ISSN 0234—0453

ИНФО

ИНФОРМАТИКА И ОБРАЗОВАНИЕ

6 1994


УРНОВ
Василий Андреевич
Президент
НЦПСО-AIST-group
ЗУБЧЕНКО
Анатолий Александрович
Генеральный директор
НЦПСО
ПЕТРОВ
Михаил Кириллович
Генеральный менеджер
AIST Inc.
ПРОНИН
Владимир Николаевич
Начальник департамента маркетинга и поддержки НЦПСО
САВАТЕЕВ
Вячеслав Юрьевич
Начальник технических служб
НЦПСО
МИРОНОВ
Михаил Александрович
Генеральный менеджер
AIST AB
КЛИМОВ
Дмитрий Юрьевич
Руководитель программных проектов AIST AB
ЮШИН
Дмитрий Анатольевич
Генеральный менеджер
НЦПСО-комплекс
БАРАНОВ
Евгений Анатольевич
Руководитель программных проектов НЦПСО-комплекс

ИНФО 6'94

ИНФОРМАТИКА И ОБРАЗОВАНИЕ

Научно-методический журнал
Учрежден Министерством образования РФ и коллективом редакции

Издается с августа 1986 г.
Выходит шесть раз в год

СОДЕРЖАНИЕ

На первой странице обложки — руководители и ведущие специалисты НЦПСО—AIST group.

Почту направлять по адресу: 103051, Москва, ул. Садовая-Сухаревская, д. 16, к. 9, журнал «Информатика и образование».

Телефон: (095) 208-30-78
Факс: (095) 208-67-37
E-Mail: info@tit-bit.msk.su

Подписано в печать с оригинал-макета 27.12.94. Формат 70×100 1/16. Бумага офсетная. Печать офсетная. Усл. печ. л. 10,40. Усл. кр.-отт. 11,70. Уч.-изд. л. 13,5. Тираж 11 000 экз. Заказ 3262.
Цена по подписке:
для индивидуальных подписчиков 1200 руб. (индекс 70423);
для предприятий и организаций 2400 руб. (индекс 73176).
В розницу цена договорная.

Ордена Трудового Красного Знамени Чеховский полиграфический комбинат Комитета по печати и информации Российской Федерации. 142300, Чехов, Московской обл.

А. П. Частиков,
профессор Краснодарского политехнического института

ИСТОРИЯ ИНФОРМАТИКИ В ЛИЦАХ: ГОВАРД АЙКЕН

> Вскоре я познакомился с первой из новых цифровых машин, построенных на реле. Она находилась в Гарвардском университете. Работы по созданию этой машины велись по правительственному заданию, и руководил ими Говард Айкен. Его успехи произвели на меня большое впечатление и вызвали чувство глубокого восхищения. Айкен рассматривал свою машину как современный вариант тех весьма примитивных вычислительных машин, которые около ста лет тому назад разрабатывал Бэббидж в Англии.
>
> *Норберт Винер, «Я — математик», 1956 г.*

В 1864 г. в одной из своих последних работ ученый, опередивший свое время, автор неосуществленного проекта Аналитической машины Чарлз Бэббидж писал: «Если кто-либо, не наученный моим опытом, попытается создать машину, способную выполнять математический анализ в объеме, равном работе целого отдела математиков, причем с использованием различных принципов или более простых механических средств, и попытка эта увенчается успехом, то мне не страшно вверить такому человеку свою репутацию, так как только он сможет сполна оценить усилия и достигнутые результаты». Реально идеи и концепции Ч. Бэббиджа смогли осуществиться только через 80 лет после написания этих пессимистических строк. И человека, который практически воплотил эти идеи, звали Говард Хетауэй Айкен. Правда, следует уточнить, что с проектом Бэббиджа Айкен познакомился только через три года после начала работ по созданию своего первого детища и был поражен настолько, что воскликнул: «Живи Бэббидж на 75 лет позже, я остался бы безработным».

Говард Айкен вырос в Индианаполисе, штат Индиана (он родился 8 марта 1900 года). После восьмого класса он был вынужден пойти работать ночным оператором в компанию по освещению и теплоснабжению, а днем посещал техническую

школу Арсенала. Руководство школы заинтересовалось способностями молодого человека и предоставило ему возможность сдать экзамены досрочно. Затем он поступает в Университет штата Висконсин и одновременно работает в газовой компании. В 1923 г. Айкен получает степень бакалавра наук, а за ночную работу — должность главного инженера газовой компании. Проработав инженером-энергетиком 10 лет, в 1935 г. он поступает в Чикагский университет, а затем становится аспирантом по физике Гарвардского университета.

Докторская диссертация Айкена «Теория проводимости пространственных зарядов», представленная отделению физики Гарвардского университета в 1939 г., изобилует выражениями разочарования по поводу того, что нельзя было решить аналитически нелинейные дифференциальные уравнения, адекватно описывающие изучаемые им явления, и что их численное решение даже для нескольких интересуемых случаев требует выполнения недоступного для человека объема вычислений. В результате этих размышлений в 1937 г. появилась его работа (он издал ее на свои средства) под названием «Предлагаемая автоматическая счетная машина», где автор описал машину в виде коммутационной доски, на которой смонтированы различные узлы вычислительной машины, причем каждая панель коммутационной доски предназначалась для выполнения определенных математических операций.

Гарвард был в то время центром «чистых» исследований, поэтому Айкен не нашел поддержки, было скорее сопротивление. Многие не верили в то, что такую машину можно создать. Его поддержали лишь астроном Харлоу Шэпли и профессор школы бизнеса Тед Браун. Айкен пытался подключить в эту работу фирму Монро, в частности, главного инженера Дж. Чейза. Чейз заинтересовался, но руководство фирмы сочло идею непрактичной и отказалось от ее реализации. Чейз рекомендовал Айкену обратиться к фирме IBM — так состоялось его знакомство с Томасом Уотсоном, президентом этой фирмы. В 1939 г. был подписан контракт, по которому фирма IBM с финансовой поддержкой военно-морского ведомства США бралась за создание машины Айкена. Спустя семь лет в мае 1944 г. машина, получившая название «Марк-I», вступила в строй. Реле, счетчики, контактные устройства, печатающие механизмы, устройства для ввода перфокарт и перфорирующие устройства, используемые в машине, были стандартными частями табуляторов, выпускаемых в то время фирмой IBM. Эта машина, работавшая с 23-значными десятичными числами, выполняла операцию сложения за 0,3 с и операцию умножения за 3 с и имела внушительные размеры (длина — 17,4 м, высота — 2,5 м).

«Марк-I» отличала высокая работоспособность (если этот термин можно применить к машине) и главное надежность. Этот колосс работал по 24 часа в сутки без выходных, выполняя главным образом расчеты по секретным проектам военно-морского флота. За первые три года работы на компьютере «Марк-I» были составлены 19 таблиц различных математических функций (функций Бесселя, функций Хенкеля, интегральных синусов и косинусов и т. д.). Многочисленные исследования, связанные с решением больших систем линейных уравнений, с помощью компьютера «Марк-I» проводил профессор отделения экономики Гарвардского университета Василий Леонтьев (позже лауреат Нобелевской премии).

Надо сказать, что «Марк-I» в 1944 г. имел черты современных компьютеров, в

частности разделения времени выполнения операций, при этом не возникали конфликтные ситуации различных устройств, широко использовались подпрограммы при программировании. А в дальнейшем были введены устройства, «обеспечивающие средства для выбора любого из нескольких ответвляющих путей операции» (т. е. операций условного перехода).

В 1947 г. в лаборатории Айкена была создана новая релейная машина «Марк-II», которая обладала необычным для того времени свойством: она могла работать как одна машина, или как две отдельные машины, решая одновременно две различные задачи. Переход с одного режима на другой производился с помощью переключателя. В машине была введена плавающая запятая.

В последние годы Говарда Айкена упрекали в том, что он при проектировании своих компьютеров неохотно переходил от электромеханических к электронным устройствам. А может быть осторожно, выжидательно: «Нет, я знал, что нужно было идти этим путем. Я не хотел зависеть от технологии. Я не хотел беспокоиться о надежности незнакомых и еще не испытанных в новом применении компонентов. Как только я увидел, что другие проекты пошли нормально и что инженеры научились работать с новой технологией, я перешел на электронику». И это случилось при создании им третьей — четвертой версии машины — «Марк-III» и «Марк-IV». Вообще с позиций сегодняшнего дня, оглядываясь назад, становится ясным, что компьютеры на основе электронно-вакуумной технологии были по существу переходной средой на пути к созданию более надежных машин с использованием реле и полупроводниковой технологии. И Г. Айкена в этом плане можно считать первооткрывателем.

Начало проектирования электронной машины «Марк-III» датировано январем 1948 г., а окончание — 1950 г. В ней числа и команды уже представлялись в двоичном коде и хранились на девяти алюминиевых барабанах, покрытых магнитным слоем.

Последний компьютер, созданный Г. Айкеном в Гарварде для военно-воздушных сил, «Марк-IV» был закончен в 1952 г. В отличие от проектируемых в то время машин на основе менее надежной электростатической памяти, память «Марк-IV» с произвольным доступом состояла из 200 сдвигающих регистров на магнитных сердечниках. Машина имела также память большого объема на магнитном барабане с отдельными секторами для 4000 16-разрядных чисел и для 1000 команд. В машине были применены индексный регистр и декодирующее устройство со специальной клавиатурой для записи программ в алгебраических выражениях, близких к обычной записи.

Большие успехи в деле автоматизации программирования принадлежат Грейс Хоппер (контр-адмирал ВМС США), которая работала с машинами «Марк» в Гарварде с 1944 по 1949 г. Ее вклад в деле создания первых компиляторов (кстати термин «компилятор» был впервые введен Грейс Хоппер) и первых языков программирования неоспорим и ее имя в тот период неразрывно связано с именем Говарда Айкена и его компьютерами.

При проектировании последней машины Айкен обратил внимание на проблемы, которые возникают при промышленном производстве компьютеров, когда они перестают быть предметом лабораторных исследований. Ему хорошо была известна пионерская работа Клода Шеннона по релейным схемам, опубликованная в 1938 году, но в 1949 г. ничего не было известно об использовании работы Шеннона для проектирования электронных схем. В 1951 г. Айкен вместе с коллегами выпустил книгу «Синтез электронных вычислительных и управляющих схем». В предисловии он писал: «Как в лаборатории, так и в учебном классе можно легко заметить, что недостаток адекватных математических методов исследования функциональных характеристик электронных управляющих схем представлял единственное и самое серьезное препятствие для быстрого развития данной области и для обучения студентов, интересующихся этим вопросом». Последующее быстрое развитие теории коммутации и управляющих систем подтвердило мудрость этого наблюдения.

Айкен первым предложил в 1947/48 учебном году ввести курс по проектирова-

нию цифровых вычислительных устройств в Гарварде. Сразу же после этого был введен курс анализа цифровых схем, ориентированный на технологии, используемые в цифровых компьютерах. В период его работы Гарвард стал одним из первых университетов, который стал присваивать ученые степени в области вычислительной техники.

Айкен был талантливым и вдохновенным учителем в самом лучшем понимании. Будучи сам волевым, независимым, целеустремленным и настойчивым, отдавая максимум времени учебе, науке, работе, он ожидал того же и от своих студентов и коллег, поэтому достигнутые ими результаты превосходили их собственные ожидания. Это был чрезвычайно активный человек, его след можно найти во многих областях науки и техники, а последний патент был выдан за год до кончины, когда ему исполнилось 72 года.

В последние годы жизни он работал консультантом фирмы Monsanlo Chemical по вопросам, связанным с исследованиями в области магнитных доменов. 14 марта 1973 г. во время поездки в Сант Луис, в штаб-квартиру фирмы Monsanlo Говард Айкен внезапно скончался от инфаркта.

Результаты его исследований и преподавательской деятельности отражены в «Трудах компьютерной лаборатории», которой он руководил в течение многих лет, во многих журнальных публикациях и многочисленных отчетах ВМФ и ВВС США, комиссии по атомной энергетике, телефонной лаборатории, Американской газовой ассоциации и других организаций. Велико значение работ, которые провели его многочисленные ученики и коллеги в развитии компьютерной науки.

Достижения Говарда Айкена были признаны во всем мире, о чем свидетельствуют многие награды, которых он был удостоен. Когда впервые в 1946 г. был учрежден Комитет по вычислительным машинам, Айкен стал одним из первых его членов вместе с Джорджем Стибицем и Джоном фон Нейманом. Он являлся членом Американской Академии науки и искусства и Института радиоинженеров; он был удостоен награды ВМФ за выдающиеся заслуги и награды ВВС за особые гражданские заслуги; его «альма матер» Висконсинский университет выдал ему диплом за выдающиеся профессиональные заслуги; Бельгия наградила его Офицерским крестом Ордена короны; Франция возвела его в кавалеры Легиона чести. Ему присвоена почетная степень доктора высшей технической школы Германии в Дармштадте, в Испании — почетного члена Высшего совета по научным исследованиям, в Швеции — члена Инженерной академии.

Профессор математики Государственного университета Гумбольта, главный эксперт проекта истории компьютерной техники Смитсоновского Национального музея истории техники Генри Тромп писал: «Я убежден в том, что Айкен был одним из провозвестников компьютерной эры. Его личные качества оказали большое влияние на многих исследователей, сотрудничавших с ним в период 40—50-х годов».

Литература

1. *Tropp H.* The effervescent years: a retrospective. IEEE Spectrum, 1974, february.

2. *Oettinger A. G.* Howard Aiken. Communications of the ACM. 1962, v. 5, no. 6.

3. *Частиков А. П.* От калькулятора до супер-ЭВМ. История компьютера. М.: Знание, 1981.

А. Р. Белкин,
старший научный сотрудник Института автоматизации и проектирования РАН
С. И. Шумов,
старший научный сотрудник Института автоматизации и проектирования РАН

АНАЛИЗ И ОЦЕНКА ТРАДИЦИОННЫХ И НЕТРАДИЦИОННЫХ МЕХАНИЗМОВ ПОЛУЧЕНИЯ И ОБОБЩЕНИЯ НОВЫХ ЗНАНИЙ

Развитие современной физики, психофизики и психологии парадоксальным образом сблизило научное знание и древнее мистическое восприятие реальности. Становится все более ясным, что ориентация только на материальный, вещественный план реальности ведет к тупику, и лишь гармоничное, сбалансированное и в материальном и в духовном плане общество имеет шанс на будущее. Поэтому так возрос интерес к древним эзотерическим учениям, оккультизму и мистицизму.

Однако далеко не все в оккультно-мистических учениях заслуживает внимания, и задача исследователя — вычленить разумные мысли и ценные идеи, отсеять зерна от плевел. Центральным при этом оказывается вопрос о соотношении традиционно научного и иных способов получения и представления знаний.

Авторами были проведены исследования, результаты которых представлены в виде двух частей. В первой дан краткий критический анализ и оценка существующих наиболее распространенных программных продуктов с точки зрения реализации в них механизмов представления знаний. Сделана попытка прогноза вероятного направления развития программных продуктов с этой точки зрения, представлен универсальный алгоритм объектно-процессной структуризации знаний, который, с одной стороны, является нетрадиционным, с другой — развивает тенденцию прикладного программного обеспечения и, возможно, указывает направление выхода из наметившегося кризиса его развития. (Автор первой части — С. И. Шумов.)

Во второй части рассматриваются механизмы разделения и сопряжения традиционных и нетрадиционных знаний. (Автор второй части — А. Р. Белкин.)

В наше время, во всех индустриально развитых странах все большее внимание уделяется внутреннему миру, субъективному аспекту человеческого бытия. От примитивного рационализма и материализма цивилизация переходит на более высокий виток развития, где рациональное, научное и мистическое знания рассматриваются не как две взаимоисключающие альтернативы, но как два более или менее равноправных, взаимодополняющих способа описания и познания мира.

Большинство существующих механизмов получения и обобщения знаний представляют собой опросник эксперта плюс некоторая база данных, в которую записываются ответы на эти вопросы. В свое время были предложены различные способы представления знаний, но суть их сводилась к следующему: попытаться зафиксировать логику мышления «если... то... иначе», то есть знания предлагалось представлять практически в виде компьютерных программ. Разумеется, вряд ли в таком случае можно говорить о получении новых знаний, скорее всего не удастся зафиксировать даже однозначно имеющиеся, поскольку знания людей достаточно противоречивы. Косвенным подтверждением некоторого кризиса в развитии способов представления знаний является отсутствие популярных компьютерных программ, в основе которых лежат получение и накопление знаний (экспертных систем).

Лет десять назад на рынке программного обеспечения появилась интегрированная система Guru, а также ее отечественный аналог Интерэксперт. С тех пор

новые программные продукты, в которых представлению знаний отводится ведущая роль, не получили сколько-нибудь серьезное распространение. Ведущие производители программного обеспечения выпустили на рынок прекрасные программные продукты, но практически все они служат для фиксирования данных и знаний более или менее традиционными способами.

Попробуем выяснить причины такого положения дел. Дадим характеристику (с точки зрения представления знаний) наиболее распространенным классам программных продуктов, указывая типичных представителей каждого класса.

Текстовые редакторы и издательские системы (MS Word, Word Perfect, Ventura Publisher) дают возможность в короткий срок и с высшим качеством изготовить традиционный линейный печатный текст. Принципиальным удобством являются механизмы работы с заголовками многих уровней, что позволяет легко просмотреть структуру текста, переставить главы и параграфы. Удобно работать способом «рекле»: РЕжим — КЛЕим. Остальные возможности направлены на печать красивыми гарнитурами различных размеров без орфографических ошибок. Но в результате все равно получается точь-в-точь такая же книга, что и в докомпьютерную эпоху.

Графические редакторы (например, PaintBrush) позволяют рисовать без красок, кистей и бумаги, но в общем-то суть осталась та же, что и при Леонардо да Винчи: в конце концов получается картина. Но если хороший текстовый редактор дает возможность найти какое-то слово или слово похожее, то в графическом редакторе это возможно только в случае представления графики в виде примитивов (векторная, или объектная графика). Графические примитивы по аналогии с текстом являются «буквами». Найти же «слово», то есть похожее изображение среди множества имеющихся в наличии, современные граф-редакторы не могут. Таким образом, итогом работы компьютерного художника является опять-таки традиционное изображение, которое также можно считать способом представления знаний.

Несколько дальше пошли **системы автоматизации проектирования** (например, AutoCAD). Первоначально они представляли собой векторные графические редакторы, затем становились все более и более интеллектуальными и начали позволять такие сложные вещи, как, например, автоматическое увеличение отверстия на чертеже, если увеличился болт, который должен в него вставляться. Однако синтез новых конструкций осуществляет все-таки специалист в данной предметной области, а при помощи компьютера он просто реализует в виде чертежей свою идею.

Другое направление развития графических средств — так называемая **метатехнология**. Иерархическая структура таких программных продуктов позволяет создавать большие многоуровневые диаграммы. Примером является MetaDesign американской фирмы Meta Software Corp. Этот изысканный графический редактор предназначен для построения блок-схем и диаграмм, презентации проектов и оформления книг и статей, а также для графического представления самых разнообразных идей. Данный программный продукт реализует теоретические разработки в области сетей Петри. К сожалению, данный класс программных продуктов не дает какой-либо регулярный способ структуризации знаний.

Базы данных (dBase, Paradox, MS Access) позволяют хранить различные виды информации: текст, изображения, звук. Извлечь из базы данных можно не более того, что в нее заложено. Как обычно, фирмы-производители программного обеспечения соревнуются здесь в удобстве пользовательского интерфейса, скорости поиска по запросу, однако ни один из существующих программных продуктов не достиг уровня базы знаний (когда решающее значение представляют связи между различными данными). Впрочем, такая задача разработчиками и не ставилась.

Электронные таблицы (MS Excel, Quattro Pro) — уникальное изобретение, поскольку очень многое можно представить в виде таблиц. Даже газетную полосу можно считать таблицей, в ячейках кото-

рой хранятся различные абзацы и иллюстрации. Возможностью мгновенного создания красивых графиков даже стали злоупотреблять: многие газеты заполняют одну-две полосы различными диаграммами, содержательный смысл которых зачастую сомнителен. Удобно создать демонстрационные слайды. Таблицы незаменимы для статистической обработки, составления прогнозов, моделирования. На наш взгляд, этот класс программных продуктов таит в себе многое такое, что еще по традиции делают другим путем, например решать дифференциальные уравнения.

Языки программирования. Собственно любой язык программирования есть язык представления знаний о функционировании той или иной компьютерной программы. Одна и та же программа может быть записана на различных языках, причем внешне в работе различия могут быть и не видны. Примерами являются образцы программ фирмы Borland, ее реализации языков C++ и Pascal идут параллельно в смысле возможностей.

Что касается особенностей разработки программ, то стало классическим использование структурного программирования. Объектно-ориентированный подход (ООП) позволяет конструировать программы из отдельных готовых блоков, при необходимости модифицировать их, не изменяя исходного текста. Недостатком ООП можно считать невозможность «межвидового скрещивания», то есть наследовать признаки от первоначально различных «пращуров». Кроме того, код программы, написанный с использованием ООП, становится практически не читаемым, кроме того и написание такового становится крупной проблемой. Неспроста большинство реализаций языков программирования обзавелись словом Visual, что означает первоначальное создание интерфейса неязыковыми средствами и дальнейшую возможность автоматической генерации кода на соответствующем языке программирования (C++, Pascal, Basic), описывающего созданный интерфейс.

Современные **системы оптического распознавания** символов (Tiger) позволяют вводить печатные и даже рукописные тексты, но не понимать их. Представление знаний — линейный текст.

Мультимедиа. За этим словом скрывается в первую очередь то, что называется мультипликацией, однако, мультипликация как бы в диалоге с человеком. Компьютер не только выдает что-либо, но и позволяет воспринимать, например человеческую речь. Тем не менее в настоящее время распознаются либо команды, либо опять-таки простая речь. По-настоящему думать компьютер еще не научился.

Особо стоит обратить внимание на **гипертекстовые системы**. Как обычно бывает со всяким модным словом, гипертекстом стали называть очень многие вещи, в лучшем случае любой текст с перекрестными ссылками. Гипертекстовыми стали подсказки в программных продуктах. Несомненно, даже в простейшем случае гипертекст — это шаг вперед по сравнению с линейным представлением текста, тем не менее оно сохраняется как таковое. Как правило, в гипертекстовых системах имеется также объектное дерево, листья которого являются отдельными текстовыми фрагментами. Еще лучше, если помимо текста имеется возможность хранения структурированных изображений (географические информационные и другие аналогичные системы). Указывая деталь такой картинки, можно получить о ней подробную информацию. В гипертекстовых системах также отсутствует механизм перехода в новое качество: синтез принципиально новых знаний на базе существующих.

В чем же причины столь примитивных способов представления знаний в наиболее распространенных программных продуктах?

Перечислим некоторые из них.

1. Программные продукты развиваются эволюционно. Новая версия, как правило, совместима с предыдущей. Новые возможности появляются постепенно. Первоначально компьютерная программа имитировала немашинную технологию плюс давала простейшие возможности. Так, текстовые редакторы изначально имитировали пишущую машинку, но давалась воз-

можность исправлять текст на экране, легко копировать отдельные фрагменты и т. п. Вообще, возможность легкого внесения исправлений является первой отличительной чертой компьютерной технологии. Вероятно, по мере совершенствования программных продуктов, способы представления знаний в них будут становиться все более изощренными, программы будут становиться все более интеллектуальными, реализуя научные разработки в этой области.

2. Научные разработки в области представления знаний начали развиваться в то время, когда персональные компьютеры отсутствовали, и поэтому поначалу они ориентировались либо на безмашинную технологию, либо на работу ЭВМ в пакетном режиме. Поскольку большинство разработок производились людьми далекими от современных технологий программирования, то эти разработки оказались слишком теоретичными и не пригодными для непосредственного использования в новых программных продуктах.

3. Непосредственные разработчики программного обеспечения, как правило, не знакомы со всеми научными разработками в области представления знаний, особенно с теми, которые производились в странах бывшего СССР. По мере интернационализации отечественной науки эти разработки займут свое место в последующих версиях программных продуктов, а также вероятно появление новых классов software.

4. Вероятным направлением «интеллектуальной» эволюции программных продуктов является постепенное использование способов структуризации знаний основанных на системном анализе.

Одно из таких направлений в течение ряда лет разрабатывалось в Российской Федерации, в частности в Институте автоматизации проектирования РАН. Доводом в его пользу является то, что рассматриваемый способ структуризации и представления знаний не противоречит существующим концепциям программных продуктов, является универсальным и может быть использован уже существующими версиями различных классов software.

Дадим краткое описание упомянутого метода структуризации, построенного на концепции **пространств выбора общего вида**.

Описание объектов и описание процессов, в которых они функционируют, рассматриваются, как правило, независимо друг от друга. Общим как в том, так и в другом случае, является построение однотипных декомпозиций, соответствующих либо объектному, либо процессному представлению систем. Наиболее полное описание достигается при слиянии процессного представления с объектным, т. е. при построении единой процессно-объектной структуры для семейства систем.

Для получения единого объектно-процессного представления семейства систем мы должны, начиная с самого верхнего уровня, при проведении декомпозиции процессов одновременно обеспечить декомпозицию и построение соответствующих структур для объектов — компонент начала и результата рассматриваемого процесса.

После того как цикл процессной структуризации повторен нужное количество раз, возникает необходимость установления связей между подпроцессами, а также указания тождественных входов или выходов объемлющих и внутренних процессов. Иначе говоря, требуется указать связи внутри уровня и между уровнями структуризации.

Создаваемые таким образом объектные структуры не будут лишь детализацией соответствующих компонент начала и результатов процессов каждого уровня. Подобъекты такой объектной структуры будут обязательно встречаться в качестве компонент начала и результата при дальнейшей детализации процессов. В итоге мы будем иметь не множество разрозненных объектных структур, сопровождающих единую процессную структуру, а единую объектную структуру. Более того, эти две структуры будут жестко связаны между собой на каждом уровне иерархии подобно тому, как связаны между собой начало, превращение и результат отдельного процесса. Совокупность этих двух взаимопроникаю-

щих структур и есть единое процессно-объектное представление семейства систем.

В качестве внемашинного представления единой процессно-объектной структуры (пространства выбора общего вида) можно использовать картотеку, реальную или виртуальную. Действительно, полученная процессно-объектная структура не содержит ничего, кроме процессов, объектов и связей, и, следовательно, для ее представления в картотеке достаточно описанных ранее средств.

Пространство выбора общего вида дает концентрированную информацию о семействе систем в целом, т. е. одновременно обеспечивает информацией как обо всех возможных системах семейства вообще, так и о любой системе данного семейства в частности. Логика использования пространств выбора общего вида при решении задачи конструирования систем представляет собой постепенный переход от рассмотрения широкого круга систем, которые можно породить на основе данного пространства выбора, к все более конкретному, узкому кругу систем и в конечном итоге к единственной системе данного семейства, наиболее полно удовлетворяющей условиям поставленной задачи. Работать сразу со всеми системами семейства, как правило, бывает невозможно. Но они, будучи в «сжатом» виде представлены с помощью предложенных средств, всегда образуют тот начальный класс систем (полное пространство выбора), с рассмотрения которого начинается процесс дедуктивного выбора, приводящий в результате к выбору единственной конструкции системы, обладающей заданными свойствами, или к выбору ограниченного множества конструкций таких систем. На каждом шаге процесса выбора результатом является некоторый класс систем, среди которых процесс выбора будет продолжаться.

Объектное описание является результатом процессного, поскольку подмножествами процессно-объектной структуры являются деревья «и/или», представляющие собой объектные описания отдельных блоков системы, причем деревья такого вида образуются входами и выходами процессов различных уровней.

Возможность получать объектное описание на основании процессного имеет важное методологическое значение, так как практика создания пространств выбора конкретных предметных областей показывает, что специалист-систематик интуитивно осуществляет сначала именно процессное описание системы (обычно даже не отдавая себе в этом отчет), а затем на основании анализа входов и выходов выделенных процессов дает признаки объектного строения системы.

В качестве итога приведем метод построения пространства выбора общего вида — циклический алгоритм структуризации конструктивно-технологических знаний в предметных областях. Ветвление осуществляется в следующем цикле:

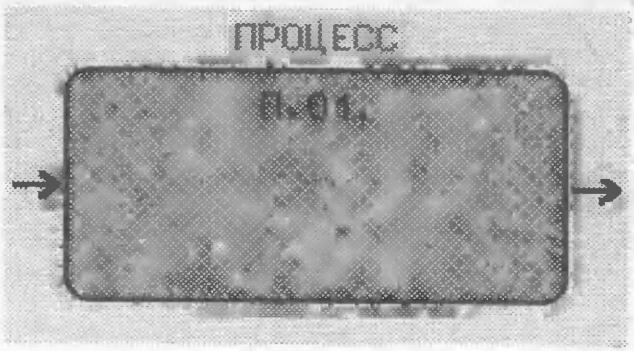


Рис. 1

а) сформулировать понятие конкретного процесса (рис. 1);

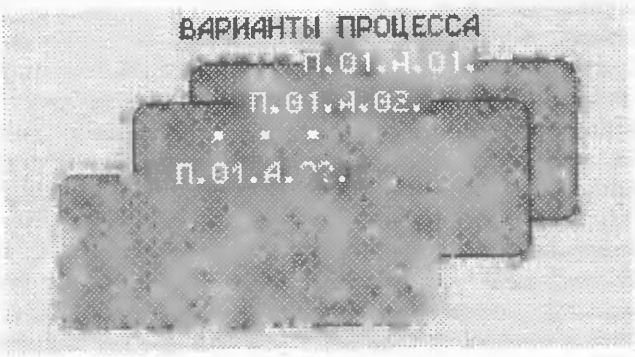


Рис. 2

b) сформулировать все понятия, соответствующие вариантам проведения процесса (рис. 2);

Рис. 3

c) сформулировать «начало», «превращение» и «результат» данного варианта процесса (рис. 3);

Рис. 4

d) сформулировать названия компонент «начала» и «результата» данного варианта процесса (рис. 4);

Рис. 5

e) сформулировать подпроцессы «превращения» данного варианта процесса (рис. 5);

Рис. 6

f) каждый подпроцесс считать процессом и описать его, начиная с пункта a) и т. д.;

Рис. 7

g) установить связи между альтернативными подпроцессами каждого «превращения» (внутриуровневая альтернативная связь) (рис. 7);

Рис. 8

h) установить связи между безальтернативными подпроцессами каждого «превращения» (внутриуровневая безальтернативная связь) (рис. 8);

Рис. 9

i) установить связи между «началом» («результатом») каждого процесса и «началами» («результатами») альтернативных подпроцессов его «превращения» (межуровневая альтернативная связь) (рис. 9);

Рис. 10

j) установить связи между «началом» («результатом») каждого процесса и «началами» («результатами») безальтернативных подпроцессов его «превращения» (межуровневая безальтернативная связь) (рис. 10).

Пользуясь этим правилом, можно, читая линейный текст, расчленить и без дублирования записать:

- все признаки строения объектов с их вариантами,
- все признаки строения процессов с их вариантами.

Принятая буквенно-цифровая индексация позволяет обеспечить следующее:

1. Каждое понятие занимает один и только один узел структуры описания.

2. Текстовые формулировки в узлах структуры выполнены на профессиональном диалекте естественного языка, поэтому сохранены все выразительные средства языка.

3. Индексация обеспечивает возможность добавления любых новых данных о строении объектов и процессов.

4. Конкретный тематический вопрос представлен в виде части структуры и может быть выдан в виде целостного документа.

5. Ветвление понятий и их взаимосвязи представлены явно в обозримой форме. Ни одна связь не выпадает из поля зрения и не может быть обойдена вниманием.

6. Можно выделять уровни конкретизации знаний, а также рассматривать только терминальные ветвления.

7. Эквивалентные серии признаков строения объектов и процессов локализованы как целое, что дает возможность обдумывать и аргументировать выбор признаков для анализа и конструирования.

8. Обеспечено главное свойство конструктивно-технологических знаний — комбинируемость между собой вариантов решения разных частей объектов и процессов.

На основе предлагаемого метода описания конструктивно-технологических знаний может быть создана система представления знаний в экспертных системах и системах автоматизации проектирования, поскольку:

1. Цикл структуризации представляет собой, по существу, алгоритм, на основе которого могут быть написаны сценарий интерактивного опроса эксперта и схема формирования соответствующих записей для базы данных.

2. Для хранения записей можно использовать стандартный индексный файл.

3. Комбинируемость позволяет генерировать варианты систем, возможно, не известные ранее.

4. Хранимые записи можно извлекать из базы знаний в требуемой последовательности.

Из существующих программных продуктов наиболее близко подошли к реализации концепции пространства выбора так называемые фотороботы, синтезирующие человеческие лица из различных начертаний глаз, носа, губ и т. д. Портреты хранятся в виде указания конкретных выбранных вариантов. Кроме того, имеется возможность поиска похожего лица из базы данных.

Что касается процессного представления, то реализованы лишь удобные графические средства описания имеющихся процессных сетей без возможности концептуального синтеза новых технологий.

По-видимому, предложенная объектно-процессная структуризация даст возможность развиваться на новом качественным уровне гипертекстовым системам, причем последние получат возможность синтеза, а их неделимыми фрагментами станут количественные описания, тем самым будет реализован практически диалектический переход количества в качество.

(Продолжение следует)

Л. Е. Самовольнова,
главный специалист отдела информатизации
Министерства образования Российской Федерации

К ВОПРОСУ ОБ АТТЕСТАЦИИ УЧИТЕЛЕЙ ИНФОРМАТИКИ

В 1995 г. все мы будем отмечать 10-летие введения курса информатики в школьные программы. Я поздравляю наших читателей с этой датой и хочу пожелать всем крепкого здоровья, вечной молодости и дальнейших успехов в творческой деятельности.

В Министерство образования Российской Федерации поступает много вопросов от учителей информатики по поводу их аттестации и установки доплат и надбавок к их заработанной плате.

В прошлом году в отделе информатизации были разработаны и направлены во все региональные органы управления образованием в качестве рекомендаций по проведению аттестации учителей и установлению надбавок к заработанной плате два документа:

1. «О порядке установления доплат и надбавок работникам учреждений образования» от 09.04.93 г. № 67-М.

2. «Об организации обучения информатике, аттестации учителей и оплате их труда» от 23.06.93 г. № 112-М.

Судя по откликам читателей, эти документы нуждаются в дополнительном комментарии.

Прежде всего, оба документа носят *рекомендательный* характер и передают компетенцию по решению этих вопросов *местным органам управления* и *администрациям школ*.

В первом документе речь идет об учителях различных специальностей и указывается, что при возложении на работника дополнительных обязанностей размер доплаты определяется учреждением образования САМОСТОЯТЕЛЬНО в пределах выделенных средств.

Понимая, что учителя информатики в учебном заведении занимают ОСОБОЕ место, выполняют значительное число дополнительных функций, требующих особой квалификации, дополнительного времени, ответственности, отдел информатизации Министерства образования Российской Федерации разработал второй документ с рекомендациями по аттестации именно учителей информатики и начислению надбавок к их заработанной плате. Он полностью соответствует первому, но содержит более подробные рекомендации, касающиеся учителей информатики не только общеобразовательных школ, но и учреждений начального и среднего профессионального образования.

Второй документ полностью опубликован в журнале «Информатика и образование» в № 3 за 1993 год и вторично перепечатан в № 1 за 1994 год.

В документе отдельно перечисляются должностные обязанности различных категорий работников образовательного учреждения, непосредственно связанных с работой в компьютерном классе и с другой вычислительной техникой, установленной в образовательном учреждении.

Приводятся отдельно должностные обязанности:

- учителя информатики (раздел I);
- заведующего кабинетом информатики (раздел III);
- заместителя директора образовательного учреждения по внедрению новых

информационных технологий (раздел III).

Должностные обязанности лаборанта, техника или инженера в письме не приводятся, так как полностью повторяют соответствующие обязанности, приведенные в документе «Тарифно-квалификационные характеристики по общеотраслевым должностям служащих» (постановление Министерства труда Российской Федерации от 10.11.92 г. № 30).

Если на учителя информатики кроме его непосредственной работы возлагаются еще дополнительные обязанности заведующего кабинетом или лаборанта, то ему следует установить надбавку к заработной плате в соответствии с объемом работы, квалификацией учителя, финансовыми возможностями образовательного учреждения.

Учитывая многочисленные письменные и устные запросы органов управления образованием, а также слушателей совещаний-семинаров по переводу работников на условия оплаты труда на основе единой тарифной сетки (ЕТС), Министерство образования Российской Федерации направило во все органы управления образованием сборник «Ответы на вопросы, возникающие при введении дифференцированной оплаты труда работников образования на основе единой тарифной сетки» (91-М от 19.05.93 г.).

В частности, на вопрос 16 о том, сохраняется ли при введении ЕТС право работника на получение доплат в размере до 12% за работу, связанную с вредными условиями, и как определяется конкретный размер доплаты, например, учителям школы, лаборантам и т. д., дается ответ, что порядок установления доплат сохраняется и доведен до органов управления образованием письмом от 12.01.93 г. № 10/32-Т. При его применении администрация учреждения прежде всего должна принять все необходимые меры, исключающие неблагоприятные условия труда для работников.

Для учителей информатики это сводится к оборудованию кабинета информатики по правилам, предусмотренным соответствующими документами (дисплеи желательно оборудовать защитными экранами и т. п.).

Размер же доплаты определяется объемом рабочего времени учителя непосредственно за экраном ЭВМ. Если, к примеру, это время составляет 70% всего рабочего времени, то надбавка будет исчисляться в 8,4% (12%×0,7=8,4) к общему окладу, включая все надбавки.

В соответствии с постановлением Госкомтруда СССР и Секретариата ВЦСПС от 15.05.90 г. № 193/7-69 «О совершенствовании организации заработной платы и введении новых ставок и должностных окладов работников учреждений и организаций системы народного образования» руководителям учреждений и организаций системы Гособразования СССР предоставлено право по согласованию с профсоюзными комитетами устанавливать доплаты рабочим, специалистам и служащим за неблагоприятные условия труда (приложение 1 к приказу Гособразования СССР от 20.08.90 г. № 579 «Положение о порядке установления доплат за неблагоприятные условия труда в учреждениях и организациях системы Гособразования СССР»).

Приводим выдержки из этого документа.

«Доплаты за неблагоприятные условия труда устанавливаются по результатам аттестации рабочих мест или оценки условий труда лицам, непосредственно занятым на работах, предусмотренных Перечнем, а также Типовыми перечнями работ с тяжелыми и вредными, особо тяжелыми и особо вредными условиями труда, утвержденными Госкомтруда СССР и Секретариатом ВЦСПС по отраслям народного хозяйства (приложение 2), и начисляются за время фактической занятости работников на таких рабочих местах или в таких условиях труда.

При последующей рационализации рабочих мест и улучшении условий труда доплаты уменьшаются или отменяются полностью.

На работах с тяжелыми и вредными условиями труда доплаты устанавливаются до 12% должностного оклада (ставки), а на работах с особо тяжелыми, особо вредными условиями труда — до 24% должностного оклада (ставки).

Аттестация рабочих мест или оценка условий труда в учреждении, организации проводится экспертной комиссией в соответствии с Рекомендациями, согласованными с Центральным комитетом профсоюза работников образования и науки (постановление Президиума от 28.04.90 г. № 37) и утвержденными Госкомитетом СССР по народному образованию 30.03.90 г..

Экспертная комиссия образуется совме-

стным решением администрации и профсоюзного комитета из числа наиболее квалифицированных сотрудников, представителей профсоюзного комитета, службы охраны труда учреждения, организации. В качестве членов экспертной комиссии могут привлекаться специалисты других служб и организаций.

Экспертную комиссию возглавляет заместитель руководителя учреждения, организации.

Руководитель подразделения совместно с профсоюзным комитетом подразделения составляет характеристики работ и условий труда конкретных сотрудников (группы сотрудников) с указанием действующих опасных и вредных производственных факторов, предусмотренных ГОСТом 12.0.003-74 «Опасные и вредные производственные факторы. Классификация», по каждому виду работы.

Ответственность за достоверность характеристик условий труда сотрудников, действующих опасных и вредных производственных факторов на рабочих местах несет руководитель структурного подразделения. О всех изменениях условий труда сотрудников руководитель подразделения должен своевременно представлять дополнительные материалы в экспертную комиссию. По заключению экспертной комиссии при последующей рационализации рабочих мест и улучшении условий труда доплаты уменьшаются или отменяются полностью.

На основании заключения экспертной комиссии руководитель учреждения, организации по согласованию с профсоюзным комитетом утверждает перечень конкретных работ, на которые устанавливаются доплаты за неблагоприятные условия труда, и размеры доплат по видам работ. Перечень работ и размер доплат включаются в коллективный договор.

Конкретные размеры доплат и их срок устанавливаются по учреждению, организации».

В приложении 2 к этому документу в перечне видов работ с тяжелыми и вредными условиями труда, на которые устанавливается доплата до 12%, имеется п. 1.164 «Работа за дисплеями ЭВМ». Этот пункт непосредственно имеет отношение к учителям информатики.

В информационно-методическом письме отдела информатизации об аттестации учителей информатики подробно рассматриваются вопросы, связанные с установлением разрядов оплаты труда по ЕТС с учетом стажа педагогической работы. При этом обращается внимание на то, что в педагогический стаж работы учителя информатики включается время работы на предприятиях, в учреждениях и организациях и время службы в Вооруженных силах по специальности, соответствующей профилю преподаваемого предмета, дисциплины (подпункт «е» пункта 5 приложения 6 к Инструкции о порядке исчисления заработанной платы работников просвещения, утвержденной приказом Минпроса СССР от 16.05.85 г. № 95, в редакции приказа Гособразования СССР от 08.06.90 г. № 400).

При очередной аттестации на присвоение квалификационных категорий рекомендуем определять соответствие учителей информатики требованиям тарифно-квалификационных характеристик с учетом следующих особенностей:

1. На II квалификационную категорию могут претендовать учителя, отвечающие общим требованиям, предъявляемым к учителям информатики, и прошедшие курс обучения базовой подготовки учителей информатики, умеющие разрабатывать собственную оригинальную методику и использовать ее в процессе обучения в школе; стаж работы в должности учителя информатики должен составлять не менее 1 года.

2. На I квалификационную категорию могут претендовать учителя, отвечающие требованиям, предъявляемым к учителям информатики II квалификационной категории, а также использующие собственную оригинальную методику в процессе обучения, которая рекомендована методическими службами другим учителям к использованию в работе; стаж работы в должности учителя информатики должен составлять не менее 2 лет.

3. На высшую квалификационную категорию могут претендовать учителя, отвечающие требованиям, предъявляемым к учителям информатики II и I квалификационных категорий, имеющие собственные методические разработки (пособия, программы и т. п.), положительно оцененные методическими службами и опубликованные в специальных научно-методических педагогических изданиях; стаж работы в должности учителя информатики должен составлять не менее 4 лет.

С. В. Русаков,
доктор физико-математических наук, Пермский государственный университет, Лаборатория информатизации образования Пермской области
И. Г. Семакин,
кандидат физико-математических наук, Пермский государственный университет, Лаборатория информатизации образования Пермской области

БАЗОВЫЙ КУРС ОИВТ: «ПЕРМСКАЯ ВЕРСИЯ» (часть 3)

Тема 5. ЭВМ и информация

В хорошо известной азбуке Морзе каждый символ (буква, цифра, знак препинания) кодируется определенным сочетанием точек и тире. Другими словами, алфавит кода Морзе содержит всего два знака: точка и тире. Такой способ кодирования можно назвать д в о и ч н ы м . Следовательно, в «морзянке» символ (точка или тире) несет 1 бит информации.

Непосредственными предшественниками ЭВМ были релейные вычислительные машины. Электромагнитное реле — это двухпозиционный переключатель, который имеет два состояния: «включено», «выключено». Каждое такое реле несет 1 бит информации. Машина содержала тысячи таких реле, которые в процессе счета непрерывно переключались. Поскольку реле — это механическое устройство, то его инерционность (т. е. задержка при переключении) достаточно велика, что сильно ограничивало быстродействие таких машин. Скорость счета лучшей отечественной релейной машины РВМ-1 составляла 50 сложений и 20 умножений в секунду.

