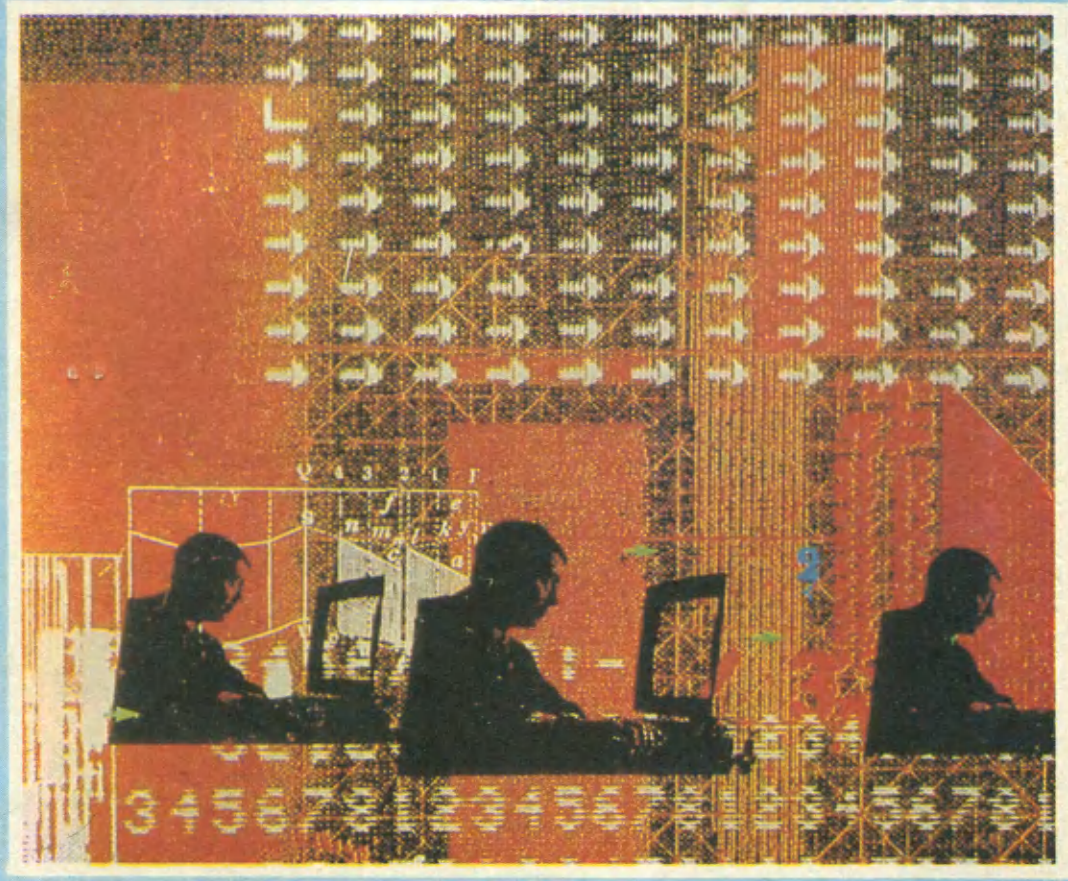
ISSN 0234—0453

ИНФОРМАТИКА И ОБРАЗОВАНИЕ

4 1989



Этому рулону бумаги
суждено превратиться
в распечатку,
на которой суммированы
мнения студентов о своих
преподавателях...

О новых формах
работы МОПИ
им. Н. К. Крупской
по
компьютеризации
образования
читайте
в интервью
с проректором
по научной
работе института
Я. А. Вагрaменко

Научно-методический журнал
Государственного комитета СССР
по народному образованию

Издается с августа 1986 г.
Выходит шесть раз в год
Москва «Педагогика» 1989

ИНФ
ИНФОРМАТИКА И ОБРАЗОВАНИЕ

4 '89
июль — август

OldPC.su
7002
музей компьютеров

Содержание

Общие вопросы

Методика обучения

КВТ

Педагогический опыт

Внеклассная работа

Точка зрения

ЭВМ в народном хозяйстве

Интервью номера

Информация

Веселый урок

Обложка *Э. Бажилина*
В оформлении номера принимали участие: *Э. Бажилин, Н. Горбунова, С. Расторгуев*

