучающихся по специальности 030100 – информатика.

Бабушкина Ирина Анатольевна
Овсянникова Марина Владимировна
Пятышева Елена Анатольевна

Практикум по программному обеспечению ЭВМ
Часть первая
Киров: Изд-во ВГПУ, 2001. – 144 с.
ISBN 5-85271-105-5

Пособие содержит материалы занятий по курсу «Программное обеспечение ЭВМ». В первой части изложены основные сведения об операционной системе Windows NT (главы 1,2), антивирусных программных средствах (глава 3), архивировании файлов (глава 4). Главы с пятой по тринадцатую посвящены основам работы в текстовом процессоре Word 2000: настройке текстового процессора, набору и форматированию текста, подготовке документа к печати.

Учебное пособие написано по материалам практических занятий со студентами педагогического университета и со школьниками.

Бабушкина Ирина Анатольевна
Овсянникова Марина Владимировна
Пятышева Елена Анатольевна

Практикум по программному обеспечению ЭВМ
Часть вторая
Киров: Изд-во ВГПУ, 2001. – 144 с.
ISBN 5-85271-105-5

В книге приводятся материалы для занятий по информатике (курс «Программное обеспечение ЭВМ»). Во второй части содержатся главы, в которых рассматриваются дополнительные возможности текстового процессора Word 2000: графические объекты (главы 1-2), таблицы (глава 3), диаграммы (глава 5), формулы (глава 6), сноски (глава 9), средства автоматизации работы с документами: автотекст и автозамена (глава 4), шаблоны (глава 7), типовые письма (глава 8).

Учебное пособие написано по материалам практических занятий со студентами педагогического университета и со школьниками.

Чеканов Дмитрий Александрович

Практикум по программному обеспечению ЭВМ
Часть третья
Киров: Изд-во ВГПУ, 2001. – 96 с.
ISBN 5-85271-102-0

Пособие является продолжением первой и второй частей «Практикума по программному обеспечению ЭВМ» И.А. Бабушкиной, М.В. Овсянниковой и Е.А. Пятышевой. Содержит материалы для занятий по курсу «Программное обеспечение ЭВМ». В первой главе изложены основные сведения о сети Интернет и навигации в ней. Вторая глава посвящена поисковым серверам. Третья и четвертая главы знакомят с электронной почтой и конференциями Интернета. В пятой главе рассматривается работа с серверами FTP. Шестая глава посвящена общению в реальном времени. В седьмой главе идет речь об Интернет-пейджере ICQ.

Учебное пособие написано по материалам практических занятий со студентами педагогического университета и со школьниками.

ДЛЯ ЗАМЕТОК