Технической базой для электронно-вычислительных машин стала электронная схема, которая называется т р и г г е р о м . Триггер состоит из двух электронных ламп (триодов), соединенных так, что при работе одна из них открыта (идет ток), другая закрыта (не идет ток). Эти состояния могут меняться. Соответственно, триггер имеет два устойчивых состояния. Поэтому его называют электронным реле. Инерционность электронного реле в тысячи раз меньше механического. Первой машиной на электронных триггерных схемах стала созданная в 1945 году американская ЭВМ ENIAC. Скорость счета этой машины по сравнению с РВМ возросла приблизительно в 100 раз.

Позже электронные лампы были заменены на транзисторы, транзисторы — на микросхемы. Многократно возросла информационная емкость и скорость ЭВМ, но двоичный принцип работы компьютера сохранился.

Согласно принципу Неймана, программа хранится в той же памяти, что и данные.

Внутренняя память ЭВМ обладает двумя основными свойствами: дискретностью и адресуемостью.

Память дискретна — это значит память состоит из некоторых «частиц». Частица памяти называется бит (так же, как единица информации). Один бит — это двоичный разряд памяти. Он хранит двоичный код (0 или 1). Слово «бит» — сокращение от английского «binary digit» — двоичная цифра.

Биты памяти — это упорядоченная последовательность двоичных разрядов. Если эту последовательность разбить на группы по 8 разрядов, то каждая такая группа образует байт памяти.

Следовательно, бит и байт — это не только единицы измерения количества информации, но и структурные единицы памяти ЭВМ. А связь между ними очень простая: 1 бит памяти хранит 1 бит информации; 1 байт памяти хранит 1 байт информации.

Объем памяти ЭВМ измеряется в килобайтах, мегабайтах, гигабайтах. Например, оперативная память школьных компьютеров УКНЦ, Корвет, Агат равна 64 Кб; Ямаха имеет ОЗУ на 256 Кб. Компьютеры серии IBM PC имеют внутреннюю память от 1 Мб и выше.

Байты в памяти ЭВМ пронумерованы. Нумерация начинается от нуля. Порядковый номер байта называется его адресом.

Адресуемость памяти означает, что любая информация заносится в память и извлекается из нее по адресам.

Имеется еще одна структурная единица памяти компьютера. Она называется машинным словом. У разных типов компьютеров машинное слово бывает разным. У самых простых типов ПК (бытовых, учебных) машинное слово равно 1 байту (8 бит). Такие машины называются восьмиразрядными (Агат, Корвет, Ямаха). Двухбайтовое машинное слово (16 бит) у школьного компьютера УКНЦ, а также у профессиональных ПК IBM PC/86, 286. У машин типа IBM PC/386, 486 машинное слово равно 4 байтам (32 бита).

Адрес машинного слова в памяти равен адресу младшего байта, входящего в это слово. Поэтому в 8-разрядных ЭВМ адреса слов в памяти меняются через единицу; в 16-разрядных — через 2, а в 32-разрядных — через 4.

Вот, например, схема адресации памяти в 16-разрядной машине:

0 слово	0 байт	1 байт
2 слово	2 байт	3 байт
4 слово	4 байт	5 байт
...	...	...

В ходе работы ЭВМ процессор непрерывно обменивается информацией с внутренней памятью.

Куда помещает процессор прочитанную из памяти информацию? Оказывается, внутри процессора есть собственная память. Она представляет собой совокупность регистров. Размер регистра процессора равен размеру машинного слова. Следовательно, обмен информацией между процессором и памятью производится словами.

На устройствах внешней памяти (магнитных носителях) информация также представлена в двоичном коде: состоянием намагниченных и ненамагниченных участков на дорожках ленты или диска. Как уже говорилось раньше, эта информация объединена в файлы. Из сказанного следует, что в компьютере имеется несколько мест, где хранится двоичная информация. Это регистры процессора, внутренняя и внешняя память. Их можно рассматривать как три уровня памяти ЭВМ.

Рис. 1

При работе компьютера происходит постоянный обмен информацией между

ЭВМ и информация

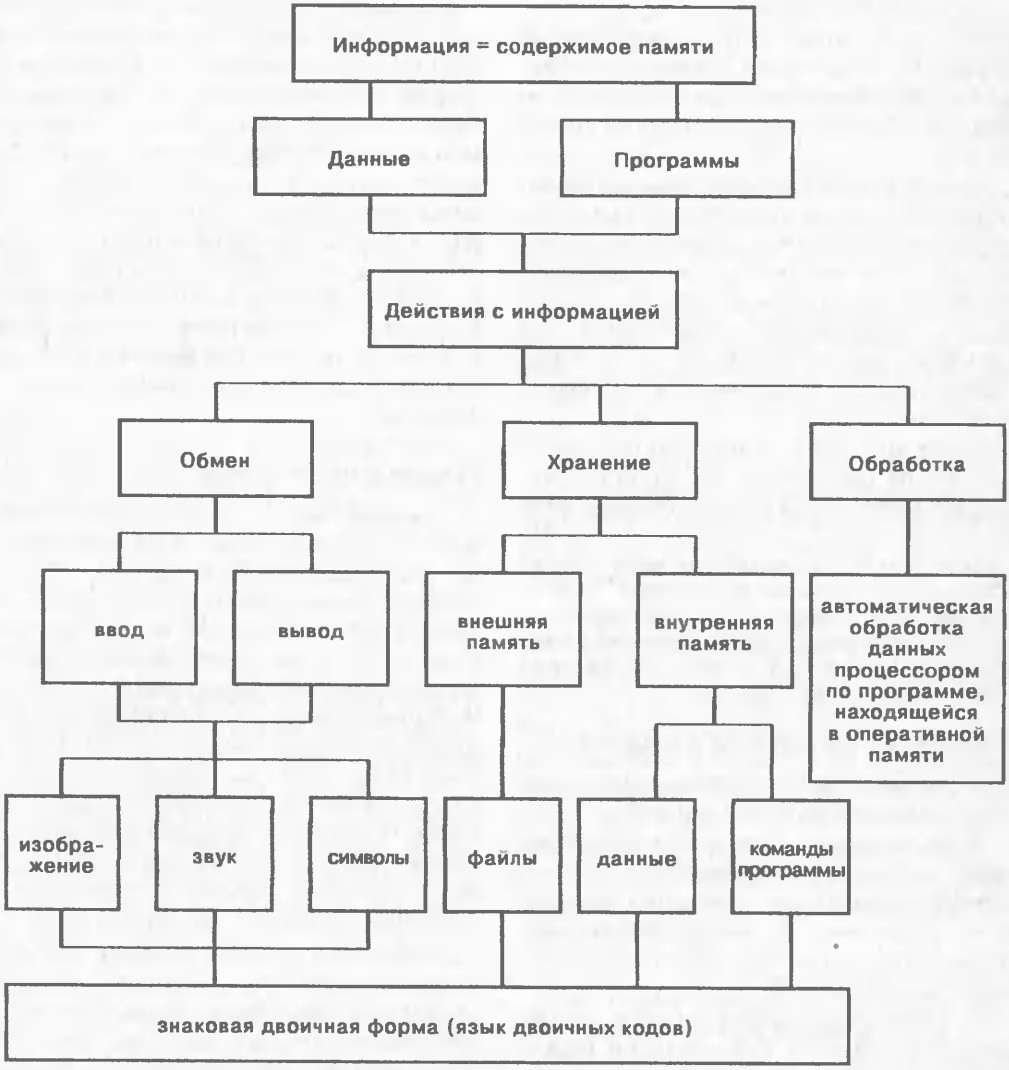


Рис. 2

тремя типами устройств памяти. Эти устройства отличаются быстродействием (временем чтения — записи) и объемом. При движении сверху вниз в этой иерархии уменьшается быстродействие и растет объем.

Кроме обмена информацией между различными узлами компьютера (*внутренний обмен*) происходит также и *внешний обмен информацией*. Внешний обмен — это прием-передача информации между ЭВМ и человеком; между ЭВМ и управляемыми ею техническими устройствами; между связанными друг с другом ЭВМ. Прием информации во внутреннюю память компьютера называется *вводом*, а передача данных из памяти ЭВМ во вне — *выводом*.

При вводе информации с клавиатуры нажатие каждой клавиши преобразуется в определенный двоичный код. При выводе

на экран дисплея или на печать двоичные коды преобразуются в символьное или графическое изображение.

На современных ЭВМ используются специальные устройства для ввода изображения и звука, которые преобразуют вводимую информацию во внутренний двоичный код.

Для обобщения изложенного материала полезно рассмотреть схему, представленную на рис. 2. Сопоставьте ее со схемой (рис. 1) в теме «Человек и информация» (№ 5). Отметим сходство этих схем, которое подтверждает тезис о том, что компьютер по своей организации представляет модель информационной функции человека.

Различие состоит в том, что современные ЭВМ не работают с образной информацией, не общаются на естественных языках.

План урока. Идея двоичного кодирования информации. Свойства внутренней памяти ЭВМ. Единицы измерения памяти. Данные и адреса. Уровни памяти ЭВМ. Внешний обмен информацией; ввод и вывод. Обсуждение схемы «ЭВМ и информация».

Контрольные вопросы и задания

1. Где впервые было использовано двоичное кодирование информации?

2. В чем удобство двоичного кодирования для вычислительных машин?

3. Что означает: «дискретность памяти ЭВМ»; «адресуемость памяти ЭВМ»; «машинное слово»; «регистр процессора»? Каков размер регистра?

4. Какие три уровня памяти можно выделить в ЭВМ? В чем различие между ними?

5. Что такое «ввод» и «вывод»?

6. Какие виды информации могут вводиться и выводиться? В какой форме они хранятся в памяти компьютера?

7. Рассмотреть схему «ЭВМ и информация». Сопоставить ее со схемой «Человек и информация». Проанализировать сходство и различие.

Тема 6. Компьютер пишет

В этом разделе мы рассмотрим применение ЭВМ для создания печатной продукции. Это одно из самых массовых приложений компьютерной техники.

Данный раздел, как и последующие, относящиеся к прикладным вопросам информатики, излагается в инвариантной форме: что изложение не ориентируется на какой-то конкретный вариант программного обеспечения (конкретный текстовый редактор). Описываются общие принципы организации и использования программного обеспечения данного типа (текстовых редакторов). Используется единая методическая схема изложения материала. Привязка к конкретным программам для различных типов КУВТ дается в табличной форме (для текстовых редакторов — в табл. 2).

Тексты в памяти ЭВМ

Всякий текст — это последовательность символов. Символьный алфавит компьютера содержит 256 знаков. Мы уже знаем, что один символ из такого алфавита несет 8 бит информации и, следовательно, в памяти ЭВМ занимает 1 байт. Все символы в алфавите пронумерованы от 0 до 255. Каждому номеру соответствует 8-разрядный двоичный код от 00000000 до 11111111. Этот код — просто номер символа в двоичной системе счисления. Подробно о системах счисления речь пойдет позже. Пока достаточно иметь представление о том, что у каждого символа имеется свой индивидуальный двоичный код.

Таблицу, в которой ставятся в соответствие символы, их десятичные номера и двоичные коды, условно называют таблицей кодировки. Для разных типов ЭВМ используются различные таблицы кодировки. Наиболее распространенной таблицей на персональных компьютерах является код ASCII (American Standart Code for Information Interchange) — Американский стандартный код для информационного обмена. Он используется на большинстве типов персональных компьютеров. Первые 128 кодов из ASCII, начиная с «00000000» и кончая «01111111», имеют международный стандарт, т. е. являются унифицированными. Это делает возможным использование кода ASCII в международных телекоммуникационных системах. Остальные 128 кодов, начиная с «10000000» и кон-

чая «11111111», используются в национальных системах кодировки, например, для кодирования символов русского алфавита. Примером другой системы символьной кодировки является код ДКОИ (Двоичный код обмена информацией). Он был введен для машин серии IBM 360/370. Ниже приводятся фрагменты таблиц кодировок ASCII и ДКОИ.

Таблица 1

№	Символ ДКОИ	Символ ASCII	Двоичный код
106	┘	j	01101010
107	,	k	01101011
108	%	l	01101100
109	–	m	01101101
110	>	n	01101110
111	?	o	01101111

Не все коды отображаются на экране в виде символов. Некоторые выполняют роль управления печатью или выводом на экран. Такие коды называются управляющими.

Текстовые редакторы

Стремление упростить работу с различными видами текстов (служебными бумагами, конспектами лекций, газетами, журналами, книгами и т. д.) привело к созданию большого количества программного обеспечения, ориентированного на решение этих проблем и называемого *текстовыми редакторами* (ТР) или *текстовыми процессорами*. Среди профессиональных ТР наибольшее распространение получили Lexicon, ChiWriter, Multy-Edit, Word, TeX. Для школьных компьютеров, используемых при изучении информатики, также имеется значительное число разнообразных ТР (табл. 2).

Общее назначение ТР — ввод текстов в компьютер и их редактирование, сохранение на ВЗУ и печать на бумаге.

При изучении ТР можно выделить следующие методические направления работы:

1) обучение практическим навыкам работы с конкретным ТР;

2) обучение принципам организации и функционирования ТР;

3) показ и уяснение всех возможностей ТР и их места в интеллектуальной деятельности человека;

4) развитие и закрепление алгоритмической линии курса ОИВТ: трактовка ТР как исполнителя с системой команд, ориентированной на определенный круг задач;

5) углубление понятия архитектуры ЭВМ: рассмотрение устройств ПК, задействованных при работе с ТР (клавиатуры, дисплея, оперативной памяти, локальной сети, накопителя на магнитных дисках, принтера);

6) углубление понятия архитектуры современного ПО: «дружественный» интерфейс, программы обслуживания сети, файловая система в составе ОС, архиваторы файлов и т. д.;

7) углубление понятия информации: рассмотрение вопросов ее хранения в памяти ЭВМ, создания зрительных образов различных символов на экране дисплея и бумаге.

Примером интегрированного подхода при изучении ТР является «Роботландия» [1], где используется редактор Микрон. Курс «Роботландия» рассчитан на учеников младших классов. На наш взгляд, восьмиклассникам этот же материал следует давать более углубленно, с изучением более развитых редакторов (учебных или профессиональных), с рассказом об используемых аппаратных и программных средствах.

За основу изложения примем концепцию исполнителя алгоритмов.

Алгоритм — это последовательность команд для исполнителя, приводящая к искомому результату. Исполнителем алгоритма может быть компьютер с соответствующим программным обеспечением (например, текстовым редактором). Возможны два режима выполнения алгоритма: *командный режим* и *программный режим*.

В командном режиме человек отдает компьютеру команды по одной в алгоритмической последовательности, компьютер их исполняет. В этом варианте, строго говоря, компьютер является *исполнителем команд*. Исполните-

Рис. 3

лем же всей работы является человеко-машинный комплекс.

В программном режиме весь алгоритм сначала заносится в программную память исполнителя, а затем автоматически выполняется.

Текстовый редактор является инструментом в руках человека, пользуясь которым человек создает нужный документ. ТР — это специализированный исполнитель, способный работать только в командном режиме.

Заметим, что в ряде ТР имеются так называемые макросредства, позволяющие запоминать небольшую последовательность команд в виде программы (макрокоманды), и затем выполнять ее автоматически.

Изложение материала будем вести по ранее предложенной методической схеме, которая в приложении к данной теме изображена на рис. 3.

Аппаратные средства

Минимальный набор, позволяющий эффективно работать с текстовыми редакторами, составляют клавиатура, дисплей, оперативная память, каналы связи локальной сети, ВЗУ на магнитных дисках, принтер.

Клавиатура. Знакомство с клавиатурой с помощью клавиатурных тренажеров позволяет ученикам освоить только ее центральную часть — клавиши, с помощью которых осуществляется ввод различных символов. В ТР активно задействованы все остальные группы клавиш: функциональные, управляющие, переключения режимов, перемещения курсора, прочие клавиши.

Дисплей. При работе с ТР пользователь имеет дело с экраном, работающим в текстовом режиме. Экран поделен на строки, в каждую из которых можно поставить ограниченное количество символов. Можно считать, что экран представляет из себя сетку, каждая клеточка которой характеризуется номерами строки и столбца (позиции в строке). В свою очередь позиция на экране представляет из себя мозаичное поле из заданного числа строк и столбцов. Зрительный образ символа формируется набором закрашенных точек. Некоторые ТР позволяют таким образом создавать свои собственные шрифты.

Оперативная память служит для хранения программы ТР и обрабатываемого текста. Представляет из себя набор адресуемых ячеек (байтов), в каждом из которых можно хранить код одного символа. При работе ТР оперативная память делится на несколько частей (листов), о назначении которых будет рассказано ниже.

Локальная сеть школьного компьютерного класса служит для загрузки ТР на рабочие места учеников с рабочего места учителя. Кроме того, по ней транспортируются текстовые файлы, сохраняемые на ВЗУ или считываемые с него. Важнейшей характеристикой таких обме-

нов является скорость передачи информации.

ВЗУ на магнитных дисках состоят из дисководов и собственно магнитных дисков. Дисководы — устройства для считывания (записи) информации с помощью считывающих (записывающих) головок. Магнитные диски — носители информации — разделены на дорожки, каждая из которых, в свою очередь, поделена на сектора (блоки), состоящие из «клеточек» (байтов). В одну клеточку помещается один символ. Число дорожек на диске, число секторов на дорожке и число байтов в секторе определяют общий объем информации, которую можно хранить на диске. Операция по разметке диска на дорожки, сектора и байты называется форматированием и выполняется с помощью специальных программ. Место расположения файла на диске фиксируется в оглавлении диска. Поиск необходимой информации компьютер выполняет следующим образом:

- по имени файла в оглавлении диска ищется его место (дорожка и номер первого сектора) на диске;
- считывающая головка подводится к соответствующей дорожке и отсчитывается нужный сектор;
- содержимое сектора считывается в буферную (оперативную) память;
- при дальнейшем чтении информации из файла производится считывание следующих секторов.

Принтер. Тексты, подготовленные с помощью ТР, могут быть распечатаны на принтере. При обучении работе с принтером отметить следующие моменты: включение, режимы On-Line/Off-Line, прогон бумаги.

Формирование зрительного образа символа с помощью мозаичного принтера аналогично принципам его построения на экране. Скорость печати текста зависит от вида шрифта, которым он печатается. Принтер имеет собственную память (буфер), что позволяет ему работать автономно от компьютера, когда текст (остаток текста) полностью помещается в буфер.

Рассмотрим теперь программную часть (рис. 1), которая, собственно, и составляет текстовый редактор.

Среда ТР

Рабочим полем ТР является экран дисплея, на котором отображаются все действия, выполняемые ТР. Важным элементом среды является интерфейс — это те средства, с помощью которых пользователь может общаться с ТР и управлять им. На сегодняшний день наиболее предпочтительным считается интерфейс в форме меню, из которого специальным маркером (выделенным цветом) можно выбирать те или иные команды ТР. К сожалению, не все школьные ТР имеют развитую систему меню.

Одновременно с меню (если оно есть) на экране высвечивается строка состояния, в которой дается информация о текущем состоянии ТР (режимы работы, позиция курсора и пр.).

Текст, обрабатываемый с помощью ТР, хранится в оперативной памяти и визуально может быть представлен в виде рулона бумаги (в некоторых ТР — разделенного на страницы), длина и ширина которого в большинстве случаев не позволяют целиком наблюдать его на экране. Таким образом, экран можно считать своеобразным окном, через которое пользователь просматривает текст. Для перемещения этого окна по тексту используются специальные клавиши. Есть ТР, позволяющие открывать несколько таких окон «над» соответствующим количеством текстов.

Кроме основной памяти (рулона), где ТР хранит обрабатываемый текст, в его распоряжении находится ряд дополнительных листов памяти, к которым относятся: лист удаленных строк; буфер («карман») для хранения копируемых фрагментов текста; справочник (подсказка, HELP); словарь.

В некоторых ТР лист памяти для удаленных строк и «карман» совмещены. В большинстве школьных ТР словарь, с помощью которого можно отыскивать орфографические ошибки, отсутствует. Некоторым эквивалентом словаря может служить режим работы «Диктант» в ТР Микрон «Роботландии». Если в ТР допускается многооконный режим, основная память делится на разделы, закрепляемые за каждым из окон.

Курсор — световое пятно на экране —

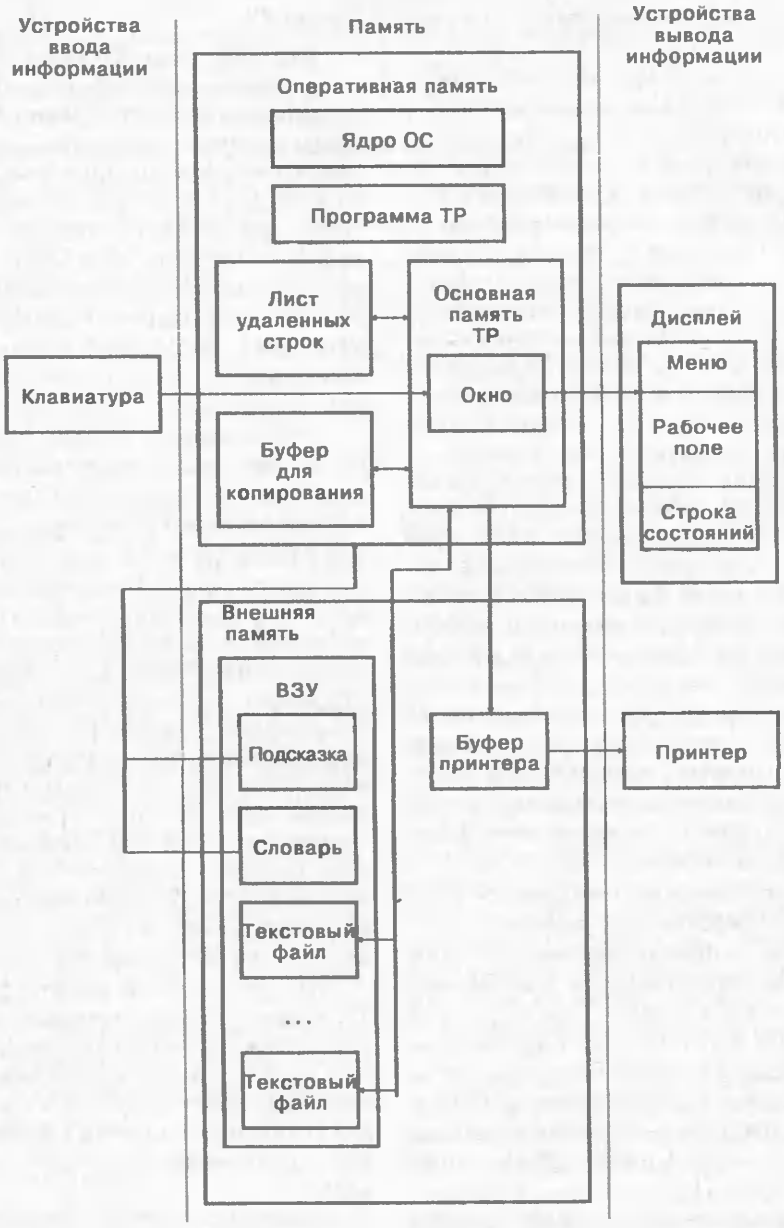


Рис. 4

место активного воздействия исполнителя на рабочее поле. При вводе символа с клавиатуры он занимает место курсора, который сдвигается на одну позицию вправо. Передвигая курсор, можно перемещать окно по основной памяти (тексту). Наличие курсора в рабочем поле указывает на то, что исполнитель находится в режиме ввода-редактирования текста. Координаты курсора в тексте (или на странице) — номер строки и номер столбца — отображаются в строке состояния исполнителя.

Архитектуру исполнителя ТР как совокупность аппаратных и программных средств схематично можно отразить следующим образом (рис. 4).

Режимы работы ТР

Ввод-редактирование текста. Состояние исполнителя, находящегося в этом режиме, отражается в строке состояний. Отметим основные компоненты состояния: координаты курсора, вставка/удаление, строчные/заглавные (нижний/верхний регистр), шрифт (рус/лат), разметка строки (абзац, разрешенное число символов), выравнивание правого края, перенос.

Последние два пункта есть не во всех школьных ТР. При наличии многооконности, каждое из окон представляет из себя самостоятельное рабочее поле.

Форматирование. Работа в этом режиме позволяет скомпоновать текст (фрагмент текста) в требуемом виде, установив отступ с левого края, красную строку, число позиций в строке и т. д.

Обращение за подсказкой. При переходе в этот режим на экране открывается дополнительное окно, содержащее краткую справочную информацию о работе в ТР. Подсказка может содержать несколько листов текста, в этом случае указывается способ перелистывания.

Орфографическая проверка. Этот режим работы возможен лишь при наличии словаря. При проверке текста фиксируются слова, отсутствующие в словаре, что является косвенным свидетельством орфографической ошибки. Режим орфографической проверки может быть совмещен с режимом редактирования.

Обмен с ВЗУ возможен в двух направлениях: сохранение (запись, SAVE) и загрузка (считывание, LOAD) текста, который при хранении на ВЗУ называется файлом (текстовым файлом). Каждый файл характеризуется своим персональным именем. Магнитный носитель может быть структурирован на каталоги (директории) и подкатологи. В этом случае возникает понятие пути к необходимому каталогу. Если компьютер имеет несколько ВЗУ, то они получают название устройств A:, B:, C: и т. д. Тогда при работе с файлами возникает дополнительная проблема выбора устройства.

Печать. В этом режиме осуществляется вывод текста на бумагу. С помощью ТР можно управлять принтером, определив шрифт, вид печатаемой страницы (левое и правое поля, число строк на странице), способ нумерации страниц (местонахождения номера страницы, порядок нумерации).

Система команд ТР

Систему команд текстового редактора можно условно разделить на следующие группы команд :

команды интерфейса — вход в меню и выход из него, перемещение маркера по меню, выбор нужного пункта меню;

команды изменения состояния ТР, позволяющие в режиме редактирования выбрать шрифт, нижний/верхний регистр, режим вставки/замены и т. д.;

команды перемещения по тексту, которые разделяются на две группы: пошаговое перемещение (стрелки вправо, влево, вверх, вниз); быстрое перемещение по тексту (переход в начало/конец строки, начало/конец текста, листание страниц, прыжок на метку и т. д.);

команды посимвольного редактирования, позволяющие внести исправления в текст (вставить/удалить/заменить символ, заменить строчную букву на заглавную и т. д.);

команды работы со строками, позволяющие манипулировать со строками (удалить, вставить пустую, рассечь на две части, «склеить» две последовательные строки);

поиск по образцу служит для отыскания в тексте мест, содержащих указанный набор символов (букву, слово, фразу и т. д.);

копирование включает в себя выделение копируемого фрагмента текста и помещение его в буфер (в некоторых ТР эти процессы идут параллельно), воспроизведение содержимого буфера в требуемом месте текста;

форматирование позволяет придать тексту требуемый вид: установить число символов в строке, выровнять

правый край, выделить поля и красную строку и т. д.

В большинстве ТР команды исполнителя реализуются двумя способами: с помощью функциональных, управляющих и прочих клавиш специального назначения; выбором соответствующего пункта в меню.

Данные

Текстовый редактор работает с символьной информацией, в которой можно выделить следующие виды структур: символы; слова; строки символов; фрагменты текста (блоки); файлы.

В системе команд имеются команды работы с каждой из этих структур как с единым целым.

Символ — минимальный объект, подлежащий обработке ТР. С символами оперируют в основном при вводе текстов и при их посимвольном редактировании.

Слово — набор символов, ограниченный слева и справа пробелами или знаками препинания; используется в командах быстрого перемещения по тексту, удаления фрагментов текста.

Строка символов — данные, занимающие в тексте одну строку, размеры которой могут быть четко очерчены форматированием. Исполнитель ТР имеет ряд команд, оперирующих строками (вставить/удалить, соединить/разорвать, перейти в начало/конец и т. д.).

Фрагмент текста — набор последовательных строк, с которым можно работать как с единым целым (удалять, перемещать, копировать и т. д.).

Понятие файла связано с хранением, считыванием и записью информации на МД. В отличие от ранее рассмотренных структур данных, файлы являются поименованными данными, т. е. каждый из них имеет свое персональное имя.

В ряде ТР для быстрого перемещения по тексту («прыжков») используются метки

При изучении этой темы рекомендуем познакомиться со следующей литературой: [1] — гл .6; [2] — введение; парагр.16—18, 21; приложение 3.3; [3] — парагр. 9—11, 49, 51, 52; [4] — парагр. 1, 34, 35; [5] — парагр. 1, 2, 19, 24; [6], [7].

Как рациональнее использовать предложенный материал?

Предлагаемое вашему вниманию пособие — своеобразный «полуфабрикат». Сохраняя общую методическую схему, необходимо адаптировать его к тем средствам, с которыми работает учитель в классе: конкретному типу компьютера, программному обеспечению (ТР). Эта рекомендация относится в равной степени ко всем последующим прикладным темам курса.

Порядок выполнения определенного задания следует записывать в форме алгоритма, по которому ученик будет управлять работой ТР. Этот алгоритм составляется из команд, входящих в СКИ ТР. Вот пример алгоритма перемещения фрагмента текста из одного места в другое:

1. Установить курсор в начало фрагмента.
2. Отдать команду «Начать выделение».
3. Установить курсор в конец фрагмента.
4. Отдать команду «Выделить фрагмент».
5. Установить курсор в позицию переноса фрагмента.
6. Отдать команду «Перенести фрагмент».

И еще одно замечание, которое касается конкретно изучения текстовых редакторов. Ученики должны вынести из этой темы понимание того, что ТР — это не просто заменитель пишущей машинки, а универсальное средство работы с текстами. Он представляет человеку разнообразные возможности манипулирования с готовыми текстовыми документами, сохраненными в файлах. Из фрагментов готовых текстов можно формировать новые тексты; тексты можно передавать по линиям связи компьютерной сети и пр. В планах 5—6-го уроков предлагаются задания на составление «досье» (можно придумать и другие примеры). На этом материале показывается, как можно использовать текстовый редактор даже в режиме несложной информационно-поисковой системы (в тех ТР, где реализован контекстный поиск).

Вариативное поурочное планирование

Урок 1

Теоретическая часть. Хранение символьной информации в памяти ЭВМ, принципы кодировки. Назначение ТР и сфера их применения. ТР — исполнитель, общая схема исполнителя (рис. 1). Определение исполнителя команд, понятие о среде исполнителя, его системе команд. Показать (на плакате), как выглядит экран в процессе работы с ТР. Объяснить назначение курсора, строки состояния ТР. Дать краткую характеристику пунктов меню (команды входа/выхода, маркер). Показать (на плакате) изображение клавиатуры и ввести терминологию для различных групп клавиш.

Практическая часть. Загрузить текстовый редактор на рабочие места учеников, обратив их внимание на сам процесс загрузки (работа ВЗУ, локальной сети, программа загрузки, программа ТР). Войдя в ТР, ученики должны набрать небольшой текст (5—10 строк) с раздаточного материала. Отрабатывается переход на следующую строку, переключение с нижнего на верхний регистр и обратно, переключение с русского на латинский шрифт и обратно. Фиксируется внимание учеников на строку состояния ТР. Дать простейшие команды редактирования (DEL, BACKSPACE). В конце урока ученики выходят из ТР, не сохраняя набранный текст. Кодировка символьной информации при ее хранении в оперативной памяти.

Урок 2

Теоретическая часть. Экран в текстовом режиме (понятие строки, позиции в строке, размеры экрана — позиции×строки). Принципы формирования зрительного образа в позиции экрана (мозаика). Место экрана в работе ТР — окно (рабочая область) над основной памятью. Основная память ТР — оперативная память компьютера.

Практическая часть. Перед загрузкой ТР показать на учительском компьютере содержимое каталога на магнитном диске (каталог, файл, имя файла, расширение имени файла). Указать на файлы, содержащие сетевой загрузчик и ТР. В качестве упражнения при работе с ТР дать задание, используя принцип мозаики, написать крупными буквами (например, в шаблонах восемь позиций на десять строк) слово или целую фразу. Обратить внимание на место в строке состояний, указывающее на объем занимаемой (свободной) оперативной памяти. Контрольный опрос: по содержимому строки состояний описать состояние исполнителя (ТР в режиме редактирования).

Урок 3

Теоретическая часть. Необходимость сохранения текстов на МД. Устройство дисководов и магнитных дисков. Элементы файловой системы ОС. Путь к файлу (полное имя файла) — устройство, каталог (подкаталог), имя файла, расширение (назначение расширения). Передача файлов по локальной сети.

Практическая часть. Работа с заготовкой текста, который считывается с МД (вызов в меню режима загрузки файла, ввод имени файла). Порядок обработки введенного текста может быть дан в виде раздаточного материала, или содержаться в самом тексте. Ученики знакомятся с возможностью обращения за подсказкой. С помощью подсказки осваивают команды работы со строками (удаление/вставка, рассечение/склейка и т. д.), быстрое перемещение по тексту. Рассматриваются возможности восстановления удаленных строк (использование листа удаленных строк). В конце занятия контролируется «правильность» итогового текста (если возможно, то автоматическим контролем типа «Диктант» в ТР Микрон «Роботландии»).

Урок 4

Теоретическая часть. Принципы работы принтера. Понятие о форматировании текста (разбивка на страницы, организация в рамках одной страницы). Копирование фрагментов текста, понятие буфера для копирования. Алгоритм копирования фрагмента текста в указанном месте.

Практическая часть. Работа с заготовкой текста в виде шаблона документа (таблицы), который считывается с МД. Итоговый документ получается в результате редактирования и копирования фрагментов заготовки. Затем текст подготавливается для печати (форматируется), записывается на МД и выдается на принтер. За полученную распечатку текста ученикам выставляется оценка.

Урок 5

Теоретическая часть. Принципы организации поиска по образцу. Использование этого режима работы для поиска нужной информации и корректировки текста. Постановка задачи на создание досье (картотеки) на учеников класса. Идея — организация запроса на информацию о конкретном ученике с помощью поиска по образцу.

Практическая часть. Работа над созданием досье. В конце урока полученные тексты записываются на МД.

Урок 6

Теоретическая часть. Дополнительные возможности ТР. Макрокоманды. Многооконный режим. Работа со словарем. Алгоритмы орфографической проверки с помощью поиска по образцу. Алгоритмы разбивки текста на фрагменты, каждый из которых помещается в отдельный файл. Алгоритмы формирования текста слиянием нескольких файлов.

Практическая часть. Продолжение работы над досье. Поиск информации не только по фамилии, но и по другим признакам. Организация выборки части списка учеников, удовлетворяющих определенным требованиям. Выдача этого списка в виде отдельного файла и распечатка его на принтере. За выполненную работу выставляется оценка.

Урок 7

Зачетный урок. Совместно с учениками обсуждается детальная структура исполнителя (компьютера, как средства обработки текстов), в которой расписывается содержимое блоков: аппаратная часть, среда, режимы работы, система команд исполнителя, данные. Команды исполнителя классифицируются по режимам работы , в которых они задействованы, и по типам данных, которые они обрабатывают.

Далее ученики отвечают (письменно) на ряд контрольных вопросов (выборочно) и сдают практический зачет, в котором контролируется их умение работать с ТР (знание системы команд, режимов работы, алгоритмы манипуляций с текстом и т. д.).

Контрольные вопросы

1. Сферы применения текстовых редакторов.
2. На какие разделы (листы) делится память ТР?
3. На какие основные группы клавиш делится клавиатура?
4. Из каких основных частей состоит ВЗУ?
5. Принципы работы принтера.
6. Основные принципы организации и функции файловой системы.

Практические задания

1. Определить объем памяти в байтах, необходимый для хранения заданной фразы.
2. Определить, сколько страниц займет текст при печати на бумаге, если задан его объем в байтах и определено количество строк на странице и число символов в строке.
3. Определить, поместится ли на дискете указанный текст (например, книга), если его ввести в память компьютера.
4. Что такое курсор ? Его роль в работе текстового редактора.
5. Будут ли одновременно видны на экране символы, стоящие в k-ой позиции X-ой строки и в n-ой позиции Y-ой строки (например: k=10, X=5, n=5, Y=39)?
6. Сформулировать алгоритм перемещения первого абзаца заданного текста в конец текста с помощью системы команд ТР.
7. Сформулировать алгоритм сборки текста средствами ТР из фрагментов, хранящихся в нескольких файлах.
8. Сформулировать алгоритм, с помощью которого можно сосчитать, сколько раз входит некоторая буква (например, «о») в заданный текст.
9. Сформулировать алгоритм подсчета числа слов в заданном тексте.
10. Сформулировать алгоритм подсчета числа предложений в заданном тексте.
11. Как можно быстро узнать, сколько строк в тексте, который обрабатывается с помощью ТР?

Задания для самостоятельной работы

1. Подготовить сборник песен, которые поют ученики хором.
2. Подготовить выпуск стенгазеты, литературного или юмористического журнала.
3. Сделать заготовки различных документов: табелей, ведомостей, приглашений на торжество и т. д.

Таблица 2

Функциональные возможности ТР	1	2	3	4	5	6	7	8
1. Интерфейс:								
меню;	–	+	+	+	–	+	+	+
справка (Help);	–	+	+	+	–	+	+	+
строка состояний.	+	+	+	+	+	+	+	+
2. Шрифты:								
алфавиты (РУС/ЛАТ);	+	+	+	+	+	+	+	+
псевдографика;	–	–	–	–	–	–	+	+
макросредства создания «иероглифов»;	–	–	–	–	–	–	+	+
конструирование собственных шрифтов.	–	–	–	–	–	–	–	+
3. Экранное редактирование:								
перемещение курсора по экрану;	+	+	+	+	+	+	+	+
«быстрое» перемещение по тексту;	+	+	+	+	+	+	+	+
посимвольное редактирование;	+	+	+	+	+	+	+	+
работа со строками;	+	+	+	+	+	+	+	+
контекстный поиск и замена.	+	+	+	+	+	+	+	+
4. Работа с фрагментами текста:								
выделение фрагмента текста;	+	+	+	+	+	+	+	+
занесение фрагмента текста в буфер;	+	+	+	+	+	+	+	+
извлечение фрагмента текста из буфера;	+	+	+	+	+	+	+	+
макросредства программирования команд.	+	–	–	+	+	+	+	+
5. Работа с внешними устройствами:								
сохранение текстов (файлов);	+	+	+	+	+	+	+	+
считывание текстов (файлов);	+	+	+	+	+	+	+	+
сборка текста из нескольких файлов;	+	–	–	–	+	+	+	+
выделение фрагментов текста в отдельный файл.	–	+	–	–	+	+	+	+
6. Форматирование:								
число позиций в строке, красных строк;	+	+	+	+	+	–	+	+
число строк на странице;	–	–	+	–	–	–	–	+
выравнивание правого края, перенос.	–	+	+	–	+	–	+	+
7. Печать текстов:								
выбор шрифтов;	–	–	+	–	–	+	+	+
выбор режима печати;	–	–	+	–	–	+	+	+
организация страниц (число строк, поля);	–	–	+	–	–	+	+	+
нумерация страниц.	–	–	+	–	–	+	+	+
8. Многооконность:								
выделение «окон»;	–	–	–	–	–	–	+	–
установка размеров и цвета «окон»;	–	–	–	–	–	–	+	–
работа с фрагментами текста в разных окнах.	–	–	–	–	–	–	+	–
9. Словарь:								
наличие словаря;	–	–	–	–	–	–	+	+
возможность расширения словаря;	–	–	–	–	–	–	+	+
10. Работа в графическом режиме:								
переход на «подстроку» («надстроку»);	–	–	+	–	–	–	–	+
«расширение» строки.	–	–	+	–	–	–	–	+

Обозначения: «–» – нет, «+» – «есть».

КУВТ-86: 1 – EDASP (Альтернатива), 2 – WORDS (Альтернатива).

Корвет: 3 – CWorker, 4 – E (Альтернатива).

УКНЦ: 5 – EDIT, 6 – WRITER.

IBM: 7 – Лексикон, 8 – CHIWRITER.

Рекомендуемая литература

1. *Первин Ю. А., Дуванов А. А., Зайдельман Я. Н., Гольцман М. А.* Роботландия. М.: 1991.

2. *Ершов А. П., Семенов А. Л., Шень А. Х.* Основы информатики и вычислительной техники. М.: Просвещение, 1988.

3. *Каймин В. А., Щеголев А. Г., Ерохина Е. А., Федюшин Д. П.* Основы информатики и вычислительной техники. М.: Просвещение, 1989.

4. *Гейн А. Г., Житомирский В. Г., Линецкий Е. В. и др.* Основы информатики и вычислительной техники. 2-е изд., М.: Просвещение, 1992.

5. *Кушниренко А. Г., Лебедев Г. В., Сворень Р. А.* Основы информатики и вычислительной техники. М.: Просвещение, 1991.

6. *Каптелинин В., Дроздова Т.* Опыт обучения работе с текстовыми редакторами//ИНФО. 1990. № 4.

7. *Утлинский Е., Смекалин Д., Додонов А.* Знакомство с прикладным программным обеспечением//ИНФО. 1988. № 1.

Редакция журнала поздравляет всех своих читателей с Новым 1995 годом! Желаем счастья, здоровья и творческих успехов.

Уважаемые читатели!

Журнал «Информатика и образование» — единственное в России периодическое издание, в котором систематически и всесторонне рассматриваются вопросы внедрения новых информационных технологий в сферу образования. Наши читатели — преподаватели школ, техникумов, вузов, студенты, школьники, словом, все те, кто учит или учится работать с компьютером.

Журнал «Информатика и образование» издается с 1986 года. За это время он стал необходимым инструментом в повседневной работе учителя информатики. Редакция поддерживает постоянный контакт с подписчиками и старается отвечать на все интересующие читателей вопросы.

Для учителя информатики мы публикуем дидактические материалы и методические приложения к учебникам информатики, методики преподавания информатики и передовой опыт учителей, нормативные документы по охране и оплате их труда, конкретные материалы по различным типам техники (IBM, Macintosh, УКНЦ, БК, «Ямаха», «Корвет» и др.) и многое другое. Журнал также служит своеобразным лоцманом в море программного и аппаратного обеспечения.

Журнал «Информатика и образование» может быть интересен не только учителю информатики:

- *директор школы найдет в журнале законодательные акты и документы Министерства образования Российской Федерации по вопросам информатизации образования (базовые учебные планы, методические рекомендации, информационные письма, нормативные документы и др.);*
- *учитель-предметник узнает, как компьютер может помочь ему в проведении урока, познакомится с лучшими программами по различным предметам;*
- *учитель начальных классов, интересующийся преподаванием информатики младшим школьникам, прочтет об опыте школ, где такие методики успешно реализованы;*
- *школьник, как и любой пользователь, найдет в журнале интересный опыт, описания и листинги программ, познакомится с новинками в области вычислительной техники.*

Вот сведения, необходимые для подписки на наш журнал:

Название журнала и индекс издания	Подписная цена на I полугодие 1995 г.	Периодичность
«Информатика и образование»		
для индивидуальных подписчиков 70423	18 000 (за три номера)	1 раз в 2 месяца
для предприятий и организаций 73176	45 000 (за три номера)	

Подписка принимается в любом отделении связи **с любого месяца**.