Главный редактор академик
В. А. МЕЛЬНИКОВ

Редакционная коллегия
И. Н. АНТИПОВ
В. Н. АФАНАСЬЕВ
И. М. БОБКО
Г. В. ГОДЖЕЛЛО
С. А. ЖДАНОВ
Б. В. ЛОМОВ
Ю. В. ЛУИЗО
(зам. главного редактора)
Н. Г. МЕЛЬДИАНОВ
И. С. ОРЕШКОВ
О. К. ПАВЛОВА
А. Ю. УВАРОВ
А. И. ФУРСЕНКО
В. О. ХОРОШИЛОВ
К. В. ШЕХОВЦЕВ
(редактор отдела)

Редактор отдела *А. Кравцова*
Научный редактор *Т. Драгныш*
Зав. редакцией *Н. Игнатова*
Художественный редактор *Л. Розанова*
Корректор *М. Суворова*

Сдано в набор 22.05.89. Подписано в печать 11.07.89. А 03611.
Формат 70×100/16. Бумага офсетная № 1. Печать офсетная. Усл. печ. л. 10,40. Усл.-кр. отт. 42,88. Уч.-изд. л. 12,70. Тираж 83 115 экз. Заказ 1283. Цена 60 коп.

Издательство «Педагогика» Академии педагогических наук СССР
и Государственного комитета СССР
по делам издательств, полиграфии и книжной торговли

Адрес для переписки: 107005, Москва, Лефортовский пер., 8.
Адрес редакции: Студенческая ул., 37.
Телефон редакции: 249-97-77

Ордена Трудового Красного Знамени Чеховский полиграфический комбинат Государственного комитета СССР по делам издательств, полиграфии и книжной торговли
142300, г. Чехов Московской обл.

© «Педагогика», «Информатика и образование», 1989

Научная программа психолого-педагогических исследований по проблеме «Новые технологии обучения и развития детей»

Программа основана на проекте психологической концепции проектирования новых технологий обучения и развития детей, опубликованной в № 3 за 1989 г. нашего журнала (оба документа разработаны одним авторским коллективом). Исследования по этой программе сформируют одно из направлений работы Всесоюзного центра по компьютеризации образования, создаваемого в рамках новой структуры АПН СССР, утвержденной на общем собрании академии 19.04.1989 г.

Деятельностный подход к обучению определяет перспективное направление психолого-педагогических исследований и разработок, ориентированных на систему непрерывного образования и направленных на создание и психологическое обеспечение информационных, принципиально новых по содержанию и формам организации технологий учебно-воспитательного процесса. Программа этих исследований должна включать разработку теоретико-практических основ построения учебной деятельности и создание учебных предметов, основанных на использовании ЭВМ. Необходимо будет провести специальный анализ объема и особенностей использования компьютера в связи с возрастом учащихся.

Так, уже сейчас считается доказанной возможность обучать детей информатике начиная с 12 лет, т. е. с того возраста, когда у ребенка формируются структуры формальных операций, а также использовать компьютеры в процессе обучения детей 6—12 лет, если это обучение не ограничивает творческих возможностей и активности самого ребенка, опирается на его игровую деятельность, развивает совместные, групповые и коллективно-распределенные взаимодействия взрослого и детей, самих детей. Особым направлением разработок станут также методы исследования успешности обучения и развития учащихся в условиях использования компьютерных систем на разных возрастных этапах школьного детства.

Предварительные психолого-педагогические исследования, выполненные в Советском Союзе в русле психологической теории деятельности человека и теории учебной деятельности, показывают, что наиболее оптимальной с возрастной точки зрения следует считать трехступенчатую модель использования компьютера в школе.

1 — младшая школа: развитие наряду с другими базовыми культурно-техническими навыками элементарной компьютерной грамотности, т. е. таких чисто пользовательских навыков и умений работать с машиной, которые обеспечивают ребенку возможность освоения знакового пространства клавиатуры и точного соотнесения элементов этого пространства с результатами собственных воздействий, появляющихся на экране компьютера. Относительная свобода программирования своих действий не должна требовать от ребенка сложных знаковых переозначений, большой глубины планирования, запоминания длинных последовательностей шагов и т. д. Вместе с тем действия ребенка с символической предметностью компьютера должны быть построены таким образом, чтобы они исходно содержали в себе возможность возврата к конкретно-практическим действиям с реальной предметностью и составляли с ней единое целое. Содержание компьютерных программ, рекомендуемых для этого возраста, не стоит строго привязывать к дифференцированным школьным дисциплинам, оно должно носить интегральный характер и обеспечивать ребенку в процессе решения задач выбор по различным критериям. Важное значение для детей этого возраста будут иметь игровые формы организации учебной деятельности. Эта деятельность будет строиться на основе широкого использования компьютерных сетей, обеспечивающих распределенный характер действий учителя и учащихся, а также самих учащихся, коммуникацию и сотрудничество в процессе совместного решения проблем и задач.

2 — средняя школа: освоение новых областей знания по принципу совместного со взрослым и сверстниками конструирования и описания «школьных» научных дисциплин. Компьютер широко привлекается в этом возрасте не только в качестве специализированной дисциплины («информатика и вычислительная техника»), но и как полуфункциональный инструмент для редактирования текстов, проведения лабораторных экспериментов, практикумов, разработок совместных проектов и т. д. На этой стадии школьники осваивают и используют такие основные назначения компьютера, как информационный источник, вычислитель, текстовый редактор, средство моделирования. Обучение программированию строится не как формирование профессионального навыка, а как умение содержательно-логически анализировать и структурировать (алгоритмизировать) определенные области знания («предметные миры»), что способствует развитию таких способностей детей к самодеятельности и самообучению, как рефлексия, анализ, планирование, воображение.

3 — старшая школа: благодаря широкому ознакомлению со всеми сферами профессиональной деятельности человека и с помощью компьютерного имитационного моделирования школьники должны научиться строить и управлять наиболее важными производственными и общественными видами деятельности людей: уметь производить экологическое и демографическое моделирование, строить аналоги конкретных промышленных производств и рассчитывать последствия их функционирования, проектировать разработку и эксплуатацию недр, проводить несложное математическое моделирование конкретных локальных проблем и предлагать их решение, моделировать ситуации выборов органов власти, развивать свое правовое и политическое сознание и т. д. На этой стадии компьютеры используются также и для профориентационной работы с учащимися, а при росте объемов самостоятельной работы — для развития навыков самообразования и переобучения.

Научная программа

Использование компьютера в средней школе позволяет поставить в качестве наиболее актуальных следующие проблемы психолого-педагогических исследований:

соотношение игровых, учебных и профессионально-ориентированных видов деятельности детей, основанных на использовании компьютерных технологий обучения, трансформация и раз-

витие этих видов деятельности в зависимости от возраста и индивидуальных особенностей учащихся;

соотношение (преемственность и различие) классных и внеклассных форм организации игровой и учебной деятельности, опосредствованной использованием компьютерных технологий обучения;

соотношение коллективных (фронтальных), групповых и индивидуальных форм обучения детей с помощью компьютера;

анализ возможностей использования компьютера в обучении детей с разным уровнем психического развития, осуществление дифференцированного подхода;

соотношение обучающих, развивающих и контрольно-корригирующих действий в процессе обучения, оценка влияния компьютерных технологий обучения на развитие детей;

подготовка учителей к использованию новых технологий обучения. Целью выполнения намеченной программы является доведение фундаментальных разработок в области психологического обеспечения новых технологий обучения до состояния тиражируемых в массовом порядке «технологических педагогических минимумов», без которых ни одна школа не должна принимать и эксплуатировать вычислительную технику. В такой пакет необходимо включать программное обеспечение, психолого-педагогические рекомендации по организации обучения с использованием компьютера и широкий набор методических разработок и материалов.

Научная продукция

В ходе исследований, выполняемых по проблеме «Новая технология обучения и развития детей» намечается разработать:

психолого-педагогические основы и принципы построения компьютерных технологий обучения и развития детей;

новые предметно-развивающие учебные среды и образцы новых учебных предметов, основанных на использовании компьютерных средств;

компьютерные модели совместной и индивидуальной учебной деятельности для учащихся начальной, средней и старшей школы;

компьютерные развивающие игры для формирования общих и специальных способностей у детей разных возрастов, особенно для детей младшего школьного возраста (6—10 лет);

компьютерные диагностические методики контроля и оценки влияния новых технологий обучения на развитие личности и познавательных процессов школьников;

рекомендации разработчикам педагогических программных средств;

рекомендации учителю по практической реализации новых технологий обучения;

рекомендации работникам психологической службы в школе и школьным психологам по компьютерной диагностике и использованию компьютерной коррекции учебной деятельности и развития учащихся.