ОБРАЩАЕМ ВАШЕ ВНИМАНИЕ НА ТО, ЧТО НА ЖУРНАЛ МОЖНО ПОДПИСАТЬСЯ ПО БЕЗНАЛИЧНОМУ РАСЧЕТУ

Телефон для справок: (095) 208-30-78
Факс: (095) 208-67-37
E-Mail: mail@infoobr.msk.su

А. Г. Юдина,
учитель информатики средней школы № 112, Москва

ИНФОРМАТИКА НА ЛОГО ДЛЯ СТАРШЕКЛАССНИКОВ

Рассуждения о том, что наиболее ценные результаты образования — это гибкость и широта мышления, способность и стремление учиться, познавать окружающий мир, стали общим местом в педагогической литературе. Однако на практике школьное обучение дает в основном лишь некоторую сумму знаний, умений и навыков, а мышление ребенка, его познавательные интересы, исследовательские, творческие качества не развиваются, и главная причина этого — в преобладании *объяснительно-иллюстративного метода обучения*: учитель объясняет, ученики выучивают и отвечают. Этот метод дешев, проверен временем, и его преобладание закономерно, хотя за последнее столетие педагогическая наука и практика выработали альтернативу — *проблемный метод обучения*, при котором ребенок не получает знания в готовом виде, а вовлекается в процесс их активного поиска, открытия. Но к сожалению, этот метод не распространен в массовой школе, и тому есть много причин: недостаток средств реализации, отсутствие систем обучающего материала, построенного по принципу проблемного обучения, а также реальные условия работы учителя, в частности — большое количество учеников в классе.

Однако в последнее десятилетие наша школа получила мощное средство воздействия на процессы обучения и воспитания детей — компьютер, использование которого идет по трем основным направлениям. Для *первого направления* характерен взгляд на компьютеры как на новое техническое средство обучения, применяемое в традиционных курсах для повышения наглядности или для контроля знаний. Воздействие на детей — эпизодическое и экзотическое; какая уж тут мультимедиа, если кодоскоп в классе увидишь не часто. *Второе направление* рассматривает компьютеры как самостоятельный предмет для изучения с целью овладения компьютерной грамотностью и информационной культурой. Целесообразно, конечно, готовить школьников к жизни в информационном обществе, но вызвать качественные изменения учебных процессов может только *третье направление*, рассматривающее компьютеры как средство влияния на способы мышления и учения. Для этого направления принципиально овладение программированием как искусством интеллектуального моделирования.

В работах ряда исследователей показано, как само научение программированию изменяет способ, которым дети изучают другие предметы, и продвигает их в интеллектуальном отношении. Интенсивно развивается формально-логическое мышление благодаря тому, что компьютер позволяет его конкретизировать.

Исследования показали, что при обучении программированию улучшаются общие способности к решению задач, так как интенсивно развивается умение осуществлять *анализ* — разбиение задачи на более мелкие подзадачи и поиск общих частей у различных задач и *синтез* — объединение решения задачи.

Попытки заставить интеллектуальную машину выполнить за человека тот или иной род работы стимулируют рефлексивное мышление о мышлении: «А как я сам это делаю?»

Изучение программирования положительно влияет даже на стиль языка, так как предполагает овладение семантикой и синтаксисом языка компьютера.

Программирование — область, в которой благодаря самой природе предмета достаточно легко организовать поисковое, проблемное обучение, предельно индивидуализированное, предоставить ребенку возможность осваивать знания через достижение осознаваемых личностных целей.

Тем самым в школе образуется как бы островок развивающего обучения, многие дети обретают уверенность, перед ними открываются новые возможности в учении и мышлении, в эмоциональном и когнитивном развитии.

А далее этот плацдарм развивающего обучения можно расширять в направлении различных школьных предметов, если со-

здать для детей возможность моделировать (используя накопленный опыт программирования компьютера) различные ситуации, процессы, среды.

Таким образом, обучение программированию может дать результаты не частные, а весьма ценные в общеобразовательном и воспитательном смысле. Проблемный, развивающий характер таких компьютерных занятий позволяет пробудить в детях желание экспериментировать, формулировать и проверять гипотезы, учиться на своих ошибках.

К сожалению, отечественная школьная информатика все больше поворачивается к утилитарно-прагматическому использованию компьютеров, к обучению работе с конторскими программными средствами. А блестящие развивающие возможности программирования практически не используются, так как перед началом содержательной работы в профессиональных языках программирования ученикам (и учителям) нужно преодолевать первоначальный барьер сложности (в основном, синтаксис), что привело к распространению мнения: программирование — удел избранных.

В то же время существует специально разработанный для развития мышления, творческих и исследовательских потенций язык ЛОГО, чей простой и естественный синтаксис плюс наглядность объектов и процессов позволяют работать с ним всем, даже слабым, ученикам. В то же время, используя его, можно целенаправленно формировать культуру мышления, способность к анализу и синтезу, абстрагированию и обобщению. ЛОГО предоставляет широкие возможности моделирования, создания условий для активного, поискового учения в таких областях, как математика (в частности, геометрия), физика, лингвистика и др. Это позволяет сделать работу действительно интересной, увлекательной, разнообразной (и не только для учеников, но и для учителя).

ЛОГО используется в наших школах, но почти исключительно в работе в младших классах. Потенциал ЛОГО далеко не исчерпан. В 4 и 5 номерах ИНФО за 1993 год напечатана краткая методика использования ЛОГО в VII—IX классах, причем не только как одного из языков программирования, но и как специально организованной среды обучения, микромира. Материал, предлагаемый в данной статье — учебное пособие, предназначенное для старшеклассников, уже знакомых со средой LogoWriter (в объеме базового курса). Принцип изложения — от частного к общему, от практических результатов к осмыслению и систематизации.

Комплекс курсов предназначен прежде всего для развития познавательной самостоятельности. Овладение программированием рассматривается не как цель, а как средство, позволяющее ученику, например, моделировать движение (а значит, глубже понять законы Ньютона), заинтересоваться закономерностями красивых геометрических построений, неожиданно явившихся результатом программы, и т. д. Оба курса — это фактически системы задач и заданий, предполагающих в основном самостоятельную практическую работу учащихся. Построение систем направлено на решение двух основных задач. Во-первых, формирование и развитие умения последовательно разбивать задачу на подзадачи, видеть общий план решения как структуру модулей. Каждый модуль-процедура получает наименование, отдельно описывается и отлаживается, а затем осуществляется сборка целого. Во-вторых, развитие умения находить аналогии, выделять общие части (модули) в разных задачах. Осуществляется это посредством описания процедур с переменными параметрами.

Эти направления — *обучение разработке общей (возможно, многоуровневой) структуры задачи еще до подробной разработки каждого модуля, а также выделение независимых факторов, влияющих на процесс, и управление процессом с использованием сразу несколько таких факторов* проходят через все темы.

Параллельно развиваются навыки алгоритмизации и продолжается совершенствование и углубление навыков анализа и синтеза.

От задания к заданию постепенно усложняются как структуры программ, так и структуры данных, причем последовательность задач составлена так, чтобы можно было использовать предыдущие задачи в качестве готовых модулей.

Базовый и профильный курсы информатики на ЛОГО могут в сочетании с накопленным опытом работы с младшими детьми составить единую линию развивающей информатики для общеобразовательной школы.

1. Типы данных. Операции над словами и списками

В языке ЛОГО (а LogoWriter — одна из версий этого языка) различаются два типа данных — слова и списки. Слово — последовательность символов без пробелов. Например:

```
175
8,036
предмет
;:,.*!
1_A
```

Список — это последовательность элементов, разделенных пробелами. Элементами могут быть слова (в частности, числа) и списки. Каждый список заключается в квадратные скобки. Например:

```
[[Петров А. И.]   [1965 г. р.]   123-00-00]
    список          список         слово

[математика   [4 3 4 5 4]   физика   [3 3]]
   слово        список       слово   список
```

Когда в качестве данного используется слово, не являющееся числом, перед ним ставят кавычки (без пробела). Сравните:

```
PR 25
PR "Имя?          (можно и PR [Имя?])
PR [Ваше имя?]
```

Команды повтора и условного ветвления используют в качестве данных списки команд или процедур, именно поэтому их необходимо заключать в квадратные скобки:

```
REPEAT 3 [FD 40 RT 90]
IF :N > 0 [STOP]
```

Что можно сделать со словом или списком? Во-первых, сосчитать количество элементов.

Пирог — это слово, в нем 5 элементов.

[Пирог] — это список, в нем 1 элемент.

[FD 40 RT 90] — это список с 4 элементами.

[1 2 [3 4]] — в этом списке 3 элемента, последний элемент сам является списком с двумя элементами.

Список может быть пустым — [].

Количество элементов слова или списка определяет операция COUNT. Например:

```
PR COUNT [1 2 3]
_3
PR COUNT [123]
_1
PR COUNT 123
_3
PR COUNT [пирог]
_1
PR COUNT "пирог
_5
```

Обратите внимание на то, что операция отличается от команды: результатом выполнения операции не являются какие-либо непосредственные действия. Если попробовать набрать в поле команд

```
COUNT "пирог
```

и нажать <ввод>, появится красная предупреждающая надпись:

```
«Не знаю, что делать с 5».
```

Вычисленное, определенное операцией значение должно быть передано другой операции или команде.

При необходимости выделить первый или последний элемент слова или списка используются операции FIRST и LAST. Например:

```
PR FIRST "шест
_ш

PR LAST [1 2 3 4]
_4

PR LAST [1234]
_1234
```

Разберем хорошо знакомую конструкцию, используемую при вводе числа:

```
MAKE "число FIRST READLIST
```

READLIST — операция, выдающая список, полученный с клавиатуры;

READCHAR — операция, выдающая символ, полученный с клавиатуры; специальной операции для ввода слова (числа) нет, поэтому приходится использовать комбинацию двух операций: FIRST READLIST (т. е. первый элемент полученного с клавиатуры списка).

Если необходимо отбросить первый или последний элемент слова или списка, используются операции BF и BL (сокр. от but first и but last).

```
PR BF "шест
_ест

PR BF BF "шест
_ст
```

Можно выделить элемент слова/списка по его номеру. Для этого служит операция ITEM, у которой два входа — номер элемента и список, в котором этот элемент нужно найти:

```
PR ITEM 3 "число
_с
```

Разберем пример использования новых операций.

Напишем процедуру СТОЛБИК, распечатывающую заданное слово по вертикали. Она должна работать так:

```
СТОЛБИК "карта
_к
 а
 р
 т
 а
```

Программа циклическая, цикл повторяется столько раз, сколько букв в слове (операция COUNT). В каждом цикле выводится один элемент слова, для этого можно использовать операцию ITEM.

```
ТО СТОЛБИК1 :слово
MAKE "номер 1
REPEAT COUNT :слово
  [PR ITEM :номер :слово
   MAKE "номер :номер + 1]
END
```

Переменным параметром цикла является номер. Перед циклом производится начальная установка параметра, в цикле параметр каждый раз меняет значение.

Возможен другой вариант, когда каждый раз выводится первый элемент слова, но само слово укорачивается на этот элемент:

```
ТО СТОЛБИК2 :слово
REPEAT COUNT :слово
  [PR FIRST :слово
   MAKE "слово BF :слово]
END
```

В этом варианте переменным параметром цикла является слово.

ЗАДАНИЯ.

Написать процедуру, которая

а) постепенно укорачивает заданное слово:

```
карта
карт
кар
ка
к
```

или

```
карта
арта
рта
та
а
```

Программа циклическая, цикл повторяется столько раз, сколько букв в слове (операция COUNT).

В каждом цикле выводится на экран значение переменного параметра — слова. А затем значение переменного параметра изменяется (команда присваивания).

Изменение (укорочение) производится операцией BF или BL;

б) печатает заданное слово «наоборот» (дом — мод).

В каждом цикле выводится на экран последний элемент (операция LAST) переменного параметра, затем переменный параметр укорачивается на последнюю букву (операция BL).

Чтобы перевернутое слово оказалось напечатанным в строчку, для вывода нужно использовать команду INSERT;

в) подсчитывает сумму заданного списка чисел. Программа циклическая, перед циклом — начальная установка. Необходимо завести переменную, в которой будет постепенно накапливаться сумма, и присвоить ей начальное нулевое значение.

Цикл повторяется столько раз, сколько чисел в списке (операция COUNT). Переменным параметром будет сам список. В цикле каждый раз значение суммы изменяется на величину первого элемента переменного параметра — списка (операция FIRST), затем переменный параметр укорачивается на первый элемент (BF).

После завершения повторений — вывод значения переменной, в которой накапливалась сумма;

г) подсчитывает количество нулей в заданном списке чисел (т. е. количество чисел, равных нулю).

Перед циклом нужно завести переменную (счетчик), в которой будет накапливаться число нулей; ее начальное значение — ноль.

В цикле каждый раз проверяется очередной элемент списка (ITEM или FIRST). Если анализируемый элемент равен нулю (команда условного ветвления), значение счетчика возрастает на единицу (MAKE "счётчик :счётчик + 1);

д) подсчитывает, сколько раз заданное число встречается в заданном списке чисел.

Если у предыдущей процедуры на входе был один параметр — список чисел, то в этом случае нужно добавить второй параметр — число.

```
to уменьшение :слово
repeat count :слово
  [pr :слово
   make "слово bf :слово]
end

to перевёртыш :слово
repeat count :слово
  [insert last :слово
   make "слово bl :слово]
end
```

```
to сумма :список
make "s 0
repeat count :список
   [make "s :s + first :список
    make "список bf :список]
pr :s
end

to подсчёт :список
make "счётчик 0
repeat count :список
   [if (first :список) = 0
      [make "счётчик :счётчик + 1]
    make "список bf :список]
pr :счётчик
end

to кол-во :эл-т :список
make "N 0
repeat count :список
   [if (first :список) = :эл-т [make "N :N + 1]
    make "список bf :список]
end
```

2. Построение гистограммы по заданному списку чисел

Часто встречается ситуация, когда полезно представить числовые данные, список чисел в графическом виде. Один из способов такого представления — гистограмма. Например, класс из 38 человек написал контрольную. Результат — список из 38 отметок. Если подсчитать, сколько раз в списке встречается каждая из оценок и затем на числовой оси построить столбики соответствующей высоты, получится гистограмма (рис. 1).

Рис. 1

Другой вид графического представления такой информации — круговая диаграмма.

Определим, какие данные нужны для построения гистограммы по списку, состоящему из натуральных чисел и нулей.

Во-первых, сам список. Во-вторых, необходимо заранее определить размеры картинки по горизонтали и вертикали, или, другими словами, каково самое большое число в анализируемом списке (количество делений по горизонтали) и какой высоты (примерно) может быть самый высокий столбик.

Алгоритм построения гистограммы:

Дано:

- список неотрицательных целых чисел;
- количество делений по горизонтали;
- количество делений по вертикали.

1. Подсчитать шаг по горизонтали и по вертикали (поделить размер экрана на число делений).

2. Нарисовать и разметить оси. Установить черепашку в начало координат.

3. Считать, что искомое число — ноль.

(П. 1—3 — начальные установки.)

4. Подсчитать, сколько раз искомое число встречается в списке.

5. Нарисовать столбик полученной высоты.

6. Передвинуть черепашку вправо на одно деление.

7. Увеличить значение, которое нужно искать в списке, на единицу.

(П. 4—7 — тело цикла, повторяются для каждого числа от нуля до наибольшего.)

Чтобы получить число повторений, надо второй параметр (наибольшее число в списке) увеличить на единицу.

Переменным параметром цикла является число, для которого ведется подсчет.

Основная часть цикла — процедура подсчета — уже готова (задание «д» из главы 1).

ЗАДАНИЕ.

Написать для пунктов 2 и 5 процедуры и отладить их по отдельности (учесть, что ширина столбика не должна превышать размер деления по горизонтали).

После этого собрать общую процедуру построения гистограммы, использующую процедуры построения осей, подсчета и построения столбика. Ее структуру можно проиллюстрировать следующей схемой:

Дополнительное задание: изменить процедуру так, чтобы гистограмма отражала, как

часто встречается каждое число в списке, в процентах от общей длины списка.

```
to оси :гор :верт
cg
pu setpos [-300 -160] pd
repeat :гор     [setx x + 500 / :гор штрих1]
setx -300
repeat :верт    [sety y + 300 / :верт штрих2]
sety -160
end

to штрих1
sety y - 3 sety y + 3
end

to штрих2
setx x + 2 setx x - 2
end


to столбик :выс :шир
repeat 2 [fd :выс rt 90 fd :шир rt 90]
end


to гистограмма :список :гор :верт
оси :гор :верт
make "эл-т 0
repeat :гор + 1
  [кол-во :эл-т :список
   столбик (:N * 300 / :верт) (500 / :гор / 2)
   setx x + 500 / :гор
   make "эл-т :эл-т + 1]
end
```

3. Итерация и рекурсия

Разберем следующее задание: подсчитать количество букв «а» в слове. Задача очень похожа на подсчет в числовом списке. Повторения можно реализовать, используя команду REPEAT:

```
to буква_1 :слово
make "N 0
make "номер 1
repeat count :слово
  [if (item :номер :слово) = "a
    [make "N :N + 1]
   make "номер :номер + 1]
pr :N
end

to буква_2 :слово
make "N 0
repeat count :слово
  [if (first :слово) = "a
    [make "N :N + 1]
   make "слово bf :слово]
pr :N
end
```

В обоих вариантах цикл повторяется заранее известное число раз; в каждом цикле производится изменение параметра цикла; перед циклом — начальные установки. Такая организация циклической программы называется итерацией.

Варианты отличаются выбором переменного параметра цикла. В первом случае это номер анализируемой буквы слова (номер-указатель как бы двигается по слову от начала к концу). Во втором варианте параметр цикла — само слово, каждый раз укорачивающееся на первую букву.

Повторение действий можно организовать и совершенно другим образом — используя рекурсию. Процедура, проверяющая первую букву слова, продолжается процедурой, проверяющей первую букву остатка слова, которая продолжается процедурой ..., и т. д. Каждый раз проверяемое слово все короче, и процесс должен остановиться, когда длина его уменьшится до нуля.

Операция, которая проверяет, не пустой ли список или слово — EMPTY?, она может быть использована командой ветвления.

```
to буква_3 :слово :N
if empty? :слово [pr :N stop]
if (first :слово) = "a [make "N :N + 1]
буква_3 bf :слово :N
end
```

Рекурсивная процедура вызывает саму себя с измененными в очередной раз параметрами до тех пор, пока параметр — слово — содержит хотя бы одну букву. (Счетчик букв «а» устанавливается в ноль при запуске процедуры как параметр.)

Можно написать рекурсивную процедуру, использующую в качестве параметра номер буквы в слове:

```
to буква_4:слово :N :номер
if :номер = count :слово [pr :N stop]
if (item :номер :слово) = "a
     [make "N :N + 1]
буква_4 :слово :N :номер + 1
end
```

Рекурсивные вызовы здесь заканчиваются, когда номер-указатель дойдет до последней буквы. При запуске счетчик устанавливается в ноль, а номер — в единицу (например, буква_4 "программа 0 1).

ЗАДАНИЕ.

Написать процедуру, подсчитывающую количество букв «а» в предложении, используя итеративную процедуру подсчета в слове. (Предложение — список, состоящий из слов.)

4. Формирование списков

Операции FIRST, LAST, ITEM выделяют один элемент слова или списка. Операции BF, BL уменьшают (укорачивают) слово или список на один элемент.

Можно сказать, что перечисленные операции «разбирают» объекты на части. В этой главе вы познакомитесь с операциями, необходимыми для сборки объектов.

Операции FPUT (first put) и LPUT (last put) удлиняют список на один элемент, приписываемый в начало или в конец.

Для этих операций нужны два параметра: первый — элемент, второй — список, к которому надо присоединить этот элемент. Результат — список.

Например:

```
PR FPUT "a [a b c]
_a a b c

PR LPUT "a [a b c]
_a b c a
```

Таким же образом можно к списку присоединить элемент, который тоже является списком:

```
          эл-т                              список
PR LPUT [ул. Лесная, 3][[А. Котов] 700–23–00]
_[А. Котов] 700–23–00 [ул. Лесная, 3]
```

В списке стало три элемента вместо двух.

Для объединения двух или нескольких списков в один общий список служит операция SE (от sentence — предложение).

```
PR SE [алгебра [4 5 3 4]] [химия [4 4]]
_алгебра [4 5 3 4] химия [4 4]
```

Было два списка по два элемента в каждом, получился один, из четырех элементов. (Если бы к тем же данным была применена операция FPUT, результат получился бы совсем другим:

```
PR FPUT [алгебра [4 5 3 4]] [химия [4 4]]
_[алгебра [4 5 3 4] ] химия [4 4],
```

т. е. в результате получен список из трех элементов).

ЗАДАНИЯ:

а) разбить список на две части по i-тому элементу (для определенности считать, что он останется в первой части).

Возможный вариант алгоритма:

1. Завести пустой список (будущая первая часть).

2. Повторить i раз следующие действия:

присоединить первый элемент исходного списка в конец нового (LPUT);

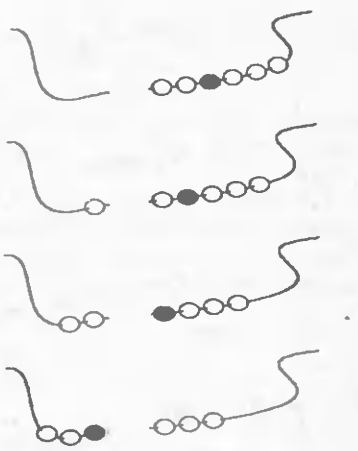


Рис. 2

укоротить исходный список на первый элемент (BF).

3. Вывести на экран полученные два списка — новый и то, что осталось от исходного.

Эту задачу можно проиллюстрировать следующим образом: предположим, из одной нитки бус нужно сделать две (разделив по черной бусине) (рис. 2).

Каждый раз первая бусина исходной нитки оказывается в конце новой нитки;

б) исключить i-тый элемент из списка.

Возможный вариант алгоритма:

1. Разделить список по i-тому элементу (использовать предыдущую процедуру).

2. Отбросить последний элемент нового списка (BL).

3. Объединить получившиеся два списка (SE) и вывести результат на экран;

в) исключить из числового списка все нули (т. е. все числа, равные нулю).

Возможный вариант алгоритма:

1. Завести новый пустой список для результата.

2. Повторить столько раз, сколько элементов в списке, анализ: если первый элемент списка не равен нулю (IF NOT (FIRST :список) = 0), приписать его в конец нового списка, а исходный список укоротить на первый элемент.

3. Вывести на экран получившийся в результате новый список.

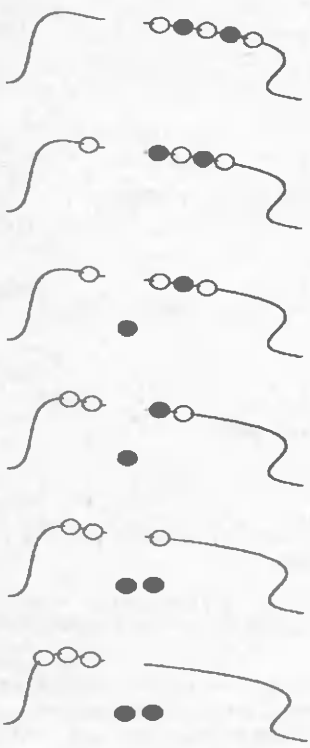


Рис. 3

Можно продолжить сравнение с бусами: как убрать с нитки все черные бусины? Нанизывать бусины на новую нитку, оставляя в стороне черные (нули) (рис. 3).

```
to деление :i :список
make "нов.список []
repeat :i
  [make "нов.список
     lput (first :список) :нов.список
   make "список bf :список]
end

to исключение :i :список
деление :i :список
pr se bl :нов.список :список
end

to нули :список
make "результат []
repeat count :список
  [if not (first :список) = 0
    [make "результат
       lput (first :список) :результат]
     make "список bf :список]
pr :результат
end
```

5. Команды и операции. Процедуры и функции

Знакомясь с разнообразными возможностями LogoWriter, вы научились использовать достаточно большое количество предписаний, или примитивов, этого языка (изначально имеющихся в нашем распоряжении).

Их можно разделить на две группы: команды и операции. Например:

команды

```
FD 70
TONE 300 1
RG
PR "кот
```

операции

```
FIRST "кот
RANDOM 100
COUNT [1 2 [3 4]]
COLORUNDER
```

И команды, и операции могут быть без параметров или использовать один или несколько параметров.

Разница между ними в следующем. Если набрать пример из первого столбика и нажать «ввод», команда немедленно выполнится — будет произведено некоторое действие: или черепашка передвинется на 70 шагов, или прозвучит нота заданной высоты и длительности, или произойдет очистка экрана, или появится заданная надпись.

Если попробовать сделать то же самое с примерами из второго столбика, в поле команд появится красная надпись: «Не знаю, что делать с ...», а вместо многоточия — результат соответствующей операции: или буква «к», или наугад выбранное число до 100, или число 3, или цвет поля под черепашкой.

Произойдет это потому, что вычисленный, определенный операцией результат должен быть передан другой операции или команде (в качестве аргумента). Например, PR COLORUNDER, или TONE RANDOM 800 1, или много раз использованное:

```
MAKE "число FIRST READLIST
```

Результат ввода с клавиатуры (операции READLIST) передается операции FIRST; результат ее выполнения — первое слово (число) — передается команде MAKE " для выполнения действия присваивания.

Таким образом, правильно записанная программа может состоять из цепочек следующего вида:

```
команда операция операция ... операция
```

Сначала в такой цепочке выполняется самая правая операция и ее результат передается (в качестве аргумента) предыдущей, ре-

зультат выполнения которой передается предыдущей, и т. д., пока не будет выполнена самая левая операция и ее результат не будет передан команде для выполнения действия. Например, чтобы выделить предпоследний элемент слова или списка, можно использовать одну за другой операции LAST и BL:

```
PR LAST BL "слово
_в
```

Операция BL передает предыдущей операции LAST слово «слов», а операция LAST передает команде вывода букву «в».

Такой достаточно простой порядок выполнения (линейный, справа налево) осуществляется только в цепочках, содержащих команды и операции с одним аргументом. В других случаях последовательность может быть более сложной.

Можно провести аналогию с обычной записью математических выражений и порядком их вычисления. Например:

$$Y = \sqrt{\sin 7.3^2}$$

первая операция — возвести 7.3 в квадрат;

вторая операция — найти sin от полученного;

третья операция — извлечь квадратный корень из полученного;

и, наконец, действие — присвоить переменной Y найденное значение.

Записать вычисление такого выражения можно следующим образом:

```
make "Y sqrt sin 7.3 * 7.3
```

В число примитивов LogoWriter входят операции (стандартные функции) SQRT, SIN, COS (извлечение квадратного корня, синус и косинус от градусного аргумента).

А вот стандартная функция тангенс отсутствует. Предположим, что в некоторой программе часто встречается вычисление тангенса, что приводит к решению описать соответствующую процедуру.

```
to tg :x
make "z (sin :x )/ cos :x
end
```

Пусть теперь нам нужно произвести вычисление, аналогичное разобранному:

$$Y = \sqrt{\text{tg } 7.3^2}$$

Если попробовать записать по аналогии

```
make "Y sqrt tg 7.3 * 7.3 ,
```

то ничего не получится.

Записать можно только в две ступени, два действия:

```
tg 7.3 * 7.3
make "Y sqrt :z
```

Описывая процедуру, мы создали новую команду, а удобнее было бы получить новую операцию — функцию.

Для описания функций служит специальная команда — OP (out put — вывод), ее использовать можно только в описаниях (в рабочем режиме нельзя). Правила описания функций такие же, как для процедур.

Используя команду OP, опишем новый вариант — функцию тангенса от градусного аргумента:

```
to tg :x
op (sin :x) / cos :x
end
```

Теперь запись

```
make "Y sqrt tg 7.3 * 7.3
```

будет правильной и приведет к желаемому результату.

Разберем еще один пример — нахождение модуля числа. Запишем и сравним два варианта: функцию (возвращающую результат при помощи команды OP) и процедуру, заканчивающуюся присваиванием результата некоторой переменной.

функция:

```
to abs :x
ifelse :x < 0   [op 0 - :x]
                [op :x]
end
```

процедура:

```
to abs1 :x
ifelse :x < 0   [make "z 0 - :x]
                [make "z :x]
end
```

Попробуем теперь выявить преимущества использования функции.

Пусть нужно вычислить и вывести на экран sin |t|:

функция:

```
pr sin abs :t
```

процедура:

```
abs1 :t
pr sin :z
```

Если то же нужно сделать для выражения |sin |t||, преимущество первого варианта становится еще более очевидным:

функция:

```
pr abs sin abs :t
```

процедура:

```
abs1 :t
make "q sin :z
abs1 :q
pr :z
```

Использование функций может сделать программу короче и яснее. Кроме того, отпадает необходимость использования вспомогательных переменных для хранения промежуточных результатов.

ЗАДАНИЯ:

а) описать функцию возведения заданного числа в квадрат. Используя эту функцию, вычислить

2^4, 2^8, 2^{10}, $\sqrt{\text{tg } 7.3^2}$;

б) написать функцию, подсчитывающую количество заданных букв в слове. Используя ее, написать функцию, подсчитывающую количество заданных букв в тексте (списке слов), и сравнить эту функцию с процедурой из задания к главе 3 (подсчет букв «а» в предложении);

в) написать функцию сравнения двух списков одинаковой длины, результат которой — количество одинаковых (равных) элементов, находящихся на одинаковых позициях.

Перед циклом необходимо завести счетчик, его начальное значение — ноль. В цикле, который повторяется столько раз, сколько элементов в списке (любом из двух), проводится сравнение первых элементов списков и в случае их равенства продвижение счетчика; затем оба списка укорачиваются на первые элементы;

г) написать функцию процентного отношения одной величины к другой и модифицировать процентную гистограмму из задания к главе 2.

```
to счёт1 :буква :слово
  make "N 0
  repeat count :слово
    [if (first :слово) = :буква
      [make "N :N + 1]
     make "слово bf :слово]
  op :N
end

to счёт2 :буква :текст
  make "K 0
  repeat count :текст
    [make "K :K + счёт1 :буква first :текст
     make "текст bf :текст]
  op :K
end

to сравнение :сп1 :сп2
make "N 0
repeat count :сп1
  [if (first :сп1) = first :сп2
    [make "N :N + 1]
   make "сп1 bf :сп1
   make "сп2 bf :сп2]
op :N
end
```

6. Поиск минимального или максимального элемента в списке чисел

Предположим, что перед вами — горка камушков и нужно выбрать самый легкий из них, используя простейшие весы.

Скорее всего, вы взвесите два первых попавшихся камушка и отложите в сторону более тяжелый. Затем на освободившуюся чашку весов положите следующий камушек и опять отложите более тяжелый из двух. Это действие вы будете повторять, пока не кончатся камушки. Последнее взвешивание определит результат.

ЗАДАНИЕ.

Написать согласно изложенному выше алгоритму перебора функцию, определяющую минимальное число в списке.

Вход один — список чисел.

Выход — наименьшее число.

Описание алгоритма.

В цикле каждый раз производится сравнение очередного элемента списка с числом, которое оказалось меньше всех предыдущих — текущим минимумом. Если очередной элемент меньше этого числа, то значение текущего минимума изменяется.

Перед началом цикла за текущий минимум принимается самый первый элемент списка, и сравнение начинается со второго элемента.

В качестве параметра цикла можно использовать номер сравниваемого элемента или сам список чисел.

Поиск максимального числа производится аналогично.

```
to min :список
make "M first :список
make "номер_тек 2
repeat (count :список) - 1
  [if (item :номер_тек :список) < :M
    [make "M item :номер_тек :список]
   make "номер_тек :номер_тек + 1 ]
op :M
end
```

7. Сортировка списка чисел

Пусть нужно упорядочить список чисел — переставить элементы в порядке возрастания.

Как поступают в таком случае? Находят минимум, выписывают, вычеркивают из исходных данных. Опять находят минимум, выписывают и т. д., пока список не кончится. То есть алгоритм можно представить так: найти минимальный элемент в списке, записать его первым в новом списке и исключить из исходного; затем найти минимум в оставшемся списке, приписать его в конец нового и опять исключить из исходного и т. д. Таким образом, постепенно, элемент за элементом, вырастет новый, упорядоченный список.

Функция нахождения минимума уже готова, можно также использовать готовую процедуру исключения элемента из списка из задания б) к главе 4. Неизвестно только пока, как определить место найденного минимума, его номер, чтобы исключить из дальнейшего рассмотрения.

Если при нахождении минимума в качестве переменного параметра цикла используется номер анализируемого элемента, его можно запоминать в те моменты, когда меняется значение текущего минимума. То есть при начальной установке необходимо завести два счетчика — один будет изменяться каждый раз в цикле (это номер анализируемого элемента, параметр цикла), а второй будет отслеживать номер текущего минимума. Начальное значение второго счетчика — единица, так как сначала за минимум принимается самый первый элемент; начальное значение первого счетчика — два, так как сравнения начинаются со второго элемента.

ЗАДАНИЯ:

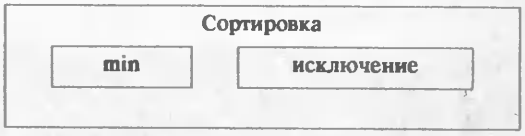
а) изменить функцию определения минимума так, чтобы в результате выдавалось не одно число (минимум), а список из двух элементов — минимум и его номер в списке («собрать» два числа в список можно командой SE);

б) превратить в функцию процедуру исключения элемента из списка;

в) написать по описанному алгоритму функцию сортировки по возрастанию.

(Повторить столько раз, сколько элементов в списке, следующие три действия: найти минимальный элемент, приписать его в конец выходного списка, исключить из исходного списка.)

Эта функция использует функцию нахождения минимума и функцию исключения элемента из списка, поэтому ее структуру можно проиллюстрировать следующей схемой:




```
to min :список
make "M first :список
make "номер_мин 1
make "номер_тек 2
repeat (count :список) - 1
  [if (item :номер_тек :список) < :M
    [make "M item :номер_тек :список
     make "номер_мин :номер_тек]
   make "номер_тек :номер_тек + 1 ]
op se :M :номер_мин
end

to искл :i :список
make "нов.список []
repeat :i
  [make "нов.список
    lput (first :список) :нов.список
   make "список bf :список]
op se bl :нов.список :список
end

to сортировка :список
make "вых.список []
repeat count :список
  [make "эл-т first min :список
   make "вых.список lput :эл-т :вых.список
   make "список искл last min :список :список]
op :вых.список
end
```

(Продолжение следует)

КуМир будет всегда !

ИнфоМир

КуМир-Гипертекст

ЗАЯВКА НА ПРИОБРЕТЕНИЕ ПРОГРАММНЫХ СРЕДСТВ

КуМир для Macintosh ____экз.

КуМир для Корвета (КорветМир) ____экз.

КуМир для УКНЦ ____экз. ФортранМир для УКНЦ ____экз.

КуМир-Гипертекст (базовый к-т) ____экз. **ДЛЯ IBM PC!**

КуМир-Гипертекст (расширенный к-т) ____экз.

Информатика-, [есять ____экз. ____экз. ПаскальМир ____экз.

Планиметрия ____экз. Функции и Графики ____экз.

Адрес:_____

_____

_____

✂

Заявки направляйте по адресу: 103051, Москва,
Садовая-Сухаревская, 16, комн.9 "Информатика и образование"

Е. А. Ерохина,
научный сотрудник Института информатизации образования, Москва

ОТ ЛОГИКИ К ПРОГРАММИРОВАНИЮ (ПРОЛОГ в курсе информатики)*

20. Системы счисления — это так просто...

Напишем программу на ПРОЛОГЕ, переводящую заданное число из десятичной системы счисления в двоичную. Сначала «научим» машину правилу деления с остатком: при делении a на b получается частное c и остаток ost, если b×c+ost=a, причем ost<b. На ПРОЛОГЕ это правило записывается так:

```
деление(a,b,c,ost')<-УМНОЖЕНИЕ(b,c,ost',a),МЕНЬШЕ(ost',b);
```

Так как диапазон чисел в ПРОЛОГЕ довольно мал, для записи результата мы будем использовать строку. В двоичной системе счисления используется всего две цифры, и поэтому для перевода их в строковое представление нам достаточно двух правил:

```
перевод(0,"0");
перевод(1,"1");
```

Правило перевода числа из десятичной системы в двоичную можно сформулировать так:

« **1) Если число больше 1, делим его на два с остатком.**
2) Затем к частному применяем то же правило перевода.
3) Повторяем эти действия до тех пор, пока не получим частное, меньшее двух.
4) Затем записываем сначала последнее частное, а затем приписываем к нему все остатки начиная с последнего».

На ПРОЛОГЕ это правило записывается так:

```
перевод(x,y)<-БОЛЬШЕ(x,1),деление(x,2,c,ost'),перевод(c,z),
              СТРЦЕЛ(ost2',ost'),СЦЕП(z,ost2',y);
```

А при помощи правила «работа» организуем ввод числа и вывод результата:

```
работа<-ВЫВОД("Введите число X:"),ВВОДЦЕЛ(x),ПС,перевод(x,y),
        ВЫВОД("Ваше число в двоичной системе равно "),ВЫВОД(y);
```

ЗАДАЧА

Написать программу на ПРОЛОГЕ для перевода заданного числа из десятичной системы счисления в систему счисления с заданным основанием (меньшим десяти).

21. Списки на ПРОЛОГЕ

Кроме хорошо знакомой нам записи данных в форме фактов, на ПРОЛОГЕ существует еще один удобный способ хранения — в виде списка. **Списком** в ПРОЛОГЕ называется упорядоченное множество констант. Список задается при помощи перечисления всех его элементов. Для обозначения списка используются квадратные скобки: []. Например, список оценок за четверть: **[5,4,4,3,2,4,5,5,3],** или список остановок поезда: **[Москва, Ленинград, Выборг, Хельсинки].**

---

* Окончание. Начало см.: Информатика и образование. 1993. № 5, 1994. № 3.

Список может быть только числовым, только символьным или содержать числа и слова одновременно. Мы можем, например, создать список станций с указанием времени стоянки поезда: **[Тула,10,Орел,15,Курск,11].**

Список может быть пустым (т. е. без элементов). Пустой список обозначается [].

Как же применять списки? Например, попробуем усовершенствовать хранение оценок в базе знаний «школа»: вместо отдельных фактов о каждой оценке будем хранить список оценок за четверть. Факты примут вид:

оценки(<фамилия>,<предмет>,<список оценок>);

Например, факты могут быть такими:

оценки(Иванова,литература,[5,5,5,5]);
оценки(Петров,химия,[3,4,5,4]);
оценки(Петров,литература,[5,4,4,5]);
оценки(Васечкин,музыка,[5]);
оценки(Васечкин,математика,[]);

Теперь попробуем задавать вопросы (для обращения к списку используется обычная переменная):

По-русски	На ПРОЛОГЕ	Ответ ЭВМ
Какие оценки у Петрова по химии?	?оценки(Петров,химия,k);	k=[3,4,5,4]?
Какие оценки у Васечкина?	?оценки(Васечкин,_,x);	x=[5] x=[]

Как мы видим, ответы на вопросы вроде бы получены. Но для более тонкого анализа (например, для выявления отличников или неуспевающих) необходимо выделять оценки из списка по отдельности.

Для работы со списками в ПРОЛОГЕ существует единственное действие: то самое, при помощи которого в книге Л. Кэролла «Алиса в стране чудес» Королева решает все свои проблемы: «Отрубить ей (или ему) голову».

На ПРОЛОГЕ эта операция называется «отсечение головы» и обозначается значком |: [x|y]*. В результате выполнения этой операции переменная x (она называется г о л о в о й списка) получает значение первого элемента в списке, а остаток списка (он называется х в о с т о м) без первого элемента попадет в переменную y. Подчеркнем, что переменная y (хвост) получает значение типа списка (даже если в нем один элемент, или вовсе нет элементов).

Например, рассмотрим результаты многократного применения операции отсечения головы к списку [1,2,3,4,5].

[1,2,3,4,5] < --- [x|y]
x=1, y=[2,3,4,5] <--- [x|y]
x=2, y=[3,4,5] <--- [x|y]
x=3, y=[4,5] <--- [x|y]
x=4, y=[5] <--- [x|y]
x=5, y=[]

Теперь научимся применять эту операцию для работы со списками: определим понятие принадлежности элемента списку. Правило запишем в виде:

принадл(<элемент>,<список>);

---

* В программах на ПРОЛОГЕ отсечение обозначается символом «¦». В тексте статьи вместо этого символа используется «|».

1. «Базис»:

Элемент принадлежит списку, если он *совпадает* с головой списка (все остальные элементы нам в данном случае безразличны):

принадл(а,[а|_]);

2. «Индуктивный переход»:

Элемент принадлежит списку, если он *принадлежит хвосту* списка (тому, что останется после отсечения головы):

принадл(а,[х|у])<−принадл(а,у);

Если ввести это правило в машину, то можно задавать вопросы:

По-русски	На ПРОЛОГЕ	Ответ
Принадлежит ли Выборг списку [Одесса,Таллинн,Выборг]?	?принадл(Выборг,[Одесса,Таллинн, Выборг]);	ДА

Интересно, а как машина ищет ответ на этот вопрос? Процесс поиска происходит так же, как и при обработке всех рекурсивных правил. Поясним поиск схемой:

?принадл(Выборг,[Одесса,Таллинн,Выборг]);

1-я копия | **а=Выборг, х=Одесса, у=[Таллинн,Выборг]** **"ДА"**

2-я копия | **а=Выборг, х=Таллинн, у=[Выборг]**

↑ ↓

Применим факт: **а=Выборг, а=Выборг** ——————┘

И наконец, применим правило принадлежности для выделения отдельных оценок из списка. Для этого введем правило «оценка ученика по предмету» (назовем его «балл»):

балл(фамилия',предмет',оценка')<−оценки(фамилия',предмет',список'),
принадл(оценка',список');

Теперь можно задать вопрос:

По-русски	На ПРОЛОГЕ	Ответ ЭВМ
Какие баллы у Петрова по химии?	?балл(Петров,химия,k);	k=3 k=4 k=5 k=4

Имея в базе знаний правило «балл», можно использовать наши старые соотношения для определения отличника и т. п.

22. Что можно сделать при помощи списков

Решим несколько задач на обработку списков.

Задача 1. Подсчет количества элементов в списке (или длины списка).

Рисуем «черный яшик»:

Правило будет иметь вид:

длина (<список>,<длина>);

Теперь напишем само правило. В качестве базиса можно взять пустой список:

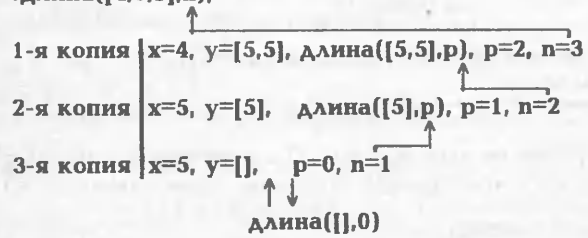
длина([],0);

Длина списка есть длина его хвоста плюс единица (длина головы):

длина([x|y],n)<−длина(y,p),СЛОЖЕНИЕ(p,1,n);

Рассмотрим процесс поиска ответа на заданный вопрос:

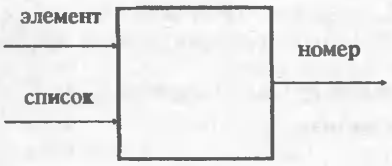
?длина([4,5,5],x);



Из рассмотренного видно, что собственно суммирование происходит при «обратном ходе».

Задача 2. Определить номер данного элемента в списке.

Рисуем «черный ящик»:



Правило будет иметь вид:

номер(<элемент>,<список>,<номер>);

Пишем правило. В качестве базиса берем голову:

номер(a,[a|b],1);

Если элемент — не голова, и известен его номер в хвосте (т. е. в списке без головы), то номер его во всем списке на единицу больше:

номер(a,[b|c],n)<−НЕ(РАВНО(a,b)), номер(a,c,p),СЛОЖЕНИЕ(p,1,n);

Примечание. Если убрать из этого правила **НЕ(РАВНО(a,b)),** то будут вычислены номера всех элементов списка, равных заданному.

ЗАДАЧИ И УПРАЖНЕНИЯ

1. Написать программу на ПРОЛОГЕ для вычисления:

а) суммы элементов числового списка;

б) произведения элементов числового списка.