Таким образом главным результатом исследований по указанной проблеме станет психологически обоснованная технология обучения, включающая применение компьютеров в преподавании учебных предметов и реализованная на примере компьютерных моделей совместной и индивидуально-организованной учебной деятельности учащихся. В рамках этой технологии будут получены важные результаты о психологических закономерностях обучения и возрастного развития ребенка-школьника в условиях использования современных информационных средств как важнейшего звена непрерывного образования.

Итоги исследований к 2000 г. будут представлены:

а) в теории проектирования компьютерных коммуникативно-ориентированных моделей учебной деятельности и системе психологических требований к разработке компьютерных учебных предметов;

б) в системе компьютерных моделей совместной и индивидуальной учебной деятельности для предметов начальной, средней и старшей школы, а также для внешкольных занятий учащихся (клубная деятельность);

в) в виде образцов компьютерных учебных курсов, включая интегральные курсы;

г) в системе научно обоснованных методов контроля за успешностью обучения и развития школьников в условиях использования новых технологий учебно-воспитательного процесса.

Разработка и психологическое обеспечение новых технологий обучения должны проводиться на основе широкой экспериментальной апробации создаваемых компьютерных моделей на базе «школы-лаборатории» и в ряде опытных школ, действующих при Всесоюзном центре по компьютеризации образования (1989—1995) с последующим распространением накопленного опыта в массовые школы страны (1995—2000).

ЧТО МОЖЕТ ЭВМ?

Сто томов за три секунды

Важнейшая характеристика компьютера — объем его оперативной памяти. Чем больше емкость оперативного запоминающего устройства (ОЗУ), тем больше программ можно скопировать в него с относительно медленного диска или неповоротливой магнитной ленты. Это значит, что время реакции программной системы, которое складывается из периода поиска необходимой программы на диске, переписи ее в ОЗУ и полезного времени работы программы, заметно сократится. Да и сами программы можно сделать удобнее, если не экономить память на их сервисных функциях.

Когда же компьютер используется для писательской или редакторской работы, большая оперативная память позволит разместить в ней целиком весь обрабатываемый текст, обеспечив таким образом быстрый доступ к любой его части. И вовсе не смогут обойтись без большой памяти компьютеры, на которых одновременно будут работать сразу несколько программистов, ведь каждому из них требуется выделить необходимый ему объем.

Западные фирмы, производящие компьютеры и их компоненты, предлагают владельцам персональных ЭВМ плату, на которой размещено ОЗУ емкостью в 16М байт. Плата похожа на папку, в которой лежит не более полусотни листов бумаги. Само же ОЗУ может вместить более 10 тыс. страниц печатного текста или сто томов по сто листов в каждом. Даже появление микрофильмов, которое в свое время произвело своеобразную революцию в архивном деле, не позволило достичь такого колоссального сжатия информации. Кроме того, в плату встроена система обнаружения и коррекции некоторых ошибок, допущенных при передаче информации. Время, необходимое для обращения к одному элементу такой памяти, составляет всего 220 наносекунд. Это значит, что просмотреть все сто имеющихся в ОЗУ томов, чтобы найти нужное слово, можно всего за 3,5 с.

Лечение или профилактика?

Даже в Советском Союзе едва ли не все газеты сообщили о «вирусной эпидемии», поразившей в ноябре 1988 г. компьютерную информационную сеть Internet. Происшествие это нельзя, однако, считать чрезвычайным — рано или поздно нечто подобное должно было случиться. Сеть Internet имеет две особенности, делающие ее весьма уязвимой для компьютерных вирусов: ориентация на общедоступность и высокая пропускная способность. Последнее обстоятельство позволяет вирусу распространиться очень широко, прежде чем он будет замечен; тщательная же защита путем кодирования передаваемой информации и усложнение системы паролей сделали бы использование сети более дорогим и менее удобным для массового пользователя.

Конечно, в некоторых случаях дорогостоящие меры защиты являются оправданными, например в сети Milnet, используемой Министерством обороны США для передачи секретной информации. Не могла обойтись без таких мер и фирма IBM, европейская часть сети которой базируется на телефонных линиях связи, частично принадлежащих ее конкурентам в области производства вычислительной техники.