2. Схематически изобразить ход поиска ответа на вопросы:

а) ? длина([Самара,Тверь,Новгород,Брянск],l);

б) ? номер(2,[3,5,2,1,2],n);

в) ? номер (мама,[бабушка,дедушка,тетя,дядя],k);

г) ? сумма ([2,4,8,16],s);

д) ? произвед([3,5,7],p);

3. Написать правила для определения:

а) количества нулей в списке;

б) максимального элемента в числовом списке;

в) суммы четных элементов списка;

г) количества четных элементов в списке.

23. Слияние списков

Рассмотрим задачу соединения двух списков в один. Для ее решения можно использовать следующую программу:

присоединить([],L,L);
присоединить([X|L1'],L2',[X|L3'])<-присоединить(L1',L2',L3');

Здесь первый и второй аргументы — исходные списки, а третий — новый список, получаемый в результате их слияния.

Поясним как работают эти правила:

1. Первый элемент первого списка X будет также первым элементом нового(третьего) списка.

2. Хвост третьего аргумента L3' — результат присоединения второго аргумента L2' к хвосту первого списка L1'.

3. Для присоединения одного списка к другому используется предикат «присоединить».

4. Так как при любом обращении к правилу удаляется голова списка, являющегося первым аргументом, то постепенно этот список станет пустым и произойдет выход на граничное условие.

Рассмотрим пример выполнения этого правила: зададим вопрос

?присоединить([1,2,3],[4,5,6],p);

1-я копия	**X=1,L1=[2,3],L2=[4,5,6],присоединить([2,3],[4,5,6],L3'), L3=[2,3,4,5,6],[X|L3']=[1,2,3,4,5,6]**
2-я копия	**X=2,L1=[3],L2=[4,5,6],присоединить([3,],[4,5,6],L3'), L3=[3,4,5,6],[X|L3']=[2,3,4,5,6]**
3-я копия	**X=3,L1=[],L2=[4,5,6],присоединить([],[4,5,6],L3'), L3=[4,5,6],[X|L3']=[3,4,5,6]**

ЗАДАЧИ

1. Выделить в отдельный список те элементы заданного списка, которые лежат в интервале от 1 до 5.

2. Перевернуть список наоборот (реверс).

3. Найти количество элементов списка, равных максимальному.

24. База знаний «Железная дорога»

Теперь мы достаточно подробно познакомились с языком ПРОЛОГ и сможем сами создать базу знаний и написать правила для поиска ответов на различные вопросы (в том числе и не предусмотренные при ее разработке).

Пусть в базе знаний содержатся факты следующих видов:

поезд(<номер>,<тип>); *где тип — "скорый" или "пасс(ажирский)",*
СВ(<номер>,<наличие>); *где наличие — "есть" или "нет",*
купе(<номер>,<наличие>); *где наличие — "есть" или "нет",*
отправление(<номер>,<со-станции>,<время>);
остановки(<номер>,<список-остановок>);
прибытие(<номер>,<список-часов-и-минут-прибытия>);
стоянка(<номер>,<список-минут-стоянки>);

Например, факты об одном поезде могут быть такими:

поезд(34,скорый);
СВ(34,есть);
купе(34,есть);
остан(34,[Тула,Орел,Курск,Симферополь]);
отправление(34,Москва,9,45);
прибытие(34,[11,30,15,20,17,50,5,40]);

час прибытия в Тулу; минуты прибытия в Тулу и т. д.

стоянка(34,[10,15,11]);

Теперь будем писать правила для поиска ответа на различные вопросы.

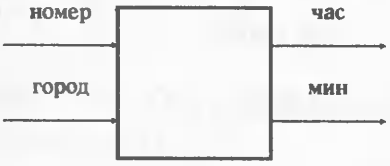
1. Какие поезда останавливаются в Орле?

останов(город', номер')<−остан(номер',список'),
принадл(город',список');

То есть поезд останавливается в данном городе, если этот город принадлежит списку остановок этого поезда.

2. Во сколько прибывает поезд с данным номером в данный город ?

Изобразим «черный ящик»(слева). Правило будет такое:



прибудет(номер',город',час',мин')<−
остан(номер',список'),номер(город',список',t),
УМНОЖИТЬ(t,2,p),ВЫЧИТАНИЕ(p,1,q),
прибытие(номер',список1)',
номер(мин',список1',p),
номер(час',список1',q);

То есть если номер этого города в списке остановок данного поезда t, то номер минут времени его прибытия туда равен p=2×t, а часа — q=p−1. Здесь используется правило для определения номера элемента в списке, составленное в разделе 22.

3. Сколько минут стоит данный поезд в данном городе?

Изобразим «черный ящик»(слева). Правило будет такое:

стоит(номер',город',мин')<−
остан(номер',список'),номер(город',список',k),
стоянка(номер',список1'),
номер(мин',список1',к);

То есть номер города в списке остановок и номер минут стоянки в списке совпадают.

4. Во сколько отправится данный поезд из данного города?

```
отправл(номер',город',час',мин',сут')<–прибудет(номер',город',час1',мин1'),
стоит(номер',город',мин2'),сумма(час1',мин1',0,мин2',час',мин',сут');
```

Здесь мы ко времени остановки прибавляем время стоянки, используя для суммирования правило «сумма», составленное в разделе 10.

5. Какие поезда прибывают в данный город ночью (т. е. после 22.00, но до 6.00)?

```
ночью(номер',город')<–прибудет(номер',город',час',мин'),
      БОЛЬШЕ(час',21);
ночью(номер',город')<–прибудет(номер',город',час',мин'),
      МЕНЬШЕ(час',6);
```

6. Каким скорым поездом можно доехать до данного города?

```
скорым(номер',город')<–поезд(номер',скорый), останов(номер',город');
```

7. Каким поездом можно доехать от одного города до другого?
Здесь надо рассмотреть два случая:
а) Первый город — пункт отправления, второй — остановка.
б) Оба города — остановки.
Соответственно, получим два правила:

```
можно(город1',город2',номер')<–отправл(номер',город1'),останов(номер',город2');
можно(город1',город2',номер')<–останов(номер',город1'),останов(номер',город2');
```

8. Каков пункт назначения поезда с данным номером?

```
назначен(номер',город')<–остан(номер',список'),длина(список',k),
      ВЫЧИТАНИЕ(k,1,p),номер(город',список',p);
```

Здесь используются правила из раздела 22 («номер» и «длина»).

ЗАДАЧИ И УПРАЖНЕНИЯ

Написать правила для ответа на следующие вопросы:

1. Определить, в каких городах останавливается поезд с данным номером (выдать города по отдельности).

2. Выяснить, верно ли, что поезд с данным номером останавливается в двух данных городах?

3. Верно ли, что поезд с данным номером останавливается во всех городах данного списка?

Записать соответствующие вопросы.

ОТВЕТЫ И РЕШЕНИЯ

К разделу 20

```
перевод(0,_,"0");
БР(а,в)<–БОЛЬШЕ(а,в);
БР(а,а);

перевод(р,н,с)<–БОЛЬШЕ(н,р),СТРЦЕЛ(с,р);
перевод(р,н,с)<–БР(р,н),УМНОЖЕНИЕ(н,т,о,р),БОЛЬШЕ(н,о),СТРЦЕЛ(л,о),
                перевод(т,н,в),СЦЕП(в,л,с);

работа<–ВЫВОД("Введите число  "),ВВОДЦЕЛ(р),
ВЫВОД("Введите основание  "),ВВОДЦЕЛ(н),МЕНЬШЕ(н,10),
перевод(р,н,с),ВЫВОД ("перевод :  ",с);
```

Примечание. Правило «БР» проверяет, что первое число больше, или равно второму.

К разделу 22

1. а) Сумма элементов числового списка:

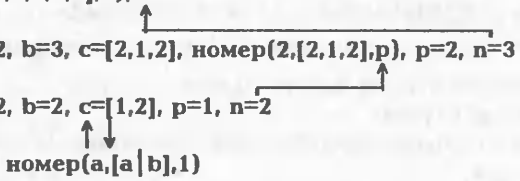
```
сумма([],0);
сумма([a|b],s)<-сумма(b,t),СЛОЖЕНИЕ(a,t,s);
```

б) Произведение элементов числового списка:

```
произвед([],1);
произвед([a|b],p)<-произвед(b,k),УМНОЖИТЬ(a,k,p);
```

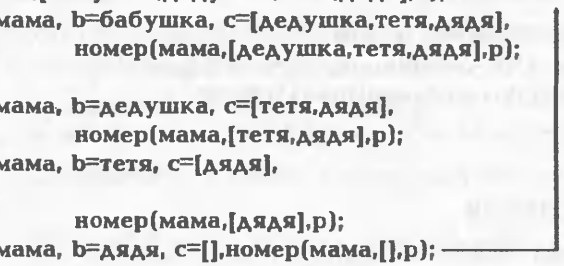
2. а), г), д) Решаются аналогично примеру, рассмотренному в тексте.

б) **? номер(2,[3,5,2,1,2],n);**



Обратите внимание, что по данному правилу находится номер п е р в о й двойки в списке.

в) **? номер (мама,[бабушка,дедушка,тетя,дядя],k);** **"НЕТ"**



Нет ни правила ни факта, который можно было бы применить. Следовательно, ответ машины: "НЕТ".

3.

а) Количество нулей в списке:

```
нулей([],0);
нулей([0|X],A)<-нулей(X,Y),СЛОЖЕНИЕ(Y,1,A);
нулей([Y|X],A)<-НЕ(РАВНО(Y,0)),нулей(X,A);
```

б) Максимальный элемент в числовом списке:

```
макс([X],X);
макс([X|Y],X)<-макс(Y,B),БОЛЬШЕ(X,B);
макс([X|Y],A)<-макс(Y,A),НЕ(МЕНЬШЕ(A,X));
```

в) Сумма четных элементов списка:

```
чет(A)<-УМНОЖЕНИЕ(X,2,0,A);
суммачет([],0);
суммачет([X|Y],A)<-чет(X),суммачет(Y,P),СЛОЖЕНИЕ(P,X,A);
суммачет([X|Y],A)<-НЕ(чет(X)),суммачет(Y,A);
```

г) Количество четных элементов в списке:

```
чет(A)<-УМНОЖЕНИЕ(X,2,0,A);
```

```
кол([],0);
кол([H|U],N)<-чет(H),кол(U,E),СЛОЖЕНИЕ(E,1,N);
кол([H|U],N)<-НЕ(чет(H)),кол(U,N);
```

К разделу 23

1.

```
в_списке(a)<-БОЛЬШЕ(a,0),МЕНЬШЕ(a,6);
выдел([],[]);
выдел([x|y],a)<-в_списке(x),выдел(y,b),присоед(b,[x],a);
выдел([x|y],a)<-НЕ(в_списке(x)),выдел(y,a);
присоед([],l,l);
присоед([x|L1'],L2',[x|L3'])<-присоед(L1',L2',L3');
```

2.

```
реверс([],[]);
реверс([x|y],a)<-реверс(y,b),присоед(b,[x],a);
```

3.

```
кол([x],1);
кол(t,[t|y],n)<-макс([t|y],t),кол(t,y,m),СЛОЖЕНИЕ(m,1,n);
кол(t,[x|y],n)<-макс([x|y],t),НЕ(РАВНО(x,t)),кол(t,y,n);
макс([x],x);
макс([x|y],x)<-макс(y,b),БОЛЬШЕ(x,b);
макс([x|y],a)<-макс(y,a),НЕ(МЕНЬШЕ(a,x));
ном(x,[],0);
ном(x,[x,y],n)<-ном(x,y,m),СЛОЖЕНИЕ(m,1,n);
ном(x,[a,b],m)<-НЕ(РАВНО(a,x)),ном(x,b,m);
```

К разделу 24

1.
```
стоянки(номер',город')<-остан(номер',список'),
  принадл(город',список');
```

2.
```
остановится(номер',город1',город2')<-стоянки(номер',город1'),
                                     стоянки(номер',город1');
```

3.
```
остановочки(номер',список2')<-остан(номер',список'),
            стоянки(номер',город'),принадл(город',список'),
            принадл(город',список2');
```

Литература

1. *Каймин В. А., Щеголев А. Г., Ерохина Е. А., Федюшин Д. П.* Информатика: Пробный учебник для 10—11 классов средних школ и техникумов. М.: Просвещение. 1989, 1990.

2. *Клоксин У., Меллиш К.* Программирование на языке ПРОЛОГ. М.: Мир. 1987.

3. *Дж. Доорс и др.* ПРОЛОГ — язык программирования будущего. М.: Финансы и статистика. 1990.

4. *Стерлинг Л., Шапиро Э.* Искусство программирования на языке ПРОЛОГ. М.: Мир. 1990.

5. Журнал «Информатика и образование»: 1987, № 4; 1989, № 4; 1990, № 2, 4, 5, 6; 1991, № 1. Цикл статей о ПРОЛОГЕ С. Г. Григорьева.

Н. Ю. Пахомова,
методист информатики и вычислительной техники
Северо-Восточного округа, Москва

ПРОБЛЕМНЫЙ МЕТОД НА УРОКАХ ИНФОРМАТИКИ*

Вечером сажусь дома за стол, думая, как завтра в классе буду объяснять новую тему. Можно поступить так: объяснить-рассказать и дать упражнения для иллюстрации изложенной информации и закрепления ее в памяти учеников, потом показать, как можно использовать полученные знания на практике. И еще надо проверить, как усвоен уже пройденный материал, отработать и закрепить навыки его применения. После того, как решены все эти учебно-воспитательные задачи, почти не остается времени на то, чтобы развить умственные способности. Ибо иллюстративно-объяснительный метод (а я имею в виду именно его) не учит думать. А еще нужно держать на должном уровне познавательную активность учащихся, приучать к самостоятельности, развивать личность. И все это надо вместить в один урок.

А что если сделать так: начать урок с решения задачи? Задачу подобрать такую, чтобы в ней требовалось использовать уже полученные знания, что будет полезно для закрепления навыков применения пройденного. Если провести коллективное решение на доске, то, пригласив учеников писать программу, я увижу, насколько хорошо усвоен материал, кто как включился в работу. Приглашать к доске буду в порядке, известном только мне, — это активизирует внимание. А первого выберу с помощью детской считалочки — это укрепит наши хорошие отношения, сделает «случайной» процедуру выбора. Решение будем писать на доске — тогда я смогу управлять (до некоторой степени) процессом создания программы, ведь вариантов программирования одной и той же задачи очень много. Запись на доске позволит акцентировать внимание на трудных и важных местах, провести обобщения.

А может быть, дать такую задачу, чтобы, решая, пришлось использовать и что-либо новое? Объяснять новое лучше в процессе решения, в момент, когда, казалось бы, все зашли в тупик и столкнулись с проблемой нехватки прежнего объема знаний. Но тогда этого нового должно не хватать совсем немножко, ровно столько, чтобы объяснение не охладило желания решать задачу.

На одном из уроков мне необходимо было рассказать ребятам, что в качестве параметра при обращении к подпрограмме в ФОРТРАНЕ можно передавать не только значения переменной и константы, находящиеся в той или иной ячейке, но и адрес самой ячейки памяти, объяснить назначение оператора EXTERNAL.

Тема важная, она приоткрывает новый уровень в программировании. Ведь подпрограмма-функция в ФОРТРАНЕ имеет имя, используемое и как имя программного модуля, и как имя ячейки памяти для возврата значения. Передача адреса такой ячейки в качестве параметра при обращении к какой-либо подпрограмме позволяет варьировать адресацию из подпрограммы к подпрограммам-функциям. Появляется новая степень свободы в программировании.

Тема эта стоит в конце изучения языка ФОРТРАН и предполагает, что курс по программированию в основном пройден, уже усвоены модульное программирование и связь подпрограмм через параметры.

---

* Данная статья представляет собой журнальный вариант главы из только что вышедшей в издательстве «Просвещение» книги Н. Ю. Пахомовой «Педагогические находки: девять граней опыта учителя информатики. Книга для учителя». По вопросам приобретения книги обращайтесь в редакцию журнала.

Рассуждаем так: в нашей программе должна быть подпрограмма, которая нуждается в вызове подпрограммы-функции, причем то одной, то другой в зависимости от выбора вызывающей подпрограмму программной единицы. Итак, минимальное количество модулей: основная программная единица, подпрограмма и две подпрограммы-функции.

ФОРТРАН — язык для программирования расчетных задач. Естественно, и наша задача скорее всего должна быть расчетной. Возможно, подпрограмма смогла бы реализовать какой-либо вычислительный метод, а подпрограмма-функция — вычисление значения функции.

Возьмем **задачу о численном интегрировании**, с которым ученики знакомятся на уроках вычислительной математики. Формула метода трапеций проста как для понимания, так и для программирования:

$$S = 1/2 \sum_{1}^{n} (f(a_i) + f(a_{i+1})).$$

В формуле $n+1$ раз вычисляется значение подынтегральной функции. Будем вычислять площади, ограниченные кривыми функций $f(x)$ и $g(x)$, изображенными на рис. 1. Пусть функция $f(x)=e^x$, а $g(x)=-x+1$. Интервал интегрирования от $-a$ до 0 и от 0 до $+a$, где a — константа.

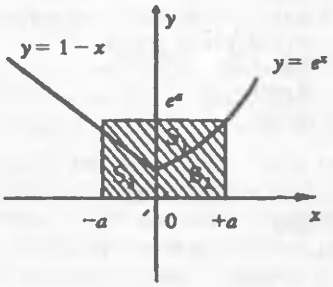


Рис. 1

Задача: вычислить соотношение площадей S_1 и S_2+S_3, используя метод трапеций, где (см. рис. 1)

$$S_2 = \int_0^a e^x\,dx,$$

$$S_3 = \int_{-a}^0 (-x + 1)\,dx,$$

$$S_1 = 2a \times e^a - (S_2+S_3).$$

Используя понятия блок-схемы связи модулей, подпрограммы метода трапеций, подпрограммы-функции, основной программной единицы, приступаем к решению задачи.

На доске появляется блок-схема:

Все задачи реализуем на Ямахе MSX.

Проанализируем получившуюся программу:

```
c     основная программная единица
      EXTERNAL FX, FE оператор, появляющийся потом
      READ (5,1) A
1     FORMAT (1X,F5.2)
      OTN=TRAP(0.,A,FE,10)+TRAP(-1.*A,0.,FX,10)
      OTN=(2.*A*FE(A)-OTN)/OTN
```

```
      WRITE (6,1) OTN
      STOP
      END
c     подпрограмма метода трапеций
      FUNCTION TRAP(A,B,F,N)
      H=(B-A)/N
      N=N-1
      TRAP=0.
      DO 2 I=1,N
    2 TRAP=TRAP+F(A+I*H)
      TRAP=H*(TRAP+.5*F(A)+.5*F(B))
      RETURN
      END
c     вычисление значения функции e^x
      FUNCTION FE(X)
      FE=EXP(X)
      RETURN
      END
c     вычисление значения функции -X+1
      FUNCTION FX(X)
      FX=1.-X
      RETURN
      END
```

Когда почти все готово, прошу ребят обратить внимание на фактические параметры FX и FE, которые получает подпрограмма TRAP в качестве значения фиктивного параметра F. Что же мы в действительности передаем? Оказывается, если не сделать специального указания транслятору, то он поймет, что передается содержимое ячеек FX и FE, а необходимо передавать адрес точек входа в подпрограммы-функции с именами FX и FE.

В этот момент появление оператора EXTERNAL остро необходимо, так как понятно его предназначение. Учащиеся получают разрешение сложившегося в задаче противоречия и прочно усваивают использование оператора. Я показываю по тексту программы место и синтаксис оператора. Его место — в числе операторов-описателей до первого выполняемого в основной программной единице. Его функция — объявить величины FX и FE адресными для последующего использования в качестве передаваемых величин фактических параметров.

А теперь давайте попробуем понять, что мы проделали на уроке. Познакомились с новым оператором ФОРТРАНА в одно касание. Но ребята усвоили оператор так, что мне больше не надо будет напоминать о его назначении и применении. Написанная нами программа — пример использования и применения этого оператора. Кроме того, ученики думали, ставили вопрос и искали на него ответ. Мы имели все, что дает на уроках программирования решение задачи — обучение деятельности в процессе самой деятельности, т. е. был использован *проблемный метод обучения*.

Однажды попробовав строить урок на основе решения задачи, я делала это вновь и вновь, замечая, насколько быстро мои ученики осваивают навыки самостоятельного мышления в программировании. И действительно, если тетрадки учеников в первые годы моей работы походили на конспекты лекций, теперь вы найдете в тетрадях, если так можно сказать, «решебник» специальной подборки задач.

Итак, в основе урока, построенного на использовании проблемного метода, лежит задача. Для школьников задача должна быть безусловно привлекательной своей формулировкой. Однако то, что интересно в одном возрасте, может не привлекать детей другого возраста.

Психологическое воздействие задачи известно. Если задача привлекательна для ученика своей сюжетной основой, если она на первый взгляд кажется ему посильной, он не пройдет мимо. Акцентированное внимание перерастет в любопытство, любопытст-

во — в азарт, азарт — в упорство преодоления. В тот момент, когда задача прочитана и осознана, возникают внутреннее противоречие, неуспокоенность тем, что надо найти решение. Мы получаем пусковой механизм мыслительной деятельности, активизирующий мышление, память, внимание.

Это замечательное свойство задач можно использовать на уроках, подбирая задачи так, чтобы извлечь максимальный обучающий эффект.

Во-первых, задача обеспечивает возникновение проблемной ситуации, которая разрешается с подачей нового материала.

Во-вторых, задача — это задание для деятельности, которой, собственно, школьники и учатся.

И, в-третьих, задача может быть использована как средство управления мыслительной деятельностью, как средство обучения думанию.

Как же найти или составить такую задачу? Я давно пришла к выводу, что задачи для своих уроков я должна придумывать сама и ничего сложного в этом нет.

Как я составляю задачу

Нам предстояло изучить тему «Математическое моделирование». Основное в понятии «математическая модель» — это условность, приближенность любой модели. Математическое моделирование — область приложения точной науки к повсюду неточной действительности.

Неточность начинается с несовершенства наших органов чувств, неточности приборов измерения и получения исходных данных. Неточность, приближенность есть и в любой попытке с помощью точных формул описать взаимозависимость вещей в природе. Это очень важное мировоззренческое понятие ложится в основу понимания математического моделирования как такового, необходимости поиска более точной модели, если результат использования прежней недостаточно точен.

Эти серьезные понятия, используемые инженерами и учеными в научной и конструкторской работе, необходимо было объяснить школьникам, упростив объяснение, сведя его к уровню их жизненного опыта и миропонимания.

С одними детьми я должна продвигаться крошечными шажками, от задачи к задаче постепенно наращивая понятия (например, по теме циклы и массивы: индекс массива как переменная, индекс массива как выражение, перебор элементов массива и их обработка и прочее).

С другими все это можно использовать сразу в одной задаче. Нам нужна такая задача, которая будет живым воплощением математического моделирования, в процессе решения которой я хотела бы закрепить ранее узнанное учащимися (исходя из потребностей программы курса).

Вот задача для первого разговора о математическом моделировании.

Задача «Укос»

Вы арендаторы и знаете, что река Одь, как и многие равнинные речки, течет, сильно извиваясь. Ее правый берег — заливной луг, ограниченный прямым как стрела шоссе.

За лето с этого луга можно получить 30 кг сена с каждых 100 м^2. Сколько сена можно было бы получить со всего луга? Посчитайте, выгодно ли брать этот луг в аренду.

Как решать такую задачу? Об интегрировании ребята еще ничего не знают.

Предлагаю им посчитать площадь как сумму площадей узеньких прямоугольных полосок.

Рассуждаем так. Площадь каждой полоски посчитать легко. Ее ширину мы выберем сами, а длина... Длина? А что, если по шоссе направить ось OX? Контур реки тогда похож на график какой-то функции, а длина отдельной полоски будет означать значение

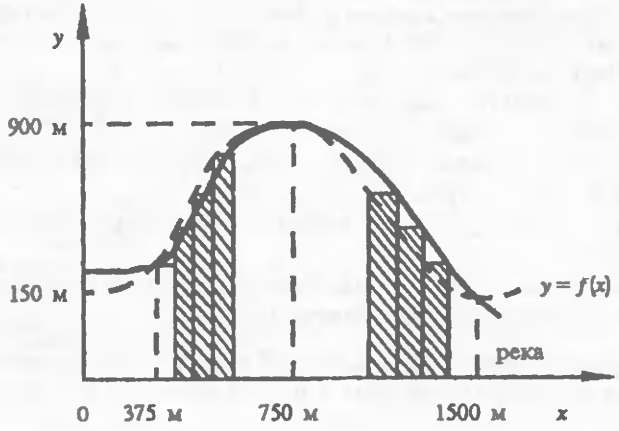


Рис. 2

функции в соответствующей точке (рис. 2) Возникает новая проблема: выбор положения оси OY и запись формулы функции-реки.

Дети уже знакомы с тригонометрическими функциями и с построением их графиков. Здесь задача обратная: по виду графика записать функцию.

Мы все в состоянии поиска, идем наощупь, рассуждаем, появляется масса попутных идей и соображений.

В конце концов мы находим формулу нашей функции:

$f(x) = k \times \sin \mu \times (x-a)+b$,

где k, μ, a, b — числовые коэффициенты, которые необходимо определить. k — коэффициент растяжения по оси OY, μ — коэффициент растяжения по оси OX, a — сдвиг вправо по оси OX, b — сдвиг вверх по оси OY. На чертеже кривая этой функции показана пунктирной линией. Ребята понимают первое допущение и приближенность такой модели.

Пусть ось OY пройдет через левый край луга перпендикулярно шоссе. Если вся длина луга 1500 м, а он кажется нам симметричным относительно середины, т. е. перпендикуляра на отметке 750 м, то первая точка перегиба находится на отметке 375 м, т. е. a=375 м. От левого до правого края кривая делает полный период — значит, 1500 м соответствуют 2π или 6,283 радиан. Коэффициент растяжения μ=6,283/1500. Минимальное удаление участка от шоссе — 150 м, максимальное — 900 м, т. е. b=150 м, $k = (900 - 150)/2$ м.

Рассуждая таким образом, мы пришли к определению значений коэффициентов k, μ, a, b, а затем и к возможности запрограммировать вычисление площади луга.

В этой задаче на глазах ребят оживают и начинают работать абстракции строгой науки. На чертеже видно, что наша река не идеальная функция — где-то она ниже, где-то выше кривой $f(x)$. Кривую $f(x)$ мы специально выделили на чертеже пунктирной линией. Ребята понимают первое допущение и приближенность такой модели. Затем сам метод измерения площадей узкими полосками приводит к еще одной неточности.

Как понять, насколько груба наша модель, как оценить степень ее приближения? Эти вопросы возникают у ребят по ходу решения задачи. Обещаю на последующих уроках рассказать им о работе с приближенными значениями и погрешностями. По опыту знаю: скучнейшая тема, если давать ее просто так, вне связи с жизнью, а мои ребята уже почувствовали потребность в этих знаниях.

Разработка же алгоритма и написание программы дело привычное, легкое.

Задача «Муха и поезда»

Эту задачу я нашла у Ж.-П. Ламуатье в «Упражнениях по программированию на ФОРТРАНЕ IV» и использую для того, чтобы вместе с учениками пережить минуты поиска перехода от *проблемной* постановки (формулировки) задачи к *математической*, пригодной для последующего программирования.

Задача. Два поезда движутся навстречу друг другу: один из пункта А со скоростью 60 км/ч, другой — из пункта В со скоростью 40 км/ч. Между ними туда и обратно летает исключительно быстрая Муха со скоростью 200 км/ч. Необходимо вычислить количество перелетов и суммарный путь Мухи. Расстояние между пунктами 160 км.

Хочу заметить, что в программировании этой задачи есть один замечательный «подводный камушек»: исчезновение порядка в процессе счета программы.

Ж.-П. Ламуатье, облегчая решение задачи, ограничил количество перелетов в программе числом *n*, а в примере распечатки результатов дал не более 25 перелетов. Таким образом, он не вышел на эту ситуацию «подводного камушка».

Ученики в процессе отладки программы столкнутся с этим явлением, будут озадачены, попытаются понять, что случилось. Я всегда буду готова прийти им на помощь. Для них будет небольшим открытием то, что в машине нет очень малых величин, что 0 там без интервала, его содержащего, что записываемые в память числа дискретны.

Формулировку задачи начнем с рисования картинки. Игровой сюжет задачи напоминает детскую игру, полусказочным существом кажется «исключительно быстрая Муха».

Самые сообразительные сразу предлагают вычислить суммарный путь Мухи. Но трезвые головы их останавливают — нам необходимо выяснить, кроме того, и количество перелетов Мухи от поезда к поезду.

Сначала Муха летит из пункта А навстречу поезду В. Вспоминаем и записываем в общем виде те формулы, по которым затем будем производить расчет. Известны начальное расстояние и скорости V и V_2 сближающихся объектов, значит, можно вычислить время встречи:

$t_1=d_1/(V+V_2)$.

Затем вычисляется путь Мухи на первом перелете:

$S_1=V\times t_1$.

и новое расстояние между поездами в момент разворота Мухи:

$d_2=d_1-t_1(V_1+V_2)$.

Теперь Муха летит обратно. Ее время до разворота на втором перелете вычисляется уже по другой формуле:

$t_2=d_2/(V+V_1)$.

Остальные формулы —

$S_2=V\times t_2$

и

$d_3=d_2-t_2\times(V_1+V_2)$ —

очень похожи.

Мы продолжаем формализацию. Выписываем группу формул для третьего перелета, пока не раздается ребячий голос, сообщающий о том, что надо прекратить выписывание, так как на четвертом перелете будут верны те же формулы, что и на втором, а на третьем и пятом те же, что на первом. Теперь для того, чтобы процесс вычисления был действительно повторяемым, надо в формулах избавиться от индексов, означающих лишь номер перелета.

Рассуждаем так. Представим, что наши буквенные математические обозначения соответствуют ячейкам памяти, содержащим соответствующие величины. Тогда в ячейке переменной t у нас будет время перелета; V, V_1, V_2 будут содержать постоянные скорости соответственно Мухи, поезда А и поезда В; d — расстояние между поездами в начале перелета; S — длина пути перелета Мухи; а в ячейках переменных Р и К будут храниться суммарный путь и количество перелетов Мухи соответственно.

Наши формулы на языке операторов ЭВМ будут похожи на математические, только

знак «=» будет означать операцию присваивания. Действительно, давайте прочтем первую из них:

$t=d/(V+V_2)$

означает: в ячейку t положить результат от деления d на сумму V и V_2. Теперь

$S=V\times t$

есть помещение произведения значений из V и из t в ячейку S. В ячейке P должен быть изначально 0, тогда

P=P+S

будет означать добавление к прежнему значению P величины S и помещение опять в P.

Если сделаем это всякий раз, как только вычислим длину перелета, то в результате накопим суммарный путь Мухи. Количество перелетов тоже надо считать. Ячейка K — сумматор количества перелетов. Вначале там должен быть 0, а затем при каждом развороте Мухи мы увеличим прежнее значение на единицу, т. е.

K=K+1.

Новый такт вычисления начинается с установления нового расстояния между поездами в момент разворота Мухи:

$d=d+t\times(V_1+V_2)$.

Здесь опять d справа от знака «=» означает прежнее расстояние (до начала предыдущего такта сближения), а d слева — новое. Далее опять вычислим t, S, R и K, зациклив выполнение прежних, уже выписанных операторов. Будем это делать до тех пор, пока... Здесь — внимание!!! — «подводный камушек», о котором я говорила выше. Условие окончания цикла вроде бы очевидно — пока d не станет равным нулю. Но в процессе вычислений t становится настолько малым, что происходит исчезновение порядка, вследствие чего d перестает уменьшаться. Складывается ситуация вечного цикла.

От этой ситуации мои ребята в полном недоумении перед проблемой. Распечатываю промежуточные результаты, чтобы всем вместе проанализировать ситуацию. Вот ее фрагмент. Здесь i — номер перелета.

i=284	t=2.4719412016293E−64	d=3.9551059226068E−62
i=285	t=1.6479608010862E−64	d=2.3071451215206E−62
i=286	t=0	d=2.3071451215206E−62
i=287	t=0	d=2.3071451215206E−62

Мы возвращаемся к анализу наших формул, думаем, пытаемся понять, когда, в какой момент получилось нулевое значение у t. Если в процессе обсуждения ситуации не возникает вопроса о минимально возможном числе в памяти ЭВМ, я в нужную минуту его подбрасываю. И мы приходим к выводу, что процесс надо остановить при $d\leq 2.3\times 10^{-62}$ на 285-м перелете (это очень малая величина). А лучше — исходя из здравого смысла, т. е. при $d\leq 10^{-5}$.

Эта задача дала возможность усвоить представление о дискретном характере хранимых в памяти ЭВМ чисел.

Программирование — это одновременно и искусство, и наука, и ремесло. Конечно, нужно поупражняться, чтобы освоить секреты ремесла, те или иные приемы алгоритмизации и кодирования. Для этого очень важно нарешать большое количество задач. Но не менее важно, развивая навыки программирования, воспитывать определенный стиль мышления, вырабатывать дисциплину ума.

Ведущими программистами мира признаны наиболее эффективными принципы модульного программирования, разработки алгоритмов сверху вниз с пошаговой детализацией, структурное программирование (Йодан, Дейкстра, Вирт и другие). Наиболее популярными среди программистов являются сейчас именно структурированные языки, позволяющие избежать при программировании алгоритмических ошибок.

Для того чтобы настроить мышление учащихся на стиль структурного программирования, удобно использовать школьный алгоритмический язык (разработанный под руко-

водством А. П. Ершова). Он не формализован, и детали его могут видоизменяться каждым учителем, но основные алгоритмические конструкции в нем неизменны.

В моей практике составление алгоритмов и запись их на школьном алгоритмическом языке (ШАЯ) предшествуют кодированию программы на БЕЙСИКЕ.

Начинающих программистов надо с первых шагов учить писать и даже мыслить структурно. Одна из хорошо работающих методик создания структурированных программ — разработка сверху вниз с пошаговой детализацией.

Когда задачи становятся не слишком элементарными, показываю приемы пошаговой детализации при разработке сверху-вниз, используя своеобразное расширение школьного алгоритмического языка.

Задача для первого знакомства с этой технологией посвящена работе с символьными данными.

Задача «Слово»

Ввести слово. Узнать:
а) сколько букв «о» в слове;
б) есть ли удвоенные буквы;
в) сколько каких букв.

Покажу разработку пункта а.

Структурная разработка задачи «Слово»
(пошаговая детализация)
I этап

```
<*1: Ввод СЛОВА *>
<*2: Посимвольный анализ слова на содержание буквы «о» *>
<*3: Сообщение о количестве букв «о» в слове *>
```

II этап

Блок <*1: не нуждается в дальнейшей детализации (одна команда на БЕЙСИКЕ).
Блок <*3: — то же самое.

Детализация блока <*2:

```
<*2.А: Вычисление N — длины слова *>
<*2.Б: Подготовка (очищение) S — сумматора *>
для i от 1 до N
нц
| <*2.В: Выделение из СЛОВА i-го символа *>
| <*2.Г: распознавание буквы «о» *>
кц
```

III этап

Все блоки <*2.А, <*2.Б и <*2.В просты (одна строка на БЕЙСИКЕ)

Детализация блока <*2.Г:

```
если СИМВОЛ=«о»
| то S=S+1
все
```

Конец разработки

Здесь в угловые скобки со звездочкой (<* и *>) заключены названия тех блоков, которые детализируются на последующих этапах. Фактически описанная последовательность действий является формулировкой задания на последующую разработку блоков

программы. На каждом шаге детализации блоки образуют весьма простые алгоритмические конструкции: линейные, разветвляющиеся, циклические.

Переход к БЕЙСИК-программе происходит без выписывания всего алгоритма на алгоритмическом языке.

```
10 input "введите слово"; D$          'ввод слова
20 N=LEN(D$)                          'вычисление длины слова
30 S=0                                'очищение сумматора
40 do i=1 to N                        'для i от 1 до N, начало цикла
50 B$=MID$(D$,i,1)                    'выделение символа
60 if B$="o" then S=S+1               'распознавание "o"
70 next i                             'конец цикла
80 print "в слове"; D$; "букв 'o' всего"; S
90 end
```

Решим еще одну задачу, закрепляя навыки обработки символьных данных.

Задача «Фраза»

Ввести фразу (строку). Узнать:

а) сколько слов во фразе;

б) есть ли повторы слов;

в) найти первое слева слово максимальной длины.

Структурная разработка задачи «Фраза»

(пошаговая детализация)

I этап

```
<*1: Ввод ФРАЗЫ *>
<*2: Посимвольный анализ ФРАЗЫ *>
<*3: Сообщение о результатах анализа *>
```

II этап

Детализация блока <*2:

```
<*2.1: Вычисление N — длины ФРАЗЫ *>
<*2.2: Очищение S — сумматора слов *>
<*2.3: Установка режима анализа ФРАЗЫ: T=0, где T=0 —выделение слова, T=1 — проход между словами *>
<*2.4: Установка счетчика позиции начала текущего слова — K=1*>
<*2.5: Установка максимальной длины слова среди уже рассмотренных — M=0 *>
<*2.6: Установка счетчика позиции начала слова максимальной длины — L=1 *>

для i от 1 до N
нц
| если T=0
| | то <*2.7: Выделение и обработка слова *>
| | иначе <*2.8: Проход между словами *>
| все
кц

<*2.9: Обработка последнего слова *>
```

III этап

Детализация блока <*2.7:

если *i*-й символ = «_» («_» — пробел)
| **то** S=S+1, T=0,
| <*2.7.1: Анализ длины слова на max *>
| <*2.7.2: Определение повтора слова *>
все

Детализация блока <*2.8:

если *i*-й символ не равен «_»
| **то** T=1, K=*i*
все

Детализация блока <*2.9:
<*2.9.1: Анализ длины последнего слова на max *>
<*2.9.2: Определение повтора последнего слова *>
S=S+1

IV этап

Детализация блока <*2.7.1:
если M<*i*–K
| **то** M=*i*–*k*, L=K
все

Детализация блока <*2.7.2:

если слово имеет повтор
| **то** печатать «повтор слова есть»
| **иначе** печатать «повтора слова нет»
все

Детализация блока <*2.9.1:

если M<N–K+1
| **то** M=N–K+1, L=K
все

Детализация блока <*2.9.2:
то же, что в <*2.7.2>

Конец разработки

Ниже предлагаются еще две задачи, первая из которых используется для введения в арсенал таких операторов, как DATA, READ, RESTORE, а вторая дается на закрепление этой темы.

Задача «Телефонный справочник»
По фамилии, полученной по вводу, выдать номер телефона из списка, хранимого в программе, или сообщение «Такого абонента нет».

Задача «Автопереводчик»
В программе создать базу данных, в которой будут храниться русские слова с их английскими значениями. Организовать вывод на экран:
а) английского значения слова по введенному русскому;
б) русского значения по введенному английскому;
в) альтернативное значение слова по введенному значению на любом языке.

Если слово неизвестно программе, должно печататься сообщение «Я не знаю перевода этого слова».

Задача «Сортировка»

Проходит урок по теме «Алгоритм упорядочения массива». К этому моменту дети умеют находить максимум, минимум, менять местами значения переменных, выполнять простейшую адресацию элементов одномерного массива.

Я не объявляю тему урока, не собираюсь рассказывать, что представляет собой алгоритм упорядочения перестановкой, хотя точно знаю, что именно этот способ найдут мои ребята, когда я предложу им упорядочить последовательность 2, 5, 444, 1, 3 в порядке убывания.

Задаю вопрос: «А что должно стоять на первом месте слева?»

Выписанные на доске числа выбраны такими, чтобы подсказать ответ. Он легко находится: 444. Это ведь номер нашей школы. Что нужно сделать для того, чтобы число 444 стояло на своем месте? Дети отвечают: «Поменяем местами числа 444 и 2». Тогда мы получаем:

444, 5, 2, 1, 3 .

Что же должно стоять на втором месте? Действительно, 5. А как мы мыслили, как определили? Верно, 444 стоит на своем законном месте, это максимальное число из всей совокупности. А число 5 — максимальное из совокупности, стоящей правее только что поставленного на свое место 444. Число 5 — максимальное и стоит в крайнем левом положении в рассматриваемой совокупности, значит, его никуда не надо переставлять.

Также мы можем действовать и дальше: на третьем месте ставим 3, меняя его местами с 2. Число 3 — максимальное из совокупности 2, 1, 3. И так далее.

444, 5, 3, 1, 2

444, 5, 3, 2, 1

Теперь, после того как мы «на пальцах» все упорядочили, прошу сформулировать в общем виде, что необходимо делать, в какой последовательности, чтобы получить нужный порядок чисел.

Получается следующая картина: первое действие — поиск максимального числа из совокупности; второе действие — постановка максимального числа в крайнее левое положение совокупности.

Сначала рассматриваемую совокупность составляют все элементы последовательности. Затем, после постановки самого левого на его законное место, в совокупность входят все остальные, т. е. элементы последовательности от второго до пятого. На следующем такте действия 1 и 2 повторяются для элементов новой совокупности с третьего по пятый. Все это повторяем до тех пор, пока не рассмотрим последнюю совокупность, состоящую из двух последних чисел.

А теперь обобщим найденный алгоритм.

Пусть мы имеем последовательность из n элементов: a_1, a_2, a_3, ..., a_n. Придуманный нами алгоритм можно записать так:

```
I этап

для i от 1 до n–1
нц
| <*1: поиск K такого, что a_K — максимальное из элементов
| с i-го по n-й *>
| <*2: постановка в крайнее левое положение a_K *>
кц
```

II этап

Детализация блока <*1:

max=a_i
для *k* **от** *i*+1 **до** *n*
нц
| **если** max<a_k
| | **то** max=a_k, T=K
| **все**
кц

Детализация блока <*2:

R=a_i
a_i=a_t
a_t=R

Конец алгоритма

Рассмотренные задачи позволили мне только приоткрыть богатые возможности обучения и воспитания через решение задач. По сути речь идет об использовании проблемного метода на уроках информатики и программирования.

Специфика проблемного урока здесь заключается в том, что он построен на основе решения специальной задачи, заранее сконструированной специально для выполнения поставленных перед данным уроком целей задач обучения и воспитания. *Технология конструирования специальной задачи* такова.

Ядром задачи становится та порция нового материала, которую мы собираемся дать. Она должна быть не очень большой, так как психологи предупреждают: усваивается от 5 до 9 единиц информации.

Новое в задаче должно применяться наиболее характерным образом, производить наиболее выгодный, запоминающийся эффект. Это объяснит наличие изучаемого в предмете, покажет его связи с уже известным.

Задача не должна быть трудоемкой, она должна решаться известными учащимся способами.

Первоначальное желание решать задачу можно усилить привлекательной для ребят формулировкой, интересной сюжетной или игровой формой. Практическая значимость, реальность решаемой задачи, перспектива применения компьютера укрепят мотивацию работы учащихся над задачей.

Учитель должен иметь возможность управлять ходом решения специальной задачи, что позволит ему контролировать правильность понимания учащимися сути задачи.

Побуждая к выдвижению разнообразных версий ответов, педагог учит выбирать правильную, оптимальную. Решая задачу, ученик учится думать.

Известный математик и педагог Дж. Пойя писал: «Наилучшие правила мышления нельзя получить как-то извне, их нужно выработать так, чтобы они вошли в плоть и кровь и действовали с силой инстинкта. Поэтому для развития мышления действительно полезным является только его упражнение».

Учимся думать — нужно решать задачи!

Н. П. Радченко,
учитель информатики средней школы № 61, Москва

ГУМАНИЗАЦИЯ КУРСА ОИВТ (Методические рекомендации и программа для VI—VII классов)

Мы стали свидетелями процесса компьютеризации всех сфер нашей жизни и в частности образования. Создана широкая сеть центров вычислительной техники. Занятия по информатике во многих школах начинаются уже с первого класса. Повсеместно работают кружки, факультативы. Во многих семьях есть персональные компьютеры. Обязательное изучение курса введено в VI—IX классах средних школ и в специальных учебных заведениях. В X—XI классах проходит углубленное изучение предмета с учетом специализации класса.