Итак, какой же путь избрать создателям новых сетей — дорогостоящую профилактику или риск «острого гриппа»? Пока соединить достоинства доступности и защищенности не удается, видимо, будут использоваться оба пути. Кроме того, так ли опасен риск? Обычные сбои компьютеров, которые случаются во всем мире и которые не являются поводом для сенсационных газетных сообщений, могут нанести не меньший ущерб. И это уже не говоря о крупных авариях, подобных пожару на коммутационном узле телефонной сети в Иллинойсе, повредившему местный компьютер и нанесшему абонентам и местным властям ущерб в 500 млн. долларов.

А. ГЕЙН, Е. ЛИНЕЦКИЙ, М. САПИР, В. ШОЛОХОВИЧ

Информатика: исполнители и алгоритмы

На уроке, посвященном понятию исполнителя алгоритмов, учитель должен донести до учащихся следующие идеи. Во-первых, человек далеко не единственный исполнитель алгоритмов. Во-вторых, любой исполнитель состоит из устройства управления и «рабочего инструмента» (понимание этого достигается рассмотрением двух-трех примеров исполнителей). В-третьих, каждый исполнитель алгоритмов обладает ограниченным набором допустимых действий (описать исполнителя — значит указать его допустимые действия). В-четвертых, для решения одних и тех же задач исполнители с более «бедным» набором допустимых действий требуют более сложных и подробных алгоритмов. В-пятых, разные классы задач требуют разных наборов допустимых действий, разных исполнителей.

На следующем уроке вводятся основные действия исполнителя Вычислитель (см. [5]) и подробно разбирается смысл понятий «переменная» и «присваивание». При этом учащиеся получают все необходимые знания для выполнения лабораторной работы, на которой они завершают решение задачи о равноускоренном движении — составляют соответствующую программу и проводят вычислительный эксперимент, используя машинную имитацию исполнителя Вычислитель. Во время этого эксперимента учащиеся увеличивают число отрезков разбиения интервала времени, следя за тем, как сближаются полученные значения пути и насколько эти значения отличаются от «истинного» значения, вычисленного по известной формуле из курса физики.

Начнем разбор задач к § 5 «Исполнители алгоритмов» из [1]. При решении этих задач следует обратить внимание учащихся на то, что каждый исполнитель алгоритмов обладает строго определенным набором действий. Умение оперировать ограниченным набором действий закрепляется решением задач 1, 2 и 3.

Задача 1. Исполнитель умеет: умножать число на 2; увеличивать число на 1.

1а). Составьте для этого исполнителя алгоритм получения числа 100 из единицы.

1б). Сколько действий в самом коротком из таких алгоритмов?

Решение

1а). Сначала следует разобрать более простые задачи, чтобы учащиеся освоились с новым исполнителем. Например, составить для этого исполнителя алгоритм получения чисел 4 и 5.

Школьники легко составляют кратчайшие алгоритмы. Один из алгоритмов получения числа 4.

Увеличить число на 1. Умножить число на 2.

Продолжение. Начало см.: Информатика и образование. 1989. № 2, 3.

По первой команде происходит сложение 1+1, получается число 2. По второй команде полученное число увеличивается в 2 раза, получается число 4.

Алгоритм получения числа 5.

Увеличить число на 1. Умножить число на 2. Увеличить число на 1.

После этого можно приступить к составлению алгоритма получения числа 100. При этом лучше всего идти не от 1 к 100, а обратно — от 100 к 1 (вместо действия «Умножить число на 2» и «Увеличить число на 1» надо использовать действия «Разделить число на 2» и «Вычесть из числа 1»). Один из всевозможных путей получения 1 из 100:

$$100 \xrightarrow{/2} 50 \xrightarrow{/2} 25 \xrightarrow{-1} 24 \xrightarrow{/2} 12 \xrightarrow{/2} 6 \xrightarrow{/2} 3 \xrightarrow{-1} 2 \xrightarrow{/2} 1.$$

8 Переворачивая этот алгоритм «с головы на ноги», получаем ответ.

Умножить число на 2. Увеличить число на 1. Умножить число на 2. Умножить число на 2. Умножить число на 2. Увеличить число на 1. Умножить число на 2. Умножить число на 2.

1б). В самом коротком алгоритме получения 100 из единицы содержится 8 действий (такой алгоритм приведен выше).

Задача 2. Исполнитель умеет из любой дроби a/b получать любую из дробей $(a-b)/b$, $(a+b)/b$ и b/a. Как получить из дроби 1/2 дробь 1/4? А как получить 67/91?