В предлагаемой вашему вниманию статье учитель ОИВТ московской школы № 61 Н. П. Радченко знакомит с разработанной ею программой гуманитарной направленности для VI—VII классов, дает методические рекомендации по работе с ней, останавливается на некоторых психологических особенностях преподавания ОИВТ. При этом автор рассматривает компьютер как мощное дидактическое средство обучения. С его помощью, считает Н. П. Радченко, можно воспитать нравственную личность, развить интеллект и эстетические чувства у детей и подростков.

В России уже создана как техническая, так и законодательная базы для обучения детей и юношества работе на компьютерах.

Компьютеры используются в играх, при получении начального представления о процессе программирования, выработке навыков в работе с прикладными программами: электронными таблицами, базами данных, текстовыми, музыкальными, графическими редакторами.

Остановимся на работе с последними. Как проходит процесс обучения при освоении детьми, допустим, графического редактора? Утрируя, можно сказать, что после объяснения типа «если нажать на клавишу «С», то на экране появится круг», роль учителя сводится к надзору за порядком. Результат такого обучения неразвитая личность ребенка. Посмотрите на рисунки детей: они однообразны и эмоционально бедны: танки, самолеты, домики-коробки, герои боевиков, а то и могила, крест, гроб на колесиках.

Несомненно, работу ребят нужно направить в другое русло, используя занятия ОИВТ для развития творческого потенциала. Для ребенка общение с компьютером — радость. Сколь долгой она будет? Это зависит от контекста такого общения. Работа на компьютерах — труд, в процессе которого ребенок должен стать не просто пользователем, а творцом. При этом надо включить ребенка в творческий процесс, обусловленный возрастными особенностями его психики. Возрождение ценностных ориентиров и национального самосознания — цель системы российского образования. Мы предлагаем одну из возможных форм гуманизации курса информатики, его включения в систему национального образования, связывая темы занятий с природой родного края, устным, музыкальным и изобразительным народным творчеством, взятом в историческом контексте.

Мы разработали программу, ориентированную на работу по ней в течение двух учебных лет — по одному часу в неделю, например: в V—VI или VI—VII классах (всего 68 часов). Она охватывает первоначальные общие знания о компьютере, предусматривает выработку устойчивых навыков работы с музыкальным, текстовым, графическим редакторами, творческое деятельное освоение духовного наследия русского народа (устный фольклор — пословицы, поговорки, былины, сказки; музыкальный фольклор — обрядовые и «трудовые» песни; декоративно-прикладное искусство — вышивка, кружевоплетение, дымковская игрушка, керамика Гжели, деревянное зодчество, резьба на деревянной посуде, городецкая роспись на дереве и хохломская, жостовские подносы, павлово-посадские платки, лаковая миниатюра Палеха; духовная музыка и храмовая архитектура).

Программа легко адаптируется к местным этническим условиям. Наши разработки окажут помощь учителям искусствоведческого цикла как общеобразовательного уровня, так и среднего специального (особенно в связи с набирающим силу направлением компьютерной живописи, анимации), учителям музыки, русского языка и литературы, истории. В них остро нуждаются педагоги дополнительного образования, руководители факультативов и кружков при обучении работе с музыкальным, текстовым и графическим редакторами.

Если вы решили использовать предлагаемую программу в своей педагогической практике и нуждаетесь в методическом пособии и дидактических материалах, обращайтесь к автору.

ПРОГРАММА

Первый год обучения: VI (V) класс (34 ч)

I–II четверти (16 ч)

ВВЕДЕНИЕ (3 ч)

Беседа: информация вокруг нас (примеры из окружающего мира). Многообразие форм представления информации (графическая, текстовая, звуковая). Как и для чего человек воспринимает информацию? Кодирование информации. Хранение информации. Примеры носителей информации от древнего мира до наших дней. Распознавание, хранение, обработка, поиск, передача информации (примеры и их анализ). Понятие об информационном взрыве и информационном обществе. Общее представление о компьютере. Техника безопасности и правила работы на компьютере. Первое практическое знакомство с компьютером. Включение и выключение ПЭВМ. Курсор и управление им. Возможности компьютера как исполнителя. Программный принцип работы компьютера. Загрузка и хранение программ. Понятие файла, структура имени. Каталог. Меню. Общая характеристика прикладного программного обеспечения. Программа «Правила дорожного движения». Игры.

Раздел I. Прикладное программное обеспечение

Тема 1. Текстовый редактор (13 ч)

Назначение программы. Загрузка одного из редакторов. Ввод символьной информации с клавиатуры. Буквы русского алфавита и их набор на клавиатуре. Переключение режимов заглавных и строчных букв. Режим латинских букв. Применение латинского алфавита.

Основные возможности по созданию и редактированию текста (пословицы и поговорки). Удаление и вставка символов и строк. Понятие о форматировании текста. Сохранение текстов и их считывание с МД.

Принтер. Печать текстов на бумаге. Выбор шрифта. Машинный диктант (отрывок из былины). Беседа: народные верования и фольклор; авторство в фольклоре; мифические персонажи; природа в устном народном творчестве; характеры героев, их духовные ценности; борьба добра со злом.

Подготовка поздравительных открыток к Дню учителя.

Самостоятельная работа с текстовым редактором (русские народные сказки: «Сивка-бурка», «Елена Премудрая и Серый волк», «Аленушка и братик Иванушка»; сказки А. С. Пушкина: «О мертвой царевне и семи богатырях», «О рыбаке и рыбке»). Беседа: единство явлений природы и настроения героев; образный язык; традиции в русской поэзии. Подготовка и издание книжек-сборников произведений (по выбору учащегося) фольклора и отрывков из сказок А.С.Пушкина (для себя и в качестве подарка первоклассникам к празднику Букваря), поздравительных открыток к Новому году.

III четверть (10 ч)

Тема 2. Музыкальный редактор (10 ч)

Звуки — информация. Назначение программы. Считывание с МД файлов с русскими народными мелодиями и их прослушивание. Беседа: музыка в жизни человека; характер русской песенности; выразительная речевая интонация как основа русского народного пения; природа — фольклорный родник; мировосприятие славян; обряды жизненного цикла; обрядовые песни; возвышенность образов; ассоциативное мышление; коллективные работы; ритм; соединение слова, напева и движения; хороводы, танцы.

Под аккомпанемент компьютера исполнение песен «Мы просо сеяли», «Во поле береза стояла», «Утушка луговая».

Основы музыкальной грамотности.

Ноты — элементы музыкальной информации. Основная характеристика звука — высота. Изображение нот на нотном стане. Скрипичный и басовый ключи. Длительность нот. Правило записи мелодий. Экранные окна, меню. Гамма и ее запись, сохранение на МД, считывание и прослушивание индивидуальных записей. Простейшее редактирование мелодий (замена, вставка, удаление, копирование). Многоголосие. Аккорд. Запись, прослушивание и исполнение хором под аккомпанемент компьютеров русских народных песен «Эх, масленица!», «Весна — красна».

Самостоятельный поиск детьми народных песен и работа с ними на компьютере. Подготовка нескольких песен, танцев под аккомпанемент компьютеров для коллективного ис-

полнения на школьном празднике, посвященном 8 Марта.

Христианство на Руси. Духовная музыка. Песня в Храме. Музыкальные традиции. Восприятие современным человеком. Прослушивание компьютерных записей.

IV четверть (8 ч)

Т е м а 3. Графический редактор (8 ч)

Значение визуальной информации. Представление и обработка графической информации на компьютере. Назначение программы «Графический редактор». Загрузка программы. Система графических меню. «Горячие» клавиши. Палитра, выбор цвета. Основные возможности по созданию и редактированию изображения. Плоскость. Основные графические примитивы: точка, линия, прямоугольник, круг.

Виды декоративно-прикладного искусства. Его роль в жизни людей. Три типа красоты окружающей человека среды. Критерии прекрасного в природе и изделиях декоративно-прикладного искусства. Основные понятия об орнаменте (происхождение слова, узор, законы симметрии, ритмы, повтор, упорядоченность, цвет, эмоциональность, назначение, связь с материалом и формой изделия). Демонстрация слайдов, фотографий, иллюстраций из книг, орнаментов на образцах изделий русского декоративно-прикладного искусства, одежде детей, сидящих в классе. Опосредованное отражение в орнаменте родной природы. Орнамент Древней Руси. Сохранение традиций в Новгороде и Пскове. Магия орнамента. Связь мотивов с мифологическими сюжетами. Геометрический, растительный, зооморфный орнаменты и их сочетания. Орнамент на квадрате и прямоугольнике, на круге, линейный, замкнутый. Копирование элементов изображения. Симметрия в орнаменте. Вышивка на полотенце. Особенности расположения орнамента на частях русского народного костюма. Понятие орнаментальной композиции. Орнамент древнерусских книг. Исполнение вариаций орнаментальных композиций из книг сказок. Перенесение файлов из графических редакторов в текстовый. Самостоятельная импровизация при подготовке поздравительных открыток к окончанию учебного года с применением знаний о симметрии, ритме повторов на прямоугольной форме. Посещение музея народного творчества и декоративно-прикладного искусства.

Второй год обучения: VII (VI) класс (34 ч)

I четверть (8 ч)

П о в т о р е н и е (2 ч)

Техника безопасности при работе на компьютере. Включение и выключение ПЭВМ. Назначение программы «Графический редактор». Основные возможности по созданию и редактированию изображения. Процесс записи на МД файлов и считывание их. Орнаменты, которые были записаны в VI (V) классе.

Т е м а 3 (продолжение). Графический редактор (6 ч)

Дополнительные возможности при создании изображения. Кривая на плоскости. Дуга, выпуклость, вогнутость. Замкнутый контур. Изделия из глины в декоративно-прикладном искусстве. Дымковская игрушка: ее прошлое и настоящее. Характерные особенности формы, цвета и орнаментального декорирования. Рисование дымковских игрушек (барыня, кавалер, крестьянка, корова, конь, козлик, курочка) по образцам или иллюстрациям. (2 ч)

Окружности, овалы, касание, разомкнутый контур, продолжение одной кривой — начало следующей. Искусство кружевоплетения. Технология изготовления. Демонстрация фотографий и образцов. Вологодские, вятские кружева. Характерные особенности. Орнамент, сюжет, традиции. Копирование подзоров с условно-растительным орнаментом, «сеточки», «звездочки», «паучки», скатертей «Морозные узоры», «В саду». Подготовка «кружев» и «игрушек» — подарков к Дню учителя. (4 ч)

II четверть (7 ч)

Т е м а 3 (продолжение). Графический редактор (7 ч)

Деревянное зодчество. Природные предпосылки. Крестьянская изба. Главный фасад, красные окна, карнизы, наличники. Орнаментальная резьба по дереву. Вариации элементов резьбы (лев, русалка-берегиня, птица-сирин) и их своеобразие; симметрия, выявление центра. Составление орнаментов из их мозаики; неотделимость орнамента от конструкции, ее подчиненность орнаменту.

Деревянная посуда. Пряничные доски. Резные орнаменты на прялках, ложках. Рельефность и пластическая выразительность орнамента. Декоративные качества дерева, физические и механические свойства древесины, ее недостатки. (3 ч)

Работа с палитрой, смешивание цветов.

Кистевая роспись. Городецкая роспись на

дереве. Крестьянское искусство. Детали растительного орнамента (травчатое письмо). Мифологические персонажи. Элементы древнерусской фресковой живописи и древнерусской книги. Изменение во времени. Жанровые сцены. Изменение их композиции. Использование фольклора. Демонстрация фотографий, иллюстраций, образцов. Повтор росписи образцов. Импровизация с мотивами и деталями городецкой росписи на зимние сюжеты. Выставка к Новому году — «Зимние забавы». (4 ч)

III четверть (10 ч)

Повторение (10 ч)

Создание изображения из кривых плавных линий.

Хохломская роспись по дереву. Демонстрация образцов (кухонная и столовая посуда), фотографий мебели, ваз. Исторические и географические сведения. Промысел. Технология изготовления. Отличительные признаки. Мотивы росписи. Растительные узоры. Мир растений средней полосы России — источник вдохновения. Связь с устным фольклором. Копирование образцов с традиционными мотивами (травка, рябина, земляника, крыжовник, смородина) и импровизация. Выполнение графических контуров, вывод на печать и ручная домашняя раскраска в традиционной цветовой гамме. (5 ч)

Керамика Гжели. Демонстрация образцов, иллюстраций. Декоративная роспись. Особенности мотивов и цветового строя. Условность. Цветочная композиция. Масштабное соотношение элементов композиции. Расположение на изделии. Круговая композиция. Копирование несложных плоской и объемной композиций. (2 ч)

Жостовские подносы. Павлово-Посадские платки. Цветы России. История промыслов. Демонстрации иллюстраций и образцов. Эстетическая выразительность цветочной росписи. Цвета и оттенки, прием оживки в росписи. Композиция на круге и квадрате. Осевая симметрия. Приемы изображения цветов, их узнаваемость (тюльпаны, лилия, купавка, мак, колокольчик, «розан», ромашка). Анализ геометрического состава композиций. Копирование и импровизация на тему «Букет». Подготовка подарков к празднику 8 Марта. (3 ч)

IV четверть (8 ч)

Тема 4. Компьютерная графика и живопись (8 ч)

Христианство на Руси. Православные храмы: назначение, архитектура, внутренний интерьер, росписи. Выполнение по фотографиям, иллюстрациям чертежей фасадов соборов и церквей Москвы и Подмосковья разных типов (шатровые, куполообразные). (2 ч)

Искусство лаковой миниатюры Палеха. Демонстрация образцов, фотографий, иллюстраций из книг. Технология производства. Судьба Палеха. Иконопись. Изобразительные традиции. Пейзаж — образ родной природы. Характерные особенности изображения деревьев, гор, воды. Новгородская и Строгановская школы письма (XX—XVII вв.). Копирование образцов.

Каменное зодчество. Рисунки палат. Сказочность изображения. Передача круглого объема. Композиция пейзажа с палатами (главное и частное). Импровизация пейзажа в стиле Палехской миниатюры (декоративность, тонкая прорисовка, условность форм).

Скопировать в Палехском стиле лошадь. Традиции Новгородского письма XV в. в прорисовке деталей. Отображения движения лошади.

Рисовать сказку — традиция палешан. Рисуем и мы: «Иван-царевич и Елена Прекрасная на сером волке». (6 ч)

Экскурсии в Кремль, Коломенское.

И. В. Марусева,
кандидат физико-математических наук, доцент, Санкт-Петербург

КОМПЬЮТЕР В ПОМОЩЬ ЛЮДЯМ С АНОМАЛЬНЫМ РАЗВИТИЕМ ЗРЕНИЯ И СЛУХА

В педагогической литературе мира признается, что изучение возможностей обучения с помощью компьютера детей с физическими недостатками относится к одной из актуальнейших тем (см., например, межвузовский сборник научных трудов: Дидактические основы компьютерного обучения. Л.: ЛГПИ, 1989, с. 16). Передача знаний (обязательный атрибут деятельности педагога) — уже сама по себе гуманный процесс. Отсюда, кстати, видится нам, некоторая гуманизационная окраска передовой тенденции в развитии информационной технологии — создание и использование в обучении систем, базирующихся на знаниях (СБЗ). Особенно важно, на наш взгляд, привитие будущим педагогам видения информационной картины мира с точки зрения усиления возможностей человечества в создании дополнительных способов передачи-получения информации для наших обездоленных в чем-то физически братьев.

Гуманизационный аспект информационной технологии в педагогике считаем, таким образом, одной из ее характерных отличительных черт и весьма интересным объектом для исследований. Представляется также полезным, в этом смысле, исследования психолого-педагогического характера о возможностях компьютера (этого социального феномена XX века) при вводе учащихся в мир компьютерной музыки, графики, рисунка. Так, наш ученик, например, не может иметь возможности посещать музыкальную школу (родители не имеют достаточных средств, нет музыкального слуха и т. д., а обучив его азам компьютерной нотной грамоты, мы откроем для него еще один вид информации (который может дать компьютер), сделаем мир вокруг него в чем-то радостней, ярче, интереснее.

Краткий обзор технических средств обучения специального применения

Одной из наибольших трудностей, которую слепые должны преодолеть во время обучения и работы, является ограниченный доступ к визуальной информации, особенно текстовой. Проблемы начинаются уже в начальной школе, где такое необходимое средство общения, как классная доска, оказывается для слепых бесполезным предметом.

Кроме доступа к информации, столь же важной проблемой для слепых является представление собственных текстов. Частично ее можно решить, овладев в совершенстве клавиатурой обычной пишущей машинки. Однако, работая таким образом, слепой лишен возможности читать написанный им текст и исправлять его.

Указанные трудности можно преодолеть, используя последние достижения техники, которые значительно расширяют область применения письма слепых. Изобретенное в 1824 г. 15-летним французом Луи Брайлем письмо стало великим благодеянием для инвалидов зрения. Основным элементом брайлевского шрифта является прямоугольная матрица из шести выпуклых точек, расположенных в трех строках и двух столбцах. Эта матрица называется для простоты шеститочием. В алфавите Брайля можно представить каждую букву, выбирая для нее соответствующую комбинацию точек в шеститочии. При этом необходимо подчеркнуть одну существенную черту письма слепых, а именно тот факт, что множество знаков брайлевского алфавита (включая пробел) насчитывает всего лишь 64 элемента. Поэтому многие графические знаки письма зрячих замещаются здесь последовательностями из двух и более символов. С некоторым приближением скорость чтения соответствует одной трети скорости чтения людей, пользующихся зрением.

Уже этого краткого объяснения наиболее существенных черт шрифта Брайля достаточно, чтобы читатель составил себе представление о его отличиях и понял, почему он почти неизвестен большей части общества.

За последние 20 лет был сделан ряд изобретений, которые хотя и не дают слепым возможности видеть, но решают ряд проблем в их жизни и профессиональной деятельности. Эти достижения стали возможны благодаря быстрому развитию электроники. Вначале появились переносные магнитофоны, что повлекло за собой возникновение новой формы изданий для слепых, какой является запись книг на кассетах. Небольшие магнитофоны служат слепым для записи лекций, а также для оперативной записи в форме высказываний собственных заметок.

В прошлом десятилетии появились читающие аппараты для слепых, а именно «Оптакон» (Optacon) и «Стереотонер» (Stereotoner).

Первый воспроизводит рисунок данной буквы при помощи вибрирующих стерженьков, которые ощущаются при прикосновении. «Стереотонер» же представляет данный графический знак в виде последовательности аккордов, каждый из которых — комбинация в рамках десяти тонов различной высоты.

Однако количество пользователей «Оптакона» составляет во всем мире не более 15000. Причиной является не только головокружительная цена (5000 долларов), но, пожалуй, в еще большей степени — высокие психофизические требования, предъявляемые к пользователю, а также относительно малая скорость чтения. Даже наиболее умелые пользователи аппарата редко читают более 70 слов в минуту.

«Стереотонер» также не нашел широкого применения, поскольку те, кто им пользовался, должны были иметь абсолютный музыкальный слух. Темп чтения был еще меньше, чем для «Оптакона» (не более 40 слов в минуту), поэтому спустя несколько лет о нем также забыли.

В конце семидесятых годов в США появилось устройство, исполняющее мечту всех слепых: машина, которая читала печатные материалы. Это оптический считыватель шрифта, снабженный синтезатором речи на английском языке. Он называется Kurzweil Talking Machine (KTM) и стоит в настоящее время 25 тысяч долларов. Высокая даже по американским меркам цена и значительные ограничения в отношении характера шрифта и формата автоматически читаемого текста обусловили то, что KTM используется на территории англоязычных стран только в учреждениях, действующих в пользу слепых.

Значительно большего успеха достигло другое изобретение, которое позволило плохо видящим людям читать большинство письменных материалов. Речь идет о так называемом телевизионном увеличителе. Принцип его действия основан на регистрации текста камерой, снабженной увеличивающим устройством с переменным фокусным расстоянием, и воспроизведении его на телевизионном экране. Поскольку приспособление увеличивает буквы в 10—20 раз, им могут пользоваться люди даже с очень плохим зрением. Количество пользователей телевизионного увеличителя во всем мире уже значительно превышает 100 тысяч человек.

Перечисленная техника полностью не решает проблем слепых людей, для которых книгохранилища обычных библиотек до сих пор остаются недоступными. Персональные компьютеры революционизировали уже множество областей деятельности человека, особенно те, которые связаны с обработкой, сбором и пересылкой информации. Многие слепые задаются вопросом, станет ли информация множества отраслей нашей жизни также и их уделом. Чтобы ответить на этот вопрос, подумаем, возможно ли в принципе общение слепого человека с компьютером. Для лишенных зрения людей необходимы устройства, которые обеспечивают двустороннюю передачу информации между человеком и компьютером, используя невизуальные средства. В одной из версий рабочее место слепого программиста снабжено двумя дополнительными приспособлениями. Первое из них можно, видимо, наиболее точно назвать брайлевским дисплеем, так как на нем, подобно обычному экранному дисплею, появляется вся текстовая информация. Она представлена в

виде брайлевских символов. Их можно читать при помощи прикосновения благодаря тому, что отдельные брайлевские точки формируются с помощью тонких закругленных на конце стерженьков, которые несколько выступают над поверхностью устройства. Для того чтобы брайлевский шрифт мог полностью передать все символы ASCII, необходимо было заменить шеститочие восьмиточием.

Такое восьмиточие часто называют брайлевским модулем. Таким образом, число всех брайлевских знаков стало равным количеству символов, которыми пользуются зрячие программисты, поскольку в модифицированном алфавите Брайля 256 элементов.

Стоит подчеркнуть тот факт, что информация, записываемая и считываемая специализированными приспособлениями для слепых, в памяти компьютера не представлена как код Брайля. Ее запись производится в коде ASCII, а в виде знаков Брайля она представлена только на вспомогательных периферийных устройствах. Здесь возникает полная аналогия со способом использования компьютера зрячими. Заметим, что текстовая информация, выводимая на экран компьютера, также хранится в памяти в виде кодов ASCII, а в удобочитаемый для человеческого глаза вид переводится только благодаря «электронике» дисплея. Описанные выше устройства дают возможность слепому создавать и воспринимать информацию, доступную каждому.

Таким образом, удалось решить главную проблему слепых, а именно, перевод брайлевского шрифта в буквы для зрячих и наоборот. Иными словами, если эти две разновидности знаков представлены в цифровом виде, то теоретически для слепых не существует ограничений.

Есть несколько фирм, выпускающих брайлевские дисплеи с разной длиной строки, т. е. с разным количеством модулей. Например, в американском устройстве Versabraille оно составляет 20. Но для слепых наиболее удобны устройства, способные представлять по меньшей мере одну строку экранного дисплея, т. е. 80 символов. Таким требованиям отвечают западногерманские устройства Brailloterm, Braille-Window и целое семейство дисплеев с общим названием Braillex. К сожалению, их цена превышает 7 тысяч долларов. Поэтому легко догадаться, почему не производятся дисплеи, которые могли бы представлять одновременно весь экран компьютера. Однако, чтобы слепой мог пользоваться условным экраном, брайлевские устройства снабжаются специальными указателями номера строки и положения курсора в строке. Сам курсор представлен в брайлевском виде как восьмиточие, т. е. прямоугольник, образованный восемью стерженьками. Например, Braille-Window, благодаря вмонтированному в компьютер специальному адаптеру, использует экранную память и может, кроме самого текста, показывать некоторые атрибуты. Правда, он не учитывает цвета. Серьезным ограничением известных брайлевских дисплеев является то, что они могут работать исключительно в текстовом режиме.

Другое приспособление, необходимое слепым для работы с компьютером, — это брайлевская клавиатура. Ее обслуживание весьма просто, так как для написания любого знака достаточно восьми клавиш.

На западном рынке существует более десятка разновидностей принтеров, которые под управлением компьютера печатают брайлевские точки на бумаге. Среди этих устройств наибольший интерес представляет норвежский принтер Braillo-400, который печатает выпуклые точки со скоростью 400 знаков в секунду, причем на обеих сторонах бумаги.

Еще одну возможность работы с компьютером предоставляет слепым использование синтезаторов речи. Устройство для генерирования синтетической речи, называемое обычно речевым выходом (англ. voice output), становится все дешевле и совершеннее. Например, речевой выход британской фирмы Dolphin Systems for the Disabled снабжен соответствующим текстовым редактором и может «рассказывать» даже о цвете, в котором представлена информация на экране компьютера. Его цена составляет 250 фунтов. Устройства с речевым выходом, благодаря своей низкой цене, более распространены среди слепых программистов, чем брайлевские дисплеи.

Подобные разработки ведутся во многих странах, в том числе и в России.

Итак, компьютеры могут быть доступны для слепых при условии использования специальных устройств, таких, как брайлевские дисплеи с клавиатурой или речевой выход, а также аппаратно-программных средств, дающих на экране увеличенные символы. Каждое из этих средств позволяет преодолеть перечисленные в начале трудности. Так, например, в качестве классной доски в школе для слепых могла бы выступать компьютерная сеть, где каждый терминал имел бы речевой выход и брайлевскую клавиатуру. Таким же образом, компьютер, снабженный брайлевским дисплеем или синтезатором речи, даст возможность слепым эффективно готовить документы в письменном виде, используя текстовый редактор. Распечатка этих текстов на принтере сделает их доступными для всех. Наконец, компьютерные брайлевские принтеры можно использовать для производства книг для слепых.

Распространяющийся все шире цифровой способ записи информации даст слепым новые шансы и сделает доступным печатное слово. Издание книг в системе Брайля или с увеличенной печатью по индивидуальному заказу слепых или плохо видящих читателей технически осуществимо уже сегодня, а широкое внедрение этих методов — это вопрос ближайших лет — для экономически развитых стран. В нашей стране, к сожалению, большие экономические трудности тормозят распространение передовых тенденций в этой области, отодвигают на значительный срок решение многих проблем людей с аномалиями в развитии. Надо иметь в виду, кроме того, что бедственное положение с экологией и прочие невзгоды современного бытия человечества ведут к тому, что людей с чистыми аномалиями зрения (слуха и пр.) практически нет (5%). В основном мы имеем дело с комплексными аномалиями, задевающими порой и центральную нервную систему.

В таблице представлена обобщенная авторами информация, почерпнутая из литературных источников, и предложен некоторый системный взгляд на рассматриваемую проблему.

Характерные черты	Первый этап	Второй этап	Третий этап
Основа способа получения и передачи информации	Ручной метод	Электромеханические и электронные приспособления	Общение с компьютером
Годы использования	1894 г. по наши дни	С начала 70-х годов нашего столетия	С 80-х годов до наших дней
Примеры способов и устройств. Краткая характеристика	Использование шеститочия Брайля 64 элемента алфавита Ск. чт. — 100 сл/мин	1. Магнитная лента 2. «Оптакон» (читающий аппарат). Цена — 5000 долларов Ск. чт. — 70 сл/мин Недостатки: высокая цена; высокие психофизические требования к пользователю 3. «Стереотонер» Ск. чт. — 40 сл/мин Недостатки: необходим музыкальный слух; невысокая скорость чтения 4. Читающая машина (КТМ) оптический считыватель шрифта с синтезатором речи. Цена — 25000 долларов Недостатки: цена; требования к шрифту и формату текста	1. Брайлевские дисплеи (БД). Модуль — восьмиточие. Алфавит — 256 символов. Сочетание ASCII и кода Брайля. Недостатки: цена; только текстовый режим 2. Брайлевская клавиатура. 8 клавиш 3. Принтеры (более 10 разновидностей). Пример: норвежский «Braillo-400». Ск. — 400 зн./сек 4. Синтезаторы речи («речевой выход») — генераторы синтетической речи. Пример: синтезаторы фирмы DSD. Цена — 250 фунтов
Устройства для слабовидящих		Телевизионный увеличитель. Увеличение в 10—20 раз	Аппаратно-программные средства, увеличивающие знаки на экране ПК

Использование азбуки Морзе

В учебной практике школ и интернатов страны для слепых детей используются, в основном, по-прежнему два типа специализированных «книг»:

а) «озвученная книга» (на основе записи учебной информации на магнитной ленте);

б) «брайлевская книга» (с нанесенными буквами Брайля на листах бумаги).

Проблема свободного доступа слепого человека к информации (и ее передаче) на сегодня еще не решена до конца. За рубежом уже созданы разнообразные типы классов со специальным оборудованием, различным в зависимости от аномального развития обучаемого. В нашей стране постоянные материальные проблемы оставляют многих людей с аномальным развитием за бортом обычной трудовой жизни.

Предлагаем дополнительный, чрезвычайно простой и дешевый способ передачи (приема) информации для подобных людей — через использование звуков азбуки Морзе (морзянки).

Для людей, лишенных зрения, мы можем передавать (принимать от них) информацию в закодированной в морзянке форме, например, при работе человека с компьютером. Информацию ему в таком случае не надо видеть, достаточно принимать ее на слух. Интересно изучить такой способ приема-передачи информации в сочетании с синтезаторами речи и другими техническими средствами.

В случае если человек имеет комплекс физических недостатков: лишен к тому же слуха, то морзянка может быть передана ему тактильным путем (например, при контакте его ладони с вибрирующим стерженьком, имитирующим своими колебаниями сочетания точек и тире).

При работе со слепыми учащимися, таким образом, код Морзе может использоваться и для поддержки при изучении физики (радиосвязи), и как еще одно средство коммуникации — дополнительно к двум вышеназванным видам книг для слепых. Мы можем обучить слепого человека навыкам радирования довольно просто. Технология передачи и приема информации для слепых радистов отрабатывалась в стране еще в 50—60-е годы. Слепые радисты обходятся без промежуточной записи услышанной закодированной информации на лист бумаги и показывают здесь высокие психофизические способности (превышающие возможности зрячих людей) (см. работы в этом направлении Л. В. Марковского).

Эта технология позволяет быстро принять и быстро передать информацию в отличие от медленной магнитной ленты.

На сегодня еще нельзя считать решенной проблему контакта слепого человека с ПК. Синтезаторы речи дороги, требуют расширения электронных возможностей ПК (см., например, разработки синтезаторов речи в Минске), а главное, они существуют зачастую только в лабораторных условиях. Малоразвитые же, упрощенные синтезаторы речи ограничены, как правило, очень небольшим словарным запасом.

Общение с ПК посредством кода Морзе дает дополнительный (очень дешевый, что так важно для нас) способ свободного (не ограниченного словарным запасом) информационного обмена. Подчеркнем, однако, что данный способ общения с ПК мы предлагаем не как альтернативу синтезаторам речи, а как дополнение к ним, как дополнительный способ общения незрячего человека с внешним миром или с персональным компьютером.

Проведенный анализ существующей проблемы позволил нам сделать выводы, что использование ПК для информационного обмена со слепым человеком должно вестись через некоторую адаптивную совокупность периферийных средств (ПС). В зависимости от психофизических особенностей человека (а это, как правило, целый комплекс отклонений в развитии человека), он может использовать тот или иной набор из данного перечня ПС. Сюда должны входить: синтезатор естественной речи, брайлевский принтер и клавиатура, брайлевский дисплей. Предлагаем к этому семейству добавить устройство, которое представляет собой специальный джойстик: ключ радиста (в традиционном исполнении или в виде одной-двух клавиш ПК — для передачи точек и тире).

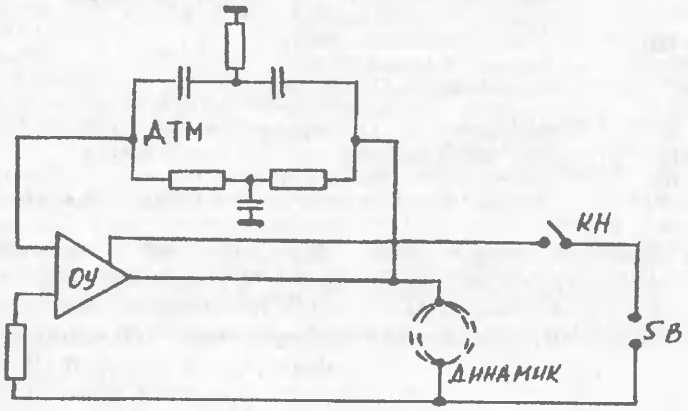
До работы с компьютером или при его

участии учащийся может овладеть грамотой радиста (изучить морзянку).

Разработан и собран макет этого несложного устройства, использованный в экспериментах. Кратко опишем его.

Для получения звуковых сигналов азбуки Морзе смонтирован звуковой генератор ЗГ на частоту 1 кГц (с допустимым выходом на динамик ЭВМ или наушник). ЗГ представляет собой операционный усилитель (ОУ) с положительной обратной связью через двойной т-образный мост (ДТМ), настроенный на частоту 1 кГц. Существует возможность перенастройки частоты. Возникновение и прерывание генерации и, соответственно, подача и прерывание звукового сигнала производится нажатием и отпусканием кнопки, замыкающей или разрывающей цепь питания ОУ. Питание схемы производится от блока питания микрокалькулятора, вырабатывающего постоянное напряжение 5 В. Блок питания включается в сеть 220 В. Схема размещена в пластмассовом корпусе.

Для слепого человека был разработан специальный редактор для работы со строкой. Редактор позволяет на слух легко вводить, выводить и корректировать текст. Ниже приведем краткую характеристику разработанного программного продукта.



Устройство звуковой передачи сигналов азбуки

Описание текстового редактора для незрячих «Script»

Текстовый редактор Script является примитивным средством ввода текстовой информации в IBM-совместимый ПК. Он написан на языке С (компилятор Borland C++ 3.0). Гарантируется его полная лицензионная чистота. Продукт тестировался на 386-м ПК под управлением DR DOS 6.0.

Редактор обладает возможностями по вводу текста, чтению его в виде морзянки, записи на диск и чтении с него. Представляемая версия редактора является самой первой. В нее не включены многие возможности, которые могут быть добавлены позднее.

Редактор также накладывает сильные ограничения на вводимый текст: текущая версия позволяет вводить только заглавные буквы русского и английского языков, цифры и некоторые специальные знаки (программист А. С. Яковлев).

Для ввода используется стандартный код Морзе, к которому добавлены некоторые символы. При добавлении символов использовались «незанятые» кодовые последовательности.

Текущая версия программы озвучивает символы русского языка на частоте 1200 Гц, а английского — на частоте 1000 Гц. Длительности точки и тире различаются ровно в три раза.

Для ввода используются клавиши 7, 8, 9 для русских символов и 4, 5, 6 — для символов английского языка. Общие для этих языков символы можно набирать, используя как «русские», так и «латинские» клавиши.

Клавиши 7 и 4 обозначают точку, 8 и 5 — тире, а клавиши 9 и 6 исполняют служебные функции знака конца кода и перехода в режим меню. Однако их использование необязательно: после истечения контрольного времени набранная последовательность точек и тире автоматически преобразуется в ASCII код.

В случае если пользователь нажал какую-либо клавишу, помимо указанных выше, редактор отзывается «недовольным» звуком низкого тона.

Двойное нажатие на клавиши 6 и 9 приводит к переходу в режим меню. В этом режиме доступны следующие операции, вызываемые набором кода одной буквы русского алфавита:

Печать (П) — вывод текста на принтер
Запись (З) — запись текста на диск
Чтение (Ч) — чтение ранее записанного текста
Имя (М) — присваивание текстовому файлу имени
Произнесение (А) — воспроизведение текущего абзаца азбукой Морзе
Произнесение (С) — воспроизведение текущей строки
Произнесение (Т) — воспроизведение всего текста
Удаление (Р) — удаление текущей строки из памяти строки
Удаление (Л) — удаление последнего слова в строке слова
Удаление (Б) — удаление всего текущего абзаца
Начало текста (Н) — переход в начало текста
Конец текста (Е) — переход в конец текста
Абзац вверх (В) — смещение на один абзац к началу текста
Абзац вниз (З) — смещение на один абзац к концу текста
Строка вверх (Й) — смещение на одну строку к началу текста
Строка вниз (Ц) — смещение на одну строку к концу текста
Конец работы (К) — завершение работы с редактором
Продолжение (О) — возврат в режим ввода текста редактирования

Эти команды обеспечивают примитивные возможности редактирования текста. Обогащение редактора возможностями — дело времени, вкуса и потребностей пользователей.

Выводы

Авторы выделили некоторые слагаемые гуманизационного аспекта в формировании мировоззрения будущего педагога современной развивающейся школы: 1) привитие понимания, осознание широких возможностей в использовании и передаче знаний на электронной основе и выработка соответственно этому эмоционально-ценностного отношения к ЭВМ; 2) формирование умений у будущего педагога раскрытия перед учениками богатства дополнительных видов информации, получаемых от компьютера — музыкальные звуки, графические изображения; 3) привитие будущим педагогам видения информационной картины мира, возможностей информационной технологии в разрезе проблемы передачи-получения информации для людей, физически обездоленных.

Предложен новый способ обмена информацией незрячих людей с компьютером — «морзянка». Способ имеет следующие достоинства: дешевизна реализации, возможность двусторонней передачи информации (к ЭВМ и от ЭВМ), неограниченность тезаурусов, возможность озвучивания и упрощение поиска и исправления ошибок.

Описанный способ информационного обмена с людьми, имеющими аномалии в развитии, может быть использован самостоятельно или в совокупности с разрабатываемой специальной периферией компьютера. Разработан текстовый редактор, облегчающий процесс ввода, отладки и поиска нужной информации в данном режиме работы с компьютером. Очевидно, что предложенный вариант информационного обмена с людьми, имеющими аномалии в развитии, может иметь и безмашинный вариант реализации.

Использование морзянки как средства обмена информацией в системе человек-компьютер одобрено преподавателями кафедры тифлопедагогики РГПУ им. Герцена (г. Санкт-Петербург). Проявлен также интерес к нему со стороны Санкт-Петербургского благотворительного фонда для слепых и слабовидящих детей.

Д. Ю. Усенков,
Москва

РЕКУРСИВНЫЙ ГЕНЕРАТОР ПЕРЕСТАНОВОК

Есть в математике такой раздел — комбинаторика, наука о перестановках. Круг ее интересов составляет определение количества возможных комбинаций расстановки различных предметов. Как правило, при этом ставится вопрос вычисления количества возможных перестановок. Но существуют задачи, в которых требуется сгенерировать все возможные варианты, например для трех цифр «1», «2» и «3»: «123», «132», «213», «231», «321» и «312». В последнем случае это нетрудно было сделать вручную, однако для большого числа элементов возникают затруднения: согласно одной из формул комбинаторики, количество перестановок для n предметов равно n! (здесь восклицательный знак обозначает факториал). И если для трех цифр существует всего шесть вариантов, то для четырех их будет 24, для пяти — уже 120 и т. д.

Можно попытаться автоматизировать генерацию перестановок, заставив компьютер переставлять символы в массиве — строке и распечатывать получаемые варианты. Требуемый алгоритм реализован на языке СИ в виде рекурсивной подпрограммы. В листинге также приведен пример ее вызова — простейшая main-программа, задающая символьный массив (строку) из n элементов и осуществляющая первоначальный вызов подпрограммы генерации перестановок. (В данном примере n выбрано равным 3, чтобы результат работы программы можно было сравнить с ранее полученным вручную.)

«Родоначальником» этого довольно хитрого алгоритма перестановок стал алгоритм «пузырьковой сортировки», очень сильно измененный. Суть его сводится к формированию цепи перестановок, каждый шаг которой организуется путем очередного рекурсивного вызова подпрограммы funct:

1) меняем местами первый символ со вторым, третьим и т. д.;

2) в оставшейся части массива (начиная со второго символа) меняем второй символ с третьим, четвертым и т. д.;

n-1) когда остаются необработанными только предпоследний и последний символы, распечатываем исходный вариант, меняем местами эти два символа и распечатываем полученный вариант.

```
/* ПРОБНЫЙ ГЕНЕРАТОР ВАРИАЦИЙ ПЕРЕСТАНОВОК С ПОМОЩЬЮ РЕКУРСИИ */

void funct(int i, int n, char *a);
#include  <stdio.h>
#include  <conio.h>

main ()
{ char a[3]={ 'A', 'B', 'C'};
  funct(0,3,a);   /* с первого символа, всего 3 символа, массив a */
}

/* РЕКУРСИВНАЯ ПОДПРОГРАММА ВЫДАЧИ ВАРИАНТОВ */

void funct (int i, int n, char *a)
{ int j;
  char ch;

  for (j=i; j<n ; j++)
   { if (i!=j)
      { ch=a[i];   /* перестановка символов местами */
        a[i]=a[j];
        a[j]=ch;
      }
     if (j!=(n-1)) funct(i+1,n,a);
        else printf("\n%s",a);
       /* если не последний проход цикла, то рекурсивный вызов,
       иначе просто печатать слово */
     if (i!=j)
      { ch=a[i];   /* обратная перестановка */
        a[i]=a[j];
        a[j]=ch;
      }
   }
  return;
}
```

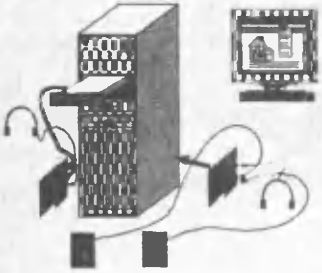
Республиканский Центр Интерактивных Средств Обучения
Телефоны: (095) 155 8737, 155 8747 Факс: (095) 155 8727

МУЛЬТИМЕДИА

При помощи средств мультимедиа Ваш компьютер сможет говорить, записывать Ваш голос, воспроизводить музыкальные произведения, имитировать различные музыкальные инструменты, показывать видеофрагменты, изображения высокого качества и многое другое. Две трети выпускаемых в США программ мультимедиа на CD-ROM используются в системе образования!

В ногу со временем!

Республиканский Центр Интерактивных Средств Обучения (РЦИСО)



поможет превратить

- Ваш IBM совместимый PC в Мультимедиа PC (MPC)
- Ваш IBM совместимый учебный класс в мультимедийный класс

поставит "под ключ"

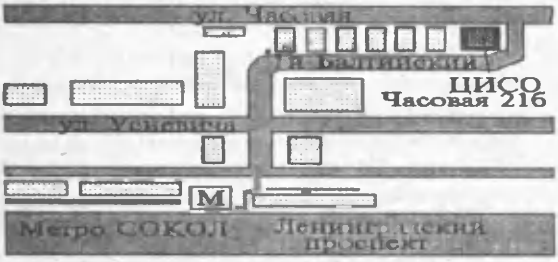
- мультимедийный PC (MPC)
- мультимедийный класс

предоставит обширный выбор программ:

- курсы иностранных языков
- энциклопедии и справочники
- новейшие программы shareware
- компьютерные игры

и многое другое.

Всего около 900 наименований. Имеется каталог с аннотациями на русском языке.



Добро пожаловать в РЦИСО!

Наш адрес: 125315, г. Москва, ул. Часовая, 21б

Проезд: станция метро "Сокол", выход к улице Усиевича, далее по схеме

Е. В. Утлинский, Э. Г. Аввакумов,
Институт общеобразовательной школы Российской академии образования, лаборатория информатики

ТЕЛЕКОММУНИКАЦИОННАЯ СЕТЬ TIT-BIT*

Выбор телекоммуникационной программы

После подключения модема к компьютеру необходимо установить и настроить программу для управления модемом. Существует множество телекоммуникационных программ, а также интегрированных пакетов, имеющих коммуникационные функции (Qmodem, MTEZ, Telix, Front Door, Frame Work и т. д.). Практически любую коммуникационную программу вы можете использовать для управления модемом. Основное требование к программе: она должна быть способной использовать современные протоколы пересылки файлов (наиболее популярный и эффективный среди них — Zmodem). Если ваш модем не имеет аппаратной коррекции ошибок, желательно использовать программу, эмулирующую режим коррекции (например, MTEZ).

В данной статье мы рассмотрим работу с телекоммуникационной программой на примере программы TIT-BIT. Ее организация характерна для большинства коммуникационных программ, но TIT-BIT к тому же имеет полный русскоязычный интерфейс.

Установка программы

В комплект поставки программы входят файлы:

install.bat — файл установки программы
tit-bit.exe — исполняемый файл программы
tit-bit.bat — файл запуска сценария автоматической работы

Обычно программа устанавливается на жесткий диск компьютера. Для установки программного обеспечения (ПО) TIT-BIT на диск C компьютера дайте команду

---

* Данная статья продолжает серию публикаций, посвященную телекоммуникационным сетям (см.: Информатика и образование. № 4. 1994). — *Примеч. ред.*

install.bat C:, для установки на диск D — команду **install.bat D:**. Файл установки программы выполнит следующие действия:

- создаст на указанном жестком диске каталог TIT-BIT;
- внутри каталога TIT-BIT создаст каталоги RECEIVE и SEND;
- поместит в каталог TIT-BIT файлы tit-bit.exe и tit-bit.bat.

После первого запуска исполняемого файла tit-bit.exe внутри каталога TIT-BIT будут созданы еще некоторые файлы, и установленное ПО TIT-BIT примет следующий вид:

RECEIVE — каталог приема информации
SEND — каталог отправки информации
tit-bit.exe — исполняемый файл программы
tit-bit.bat — файл запуска сценария автоматической работы
tit-bit.log — файл ведения журнала автоматической работы
tit-bit.cnf — файл настройки программы
tit-bit.fon — файл телефонной книжки
tit-bit.scr — файл сохранения экрана

Режим работы «Терминал»

Для запуска ПО TIT-BIT включите модем (последовательность включения модема и компьютера несущественна) и запустите **tit-bit.exe.** Как и любая современная телекоммуникационная программа, ПО TIT-BIT имеет три основных режима работы: «Терминал», «Настройка» и «Телефонная книжка». В каком бы режиме вы ни находились, в нижней подсвеченной строке экрана всегда выводится список разрешенных в данный момент клавиш или краткое пояснение. Находясь в режиме «Терминал» можно завершить работу с программой, нажав клавишу <F10>.

Когда ваш модем не находится на связи с удаленным модемом, режим «Терминал» служит для управления состоянием модема с помощью набора Hayes-команд (AT-команд). После установки связи с удаленным модемом данный режим используется для обмена информацией, и вводимые символы интерпретируются модемом уже как данные, а не как команды.

Обычному пользователю нет необходимости использовать режим «Терминал» для ввода команд, но некоторое представление о наборе команд модема иметь следует.

Почти каждая команда, которую способен понять Hayes-совместимый модем, должна начинаться с префикса **AT** (attention — внимание), написанного прописными или строчными буквами, и завершаться нажатием клавиши <ENTER>.

Простейшая команда, которую вы можете дать модему для проверки его готовности, — это просто **AT** или пустая команда. Если модем исправен и правильно подключен к компьютеру, вы увидите на экране ответ модема **OK** или **0**. Форма ответа (словесная или числовая) зависит от настройки модема. Для работы большинства коммуникационных программ необходимо, чтобы была установлена словесная форма ответа модема. В качестве примера отметим, что изменить форму ответа модема можно с помощью команд:

ATV1 — установить словесную форму ответа
ATV0 — установить числовую форму ответа

С помощью AT-команд можно управлять звуком встроенного в модем динамика:

ATM0 — звук всегда выключен
ATM1 — звук включен до установки соединения

ATM2 — звук всегда включен
ATM3 — звук включается после набора номера и выключается после установки соединения

Заставить модем набрать телефонный номер 924-7864 можно с помощью команды **ATDP924-7864**. Здесь команда DP означает импульсный набор номера (используется подавляющим большинством отечественных АТС). Если же вы подключены к АТС с тональным набором номера, необходимо использовать команду **DT**. При записи телефонного номера можно использовать знаки «-» и пробелы.

Как известно, чтобы сделать междугородний звонок, надо набрать цифру 8, дождаться длинного гудка, затем набрать код города и телефонный номер. Для ожидания длинного гудка используется разделитель W. Так, если вы звоните из какого-либо города в Москву (код 095) по телефону 924-7864, команда дозвона будет выглядеть следующим образом:

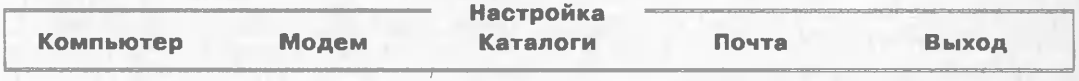
ATDP8 W 095 924-7864.

Кроме того, модемы имеют набор регистров (S0...S95), в каждом из которых может храниться 1 байт информации. Например, в регистре S7 содержится значение времени ожидания ответа удаленного модема (в секундах) после набора номера. Вывести на экран значение регистра S7 можно с помощью команды **ATS7?**. Заводская установка этого регистра — 30 с. Обычно при дозвоне по междугородней линии связи 30 с на установку связи недостаточно. Изменить значение этого регистра на 60 с можно с помощью команды **ATS7=60**.

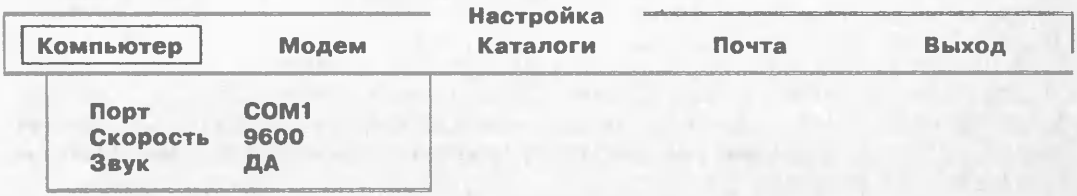
Большинство современных модемов имеют энергонезависимую память (NOVRAM), т. е. они способны хранить текущую конфигурацию модема даже при отключении питания. Произвести запись текущей конфигурации модема в энергонезависимую память можно с помощью команды **AT&W**, загрузить содержимое энергонезависимой памяти в качестве текущей конфигурации модема можно с помощью команды **ATZ**. Восстановить заводскую конфигурацию модема можно с помощью команды **AT&F**. Полное описание набора АТ-команд приведено в документации к каждому конкретному модему.

Режим работы «Настройка»

Данный режим предназначен для установки параметров, необходимых для правильной работы модема и телекоммуникационной программы. Перейти из режима «Терминал» в режим «Настройка» можно, нажав <F3>, обратно — с помощью клавиши <ESC>. При входе в режим «Настройка» вы увидите горизонтальное меню:



Вы можете перемещать курсор по пунктам этого меню с помощью клавиш <ВПРАВО>, <ВЛЕВО> и открывать подменю с помощью клавиши <ENTER>. Выберите пункт «Компьютер» и откройте подменю:



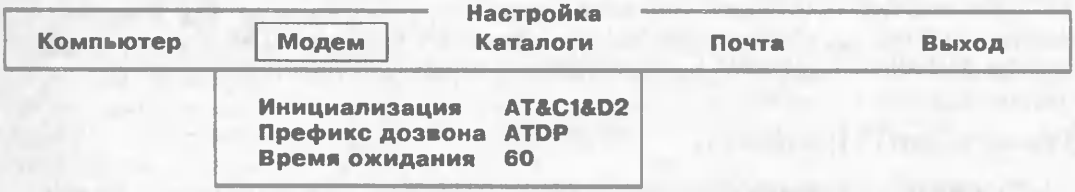
Вы можете перемещать курсор по пунктам этого подменю с помощью клавиш <ВВЕРХ>, <ВНИЗ> и входить в режим редактирования с помощью клавиши <ENTER>.

В пункте «Порт» следует установить номер последовательного порта компьютера, к которому подключен ваш модем (от COM1 до COM4). При верной установке номера порта, после выхода из режима «Настройка» в режим «Терминал» в ответ на команду **AT** модем должен возвращать **OK**.

В пункте «Скорость» устанавливается скорость выбранного последовательного порта для обмена данными между модемом и компьютером. Возможные значения скорости порта (бит/с): 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200. Для современных модемов, использующих протокол сжатия данных V.42bis (теоретический коэффициент сжатия 4:1), рекомендуется устанавливать скорость последовательного порта компьютера в 4 раза большей, чем скорость самого модема. Это связано с тем, что, приняв 1 блок информации от удаленного модема, после декомпрессии модем передает в компьютер 4 блока информации. Так, для модема, работающего на скорости 2400 бит/с, рекомендуется устанавливать скорость последовательного порта компьютера 9600 бит/с, для модема 14400 бит/с — скорость порта 57600 бит/с. При приемлемой установке скорости порта после выхода из режима «Настройка» в режим «Терминал» в ответ на команду AT модем должен возвращать OK, в противном случае скорость последовательного порта следует понизить.

Пункт «Звук» позволяет включить или выключить звуковое сопровождение программы.

Теперь рассмотрим пункт горизонтального меню «Модем» и соответствующее ему подменю:

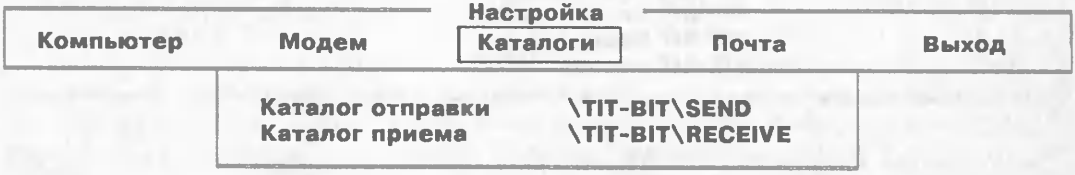


В пункте «Инициализация» записывается строка инициализации модема, содержащая AT-команды, которая посылается в модем при загрузке ПО TIT-BIT. Для большинства модемов подходит инициализационная строка **AT&C1&D2**. Если вы звоните по междугородней линии, можно в инициализационной строке задать значение регистра S7, например **AT&C1&D2S7=60**. Некоторые коммуникационные программы могут требовать, чтобы строка инициализации модема завершалась специальной последовательностью символов (например, ^M).

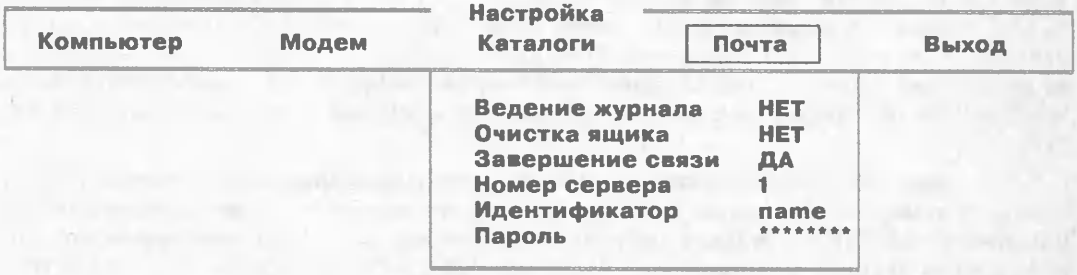
«Префикс дозвона» определяет команду, которая должна использоваться модемом для набора телефонного номера. Обычно используется префикс **ATDP** для импульсного набора номера (принятого в России). Если вы подключены к АТС с тональным набором номера, используйте префикс **ATDT**.

В пункте «Время ожидания» задается промежуток времени, в течение которого программа ожидает ответ модема об установке связи (не путать со временем, хранящимся в регистре S7 модема, в течение которого модем ожидает ответа удаленного модема). Данный параметр программы рекомендуется устанавливать не меньшим, чем значение, содержащееся в регистре S7 модема.

Пункт горизонтального меню «Каталоги» предназначен для установки каталогов отправки и приема информации. Файлы для отправки необходимо помещать в каталог SEND, принимаемые файлы будут помещаться в каталог RECEIVE:

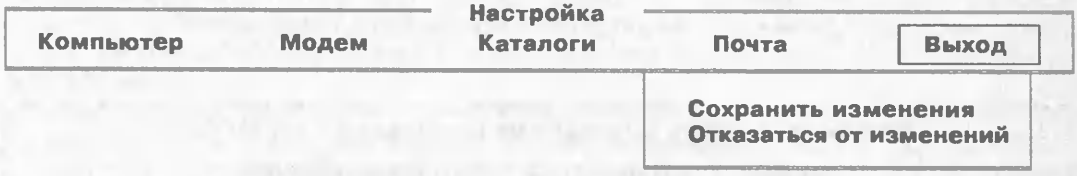


Пункт горизонтального меню «Почта» предназначен для установки параметров автоматического обмена электронной почтой с сервером сети TIT-BIT:



Данное подменю имеет значение только для зарегистрированных абонентов сети TIT-BIT, так как только они могут производить обмен почтой с серверами сети. Вопросы использования электронной почты будут рассмотрены в следующей статье, сейчас же отметим, что зарегистрированный абонент может занести в данное подменю порядковый номер сервера из телефонной книжки своего ПО TIT-BIT, до которого он будет дозваниваться, свой идентификатор и пароль входа на сервер (при вводе пароля вы увидите на экране лишь знаки «*»).

И наконец, последний пункт горизонтального меню «Выход» позволяет вернуться в режим «Терминал» с сохранением или без сохранения произведенных изменений:



Режим работы «Телефонная книжка»

Данный режим предназначен для занесения в программу названий телекоммуникационных серверов и их телефонных номеров, а также для осуществления дозвона до одного или нескольких выбранных серверов. Перейти из режима «Терминал» в режим «Телефонная книжка» можно, нажав <F2>, обратно — с помощью клавиши <ESC>. При входе в данный режим вы увидите на экране телефонную книжку (на рисунке приведена только часть книжки):

Телефонная книжка

N	Название сервера	Телефонный номер
1	TIT-BIT Москва	8 w 095 924-78-64
2	TIT-BIT Ярославль	8 w 0852 32-11-83
3	TIT-BIT Архангельск	8 w 8182 49-06-26
4	TIT-BIT Переславль	
5	TIT-BIT сервер «ЭКСИ»	

Телефонная книжка ПО TIT-BIT позволяет хранить до 15 названий и телефонных номеров серверов. Вы можете перемещать курсор по книжке с помощью клавиш <ВВЕРХ>, <ВНИЗ>.

С помощью клавиши <F4> можно войти в режим редактирования текущей строки. Находясь в режиме редактирования, вы можете сменить поле с помощью клавиши <TAB>, завершить редактирование, нажав <ENTER>, или отказаться от произведенных изменений, нажав <ESC>. Например, если вы находитесь в Москве и хотите позвонить на московский сервер, то вам следует отредактировать первую строку книжки, убрав из телефонного номера фрагмент «8 w 095», обеспечивающий выход на междугородную связь.

Допустим, вы решили позвонить на телекоммуникационный сервер, указанный первым в телефонной книжке. Для этого установите курсор на первую строку книжки и нажмите <ENTER>. У модема должен загореться индикатор ON, показывающий, что модем «снял трубку» и занял линию. Во время набора телефонного номера индикатор ON будет мигать. В нижней части экрана появится строка «Время дозвона», в которой вы сможете видеть оставшееся время ожидания установки связи с удаленным модемом. Если время дозвона истекло, а связь не была установлена, программа заставляет модем набирать номер еще и еще раз до тех пор, пока связь не будет установлена или пока пользователь не прервет дозвон, нажав <ESC>. В случае неудачных попыток дозвона программа выводит на экран сообщения «ЗАНЯТО», «НЕТ ГУДКА», «НЕТ ОТВЕТА». Инициировать повторный набор номера, не дожидаясь истечения времени дозвона, можно с помощью клавиши <ENTER>.

При установке связи с удаленным модемом загорается индикатор CD, на экран выводится сообщение «СОЕДИНЕНИЕ», и программа автоматически переключается в режим «Терминал» для обмена данными. В случае установки некачественной связи (когда на экран начинает выводиться всякий «мусор») следует прервать связь, нажав <F9>, войти в режим «Телефонная книжка» с помощью клавиши <F2> и повторить дозвон. Если же установлена нормальная связь, вы увидите на экране заставку сервера:

Добро пожаловать в телекоммуникационую сеть TIT-BIT!

Модем: (095) 924-7864 19200/V.42bis/MNP5
станция работает круглосуточно

Операторская служба: (095) 923-01-61

электронная почта
телеконференции
новости образования, науки, бизнеса
реклама Ваших разработок

Гостевой идентификатор: TIT-BIT
Подсказка по команде:HELP

Поздравляем вас! Вам удалось войти в телекоммуникационный сеанс, а это означает, что вы разобрались во многих вещах: как выбрать модем, как подключить модем к компьютеру, как выбрать, настроить и использовать коммуникационную программу.

Используя ПО TIT-BIT во время телекоммуникационного сеанса, вы всегда можете посмотреть строки, «ушедшие вверх экрана», с помощью клавиши <ВВЕРХ>, а также включить режим записи всей выводимой на экран информации в файл **tit-bit.scr** с помощью клавиши <F5>.

В следующей статье мы рассмотрим, как работать с серверами телекоммуникационных сетей, как производить пересылку файлов и что такое электронная почта. Но если вы опытный пользователь или воодушевлены первым успехом, вводите гостевой идентификатор TIT-BIT и входите на сервер. Получить подсказку по работе с сервером вы сможете по команде HELP. Напоминаем, что в телефонной книжке указаны действительные номера серверов сети TIT-BIT (см. выше).

ПРИЛОЖЕНИЕ

Как стать абонентом телекоммуникационной сети

Для подключения к какой-либо сети вы должны иметь следующее:

- IBM-совместимый компьютер с последовательным портом передачи данных;
- Hayes-совместимый модем;
- телефонную линию.

После этого попробуйте узнать, есть ли у вас в городе или области региональный узел какой-либо телекоммуникационной сети. Позвонив оператору узла, вы сможете узнать условия подключения к сети. После заключения договора на обслуживание и оплаты информационных услуг вы получите электронный адрес, и оператор узла поможет вам подобрать необходимое для работы программное обеспечение.

Как организовать региональный сервер сети

Для организации регионального сервера какой-либо сети прежде всего необходимо связаться с администрацией интересующей вас сети и узнать, какие технические средства и программное обеспечение необходимо иметь и каковы условия вхождения в сеть в качестве регионального сервера.

Как правило, для организации сервера необходимо иметь:

- IBM-совместимый компьютер не ниже AT 386DX. Количество последовательных портов компьютера определяет, сколько абонентов смогут одновременно работать с сервером;
- Hayes-совместимые модемы. Модемы должны соответствовать современным стандартам V.42bis и MNP5. На сервере обычно используются модемы со скоростью передачи данных не медленнее 14400 бит/с. Количество модемов определяет, сколько абонентов смогут одновременно работать с сервером;
- каналы связи для связи вашего сервера с вышестоящим сервером сети и связи абонентов с вашим сервером. Как правило, для связи абонентов с сервером используются обычные (коммутируемые) телефонные линии, для связи регионального сервера с вышестоящим сервером сети используются обычные телефонные линии, выделенные телефонные линии или каналы телефонной сети повышенного качества «Искра». Количество телефонных линий определяет, сколько абонентов смогут одновременно работать с сервером.

Б. И. Глазов,
заслуженный деятель науки и техники Российской Федерации, доктор технических наук, профессор Академии им. Ф. Э. Дзержинского, академик Международной академии информатизации
Д. А. Ловцов,
кандидат технических наук, доцент Академии им. Ф. Э. Дзержинского
С. Н. Михайлов,
кандидат технических наук, преподаватель Академии им. Ф. Э. Дзержинского
А. В. Сухов,
кандидат технических наук, преподаватель Академии им. Ф. Э. Дзержинского

КОМПЬЮТЕРИЗИРОВАННЫЙ УЧЕБНИК

Компьютеризация обучения и образовательного процесса, разработка и внедрение автоматизированных систем интенсивного обучения [1...5] включает, в частности, создание компьютеризированных пособий и учебников (КПУ). Под КПУ будем понимать экспертную информационную систему — структуризованную совокупность упорядоченных знаний и данных, обеспечивающую внедрение новой информационно-педагогической технологии решения дидактических задач обучения (самообучения) [6].

1. Назначение компьютеризированного учебника

1. Полное использование в учебном процессе технических возможностей современных средств электронно-вычислительной техники, таких, как:

- машинная имитация реальных сложных (многоразмерных) материальных объектов (систем; физических, химических и других процессов) с образно-художественным представлением (визуализацией) на табло и экранах дисплеев анимационных (динамических) результатов имитационного моделирования и гипертекстов в реальном времени обучения;
- реализация игровых форм и методов обучения (деловых, исследовательских, проектных, имитационных, контрольно-тестирующих, коммерческих и других игр);
- управление процессом обучения в диалоговом человеко-машинном режиме и пр.

2. Автоматизация и интенсификация педагогического труда при

- проектировании систем обучения;
- разработке учебных дисциплин;
- подготовке к занятиям и отборе учебного материала в соответствии с поставленной дидактической задачей обучения [1, 4];
- формировании дидактических материалов (видеограмм, кадропланов, опорных конспектов, раздаточного материала и др.).

3. Обеспечение возможности самостоятельного тиражирования (полного и выборочного) и коллективного независимого совершенствования имеющейся авторской версии КПУ путем экспериментального (на ЭВМ) анализа и проверки на практике адекватности представленных соавторами КПУ аналитических и имитационных моделей, их доработки, создания (включения в КПУ) новых и их имитационного исследования.

4. Индивидуально-целевое (преподавателями, обучающимися, специалистами и др.) глубокое освоение современных средств автоматизации, вычислительной и оргтехники.

2. Концепция компьютеризированного учебника

Вопросы разработки концепций построения компьютеризированных учебников вызывают острую полемику [7,8], но вместе с тем стороны, выступающие в этом споре «за» и «против» компьютеризации обучения, единодушны в одном — как элемент новизны и прогресса различные кон-

цепции построения компьютеризированных учебников имеют право на жизнь.

Этап развития, переживаемый в настоящее время системой высшего образования, сравнивают с возникновением книгопечатания. Оно в свое время привело к открытию Яном Коменским классно-урочной системы обучения. Сегодня вычислительная техника призвана совершить революционное изменение в технологии обучения [2, 9]. Приход в систему образования компьютера стал своего рода рубиконом, отделяющим эпоху традиционного обучения от эры компьютеризированного образования.

В связи с этим встает вопрос: в какой мере внедрение новых технических средств в учебный процесс потребует пересмотра старых методов преподавания и общения с обучаемыми? Нужно отметить, что безосновательны как чрезмерные упования на «новую информационно-педагогическую технологию», так и скепсис, сопровождающий ее приход. Можно утверждать, что эта технология не уменьшает требований к преподаванию базисных наук, а повышает их. Причем особенно важной становится подготовка преподавателей к овладению средствами современной информационно-педагогической технологии [9].

Базовым элементом новой информационно-педагогической технологии является компьютеризированный учебник. Так как материал, излагаемый в КПУ, должен быть систематизированным изложением знаний для формирования умений и навыков из предметной области, то КПУ должен представлять собой экспертную информационную систему, базирующуюся на знаниях (ЭИСБЗ), которая, как известно, включают три основных компонента: базу данных и знаний (БДЗ), подсистему принятия решения, пользовательский интерфейс [6].

Под БДЗ понимают интегрированную информационно-программную систему. Подсистема принятия решений в ней предназначена для извлечения из БДЗ ответов на вопросы и получения решения задач, сформулированных в терминах БДЗ. Пользовательский интерфейс обеспечивает работу БДЗ и подсистемы принятия решения на языке, приближенном к профессиональному языку специалистов прикладной области, для которой создан КПУ.

Таким образом, можно утверждать, что **КПУ следует представлять в виде ЭИСБЗ, обеспечивающей использование знаний и техники рассуждений человека-эксперта в процессе обучения, реализуемого на основе новых информационных технологий при использовании компьютеров без посредничества профессиональных программистов в целях освоения знаний и развития умений в прикладной области.**

Для повышения эффективности усвоения учебного материала в КПУ следует учитывать психофизиологические возможности человека-пользователя по приему и переработке воспринимаемой информации, в частности характеристики зрительного анализатора человека, путем дифференцирования цветового фона для различных дидактических материалов (элементов) и увязки объема отображаемой «дозы» информации с интенсивностью световой гаммы.

3. Общая характеристика компьютеризированного учебника

Представим общую характеристику КПУ особенностями его содержания и формы (структуры) [1, 2].

Кратким содержанием КПУ является оглавление, составляющее «меню» ЭИСБЗ в виде структурного концептуального фрейма-прототипа [1, 10].

Совокупность оглавления, основных понятий и описания внешних связей КПУ образуют его ЛОС — лист основного содержания [3, 4, 5], представляющий модель предметной области учебника. Совокупность основных понятий и связей между ними удобно представить в виде предметного тезауруса.

По способу представления материала в памяти компьютера КПУ отличается от традиционных — в память ЭВМ может быть заложен как полный текст учебника, так и только минимальное его описание — фрейм учебника, его опорный конспект, предназначенный для использования преподавателем или обучающимся. *По способу использования* КПУ кроме традиционного (книжного) выделяется алгоритмический

способ, предлагающий использование сценария диалога «КПУ — пользователь», построенного на основе пользовательского интерфейса. *По способу совершенствования (развития) материала* КПУ, кроме традиционного авторского совершенствования, выделим пользовательское совершенствование, когда все предложения и замечания пользователей запоминаются компьютером для дальнейшего анализа автором учебника и включения в последующие версии с указанием соавторов. Такая возможность особенно ценна в отношении совершенствования алгоритмической части знаний (комплекса прикладных программ) КПУ.

На практике КПУ сначала «проектируется и рисуется на бумаге» [1], и его содержание в значительной степени зависит от знаний, умений и навыков разработчиков. При разработке КПУ следует так организовать этапы этого процесса, чтобы можно было параллельно созданию практически использовать создаваемую ЭИСБЗ как развивающуюся информационную систему. При этом в качестве первого этапа представляется целесообразным создать тезаурус предметной области с библиографией и включением в него неалгоритмических (декларативных) знаний, взятых из текста учебника, и процедурных знаний, носителями которых являются эксперты, в том числе преподаватели. Тезаурус КПУ необходимо дополнить ЛОСом, в виде последовательности «доз» обучения.

Содержание КПУ можно представить в виде концептуального фрейма минимального описания или в виде главного меню. Первым приближением обеих форм представления может рассматриваться оглавление КПУ.

В качестве примера рассмотрим оглавление КПУ по учебной дисциплине «Системная информатика с приложением в космонавтике». Выделим в системной информатике [11] (информатике, построенной на принципах системологии) два основных направления — общесистемное и прикладное. К общесистемному отнесем методологические основы и информационные системы общего назначения. К прикладному — космические информационно-кибернетические системы. В соответствии со сказанным КПУ по системной информатике включает три части:

- системологические основы информатики;
- информационные системы общего назначения;
- космическую системную информатику.

Каждая часть состоит из разделов, разделы — из глав (тем), темы — из параграфов и подпараграфов (учебных вопросов).

Структуру КПУ удобно представить

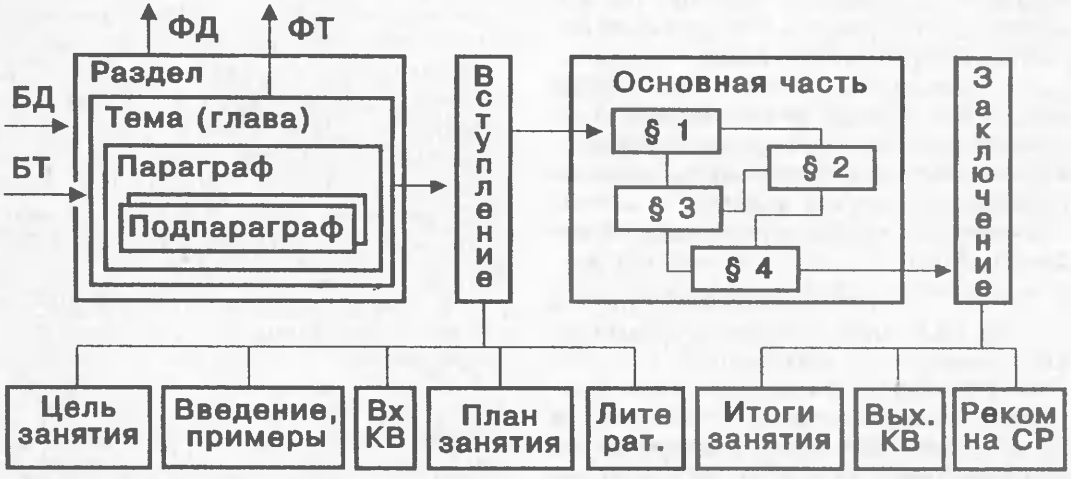


Рис. 1. Внутренняя организация учебника

описанием его внешних связей [12] и внутренней организации [2].

Внешние связи могут быть представлены указанием базовых дисциплин (БД) и тем (БТ), на основе которых строятся данная учебная дисциплина и формируемые дисциплины (ФД) и темы (ФТ), обеспечиваемые учебной дисциплиной (учебником). При этом первые образуют входы, а вторые — выходы учебника и следуют из его целевого назначения. Например, для КПУ по системной информатике БД являются высшая математика, теория информации, физика, теория систем и др., а ФД — вычислительные системы, автоматизированные системы управления, радиотехнические системы и др.

Внутренняя организация КПУ предполагает деление его на составные части: разделы, темы (главы), параграфы и подпараграфы. При этом каждую отдельную часть можно представить инвариантной структурой (рис. 1). Применительно к теме (главе) — между параграфами образуются характерные связи, на которых создается структура; во вступлении формулируются цель темы и входные контрольные вопросы (КВ), приводятся практические примеры, называются рассматриваемые вопросы и перечисляется литература по теме; в заключении подводятся итоги темы, формируются выходные контрольные вопросы и даются рекомендации по организации самостоятельной работы (СР).

4. Фрейм-сценарий постановки и решения дидактических задач

Для описания и реализации диалоговых процедур использования КПУ требуется наличие в памяти компьютера логико-лингвистической модели (ЛЛМ) процесса переработки качественной (лингвистической) информации о дидактических ситуационных особенностях решения конкретных задач обучения (самообучения). Для построения такой ЛЛМ наиболее приемлемы фреймовые описания [10], поскольку фрейм представляет собой модульную структуру знаний и данных со всей совокупностью включенных в нее процедур для распознавания стереотипных ситуаций (например, ситуаций обучения); язык фреймов относится к открытой системе, в которой можно добавлять новые компоненты, исключать или заменять старые (малоэффективные на практике) без выключения ЛЛМ из эксплуатации; для соединения фреймов используется естественный язык. Целесообразно использование фреймовых описаний (моделей) для диалоговой оперативной постановки конкретной дидактической задачи обучения (ДЗО), в частности для отбора учебного материала из КПУ и определения последовательности его изучения.

Фрейм-сценарий обучения представляется совокупностью [1]:

- концептуального фрейма-прототипа «Маршрут сценария обучения» в виде иерархического графа «И/ИЛИ», содержащего слоты для задания определенных элементов (цели, задачи, этапы, условия, ...) ДЗО;
- структурно-логической схемы компьютерного ЛОС, содержащей соответствующие возможные маршруты изучения учебного материала (последовательности видеограмм ЛОС с кратким описанием действий пользователя КУ).

Диалоговая человеко-машинная активизация фрейма-прототипа постановки конкретной ДЗО позволяет сформировать соответствующий терминальный фрейм, последующее машинное сравнение которого с имеющимися в ЭИСБЗ (возможными) терминальными фреймами-образцами обеспечивает выбор соответствующего данной ДЗО маршрута сценария обучения на схеме компьютерного ЛОС и учебно-методического материала из КПУ.

Вершинами-слотами первого яруса (непосредственно связанными с корнем графа) фрейма-прототипа являются следующие элементы ДЗО: цель обучения, этапы обучения, возможности учебно-материальной базы, требуемый уровень овладения материалом, форма использования КПУ (виды занятий), исходный уровень подготовки пользователя, состав библиографии, формы контроля знаний при работе с КПУ, ресурс времени и др.

Каждый из перечисленных слотов снабжается набором возможных заданий в виде меню выбора, включая «задание-отсутствие» (ЗО) [10] — ответ пользователя

в случае недостатка информации для однозначного заполнения задания. Диалоговая процедура оперативной постановки конкретной ДЗО осуществляется в процессе активизации фрейма-прототипа с использованием следующих возможных цепочек вопросов-ответов (приведены в круглых скобках) фрейм-сценария обучения (где V — знак логической операции «ИЛИ»):

< 1 слот (Цель обучения) > := < 11 (уметь классифицировать объекты) > V < 12 (уметь формулировать задачи) > V < 13 (уметь представлять модель) > V < 14 (уметь выбирать модель) > V < 15 (уметь обосновывать методы) > V < 16 (уметь оптимизировать решения) > V < 17 (уметь проектировать системы) > V < 18 (уметь применять методы на практике) > V < 19 (знать понятия, принципы, законы, теоремы) > V < 110 (знать краткое содержание темы) > V < 111 (ЗО) >;

< 2 слот (Этапы обучения) > := < 21 (начальные этапы — общая ориентировка, ориентировочная основа действий) > V < 22 (рабочие этапы) > V < 23 (заключительные этапы) > V < 24 (ЗО) >;

< 3 слот (Возможности учебно-материальной базы) > := < 31 (дисплейный класс) > V < 32 (табло, экран дисплея) > V < 33 (автоматизированный имитационно-моделирующий комплекс) > V < 34 (автоматизированная система обучения, комплекс программ обучения) > V < 35 (ЗО) >;

< 4 слот (Требуемый уровень овладения материалом) > := < 41 (умение решать типовые задачи) > V < 42 (умение решать нестандартные задачи) > V < 43 (умение решать творческие задачи) > V < 44 (ЗО) >;

< 5 слот (Форма использования КПУ — виды занятий) > := < 51 (лекция) > V < 52 (практическое занятие) > V < 53 (самостоятельное занятие) > V < 54 (семинар, дискуссия, конференция) > V < 55 (ЗО) >;

< 6 слот (Исходный уровень подготовки пользователя) > := < 61 (с опорой — с конспектом) > V < 62 (без опоры) > V < 63 (навык) > V < 64 (ЗО) >;

< 7 слот (Состав библиографии) > := < 71 (основная) > V < 72 (дополнительная) > V < 73 (учебно-методическая) > < 74 (полная) > V < 75 (ЗО) >;

< 8 слот (Формы контроля обучения при работе с КПУ) := < 81 (входной контроль) > V < 82 (фоновый контроль — учебного вопроса темы) > V < 83 (фронтальный контроль — всех учебных вопросов темы) > V < 84 (выходной контроль) > V < 85 (ЗО) >;

< 9 слот (Ресурс времени) > := < 91 (неограниченный) > V < 92 (ограниченный) > V < 93 (2 академических часа) > V < 94 (практически отсутствует) > V < 95 (ЗО) >.

Задача формирования терминального фрейма заключается в выборе из меню конкретных заданий-слотов. После машинного сравнения полученного терминального фрейма и фреймов-образцов выбирается наиболее подходящий фрейм-образец и соответствующий маршрут сценария изучения учебного материала, на основе которого формируется конкретная «доза» обучения (в памяти компьютера).

Доза обучения по теме содержит следующие компоненты:

- текстовая (файловая) часть КПУ;
- маршрут-сценарий изучения отобранного учебного материала в виде определенной последовательности видеограмм компьютерного ЛОС;
- набор контрольных вопросов (входных, выходных, контекстных);
- состав периферийного оборудования;
- набор имитационных и расчетно-логических задач;
- упорядоченный набор программных средств в соответствии с формой обучения;
- перечень рекомендованной литературы.

На этом этапе декомпозиции фрейм-сценария обучения существенны два возможных подхода:

- гибкое формирование пользователем (преподавателем, обучающимся, специалистом) дозы обучения, компоненты которой определяются каждым пунктом меню выбора фрейма-прототипа;
- выбор дозы обучения пользователем из заранее сформированных комплексных наборов компонентов. При этом множество возможных состояний фрейма-прототипа разбивается на конечное число подмножеств, каждому из которых ставится в соответствие конкретная доза обучения.

5. Организация пользовательского интерфейса и сценарий программного продукта

Основной задачей сценария программного продукта (ПП) является организация пользовательского интерфейса ПП в соответствии с решаемой им целевой задачей. Принципы сценария определяют алгоритм синтеза ПП. Цель их применения — сфор-

мулировать задачу группе программистов — разработчиков ПП. Целью, преследуемой разработчиком ПП при использовании принципов сценария, является *обеспечение согласованного пользовательского интерфейса, максимально удовлетворяющего современным эксплуатационным, эргономическим и другим требованиям* [13, 14]. Эти принципы должны удовлетворять правилам формирования терминальных фреймов, чтобы не нарушать концептуальную основу изучения учебной дисциплины.

Основные положения сценария определяются материалом конкретной учебной дисциплины, состоящим из текстовой части, сборника задач, примеров, справок с выделенной иерархией целей обучения.

Пользовательский интерфейс — это порядок человеко-машинного диалога [14]. Способ взаимодействия компьютера с пользователем реализуется в прикладной программной системе, управляющей доступом, переработкой информации и представлением ее в понятном и удобном для пользователя виде. Взаимодействие пользователя с компьютером определяется возможностью пользователя понять, проанализировать информацию, предоставляемую компьютером, и перейти к ответу посредством интерактивной технологии интерфейса (инструментальных программных и физических средств), что составляет язык действий.

Основными показателями согласованности пользовательского интерфейса являются его *эффективность* (способность развития у пользователя максимально оперативной и простой концептуальной модели взаимодействия), *конкретность* и *наглядность*. Последнее достигается использованием наглядных выразительных средств обеспечения человеко-машинного диалога. Кроме того, согласованность пользовательского интерфейса рассматривается в трех аспектах — физическом, синтаксическом и семантическом [13, 14].

Физический аспект согласованности основывается на рациональном использовании технических средств. Для организации пользовательского интерфейса необходимо использовать удобные и привычные для пользователя клавиши и комбинации клавиш. Так, например, клавиша «F1» компьютера традиционно задействуется под функцию вызова помощи, основная функциональная кнопка «мыши» находится под указательным пальцем и др. *Синтаксический аспект согласованности* связан с рациональностью визуализации информации, включающей определение экранных участков и порядка отображения основных пунктов, подпунктов, заголовков и т. п., языка действий. *Семантический аспект согласованности* характеризуется понятностью, доступностью, ясностью диалога, качеством и полнотой контекстных подсказок.

Согласованный пользовательский интерфейс обеспечивает экономию как времени пользователя, так и ресурсов компьютера, а также страховку пользователя от ошибок применения ПП, что, в свою очередь, определяет конкурентоспособность ПП.

Разработка пользовательского интерфейса состоит из проектирования панелей, диалога и «окон» (отдельных ограниченных частей экрана).

При проектировании в каждой панели выделяют три части: меню действий с выпадающим меню, «тело» панели и область функциональных клавиш. Под диалогом понимается рекуррентная процедура «вопрос—ответ» между пользователем и компьютером, включающая запросы на переработку информации и навигацию по приложению КПУ. Действия диалога: ввод, отказ, выход из функции, выход из приложения.

Для КПУ используются три типа окон:

- первичное окно (для диалога);
- вторичные окна (вызываются из первичных и предназначены для параллельного основному продолжения диалога);
- всплывающие окна (участки экрана, расширяющие диалог пользователя с первичным или вторичным окном, и окна сообщений).

Принципы создания пользовательского интерфейса основаны на формировании оконного интерфейса с вызовом главного меню и всплывающих окон подменю. Наиболее перспективным является принцип «объект—действие», который обусловливает минимизацию числа режимов ПП и страховку пользователя от возникающих неудобств в работе с ПП. Пункты главного

меню вызывают всплывающие подменю, а те, в свою очередь, определяют последующие действия. Например, пунктами главного меню КПУ по системной информатике [2] являются:

1. **Раздел учебника.** Выбор номера раздела и темы занятия. Выбор темы занятия приводит к заполнению текстом основного окна. Просмотр полного текста КПУ осуществляется по выходе из главного меню (клавиша «ESC» или «КЛЮЧ») при помощи клавиш листания страниц и стрелок.

2. **Просмотр рисунков.** Выбор рисунков для темы.

3. **Литература.** Вызов библиографической справки по теме. Справка может быть полной (с расширенным списком литературы по разделу), сокращенной (с минимальным перечнем литературы по данной теме) и дополнительной (дополнительная литература).

4. **Задачи.** Выбор и запуск контрольных и демонстрационных задач темы.

5. **Контроль.** Выбор и задание контрольных вопросов для осуществления входного, фонового, фронтального и выходного контроля. По результатам выходного контроля формируется оценка (по количеству правильных ответов) и сохраняется в правом верхнем углу экрана).

6. **«Записная книжка».** Формирование справочного материала, конспекта, опорного конспекта по основному тексту КПУ с последующей записью его на магнитный диск или выдачей на принтер.

7. **Выход.** Окончание работы основного модуля ПП.

Пользовательский интерфейс может быть текстовым или графическим. *Текстовый режим* имеет преимущества в скорости вывода информации на экран и позволяет использовать сравнительно простые драйверы клавиатуры, не требует специального расширения оперативной памяти. Его целесообразно применять на компьютерах с небольшими по быстродействию и оперативной памяти ресурсами и графическими адаптерами. *Графический тип* пользовательского интерфейса в зрелищном восприятии может намного превосходить текстовый тип, но эти преимущества проявляются только при наличии нескольких мегабайт расширенной или верхней памяти на суперкомпьютерах с качественными графическими терминалами высокого разрешения. К дополнительным атрибутам пользовательского интерфейса относится его согласованность с другими ПП. Разумно предусматривать возможность использования «мыши» для работы с ПП во всех функциональных режимах.

Работа пользователя с программным продуктом КПУ заключается в выполнении следующих операций:

1. **Загрузки ПП.** Из каталога файлов ПП (рис. 2) производится запуск .EXE модуля. Эта операция приводит к появлению на экране дисплея основного окна ПП с главным меню выбора).

2. **Загрузки текста по теме занятия.** Активизируются пункт меню «Раздел» и соответствующие пункты всплывающего подменю для выбора подраздела, темы, учебного вопроса. Выбор и выполнение пунктов меню происходят при нажатии клавиши «Enter» («ВВОД»). В результате выполнения операции основное окно заполняется материалом темы занятия.

3. **Просмотра рисунков, получения библиографической справки, выполнения задачи.** В тексте по теме занятия предусмотрены обращения к рисункам. Для выбора конкретного рисунка необходимо активизировать главное меню клавишей «F10», перейти при помощи стрелок к пункту «Рисунок» и указать во всплывающем подменю конкретный рисунок для просмотра. В результате выполнения операции инициируется графический режим и визуализируется соответствующий динамический или статический графический объект. Клавиша «ESC» возвращает предыдущий текстовый образ. Аналогичным образом вызываются разделы меню «Задачи» и «Литература».

4. **Контроля усвоения материала.** Вызывается соответствующий пункт меню для самопроверки готовности к конкретному занятию и анализа качества усвоения материала. Вызов процедуры контроля и выход из нее происходят аналогично пункту 3. Режим контроля может задавать и преподаватель. Контрольная оценка не исчезает с экрана дисплея до выхода из материала по теме занятия.

5. **Отбора учебного материала**. Вызыва-

ется пункт меню «Записная книжка», который предоставляет пользователю возможность формировать текстовый файл, являющийся произвольным подмножеством материала по теме занятия, или согласно решаемой ДЗО. При активизации данного пункта меню пользователь при помощи клавиш «Alt+B» («ДОП+B»), «Alt+E» («ДОП+E») формирует начало и конец очередного фрагмента для сохранения. Отобранный материал можно распечатать на принтере или сохранить в отдельном файле.

6. **Выхода из ПП.** В главном меню вызывается соответствующий пункт, приводящий к выгрузке модулей ПП из оперативной памяти компьютера и завершению работы программ КПУ.

Заключение

По мнению авторов, компьютеризированный учебник дает преподавателю новые возможности по организации учебного процесса, а обучающимся — по развитию творческих способностей. Первый практический опыт применения КПУ по различным учебным дисциплинам в Академии им. Ф. Э. Дзержинского показал множество преимуществ КПУ по сравнению с традиционными учебниками:

- обеспечение обратной связи «обучаемый — преподаватель» для постоянного творческого совершенствования КПУ;
- значительное сокращение времени на изучение учебных дисциплин;
- создание (за счет анимационных иллюстраций) иллюзии постоянного присутствия педагога рядом с обучаемым и «оживления» сухих страниц учебника-книги (это особенно важно для заочников, экстерников и пропустивших плановые занятия);
- создание приятного психологического настроя у обучающихся (в трудный момент «машина нам поможет»);
- обеспечение индивидуализации обучения за счет отбора каждым обучаемым учебного материала из КПУ и изменения последовательности его изучения с учетом своих психофизиологических особенностей, возможности неоднократного возврата к трудным вопросам и самоконтроля при выборе и решении задач различной степени трудности и др.

Создание компьютеризированного учебника — учебника будущего — является, на наш взгляд, одной из интереснейших и перспективных задач педагогики.

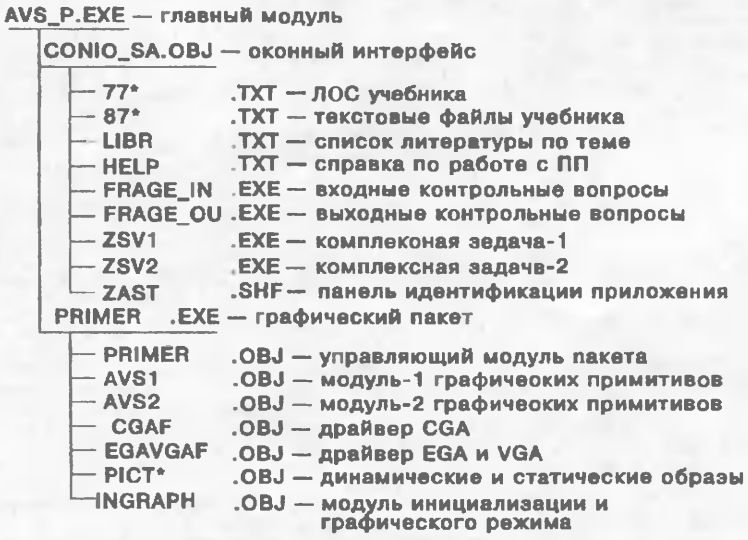


Рис. 2. Файловая структура программного продукта

Литература

1. *Глазов Б. И., Ловцов Д. А., Сухов А. В.* Системная информатика с приложением в космонавтике: Фрагмент компьютериз. учебника. М.: Российская ассоциация ИРСНИО «Кадры», 1992.

2. *Глазов Б. И., Ловцов Д. А., Сухов А. В, Михайлов С. Н.* Концептуальные основы создания компьютеризированных учебников//Компьютерная технология обучения в академии: Научно-метод. материалы. М.: Академия им. Ф. Э. Дзержинского, 1994. Вып. XXXVIII. С. 38—44, 118—120.

3. *Золотарев А. А.* Разработка интенсивных дидактических систем//Научные основы определения содержания подготовки специалистов. М.: Академия имени Дзержинского, 1981. С. 25—32.

4. Проект концепции системы интенсивного обучения в ввузах. М.: Российская ассоциация ИРСНИО «Кадры», 1992.

5. Сборник методологических рекомендаций по разработке содержания обучения и дидактических систем/Под ред. А. А. Золотарева. М.: МИИГА, 1988.

6. *Поспелов Г. С.* Искусственный интеллект — основа новой информационной технологии. М.: Наука, 1988.

7. *Горелов И. Н.* Разговор с компьютером: психолингвистический аспект проблемы. С послесловием Д. А. Поспелова. М.: Наука, 1987.

8. *Клейман Г. Н.* Школы будущего: компьютеры в процессе обучения: Пер. с англ. М.: Радио и связь, 1990.

9. Индустрия программных средств. М.: Знание, 1989. (Новое в жизни, науке и технике. Сер. «Вычислительная техника и ее применение», № 4.)

10. *Поспелов Д. А.* Логико-лингвистические модели в системах управления. М.: Энергоиздат, 1988.

11. *Глазов Б. И.* Методологические основы информационно-кибернетической системотехники. М.: Академия им. Ф. Э. Дзержинского, 1992.

12. *Динамов Б. С.* Методика установления междисциплинарных связей учебных дисциплин. М.: МИИГА, 1988.

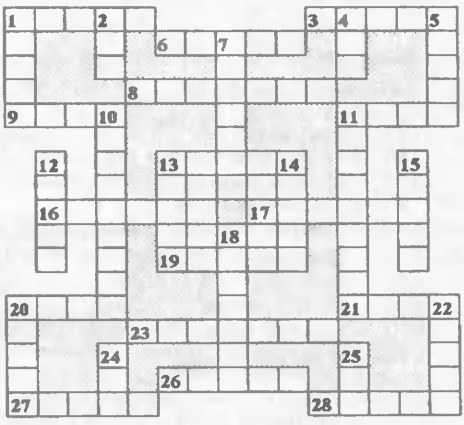
13. *Полоцкий В. К.* Работаем с системой Clipper. М.: Мир, 1990.

14. Проектирование пользовательского интерфейса на персональных компьютерах. Стандарт фирмы IBM/Под ред. М. И. Дадашова. М.: А/О «ЛЕВ», 1992.

* * *

КРОССВОРД

Составитель Д. М. Златопольский



По горизонтали:

1. Результат арифметической операции.
3. Один из контактов транзистора типа МОП («Металл – Оксид – Полупроводник»).
6. Машина с человекоподобными действиями, управляемая по программе.
8. Марка игровой приставки к телевизору.
9. Часть экрана монитора, используемая по специальному назначению.
11. Операция переработки информации в компьютере.
13. Знак арифметической операции.
16. Цифра десятичной системы счисления.
17. Число, определяющее номер элемента массива.
19. Порядковый номер байта оперативной памяти.
20. Компонент данных типа запись.
21. Степень, уровень значимости чего-либо.
23. Язык программирования.
26. Небольшая по размерам программа специфического назначения.
27. Символ на клавиатуре.
28. Прямоугольник, ограничивающий меню или т. п.

По вертикали:

1. Два смежных байта оперативной памяти.
2. Положение в компьютерной игре, имитирующей древнеиндийскую игру.
3. Состояние, в котором вы можете быть президентом фирмы IBM.
5. Электрод электронной лампы.
7. Фирма – разработчик систем «Турбо» для языков программирования.
10. Число, определяющее систему счисления.
11. Жесткий магнитный диск для хранения информации в персональном компьютере.
12. Вид связи.
13. В языкаx программирования – идентификатор, позволяющий именовать некоторый оператор программы.
14. Тригонометрическая функция, расчет значения которой предусмотрен в языках программирования.
15. Величина, с помощью которой осуществляется счет.
18. Устройство для вывода информации в персональном компьютере.
20. Комплекс программ одного назначения.
22. Часть книги.
24. То, без чего не может работать исправный и включенный в сеть компьютер.
25. Язык программирования.

Классы

Проектирование конфигурации и поставка "под ключ" компьютерных классов и комплексов на базе IBM-совместимой техники

Программы

Обеспечение системными, инструментальными, программно-педагогическими и административными средствами для IBM-совместимой и техники типа УКНЦ, УКНЦ-01, КУВТ-86, ДВК

Мебель

Изготовление современной мебели для учебных заведений и офисов

Сервис

Комплексное программно-техническое обслуживание и ремонт средств вычислительной техники (КУВТ-86, УКНЦ, УКНЦ-01, ДВК, БК, Партнер, Корвет, Агат, IBM-совместимая

107005, г.Москва, Волховский пер., 11,
АОЗТ фирма "ЭКСИ" (бывш. "Колледж")
Тел./факс: (095) 265-62-65
Тел.: (095) 267-70-58

ЭКСИ
АОЗТ Фирма "ЭКСИ"

MultiVision

"Замечательное средство для создания презентаций
с использованием красочных анимаций, текстов и музыки"
"PC WORLD"

Универсальная среда для
гибкой генерации и сборки учебных
курсов, обучающих программ и
компьютерных шоу.

ИНТЕРВЬЮ

НА COMDEX'94 МЫ СТОЯЛИ РЯДОМ С PANASONIC

УРНОВ Василий Андреевич, президент НЦПСО—AIST group, член редакционной коллегии журнала «Информатика и образование».

— Василий Андреевич, расскажите, пожалуйста, как Вы пришли в систему образования?

— В 1984 г. я закончил механико-математический факультет МГУ им. М. В. Ломоносова по кафедре топологии, по образованию я математик-тополог. Сложилось так, что, учась в университете, я уже со второго курса работал с маленькими детьми. Мне это было очень интересно, и работа с детьми получалась. Поэтому сомнений не было: я решил сразу пойти работать в школу, тем более что после университета это было возможно. В 1984 г. я начал работать учителем математики и, отработав около трех лет, был классным руководителем от четвертого класса до десятого. Успел познакомиться со всеми прелестями школьной жизни. К началу «великой программы компьютеризации» (1985—1986 гг.) в СШ № 209, где я работал (сейчас это УВК № 1601, наша основная экспериментальная площадка) появились первые машины. Мы открыли межшкольный вычислительный центр. Нагрузка была 42—46 часов в неделю, по 7 уроков каждый день, не говоря уже о проблемах, связанных с обслуживанием компьютеров. Но работать было очень интересно!

Очень скоро мне стало понятно, что нужны квалифицированные сотрудники, и я начал привлекать единомышленников. Два человека, которые первыми пришли в эту школу по моей просьбе — Дмитрий Юрьевич Климов (сейчас главный эксперт НЦПСО по науке) и Вячеслав Юрьевич Саватеев (начальник технических служб). Я с ними учился, а так как мы были выпускниками МГУ, то Дмитрий Юрьевич, как человек очень «методический», сразу сказал, что нужно подводить под нашу самодеятельность базу. И мы быстро написали учебник. Его печатали большими тиражами. Но самым интересным мы считаем хорошую подборку задач. В этом помогал Михаил Кириллович

Петров (генеральный директор AIST Inc., нашей дружественной фирмы).

– Мы познакомились, когда Вы уже работали в городском управлении народного образования, в отделе информатизации...

— Теперь это Московский департамент образования. Тогда не было отдельного отдела по компьютеризации, а было управление труда и информатики.

Все разворачивалось довольно быстро. Был организован отдельный отдел. В школы активно пошла техника: и БК, и УКНЦ. В то время было принято решение оснастить московские школы УКНЦ. Самой большой проблемой у этих машин было отсутствие программ и то, что они нужны, не все понимали. Я не помню фамилию главного финансиста ГУНО, но я ему месяца три доказывал, что программы нужно писать профессионально, а он отвечал примерно следующее: учителям дали компьютеры, вот пусть они и пишут. Мы его долго убеждали, что это некий промышленный процесс, производство, для которого нужны самые разные специалисты самых разных уровней и самых разных областей, начиная от программистов и заканчивая учителями-методистами, художниками.

Вторая проблема была связана с установкой и техническим обслуживанием — т. е. с железом. Железо было безобразное, сырое, несмотря на то, что вообще-то машина УКНЦ хорошая, но у нее есть один существенный недостаток — она ни с чем, кроме себя, не совместима.

– Как родилась идея создания НЦПСО?

— Эта идея пришла к Анатолию Александровичу Зубченко, по специальности детскому психологу. Он работал, как теперь говорят, в ВПК, мы сидели в школе и ГУНО и пытались писать какие-то программы. В рамках постановления № 601 МГК ВЛКСМ появилась юридическая возможность открыть что-то, похожее на самостоятельную структуру, которая могла бы работать по интересному направлению на хозрасчетной основе. У Анатолия Александровича были идеи о реализации определенных психологических концепций в образовании, а у меня были идеи по технической поддержке компьютеров. Это был 1988 год. Прошло уже семь лет — мы, наверное, одно из самых старых предприятий, работающих в сфере информатизации образования.

– НЦПСО – ведущая фирма, которая много работает с зарубежными партнерами. В чем отличие в подходах к информатизации?

— Мы начали информатизацию позже, чем, например, Америка и Европа. На федеральном уровне эти процессы регулировались в Америке и Японии, которые мало чем отличались по официальному старту. В Америке и Японии информатика начиналась со всеобщего обучения компьютеру. В то время практически не было никаких пользовательских средств, т. е. не было сегодняшнего подхода к компьютеру, как к бытовому устройству. Компьютер был довольно сложным, весьма дорогим, его использовали в основном для программирования, и именно поэтому начинали учить языки программирования. Правда, к этому времени существовала такая методика, как LOGO. Она была реализована еще до персональных машин на фреймах — больших машинах, но была слабо распространена, хотя это и была единственная педагогическая методика, нацеленная на обучение.

И в Америке, и в Японии в информатизации образования довольно быстро появились коммерческие тенденции. В Америке — фирма APPLE, которая действительно построила блестящий персональный компьютер, в то время просто потрясающий по своим идеям и решениям. В Япо-

нии — линия NEC вместе с MS-линией. Но и в Америке и в Японии государство плохо разбиралось, что такое информатизация образования. Была некая генеральная линия, а также экспертные оценки, что все это, наверно, правильно. Поэтому найдя генерального подрядчика, государство с удовольствием отдало информатизацию ему на откуп. Другое дело, что там можно за все спрашивать: что делает фирма, как, особенно за федеральный заказ. С другой стороны, все было довольно демократично, так как появился IBM персональный компьютер, следовательно, появился выбор, и покупали и то, и другое.

Американские школы были оснащены техническими средствами лучше наших: видеомагнитофоны и телевизионные системы, системы внутренних коммуникаций и т. п. Поэтому компьютер в школе был у них встречен более обыденно, чем у нас. Коммерческие фирмы стали привлекать различных специалистов для разработки этого направления. Фирма APPLE на этом просто росла, и все было хорошо до тех пор, пока кто-то из американских социологов не заинтересовался: зачем нужно столько программистов? И действительно, крен в эту сторону был большой, а программист — достаточно узкая специальность. Мы повторили их путь.

В настоящее время, я думаю, что у нас ситуация лучше, потому что в Америке, когда не оправдались надежды от внедрения компьютера в образовательную сферу, финансирование стало сокращаться и работы сворачиваться. До сих пор в американских школах на различных машинах работают учителя, на федеральном же уровне регулирования уже нет. Но на университетском уровне информатизация разворачивается, там образование принципиально отличается от школьного. На мой взгляд это очень правильно: в среднем образовании нужно учить работать и думать, а на уровне высшего — о чем думать и над чем работать. Потому что в том возрасте, когда дети получают среднее образование, они не учатся, а живут, а получая высшее — учатся. Это совершенно разные вещи и разные психологические концепции.

Еще одна причина динамичного развития информатизации образования, на мой взгляд, состоит в том, что у нас эта область стала «зоной свободного творчества». В информатизацию пришли творческие, способные специалисты, которые могут экспериментировать, хотя и с определенным риском. И здесь наше отставание по технологической линии сработало в плюс, потому что те идеи, которые стали появляться в головах, особенно последние 2—3 года, смогли быть реализованы, техника под них успела. Идея мультимедиа — идея новых технических решений — получила промышленную реализацию, стала реальностью!

В России сегодня существует несколько принципиальных технологических решений в области информатизации образования, около десяти серьезно работающих фирм в этой области, развивается компьютеризация предметов. И в эту сторону двигаемся мы, в эту сторону двигаются ИНФОМИР, ИНТ, КОЛЛЕДЖ и т. д.

НЦПСО предлагает свои подходы, которые, кстати, получили высокие оценки у американских и японских специалистов: концепции сред, миров, АРМ учителя, ученика. Мы стараемся сделать компьютер инструментом не чужеродным в образовании, а инструментом генерации предметных сред, т. е. предметных обучающих курсов, интерактивных обучающих мультимедийных предметных систем, которые были бы предметно независимы, т. е. когда программно-аппаратное решение не зависит от того, на каком предмете оно используется:

физика, химия, математика — оно подготовлено к любому из этих предметов. Мы все упираемся в нехватку ресурсов, людей, денег, в первую очередь, на массовую кадровую подготовку. И нам — кто это производит, и тем, кто пишет, и тем, кто все это придумывает, нужен грамотный пользователь. Но нам важно, чтобы такие учителя были не только из области информатики, но и из предметной области.

Кроме того, мы хотим, чтобы эти технологии были во всех школах, чтобы информационная грамотность стала абсолютно всеобщей.

– Чего нам ожидать в ближайшие 2–3 года?

— Если говорить о предметах, то с моей точки зрения в связи с быстрым развитием мультимедийных систем, связанных с видеотехнологиями, с интерактивными технологиями будет развиваться поддержка гуманитарных предметов. Изначально казалось, что легче поддержать предметы естественного цикла: математику, физику, технические дисциплины. Но мы считаем, что это не так. Предметы, в которых уровень наглядности и абстракции выше — история, биология, химия и т. п. поддерживаются интерактивными демонстрационными средами на порядок лучше. Ведь для этих предметов гораздо более важен эффект зрительного и слухового, а, фактически, мультимедийного восприятия. У нас есть некоторые решения и для предметов естественного цикла, например: в физике — это моделирование эксперимента, в математике — алгоритмы создания программ, а в гуманитарных областях, как я говорил выше, — демонстрационные интерактивные среды, уменьшающие степень абстракции восприятия и улучшающие усвоение знаний.

Что касается информатики — этот предмет существует не во всех странах, и мне не кажется, что он нужен как общеобразовательный, но он должен существовать, как центр поддержки внедрения во все остальные предметы, в делопроизводство, в управление и др. Курс информатики должен стать базой для внедрения компьютера в учебный процесс, создания информационной школьной среды. А сегодня вводить его надо, начиная с VI класса.

– А что преподавать в таком курсе информатики с VI класса?

— У нас есть целый набор готовых объектов, программ и технических решений. В этом курсе компьютер надо рассматривать как некий черный ящик. Ребенок должен относиться к нему как к телевизору или стиральной машине и знать, что он может, не понимая, как он это может. Дальше, когда ребенок подрастет и сможет сам определять, что ему нравится и чем бы он хотел заниматься (а это происходит в VII, VIII классе) уже нужно смотреть, что ему ближе: алгоритмическая сторона или изобразительная. Существующие среды на IBM-совместимых машинах позволяют ему это реализовать. В результате лишь 2, 3, 4 человека займутся тем самым программированием, которое им на роду написано, кто-то займется базовыми математическими вещами, кто-то уйдет в техническую, кто-то — в область, связанную с гуманитарными вещами, но у них есть базовый набор знаний по информационным технологиям.

– Сейчас уже существуют курсы информатики в начальной школе, а Вы говорите, что этот курс надо вводить с VI класса?

— Хорошо бы его вводить и в начальной школе, но это очень страшно. Лично я за это не берусь, потому что пока и мировой опыт, и наш отечественный еще достаточно малы, а наломать дров здесь можно.

– В чем это выражается?

— Мы не только с малышами, но и со взрослыми зачастую попадаем в следующие ситуации. В школе появляется информатика, компьютеры, молодые ребята, которые с интересом с утра до ночи этим занимаются, интересные творческие люди, — создается некое сообщество. Затем эти дети начинают втягиваться в информатику все сильнее, заканчивают школу, поступают в технические институты и через два года выясняют, что многие из них — люди с классическим гуманитарным уклоном, а время было безвозвратно упущено, ведь то, что они могли бы взять по гуманитарной линии, они не взяли, а все что взяли — это не их. Мы как преподаватели и методисты сталкивались с этим не раз.

Когда мы это поняли, то сразу стали вводить видеотехнологии в гуманитарных науках, даже когда еще не было «железных ресурсов». Например, Е. А. Горский (сегодня — заместитель генерального директора НЦПСО, в 1988—90 гг. — учитель истории той же школы № 209) два года собирал огромную видеотеку. Мы старались делать все, чтобы сбалансировать интерес к компьютеру «гуманитарной составляющей».

А маленькие дети не умеют отбиваться от учителя...

– Но проводят же эксперименты по использованию компьютера в детском саду?

— Вот к этому я отношусь крайне отрицательно, так как в тех детских садах, которые я видел, подход чисто технологический. Информатикой в начальных классах действительно здорово в нашей стране занимается Ю. А. Первин. У него и его коллег в «Роботландии» преобладают содержание, цели, а не технические решения.

Наша реализация пакета JUNIOR для младшей школы идет от базовых философских вещей, которые не могут повредить. Философский подход в «Роботландии» гарантирует ее от опасных ошибок. Ведь «Роботландия» — это подборка гениальных задач человечества, курс, в основе которого лежат задачи для разного возрастного уровня: от самых маленьких детей до больших: «Мудрый кролик», «Задача о 4 конях», «Морской бой».

Например, «Морской бой» — это введение в теорию матриц, в систему координат, систему индексов, систему функций, последовательностей — а ребенок-то воспринимает как морской бой. Это задача базовая.

– Что такое базовые задачи?

— Система координат — это понятие из области математических или общекультурологических знаний? Конечно, из второй.

А вопрос: почему компьютер работает в двоичной системе? Когда я начинал в школе объяснять двоичную систему счисления, то обнаружил, что ученики IX—X классов не понимают, что такое вообще системы счисления. Их как громом поразило, что можно считать по-другому. А это базовое понятие. Те, кто это понял, никогда цифру числом не назовут. Это тоже из области общечеловеческих культурологических ценностей.

– Какая техника должна стоять в школе: профессиональная или специальная учебная?

— Я думаю, что в школе должна стоять техника, специализированная и ориентированная на учителя, но в то же время предельно профессиональная. Особенно это касается новейших технологий мультимедиа, которые являются основной линией сегодняшних производителей ВТ. «Гуманизация компьютера» — лозунг фирмы INTEL, выдвинутый на выставке COMDEX'94. «PENTIUM-компьютер — это будущий домашний компьютер Америки» — декларирует MICROSOFT. Компьютер для широчайшего пользования, для интерактивного телевидения и всего, что с

этим связано — стремление ведущих компьютерных фирм.

Но техника должна быть обязательно обкатанной в смысле промышленного производства.

Почему мы не поставляли в 1994 г. аппаратуру, связанную с видеокомпьютерной обработкой? Ответ однозначный: потому что в 1994 г. существовало около десятка профессиональных устройств для компьютеров с видеообработкой, но ни одно из этих устройств не было законченным. И может возникнуть рисковая ситуация когда пользователь-покупатель скажет: «У фирмы все такое!» Повторяю — в системе образования должны использоваться последние достижения, но они должны быть промышленно обкатаны и должны отвечать соответствующим гарантиям.

Мы принципиально отличаемся от всех фирм, работающих на рынке образования, тем, что подходим к тому, что делаем, как к промышленному производству. Мы считаем, что в школу не может поступать «сырое» изделие, экспериментальный образец. Поэтому наш комплекс — промышленный и на уровне оборудования, и на уровне программ.

– Расскажите о выставке в Америке.

— Самое существенное — то, что в этом году выставка была меньше, чем в прежнем, хотя она была более спокойная, более красивая, более презентабельная. Возник вопрос: отчего упал интерес? На самом деле прошлый год был последним годом неосознанного роста интереса.

В этом году усилия крупнейших мировых компаний (MICROSOFT, INTEL, BORLAND и т. д.) были направлены на то, чтобы люди поняли, что происходит в области новых информационных технологий, на обучение людей этому пониманию, на информирование о том, что будет завтра, какие будет перспективы, и на выставку приехали уже те, у кого были новые вопросы, кто серьезно смотрит вперед. Информации достаточно. Основная заслуга в этом конечно фирмы MICROSOFT, которая проводит гениальную политику: бесплатные консультации, огромное количество серьезно отточенной информации о своих пакетах, сделанных профессионально для простого пользователя, стандартизация оборудования, программ и всего остального.

Второе отличие состоит в том, что на сегодняшнем этапе компьютер уже настолько близок гуманитарному человеку, что будущие технологии стали уже вполне реальной вещью для экономических обоснований, для инвестиций и вкладов, для дальнейших принципиальных разработок. Они становятся реально интересны коммерческим структурам.

На сегодняшний день основной проект MICROSOFT — **интерактивное телевидение.**

Представьте себе, что вы включаете телевизор и имеете не конкретную сетку программ, а очень простое устройство, с помощью которого вы можете обратиться на станцию, где на серверных системах запущены сотни тысяч программ: от обучающих до самых разнообразных, плюс прямой эфир. Вы можете на сегодняшний день выстроить свою конкретную программу так, как вам нравится — она будет только ваша. Вы можете посмотреть прямую трансляцию, но в любой момент можете ее прервать и посмотреть потом, поскольку она уже будет записана на сервере. Конечно, это не только связано с денежными вложениями, но и со временными рамками. Но вы строите только то, что вам, жене, ребенку нужно. Существует огромная подборка сконцентрированных по определенным структурным и социальным законам материалов: от того, чем кормить собаку, до того, как лечить ребенка.

Это великий проект, который

можно реализовать только посредством компьютерных систем мультимедийного направления, связанного с лазерными CD-технологиями, технологиями передачи информации, работы с графикой. Но это уже становится реальностью!

Из более близких тенденций — развитие мультимедийных технологий. Мы привезли с выставки несколько решений, точнее, четыре решения в рамках своего мультимедийного комплекса MultiVision Pro — от дешевого до предельно профессионального. Произошла стабилизация мультимедийных технологий, они стали действительно пользовательскими и надежными.

Сегодня мы можем наблюдать сложности у фирмы APPLE. Их проекты идут плохо, особенно это связано с проектом Ньютон, который проигрывает конкуренцию по причине не самых хороших технологических решений. Большие проблемы связаны с PowerMacintosh, в котором существует больше операционных систем, чем пользовательских приложений, и они очень проигрывают системе Windows. Проект, который ведет MICROSOFT и INTEL в связи с новой операционной средой Windows 95 который до этого назывался проект Chicago, сейчас уже существует и прекрасно смотрится. Главное в том, что в 1994 г. MICROSOFT была проведена работа по переносу всех прикладных пакетов под эту среду, и аппаратура оказалась к ней готова. Это позволит абсолютно безболезненно перейти в 1995 г. по прогнозам MICROSOFT и INTEL 65% пользователей в эту среду. Мы уже под нее работаем.

К такой конкуренции оказалась не готова фирма APPLE, хотя она по-прежнему лидирует в некоторых областях, например, в издательской деятельности.

Таким образом, все новые решения в области мультимедиа ложатся на плечи компьютеров направления INTEL. MOTOROLLA тоже уходит на специализированный рынок — производство систем мобильной связи.

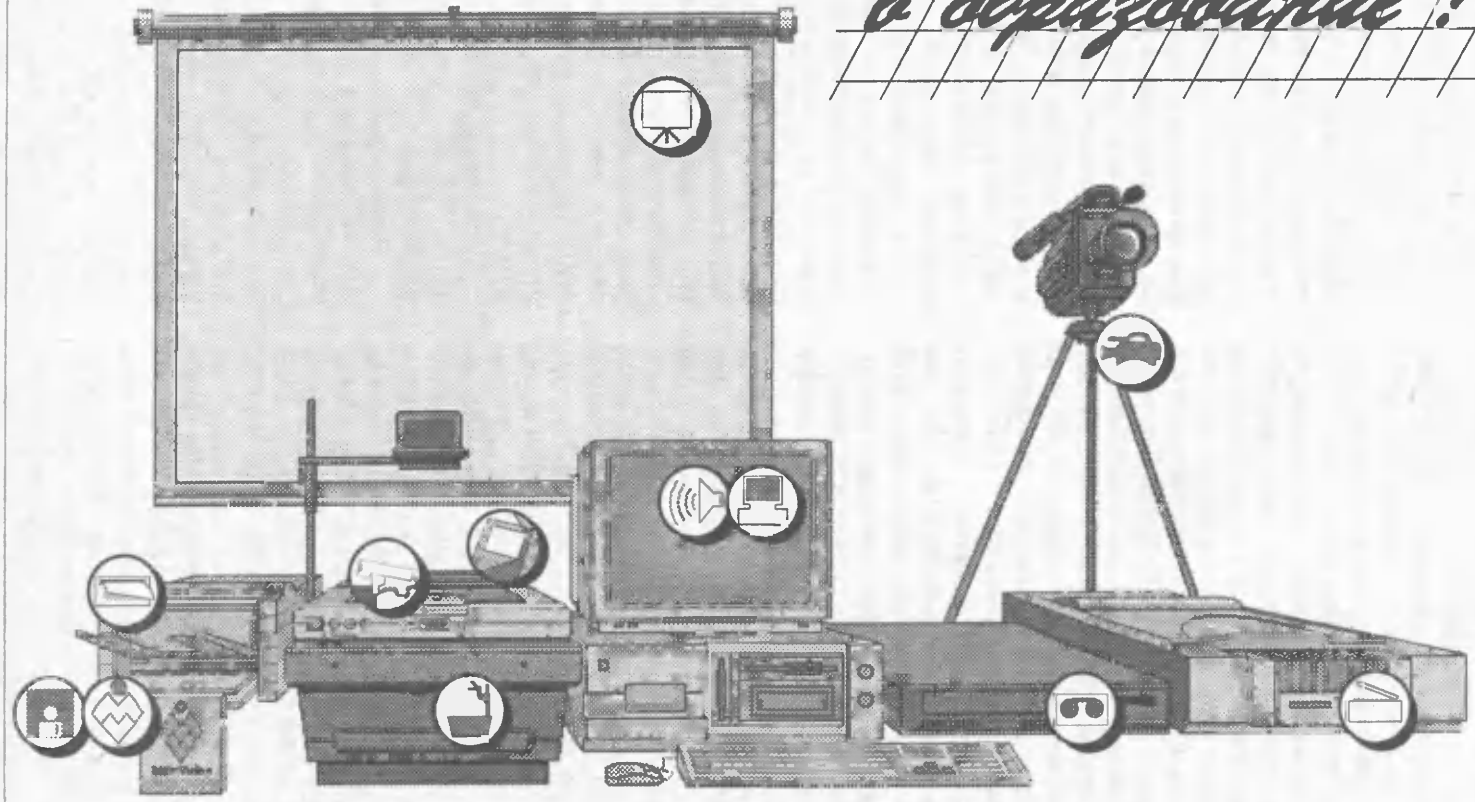
На выставке в Лас-Вегасе мы выставляли свою продукцию, свои программистские решения. Я думаю, что они вполне конкурентноспособны, поддержаны реальным производством, серьезной проработкой инфраструктуры. Первый раз мы попытались выйти на Запад в 1990 г. Все эти годы мы учимся, как это делается в нормально экономически устроенных странах, на каких принципах, что такое западная реклама, что такое западное производство, что такое западные стандарты. В этом году мы имели свой собственный большой стенд на COMDEX (это вообще большое достижение — из России стенд был только у STEEPLER). Наша делегация вместе с американскими коллегами составляла около 15 человек. Это был серьезный проект, который уже сегодня дает результаты. Мы ведем сейчас более 60 переговоров о контрактах с разными странами, не только США.

Выставка в Лас-Вегасе собирает около 200 тыс. человек. Она состоит из нескольких павильонов.

Центральный павильон — место, где собираются наиболее серьезные участники, и мы стояли именно там, благодаря хорошим и давним контактам с американскими партнерами. Я испытывал просто детское удовольствие, что рядом с PANASONIC, SUMSUNG и т. д. стоим мы.

MultiVision PRO

Мультимедиа – в образование!



IBM PC/AT-486 'Multivision Pro'

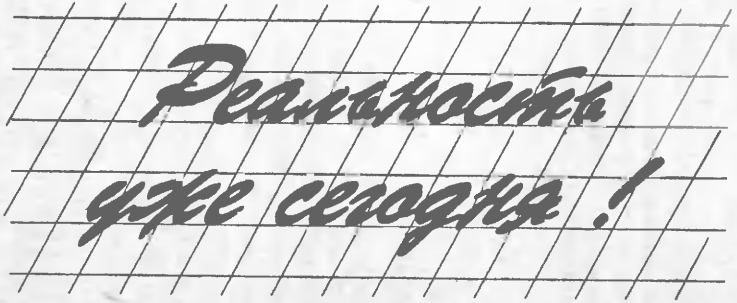
Базовой составляющей комплекса MultiVision Pro является мультимедийный компьютер IBM PC-486 "MultiVision Pro". Открытая архитектура, встроенный источник звука, CD ROM, режим PC TV - далеко не полный перечень его возможностей сегодня и завтра.

Проекционное оборудование

Демонстрационные возможности комплекса базируются на сочетании уникального оборудования. Цветная жидкокристаллическая панель *Proxima Ovation 820* - это 2 миллиона цветов видеокомпьютерного изображения без каких-либо задержек на экране размером до 3 метров по диагонали. *Proxima Cyclops 2030* - интерактивная система, возможность управлять компьютерной программой непосредственно у настенного экрана или на расстоянии до 8 метров. *Overhead Projector Medium 10K* - мощный и ровный световой поток, надежная защита красок панели, возможность работы со стандартными "прозрачными" носителями.

Мебель

Современный дизайн *мебели,* элегантные технологические решения позволят Вам компактно разместить оборудование комплекса, легко перемещать его по аудитории и создадут необходимый комфорт в работе.



Genicom Color Printer

Печать высококачественных графиков, диаграмм и иллюстраций обеспечит Вам цветной принтер с термомопереносом *Genicom 7025*. Технические возможности принтера позволяют получить полноценный отпечаток как на бумаге, так и на прозрачной слайдопленке.

Hewlett Packard Colour Scanner

Цветное сканирующее устройство *HPSJ IICx* поможет Вам легко и просто создавать библиотеки компьютерных изображений. ***Внимание!!!*** Вы можете, используя сканер и проекционное оборудование комплекса, получить изображение на большом экране с любого бумажного носителя.

Видеооборудование

Высокое качество видеоизображения обеспечивает оборудование фирмы Panasonic. *Видеомагнитофон* - VHS формат, видеостандарты PAL, SECAM, воспроизведение сигналов NTSC. *Видеокамера* - съемка в формате VHSC с 8-ми кратным объективом, работа в видеорежимах SP и LP, ручная и автоматическая настройка фокуса.

Tripod Screen Professional

Качество, степень яркости, контрастность видео и компьютерных изображений, проекций слайдов и кодограмм станут значительно выше, если Вы пользуетесь экраном *Medium Tripod Screen Professional*.

Программное обеспечение

Создание обучающих программ, учебных курсов и компьютерных демонстраций будет достаточно простым, если Вы работаете с пакетом *MultiVision* - основой комплекса MultiVision Pro. Красочные анимации, яркая графика, тексты, музыка и компьютерное видео сделают Ваши уроки и лекции незабываемыми.

Научный центр программных средств обучения
1090004, Москва, Б.Коммунистическая ул., 9-а
Тел. 272-26-71, 214-46-49 Факс 271-04-28

MultiVision Version 4.5

ВЫ МОЖЕТЕ СОЗДАТЬ

- Образовательные программы для демонстрационной поддержки учителя и самостоятельного изучения материала учащимися.
- Интерактивные шоу для докладов и лекций.
- Демонстрационные ролики для выставок.
- Путеводители, компьютерные справочники, каталоги и многое другое.

ВОЗМОЖНОСТИ

В своих обучающих или демонстрационных программах Вы можете использовать:

- мультфильмы, для создания которых не требуется профессиональных навыков;
- картинки, нарисованные или считанные с помощью сканера;
- текст, с разнообразным шрифтовым оформлением;
- звук, воспроизводимый с высоким качеством.

ПРОСТОТА В ИСПОЛЬЗОВАНИИ

Для создания собственных программ совершенно не обязательно быть программистом. MultiVision позволяет, используя одну только мышь, шаг за шагом воплощать Ваши замыслы, пока Вы не добьетесь желаемого результата.

МОДУЛИ

Всю необходимую информацию для Ваших программ можно подготовить в специальных модулях MultiVision - редакторах.

- Animator позволяет быстро создавать мультфильмы. Вы только задаете ключевые кадры, их преобразования и траектории движения.
- Imager собирает изображения и мультфильмы в единые библиотеки.
- PhotoMaster - инструмент для сканирования изображений.
- Painter - мощный графический редактор с уникальным набором возможностей.
- Writer - специализированный редактор для подготовки красочных экранных текстов. Импорт текстов из MS Word 5.0, WinWord 2.0, Word Perfect 5.1. Поддерживает шрифты в формате TrueType.
- Designer - это основной модуль MultiVision, в котором отдельные объекты компануются в законченную программу. Здесь определяется порядок просмотра информации и ее расположение на экране.

РЕЗУЛЬТАТ

Результатом работы MultiVision является не набор файлов, а единая самостоятельная программа, работающая под DOS или MS Windows. Такая программа может быть легко перенесена на любой другой компьютер и распространяться без всяких ограничений.
При желании Вы можете записать программу в виде проигрываемого ролика.

НЦПСО

Научный Центр Программных Средств Обучения
Адрес: 109004, г.Москва, ул. Б.Коммунистическая, 9-а
ТЕЛЕФОН: (095) 272 11 25, 214 46 49. Факс (095) 271 04 28

ТОЧКА ЗРЕНИЯ

Д. Ю. Климов,
ведущий специалист НЦПСО

РИСКОВАННЫЕ ИГРЫ В POWERPC

Персональный компьютер — устройство для работы с Microsoft Windows.

Из толкового словаря 2000 года

Климов Дмитрий Юрьевич — окончил в 1984 г. механико-математический факультет МГУ им. М. В. Ломоносова, по специальности математик-тополог. С 1988 г. ведущий разработчик НЦПСО, является автором учебника «Преподавание информатики в компьютерном классе», изданном в издательстве «Просвещение» в 1989 г. Автор программных пакетов: LOGO для УКНЦ, Multivision, Mediamaster.

В ноябре 1994 г. специалисты Научного центра программных средств обучения (НЦПСО) уже третий раз побывали на крупнейшем компьютерном шоу мира — осенней выставке COMDEX.

Выставка эта традиционно проходит в Лас-Вегасе — крупнейшем центре азартных игр не только США, но и всего мира. Лас-Вегас выбран в качестве места проведения выставки не потому, что в компьютерной индустрии такие азартные люди, а из-за того, что трудно найти где-либо еще столько гостиниц, чтобы разместить 200 тысяч человек, и столько выставочных площадей, чтобы разместить все экспонаты.

Даже краткое изложение вынесенных впечатлений могло бы занять сотни страниц. Но сейчас хотелось бы коснуться одного узкого вопроса, представляющего, тем не менее, большой интерес для российской аудитории. Речь пойдет о так называемом компьютере будущего, построенном на микропроцессоре PowerPC.

Пейзаж перед сражением

В течение многих лет технологической основой компьютерной индустрии являлась «сладкая парочка» Intel-Microsoft. Компьютеры на микропроцессорах Intel 80x86 с операционной системой Microsoft DOS (в последнее время и Windows) захватили львиную долю рынка. Все прочие архитектуры и соответствующее программное обеспечение (ПО) занимали ниши поскромнее. Так, рабочие станции с UNIX использовались в основном в научных кругах, а машины Macintosh — в издательской деятельности.

Но не реже чем раз в два года в компьютерной прессе поднимался шум о конце владычества Intel-Microsoft. То предсказывали эру UNIX, то новые мощные Macintosh и их дружественный интерфейс должны были вытеснить «медленные» IBM PC и «корявый» MS DOS, а то IBM прочила свой тандем PS/2—OS/2 в компьютеры будущего. Немало шуму наделал и компьютер NeXT года три назад.

Ни одна из этих попыток сбросить с пьедестала «сладкую парочку», увы, не имела успеха.

В настоящий момент мы присутствуем при, возможно, наиболее масштабной попытке свернуть компьютерную индустрию с проторенного пути. Консорциум гигантов — IBM, Motorola и Apple — ставят на кон не только огромные денежные средства, но и само свое будущее.

Чтобы ясно представить себе опасность этой игры, следует ясно понимать, что за спиной Intel—Microsoft стоят почти все крупные производители компьютеров по всему миру, тысячи программистских фирм, едва ли не сто миллионов установленных компьютеров.

Зачем это им нужно?

Фирма Apple в течение многих лет имела свою устойчивую нишу — до 10 процентов рынка персональных компьютеров — и лидирующее положение во внедрении новых компьютерных технологий. Ее компьютеры Macintosh обладали уникальным программным обеспечением.

Микропроцессоры 680x0 фирмы Motorola, на которых основаны Macintosh, были мощнее аналогичных изделий Intel.

Фирма IBM, изобретатель IBM PC, была крупнейшим производителем персональных компьютеров (и, заметим, конкурентом Apple).

Однако за последние несколько лет произошли значительные изменения в раскладе сил. Все перечисленные преимущества исчезли. Компьютеры Macintosh утратили свою уникальность — благодаря Windows и новым процессорам Intel. Выпуск персональных компьютеров на традиционных процессорах Motorola 680x0 практически прекращен. Огромные средства, потраченные на разработку кристалла 68060, потрачены впустую, и его появление прошло незамеченным. Разработчики программ осознали, что производить программы для Windows гораздо выгоднее, чем для Macintosh, и больше не существует программ для Macintosh, которые нельзя было бы найти для Windows — с равными или даже большими возможностями. IBM же была оттеснена на периферию рынка персональных компьютеров благодаря энергичным усилиям более молодых фирм, таких, как Compaq, Dell, и напористых тайваньских производителей.

RISC

Архитектура CISC (архитектура со сложным набором команд) появилась 16 лет назад, и именно она лежит в основе процессоров семейства 80x86 (на которых построены IBM PC) и 680x0 (основа Apple Macintosh). Архитектура RISC (сокращенная система команд) считается более современной и дешевой в производстве.

Разработанное совместными усилиями компаний IBM, Motorola и Apple семейство RISC-процессоров PowerPC должно обеспечить высокую производительность при меньшей, чем у изделий фирмы Intel, стоимости. На первый взгляд выглядит многообещающе: завтра же все пользователи Pentium выбросят свои дорогостоящие устаревшие «калькуляторы» и выстроятся в очередь за светлым будущим всей компьютерной индустрии. На деле все обстоит несколько сложнее.

Ценность любой вещи не может определяться каким-либо одним фактором. Чем сложнее предмет, тем большее количество факторов должно быть принято к рассмотрению. Приведем пример из области автомобилестроения: всем известно, что бензиновый двигатель плох. Имеется большое количество альтернативных двигателей — дизель, электромотор, газовое топливо, разного рода маховики, педальный привод и т. д. В прессе периодически возникает шумиха о смерти бензинового двигателя. Тем не менее даже журналисты, раздувающие эту шумиху, имеют автомобили именно с бензиновыми двигателями. Бензиновые двигатели наверняка сойдут с арены, но произойдет это не завтра и даже не через 30 лет. Кроме того, инженеры непрерывно совершенствуют традиционный двигатель, и в скором будущем он станет чище и экономичнее, чем любые имеющиеся сейчас альтернативы.

Рассмотрим главное, по утверждению рекламы, преимущество семейства PowerPC — хорошее соотношение цена/производительность.

В настоящее время доступен только первый из этого семейства микропроцессоров — PowerPC 601. Тесты показывают, что компьютеры с этим процессором превосходят Pentium только при интенсивных вычислениях с плавающей точкой, которые редко используются в прикладных программах. В целом на типовых задачах 66 МГц Pentium превосходит 66 МГц PowerMac на 15—45%.

Странно, не правда ли? Дело в том, что большинство прогнозов по PowerPC основывается на сравнении будущих, еще никем не виданных версий этого процессора и реально существующих на рынке изделий Intel. Здесь сказывается различие рыночной политики этих производителей: про будущие процессоры Intel сейчас *наверняка* известно только то, что они будут полностью совместимы со всеми ранее выпускавшимися моделями, а про грядущие PowerPC пресса раструбила больше, чем известно самим инженерам фирмы Motorola.

Что же касается стоимости процессора, то она зависит в основном от объема производства, а по этому показателю никто и близко не подходит к Intel. Процессоры Pentium становятся базовыми для индустрии, и цены на них весьма быстро падают. Кроме того, несколько независимых фирм начинают поставки совместимых процессоров класса Pentium, что усугубит этот процесс. Заметим также, что стоимость самого процессора обычно составляет весьма небольшую часть от стоимости всей системы, а с повышением требований к объему памяти и стандартной периферии (особенно мультимедийной) доля стоимости процессора будет неизбежно падать. К тому же многие обозреватели отмечают, что в настоящее время Intel имеет возможность продавать свой Pentium по явно завышенной цене (в отличие от Motorola). Так, например, из собственного опыта мы знаем, что за последние шесть месяцев цена процессора 486DX2 66 МГц упала с 380 USD до 175.

Неявно производители PowerPC вынуждены признать такое положение дел. Грустную улыбку вызывают разного рода аппаратные эмуляторы 80x86 для машин PowerPC, значительно увеличивающие стоимость и сложность системы. (Совершенно ясно, что никому и в голову не приходит решать обратную задачу для IBM PC.)

Также ведется много разговоров о преимуществах архитектуры RISC перед CISC; на этом построены многие рекламы PowerPC. Однако архитектурные различия между RISC и CISC становятся весьма условными. Не существует архитектурных ухищрений в мире RISC, которых нельзя было бы найти в CISC. В действительности, Pentium уже представляет собой суперскалярный RISC процессор с небольшим блоком совместимости с 80x86. Грядущие версии процессоров Intel разрабатываются в сотрудничестве с признанным производителем суперкомпьютеров с архитектурой RISC — фирмой Hewlett-Packard; наследник Pentium — микропроцессор P6 уже анонсирован (ожидается, что он будет в 10 раз мощнее первого Pentium), следующий за ним P7 находится в разработке. С другой стороны, и PowerPC нельзя признать «типичным» RISC, поскольку его 200 команд не вписываются в классические 127.

В дело вмешивается еще один критический фактор (одного его достаточно, чтобы погубить будущее нового процессора) — неспособность Apple и IBM договориться между собой. Дело в том, что разный порядок байтов в слове делает *фундаментально* несовместимыми PowerMac фирмы Apple и новые системы PReP фирмы IBM. Программная несовместимость никогда не позволит этим машинам захватить заметную долю рынка.

В общем, мы приходим к выводу, что Apple и Motorola сделались поклонниками RISC не в силу преимуществ этой архитектуры, а потому, что, в силу поражения Motorola, не имели другой перспективы. Если бы процессор 68060 вышел хотя бы на год раньше (а не позже!), чем Pentium, они и сейчас были бы в лагере CISC.

Мягкое подбрюшье новых компьютерных технологий

Производители аппаратуры (каковыми, в частности, являются IBM, Apple и Motorola) слишком часто склонны недооценивать фактор программного обеспечения. Как показывает практика, цикл обновления программного обеспечения составляет около 10 лет, что значительно больше, чем цикл смены аппаратуры.

Возвращаясь к сравнению быстродействия процессоров Intel и PowerPC, нельзя не отметить еще одну тонкость, заключающуюся в том, что все сопоставления относятся к «родным» программам, т. е. написанным специально для данного процессора. Но в то время как для Pentium таких программ существуют десятки тысяч, для PowerPC счет пока идет на десятки. Суровая реальность состоит в том, что для использования ваших любимых программ на PowerPC необходимо применение специальной программы-эмулятора Macintosh, которая на лету разбирает команды 680x0 и исполняет их на PowerMac. Более того, сама операционная система System 7.1.2 на 85% состоит из старого 680x0 кода. Все это приводит к тому, что на реальных задачах PowerMac в режиме эмуляции работает в 3—4 раза (а на некоторых задачах и 10 раз) медленнее, чем Pentium с такой же тактовой частотой (т. е. как домашний PC стоимостью 1000 долларов).

Помимо того, «настоящий» PowerPC модели 620 еще реально не существует. Когда же он все-таки появится, то для него опять придется писать новое программное обеспечение, так как этот процессор 64-битный. Что также означает, что этот процессор никогда не будет использован в полную силу (как это было, например, с Intel 386 — ведь настоящие 32-битные ОС подоспели только к выходу процессора Pentium!).

Миллионы пользователей уже сейчас не воспринимают качественную разницу между System 7 и Windows 3.1. После выигрыша процесса о графическом интерфейсе с Apple руки у Microsoft развязаны, и он радикально изменил интерфейс Windows 95, не оставив никаких преимуществ для Apple. Более того, и в этой области время работает не на Apple. Новые операционные системы от Microsoft — Windows NT и ее «облегченный» вариант Windows 95 находятся далеко впереди System 7 по многим критическим параметрам — многозадачности, защите памяти, поддержке многопроцессорности, переносимости. Внесение таких изменений в System 7 потребует полной ее переделки. Лишь часть из них ожидается в версии с условным названием Copland только в 1996 г. Ясно, что два года — большой срок, и Microsoft не упустит возможность использовать эту фору. (Заметим в скобках, что даже «победа» RISC не означала бы победу Apple, так как у Microsoft и здесь «все схвачено»: Windows NT работает на любых RISC-процессорах, в том числе и на PowerPC.)

Некоторые обозреватели, впрочем, полагают, что новые технологии Apple (OpenDoc и Taligent, позволяющие облегчить интеграцию программ и сделать их компактнее) позволят ей сохранить технологическое лидерство. Но Microsoft движется в том же направлении со своей технологией OLE. Кроме того, скоро обе эти технологии будут реализованы (самой же Apple) и для Windows.

Мы, как представители программистской фирмы, видим еще одну серьезную проблему разработки для PowerMac. Для

этой платформы не существует современных средств разработки. Во-первых, все компиляторы в настоящий момент представлены в бета-версиях. Учитывая, что первые версии любой программы всегда неустойчивы, все программное обеспечение для нового PowerMac (а первые два-три года это будут именно *первые версии*), изготовленное на бета-компиляторах *первых версий*, да еще под управлением *первых версий* новой операционной системы, неизбежно будет весьма неустойчивым. Во-вторых, для System 7 нет средств, подобных Microsoft Visual Basic (HyperCard слишком устарел), дающих возможность мелким производителям быстро и недорого делать огромное количество простых программ и интегрировать «настоящие» аппликации по заказу. Таким образом, переход на PowerPC для небольших производителей программного обеспечения будет дорогим и болезненным, не говоря уже о параллельной разработке под две и более платформы.

Ставка больше, чем жизнь

Итак, можно сделать следующие выводы.

Победа PowerPC (т. е. захват значительной доли рынка) крайне маловероятна. Скорее всего, Apple и Motorola сумеют удержать свою часть рынка (не более 10%), что и является, по всей вероятности, их основной целью.

Не следует ожидать, что процессоры PowerPC в обозримом будущем смогут существенно превзойти изделия Intel по производительности.

Лишь очень небольшая часть производителей программного обеспечения рискнет поставить на PowerPC, ибо как можно ставить свою судьбу на консорциум, участники которого не могут договориться между собой? В основном это будут фирмы, традиционно производившие программное обеспечение для Macintosh. Без программного же обеспечения любой, самый лучший компьютер — лишь бесполезная железка.

Выставка COMDEX'94 оставляет впечатление, что основные баталии проходят и будут проходить между Intel и его прямыми конкурентами — Cyrix, AMD и другими производителями Pentium-совместимых микропроцессоров. Основной операционной системой во второй половине десятилетия станет Windows NT, в том числе и для машин PowerPC.

Литература

1. *Tom R. Halfhill.* Apple's High-Tech Gamble//Byte. 1994. December.

2. *Phil Lemmons.* IBM and Apple Fumble Windows Challenge//PC World. 1994. October.

3. *Michael Miller.* Another Microsoft Magic Show?//PC Magazine. 1994. December 6.

4. *John Dvorak.* Windows NT Is Your Future//PC Magazine. 1994. November 22.

5. *Mike Feibus.* The Gowing PowerPC Family//PC Magazine. December 20.

6. *Eric Knorr.* Power Mac Proves No Contest for Pentium PC//PC World. 1994. December.

* * *

ОТВЕТЫ НА КРОССВОРД, ОПУБЛИКОВАННЫЙ НА СТР. 94

По горизонтали: 1. Сумма. 3. Исток. 6. Робот. 8. «Микроша». 9. Окно. 11. Ввод. 13. Минус. 16. Девять. 17. Индекс. 19. Адрес. 20. Полв. 21. Ранг. 23. Фортран. 26. Вирус. 27. Точка. 28. Рамка. **По вертикали:** 1. Слово. 2. Мат. 3. Сон. 5. Катод. 7. Борланд. 10. Основание. 11. «Винчестер». 12. Радио. 13. Метка. 14. Синус. 15. Число. 18. Принтер. 20. Пакет. 22. Глава. 24. Ток. 25. Ада.

КЛУБ «АГАТ»

С. В. Фролов,
Санкт-Петербург

СОВМЕСТИМ ЛИ SOFT?

С момента выпуска ПЭВМ «АГАТ» и по настоящее время создано большое количество версий системы программирования БЕЙСИК, в которых точки входа в подпрограммы различны. Кроме того, существуют и системы, отличные от БЕЙСИКа, такие, как система «Школьница», «Best tool kit», «Onix» и др. Программы, в которых есть ссылки на подпрограммы в области ПСЕВДО-ПЗУ, обычно используются для какой-либо одной системы и не будут работать в другой.

В заметке А. Кривцова «Агат в образовании 1983—1987» (ИНФО. 1988. № 2. С. 29) обещано, что «соответствие основных точек входа интерпретатора БЕЙСИКа-60 и БЕЙСИКа-67 предполагается опубликовать в ближайших номерах», однако такое соответствие опубликовано не было.

Вместе с тем работа по написанию программ, адаптация их с ПЭВМ «APPLE II» побудили меня самостоятельно составить такую таблицу, которая позволит читателям использовать опубликованные программы на своих компьютерах и адаптировать программы, написанные для других диалектов БЕЙСИКа.

Небольшое пояснение к таблице. По вертикали расположены более или менее стандартные названия подпрограмм, по горизонтали — названия систем программирования, для которых сопоставлены адреса.

Примечания

1) АГАТ — системный монитор, входящий в стандартное ПЗУ и работающий сразу после включения ПЭВМ.

2) Система BASIC-67 распространялась также с адресами монитора типа BASIC-60.

3) СЕТЕВ. — «Сетевой» БЕЙСИК (включен в систему ИКП).

4) T.KIT — отладчик под названием «BEST TOOL KIT 89.2 и 89.3», автор А. Голов.

5) Акк. — регистр аккумулятора.

6) Действие подпрограмм может незначительно отличаться в разных диалектах.

Пояснения и использование подпрограмм

BASCALC — позиционирование курсора (номер строки в Акк.) на начало строки;

BELL — вывод звукового сигнала «БИП»;

BS — перемещение курсора на одну позицию влево;

CLEOP — очищает экран от курсора до конца экрана;

CLEOP1 — то же, но координаты X и Y помещены в Акк. и рег. Y соответственно;

CLREOL — очистка экрана от курсора;

COUT — выводит символ из Акк. на устройство вывода (обычно — JMP COUT1);

COUT1 — выводит символ на текущую позицию экрана;

CR — переводит курсор на начало следующей строки;

CROUT — передает символ $8D подпрограмме COUT;

CROUT1 — то же, но с предварительь-

	АГАТ	BAS-60	BAS-67	СЕТЕВ.	APPLE	РАПИРА	Т. КІТ
BASCALC	FB7D	FB8E	FB26	FB2D	FBC1	1AF4	D969
BELL	FCA5	FCB4	FF3D	FF3D	FF3A	19A4	D89B
BS	FC23	FC1F	FC21	FC21	FC10	1915	D81F
CLEOP1	FC49	FC4B	FC4F	FC4F	FC46	1952	D85C
CLEOP	FC45	FC47	FC4B	FC4B	FC42	194E	D858
CLREOL	FC05	FC0C	FCA5	FCA5	FC9C	198D	D288
COUT	FDD4	FDDC	FDE0	FDE1	FDED	18C6	D7E0
COUT1	FDD7	FDDF	FDE3	FDE4	FDF0	18C9	D7E3
CR	FC5B	FC67	FC6B	FC6B	FCB2	1964	D86F
CROUT	FD75	FD8E	FD92	FD92	FD8E	18C4	D7DE
CROUT1	FD45	FD89	——	——	FC48	18C1	D7DB
GETLN	FD55	FD6E	FD72	FD72	FD6A	1877	D4CB
GETLNZ	FD4F	FD6B	FD6F	FD6F	FD67	1874	D4C8
HOME	FC3B	FC3D	FC$1	FC41	FC58	1946	D850
INIT	FB3C	FBED	FBEF	FBEF	FB2F	1AAE	D93F
KEYIN	FD07	FD0D	FD11	FD11	FD1B	1813	D504
LF	FC5F	FC6B	FC6F	FC6F	FC66	1980	D87C
PRBYTE	FDC1	FDC9	FDCD	FDCE	FDDA	18EC	D953
PRHEX	FDDA	FDD2	FDD6	FDD7	FDE3	18F5	D95C
RDCHAR	FB66	FB7A	FB80	FB74	FD35	1ADD	D549
RDKEY	FD04	FD0A	FD0E	FCCB	FD18	1810	D501
REASS	FE60	FE5E	FE5E	FE63	FE5E	——	E1D6
RESET	FA48	FA48	FA4C	FA51	FA62	1806	0384
SCROLL	FC69	FC75	FC79	FC79	FC70	19FA	D8D9
SETINV	FE77	FE75	FE75	FE7A	FE80	19F7	D8AF
SETNORM	FE7B	FE79	FE79	FE7E	FE84	19BD	D8B4
SETTXT	FB2E	F846	F846	F846	FB39	1AA1	D7CD
TABV	FB50	FBFF	FC01	FC01	FB5B	1ABF	94EF
UP	FC27	FC33	FC37	FC37	FC1A	1929	D833
VTAB	FC2F	FC5B	FC5F	FC5F	FC22	1931	D83D
VTABZ	FC31	FC5D	FC61	FC61	FC24	1933	D83D
WAIT	FB94	FB2F	FB94	FB94	FCA8	1B07	D98E

ной очисткой экрана до конца строки;

GETLN — вводит строку символов;

GETLNZ — то же, но с предварительным переходом на новую строку;

HOME — очищает экран и переносит курсор в левый верхний угол;

INIT — инициализирует параметры экрана (границы, размер и пр.);

KEYIN — берет входной символ;

LF — переносит курсор на строку вниз;

PRBYTE — печатает шестнадцатеричный байт (значение в Акк.);

PRHEX — печатает младший полубайт из Акк. как одну 16-ричную цифру;

RDCHAR — то же, что KEYIN, но воспринимает управляющие коды (стрелки и др.);

RDKEY — ввод символа с устройства ввода (обычно JMP KEYIN);

REASS — вывод 28 строк в мнемокоде ассемблера;

RESET — обработка реакции на клавишу УПР-СБР;

SCROLL — сдвигает экран на одну позицию вверх и очищает нижнюю строку, на которую помещается курсор;

SETINV, SETNORM — устанавливает инверсный и нормальный режимы соответственно;

SETTXT — устанавливает и включает текстовый режим с адресом начала экрана, старший байт которого расположен в яч. $19;

TABV — помещает курсор на строку, номер которой находится в Акк.;

UP — перемещает курсор на строку вверх;

VTAB — перемещает курсор на начало текущей строки;

VTABZ — то же, но номер строки находится в Акк.;

WAIT — выполняет задержку длительностью (5*Акк.*Акк+27*Акк.+26)/2 мкс.

Показанная выше таблица свидетельствует о том, что программы, в которых будет использовано непосредственное обращение к адресам системы, будут работать только в одной этой системе. Поэтому для того, чтобы программы могли работать в других системах, в них следует использовать подпрограммы ввода/вывода, написанные самостоятельно. Предложенные ниже подпрограммы, написанные на языке ассемблера для ПЭВМ «АГАТ», я использую практически во всех своих программах как базовые. Перед программами дан пример, который на экране демонстрирует их возможности. (Листинги компилируются в ассемблере системы «Школьница» или «ИКП».)

```
    ORG $2800                    начало программы
    OBJ $2800                    адрес размещения в памяти ассемблера

    LST OFF                      выкл. вывод на экран листинга

    DSECT;

    ORG $D0
; ячейки нулевой страницы
;
BASL    DS 1 \
BASH    DS 1 /       вместе, не разъединять
BAS2L   DS 1 \
BAS2H   DS 1 /       вместе, не разъединять
;
; ниже ячейки могут быть не в нулевой странице
;
YSAV    DS 1
XC      DS 1
YC      DS 1         рабочие ячейки
```

```
XSAV    DS 1
SAV0    DS 1
INVFLG  DS 1          номер цвета
ВЫСОТА  DS 1
ДЛИТЕЛ  DS 1
CH      DS 1          коорд. X*2
CV      DS 1          коорд. Y

   DEND

TXPAGE EQU $1000                  начало текстовой страницы
;                                 (обычно бывает равно $7800)

НАЧАЛО

   STA $C70A                      включение экрана с адреса $1000
;                                 для экрана с $7800 - STA $C73E

   JSR RAMKA                      вывод рамки
   DFB 30                         ширина
   DFB 30                         высота
   DFB 0                          коорд. X
   DFB 0                          коорд. Y
   DFB 4                          цвет синий

   JSR PRINT                      вывод надписи
   DFB 0                          признак ввода координат
   DFB 18                         коорд. X
   DFB 2                          коорд. Y
   DFB 3                          цвет желтый
   ASC .** .                      вывод звездочек
   DFB 6                          цвет голубой
   ASC .ЗАГОЛОВОК.
   DFB 3                          цвет желтый
   ASC . **.                      выведена надпись: ** ЗАГОЛОВОК **
   DFB 0,4,29                     новые координаты
   DFB 7                          цвет белый
   ASC :С.ФРОЛОВ  V0.0  1993  <F34MS>:
   DFB $81                        конец вывода текста

   JSR RAMKA                      вывод второй рамки
   DFB 21,5,8,12,5                параметры рамки

   JSR PRINT                      вывод надписи во второй рамке
   DFB 0,12,15,3                  новые координаты
   ASC .НАЖМИТЕ КЛАВИШУ .
   DFB 1                          цвет красный
   ASC .РЕД.
   DFB $81                        конец вывода

METKA JSR KEY                     ожидание ввода клавиши
   CMP #$9B                       код клавиши РЕД ?
   BNE METKA

   JSR HOME..                     очистка экрана

   RTS   выход

;-----------------------------------------------------------

**—ПОДПРОГРАММЫ—**

;-Рисование рамки-*
;
```

```
; Формат вызова:
;
;JSR RAMKA
;DFB ширина рамки (кол-во пробелов внутри рамки) до 30 (62)
;DFB высота рамки до 30
;DFB координата X левого верхнего угла рамки
;DFB координата Y
;DFB цвет рамки

RAMKA PLA                       сохранение адреса возврата
    STA BAS2L
    PLA
    STA BAS2H
    LDY #1
    LDA (BAS2L),Y               считывание байта из программы
    STA XC                      (параметры)
    INY
    LDA (BAS2L),Y
    STA YC
    INY
    LDA (BAS2L),Y
    TAX
    INY
    LDA (BAS2L),Y
    PHA
    INY
    LDA (BAS2L),Y
    PHA
    TYA                         вычисление адреса обратного перехода
    SEC                         (на адрес сразу за параметрами)
    ADC BAS2L
    STA BAS2L
    BCC RAMKA.4
    INC BAS2H
RAMKA.4 PLA
    STA SAV0
    PLA
    TAY
    LDA SAV0
    JSR BASCALC1.               перенос курсора в левый верхний
    LDA YC                      угол рамки
    BEQ RAMKA.2
    LDA #' '                    или символ левого верхнего угла (для АГАТ-9)
    JSR COUT                    вывод символа
    LDA #' '                    или символ горизонтальной черты
    LDX XC
    LDY CH
    JSR POVTOR.                 вывод горизонтальной черты
    LDA #' '                    или символ правого верхнего угла
    JSR COUT                    вывод символа
    DEY
    DEY
    STY CH
RAMKA.1 INC CV                  переход на следующую строку
    JSR BASCALC..
    LDA #' '                    или символ вертикальной черты
    JSR COUT
    LDA INVFLG
    PHA
    LDA #$23                    цвет заполнения внутри рамки
    STA INVFLG
    LDX XC
    LDA #' '                    или символ заполнения внутри рамки
    LDY CH
```

```
    JSR POVTOR.                 вывод строки из символов заполнения
    PLA
    STA INVFLG
    LDA #' '                    или символ вертикальной черты
    JSR COUT                    вывод символа вертикальной черты
    DEY
    DEY
    STY CH
    DEC YC                      повторять, пока не будет выведена
    BNE RAMKA.1                 вертикальная часть рамки
    INC CV
    JSR BASCALC..
    LDA #' '                    или символ левого нижнего угла
    JSR COUT
    LDA #' '                    или горизонтальный символ
    LDX XC
    JSR POVTOR.                 вывод нижних горизонтальных символов
    LDA #' '                    или символ правого нижнего угла
    JSR COUT
RAMKA.2 LDA INVFLG              начало вывода звука
    AND #$F
    EOR #$F
    ASL A
    ASL A
    ASL A
    LDY #8
    JSR PIC                     вывод небольшого звука (можно убрать)
RAMKA.3 JMP (BAS2L) -RTS

;-------------------------------------------------------

**—Очистка экрана—**
; Точки входа:
; HOME  - очистка экрана текущим цветом
; HOME. - цвет символов в Акк.
; HOME.. - цвет символов - $23 (нормальный, желтый)

HOME.. LDA #$23                 номер цвета
HOME. STA INVFLG
HOME LDA #<TXPAGE               адрес начала текстовой (графической)
    STA BASH                    страницы
    LDX #8                      количество страниц очистки (20 для графики)
    LDY #0
    STY BASL
HOME.1 LDA #$A0                 символ очистки (=0 для графики)
    STA (BASL),Y
    INY
    LDA INVFLG     *            цвет для очистки  (* помечен код,
    STA (BASL),Y   *            который следует исключить для графики)
    INY            *            и режима 64*32
    BNE HOME.1
    INC BASH
    DEX
    BNE HOME.1
    RTS

;-------------------------------------------------------

**—Вычисление адреса—**

; Точки входа:
;
;BASCALC   -                     координаты в регистрах X и Y. Адрес точки
;                                помещается в ячейки BASL,BASH (мл.,ст.)
```

```
;BASCALC.  -                    то же, но координаты берутся из ячеек
;                               CH и CV (X и Y соответственно)
;BASCALC..  -                   то же, что BASCALC. ,но в BASL,BASH помеща-
;                               ется адрес начала строки и далее можно
;                               выводить символы подпрограммой COUT
;BASCALC1  -                    то же, что BASCALC.. ,но координаты X и Y
;                               сохраняются в ячейках CH и CV
;BASCALC1.. -                   то же, что BASCALC1, но дополнительно из
;                               Акк. сохраняется номер цвета

BASCALC1. STA INVFLG
BASCALC1 STX CH
    STY CV
BASCALC.. LDX #0
    DFB $2C
BASCALC. LDX CH
    LDY CV
BASCALC TYA
    AND #$1F
    STA BASH
    LDA #$00
    LSR BASH
    ROR A
    LSR BASH
    ROR A
    STA BASL
    LDA #<TXPAGE
    ADC BASH
    STA BASH
    TXA
    ADC BASL
    STA BASL
    RTS

;-------------------------------------------------------

**—Ввод символа—**
;
;Вводит символ с клавиатуры без курсора и без
;декодирования русской клавиатуры

KEY STA $C010          очищаем строб
    LDA $C000          берем код из буфера клавиатуры,
    BPL *-3            пока он не станет больше $80
    RTS

;-------------------------------------------------------

**—Вывод сообщения—**
;
;Выводит строку сообщения сразу после команды
;обращения до кода $81 (УПР-А или клавиша f0)
;после кода 0 в следующих двух байтах вводится
;новая координата X*2 и Y
;
; JSR PRINT
; DFB 0,12,22                   переход на позицию 6/22
; ASC .Текст сообщения.
; DFB $81

PRINT PLA                       принцип действия такой же, как
    STA BAS2L                   и подпрограммы RAMKA
    PLA
    STA BAS2H
```

```
    LDY #1
PRINT.1 LDA (BAS2L),Y
    BNE PRINT.4
    INY
    LDA (BAS2L),Y
    TAX
    INY
    LDA (BAS2L).Y
    STY YSAV
    LDY #$C6
    LDY #$D2
    LDY #$D3
    TAY
    JSR BASCALC1
    LDY YSAV
    JMP PRINT.5
PRINT.4 CMP #$81
    BEQ PRINT.2
    JSR COUT
PRINT.5 INY
    BNE PRINT.1
PRINT.2 TYA
    SEC
    ADC BAS2L
    STA BAS2L
    BCC PRINT.3
    INC BAS2H
PRINT.3 LDX XSAV
    JMP (BAS2L)

;-------------------------------------------------------

**—Вывод символа—**

;Выводит символ из аккумулятора в ячейку памяти с
;адресом (BASL)+CH - в BASL - адрес начала строки
;CH - координата по X (удвоенная)
;Если в Акк. число, меньшее $10, то оно воспринима-
;ется как номер цвета для вывода следующих символов,
;причем если номер  8, то он инверсный (-8), а
;если  8, то нормальный

COUT PHA
    CMP #$10                    символ <$10?
    BCC COUT.2                  да, это цвет, переход
    STY YSAV
    LDY CH                      коорд. X
    STA (BASL),Y                вывод
    INY
    LDA INVFLG                  цвет
    STA (BASL),Y                вывод
    INY
    STY CH                      сохранение новой коорд. X
    LDY YSAV
COUT.RTS PLA
    RTS

COUT.2 CMP #8
    BGE COUT.3
    ORA #$20
    STA INVFLG                  сохранение нормального цвета
    PLA
    RTS
```

```
COUT.3 SBC #8
    STA INVFLG                    сохранение инверсного цвета
    PLA
    RTS

;-------------------------------------------------------

**—Пик-генератор—**
;
; Выводит обычный звуковой сигнал

PIC STA ВЫСОТА
    STY ДЛИТЕЛ
    LDY #0
P2 LDX ВЫСОТА
    STA $C030
P3 DEY
    BNE P4
    DEC ДЛИТЕЛ
    BEQ P5
P4 DEX
    BNE P3
    BEQ P2
P5 RTS

;-------------------------------------------------------

*—Вывод шестнадцатеричного байта (из Акк.)

PRBYTE PHA
    JSR OBRAB.1
    JSR COUT
    PLA
PRHEX JSR OBRAB.2
    JMP COUT

OBRAB.1 LSR A
    LSR A
    LSR A
    LSR A
    DFB $2C
OBRAB.2 AND #$F
    ORA #$B0
    CMP #$BA
    BCC OBRAB.1A
    ADC #6
OBRAB.1A RTS

;-------------------------------------------------------

*—Повтор символа

; X   - количество символов
; Акк - символ
; Y   - номер цвета

POVTOR STY INVFLG
POVTOR. JSR COUT
    DEX
    BNE POVTOR.
    RTS
```

ВСЕМ РУКОВОДИТЕЛЯМ ШКОЛ, ПТУ, ТЕХНИКУМОВ,
УЧИТЕЛЯМ И МЕТОДИСТАМ,
использующим вычислительную технику в учебном процессе

Не спешите расходовать огромные средства на IBM-совместимые учебные классы. Вы можете сэкономить и найти Вашим деньгам лучшее применение!
Ваши терминальные классы КУВТ-86, УКНЦ, «Корвет» еще могут хорошо послужить и обеспечить Ваш учебный процесс на уровне современных требований. Для этого достаточно укомплектовать эти классы комплексными программно-методическими пакетами.

!!! СРЕДНЯЯ СТОИМОСТЬ ОДНОГО ПАКЕТА — 70 000 РУБЛЕЙ !!!

НАУЧНО-ТЕХНИЧЕСКОЕ ПРЕДПРИЯТИЕ

« НОВАЯ АЛЬТЕРНАТИВА »

предлагает Вам приобрести по самым умеренным ценам следующие программно-методические пакеты:

Для классов КУВТ-86 (любых типов):

1. Пакет АЛЬТЕРНАТИВА-2 (КУВТ). Назначение — изучение информатики. Состав — 7 полностью записанных дисков. Включает: новый сетевой монитор, обеспечивающий все виды пересылок, ТУРБО-ПРОЛОГ, новейшие версии текстовых, музыкальных, графических редакторов, новую систему управления базами данных, клавиатурные тренажеры, электронные таблицы, исполнители, пакет новых игровых программ, подробную техническую и методическую документацию.
2. Пакеты учебных программ РОБОТЛАНДИЯ и ЛОГО, предназначенные для изучения основ информатики с младшими школьниками. Являются полными аналогами пакетов, хорошо себя зарекомендовавших на классах ЯМАХА.
3. Система программирования ТУРБО-ПАСКАЛЬ, подобна используемой на IBM PC.

Для класса УКНЦ (любых типов):

Пакет АЛЬТЕРНАТИВА-2 (УКНЦ). Назначение — изучение информатики в старших классах. Состав — 5 полностью записанных дисков. Включает: файловый монитор типа NORTON, сетевой монитор, обеспечивающий все типы пересылок по сети, текстовые, музыкальные, графические редакторы, систему управления базами данных, систему ТУРБО-ПАСКАЛЬ с графикой, транслирующую в коды, ТУРБО-ПРОЛОГ, электронные таблицы с графическим выводом, пакет игровых программ, подробную техническую и методическую документацию.

Новинки:

1. Учебная система программирования БЕТА-ЛОГО — аналог системы LOGOWRITER для классов PS/2.
2. Учебная система программирования АЛТ-ПАСКАЛЬ — аналог системы TURBOPASCAL v.6 для IBM.

Для классов «Корвет» (любых типов):

Пакет АЛЬТЕРНАТИВА-2 («Корвет»). Назначение — обеспечение изучения информатики в старших классах. Состав — 5 полностью записанных дисков. Включает: сетевой монитор, обеспечивающий быструю пересылку по сети операционной системы CP/M и все виды работы с сетью; текстовый, музыкальный, графический редакторы; систему управления базами данных, систему ПАСКАЛЬ с графикой, транслирующую в коды; электронные таблицы, систему ТУРБО-ПРОЛОГ, пакет игровых программ, подробную техническую и методическую документацию.

Для классов БК:

Программный пакет АЛЬТЕРНАТИВА-2 (БК-0011 всех типов). Состав: сетевой монитор, текстовые редакторы, электронные таблицы, базы данных, ТУРБО-ПАСКАЛЬ, игры, техническая и методическая документация.

Условия поставки:

Высылка по почте. Форма оплаты — любая (безналичная предоплата по счету, наложенный платеж и т. д.).

Любой, приславший нам заявку, впредь регулярно и безвозмездно получает проспекты наших новых разработок.

Наша ставка — на высокое качество и большое число заказчиков, а не на высокие цены.

Наши пакеты — это то, что Вам доступно и так необходимо сегодня!

Не теряйте драгоценного времени. ВЫШЛИТЕ НАМ ЗАЯВКУ СЕГОДНЯ ЖЕ!

Наш адрес: 656057, г. Барнаул, а/я 2513. НОВАЯ АЛЬТЕРНАТИВА.
Директор Гриценко А. Н.

КЛУБ «КОРВЕТ»

В. М. Нелин,
Армавирский педагогический институт

АДАПТАЦИЯ DBASE II V. 2.5 К КЛАВИАТУРАМ НЕ QWERTY ТИПА

Первоначальный вариант широко распространенной СУБД dBASE — dBASE II предназначался для 8-разрядных компьютеров, работающих под операционной системой CP/M. К типичным представителям этой группы относятся такие КУВТ, как «Ямаха» и «Корвет».

Основной файл (dBASE.COM — 19584 байт) ориентирован на работу с клавиатурой типа QWERTY. При использовании клавиатур иного типа, например клавиатур «Ямахи» или «Корвета», возникает масса неудобств: не работают клавиши управления курсором (заменяющие эти клавиши нажатия Ctrl-X, Ctrl-E, Ctrl-D, Ctrl-S неудобны из-за некомпактного расположения клавиш X, E, D, S на клавиатурах не QWERTY типа), клавиши RETURN, INS, DEL, словом, работать с клавиатурой оказывается практически невозможно. Стоит отметить, что даже в случае клавиатуры QWERTY нормальная работа возможна лишь в том случае, когда клавиши управления курсором ↓, ↑, →, ← соответствуют нажатиям Ctrl-X, Ctrl-E, Ctrl-D, Ctrl-S.

Детали дальнейшего рассуждения относятся к КУВТ «Ямаха», хотя могут быть распространены и на другие КУВТ, в частности КУВТ «Корвет» (при этом необходимо учесть отличие управляющих ASCII-кодов «Ямахи» и «Корвета»).

В файле dBASE.COM адреса подпрограмм, на которые передается управление при выполнении управляющих кодов Ctrl-A, Ctrl-B, Ctrl-C и т. д., сведены в таблицу, расположенную по адресам 02BE—02FD, т. е. занимающую 64 байта (считается, что файл dBASE.COM загружен в отладчик с адреса 0100). Для работы с этой таблицей файл dBASE.COM достаточно загрузить в один из отладчиков, работающих под CP/M, например DDT или SID (для «Ямахи» — ZSID или DBG).

Первоначальный вид таблицы таков:

```
02B0: .. .. .. .. .. .. .. .. .. .. .. .. .. .. 7A 02
02C0: 4E 03 44 06 54 05 A4 03 4E 03 B0 05 78 04 77 03
02D0: 7A 02 B0 05 4E 03 A4 03 6A 05 0E 06 7A 02 7A 02
02E0: FE 02 5F 05 77 03 1B 06 E2 05 5A 04 28 06 B0 05
02F0: 22 05 31 06 7A 02 7A 02 7A 02 7A 02 7A 02 .. ..
```

Выполнение нажатий Ctrl-A, Ctrl-B, Ctrl-C и т. д. приводит, соответственно, к передаче управления по адресам 27A, 34E, 644 и т. д.

Для нормальной работы клавиатуры «Ямахи» достаточно изменить несколько адресов этой таблицы (изменяются адреса, соответствующие ASCII-кодам клавиш RETURN, INS и т. п.). Ниже приводится вид откорректированной таблицы:

```
02B0: .. .. .. .. .. .. .. .. .. .. .. .. .. .. 7A 02
02C0: 4E 03 44 06 54 05 A4 03 4E 03 B0 05 78 47 77 03
02D0: B0 05 B0 05 31 06 44 06 B0 05 0E 06 5F 05 6A 05
02E0: FE 02 5A 04 77 03 1B 06 E2 05 5A 04 28 06 B0 05
02F0: 22 05 31 06 7A 02 A4 03 77 03 4E 03 B0 05 .. ..
```

Нам пишут

К описанным в ИНФО № 2 за 1992 г. возможностям принтера CPF-H80 можно добавить следующее. Строка

LPRINT CHR$(27)CHR$(106)CHR$(число)

позволит осуществить возврат на несколько строк назад (количество строк зависит от числа).

Используя программу печати PRTOOLS.COM, можно вставлять прямо в текст небольшие графические изображения.

В. Ф. Казначейский (г. Караганда, СШ 29)

ДОСКА ОБЪЯВЛЕНИЙ

Программы для «Агата». Вышлем каталог и демонстрационный диск.
630117, Новосибирск, а/я 427.

ОТ РЕДАКЦИИ

В статье О. Н. Марковой «Обработка массивов на языке БЕЙСИК» в листингах примеров по не зависящим от редакции техническим причинам пропущен знак «¤». Следует читать:

Данные в массиве
Листинг 3, строки 10, 30 и 60: после имени массива A¤.
Листинг 4, строки 20 и 30: после имени переменной X¤ и функций STR¤ и MID¤.

Способы заполнения массивов
Листинг 3, строки 20, 40 и 90: после имени массива A¤.

Задачи на обработку символьного массива
Листинг 1а, строки 20, 40 и 50: после имени массива A¤ и функции MID¤.
Листинг 1б, строки 20—50 и 80: после имени переменной A¤ и функции MID¤.
Листинг 2, строки 20, 40 и 120: после имени массива A¤.

Кроме того, в листинге 3 (подзаголовок *«Задачи на сортировку массива»*) в строке 150 пропущен знак «<». Следует читать:

```
150 IF A(J,Y)<A(M,K) TH 180 EL 170
```

НАПЕЧАТАНО В 1994 г.

ИНФОРМАТИКА В МЛАДШИХ КЛАССАХ

ВСТУПИТЕЛЬНЫЕ ЭКЗАМЕНЫ ПО ИНФОРМАТИКЕ

ОЛИМПИАДА ПО ИНФОРМАТИКЕ

НОВЫЕ ПРОЕКТЫ

ИНТЕРВЬЮ

ТОЧКА ЗРЕНИЯ

ПРОФЕССИОНАЛЬНЫЙ КОМПЬЮТЕР

ЯЗЫКИ ПРОГРАММИРОВАНИЯ

ЛЮДИ, ИДЕИ, РЕШЕНИЯ

КЛУБ БК

КЛУБ УКНЦ

КЛУБ «КОРВЕТ»

ИСТОРИЯ

НЦПСО АИСТ

В 1985 году группой программистов и учителей информатики были начаты работы по созданию компьютерных программ для сферы образования.

1987

Накопленный за два года опыт работы позволил в 1987 году создать НЦПСО (Научный Центр Программных Средств Обучения) - структуру, призванную решать задачи информатизации среднего и среднего специального образования г. Москвы.

С 1985 по 1990 год НЦПСО разработано более 100 программ для различных типов отечественной вычислительной техники. С самого начала своей деятельности НЦПСО занимается установкой и технической поддержкой компьютерных классов, а с 1993 - и поставкой мультимедийных комплексов в школы страны. Сегодня более 70% школ Москвы (более 1 000 школ) - постоянные партнеры НЦПСО. Каждая десятая школа России (более 10 000 школ) использует программные средства, разработанные в НЦПСО.

1991

В 1991 году совместно со шведской фирмой IST AB была создана новая фирма - AIST AB (Advanced Instructional Software Trading AB). Основой деятельности AIST AB стала разработка программного обеспечения в области динамической компьютерной графики.

За три года фирмой было создано 6 программных пакетов для IBM-совместимых компьютеров, среди которых наиболее известны "TeachCAD"(1991), "MultiVision"(1992), "MediaMaster"(1994).

Ориентация на международный рынок изменила подход к разработке программных средств. Стратегия деятельности на Российском образовательном рынке в 1991-1993 годах оставалась, в целом, неизменной.

1994

В конце 1993 - начале 1994 года, в связи с выработкой концептуально новых подходов к проблемам информатизации образования, а также возросшей необходимостью активной маркетинговой деятельности на отечественном и зарубежном рынках, произошла реорганизация структуры НЦПСО-АИСТ.

В рамках НЦПСО создано специальное подразделение "Проект "Образование", обеспечивающее одну из наиболее перспективных разработок в мультимедиа технологиях обучения - комплекс MultiVision Pro.

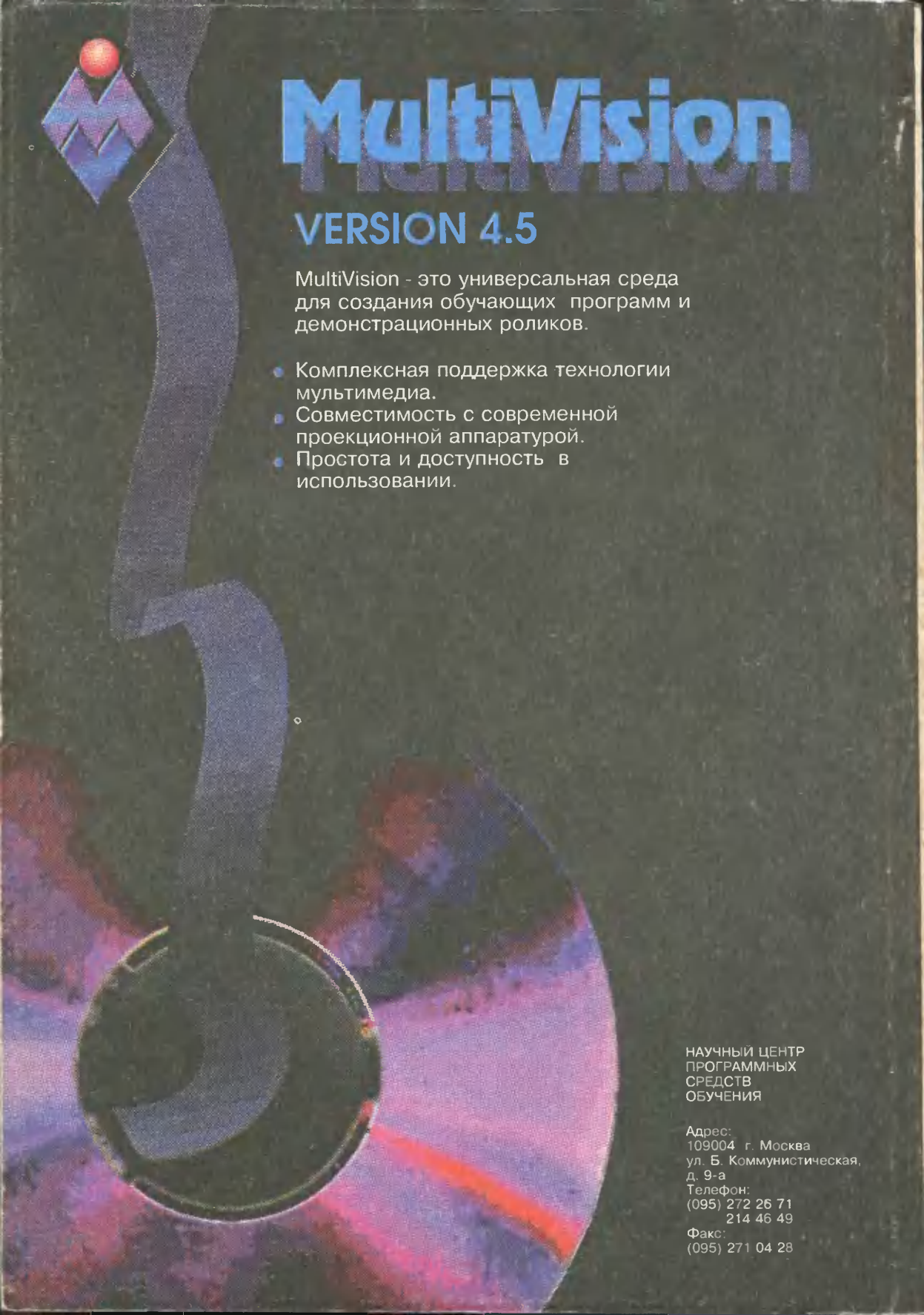
Образована компания НЦПСО-комплекс, осуществляющая поддержку и версификацию программных продуктов, разработанных в AIST AB и НЦПСО. Планируется открытие служб "Горячая линия" и BBS.

Начала самостоятельную деятельность компания AIST Inc. (Agency of Informative Systems and Technologies, Inc.). Основные направления - современные средства компьютерной презентации (программы презентационной и демонстрационной графики, проекционное оборудование), маркетинг программных продуктов фирм НЦПСО и AIST AB.

ОСНОВНЫЕ ПРОДУКТЫ

- Программное обеспечение для образования
- Комплексы мультимедиа для образования
- Проекционное оборудование
- Программное обеспечение для презентаций и видеомонтажа

	НЦПСО	AIST AB	НЦПСО комплекс	AIST Inc.
☎	272 26 71	229 67 06	214 46 49	271 29 89
FAX	271 04 28	229 76 53	271 04 28	271 04 28


MultiVision
VERSION 4.5
MultiVision - это универсальная среда для создания обучающих программ и демонстрационных роликов.
• Комплексная поддержка технологии мультимедиа.
• Совместимость с современной проекционной аппаратурой.
• Простота и доступность в использовании.
НАУЧНЫЙ ЦЕНТР ПРОГРАММНЫХ СРЕДСТВ ОБУЧЕНИЯ
Адрес:
109004 г. Москва
ул. Б. Коммунистическая, д. 9-а
Телефон:
(095) 272 26 71
214 46 49
Факс:
(095) 271 04 28