Решение

Лучше всего решать эту задачу, как задачу 1а), с конца, т. е. не от 1/2 переходить к 67/91, а, наоборот, от 67/91 к 1/2.

Задача получения из дроби 1/2 дроби 1/4 более легкая. Ее решение:

$$\frac{1}{2} \to \frac{2}{1} \to \frac{3}{1} \to \frac{4}{1} \to \frac{1}{4}.$$

Превращение дроби 1/2 в 67/91 производится аналогично.

Задача 3. На столе лежат две двухкопеечные монеты и три пятака (см. рис. 9).

9 

Исполнитель может одновременно перемещать две соседние монеты разного достоинства в начало или конец цепочки либо на освободившееся в результате таких перемещений место в середине цепочки (не раздвигая и не переворачивая монеты). Например, перенеся вторую и третью монеты к правому концу цепочки, получим такое расположение монет, как на рис. 10.

10 

При этом исполнитель не может раздвигать монеты или менять их местами. Составьте для этого исполнителя алгоритм преобразования исходной цепочки монет в такую цепочку, как показано на рис. 11.

Указание к решению

Как и в предыдущей задаче, основная идея заменить прямую задачу обратной, т. е. искать путь от рис. 11 к рис. 9.

11 

Следует предупредить возможные ошибки: 1) ученики часто используют действия, не указанные в условии задачи (например, в задаче 2 умножают числитель и знаменатель дроби на одно число, а в задаче 3 перемещают пары монет одинакового достоинства); 2) в качестве исходного числа в первой задаче ученики берут не 1.

Эти задачи имеют еще одну особенность: соответствующие алгоритмы можно (и нужно) представлять себе *как пути движения от исходного положения к конечному положению*. При «прокладке» этих путей оказывается более простым движение не от исходного положения к конечному, а наоборот — от конечного положения к исходному. Такой подход акцентирует внимание учащихся на анализе путей достижения результата, что является пропедевтикой материала следующего параграфа — «Дости-

жимые цели». Поэтому при обсуждении решения этих задач целесообразно задавать учащимся вопросы типа: какую цель мы должны достичь?, каков путь для достижения цели? и т. п.

Большую методологическую нагрузку несет задача 4. Здесь фактически впервые проявляются идеи вспомогательного алгоритма и пошаговой детализации: а именно алгоритм, составленный при решении задачи а), должен рассматриваться как вспомогательный при решении задачи б). Решая задачу а), мы как бы обучаемся новому действию — нахождению центра тяжести прямоугольника, а затем применяем его при решении более сложной задачи. Кроме того, пункт а) является, по сути дела, подсказкой для пункта б). Ученики должны догадаться, что для нахождения центра тяжести фигуры из пункта б) ее надо разбить на два прямоугольника.

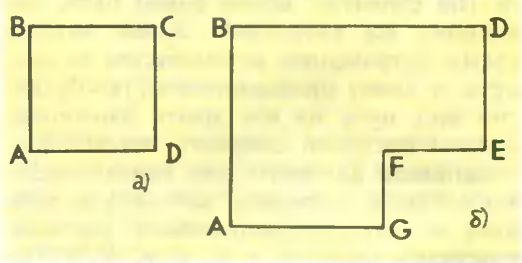
Задача 4. Составьте алгоритм нахождения центра тяжести фигур на рис. 12а и 12б с помощью карандаша и линейки.

Решение

Поскольку решение задачи 4а) очевидно (надо провести диагонали прямоугольника и найти точку их пересечения), приведем решение задачи 4б), используя «сложное действие» нахождения центра тяжести прямоугольника.

Воспользуемся известным свойством: центр тяжести фигуры, составленной из двух фигур, лежит на отрезке, соединяющем центры тяжести этих фигур.

12



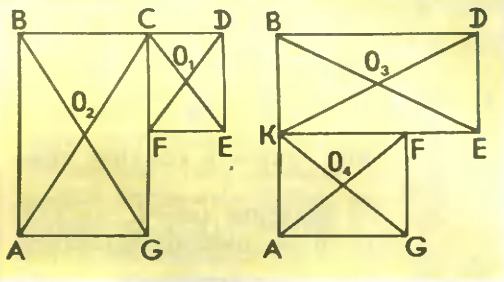
Данную фигуру (рис. 12б) можно двумя способами разбить на два прямоугольника (см. ниже). Каждое такое разбиение даст нам отрезок, на котором лежит центр тяжести всей фигуры. Значит, точка пересечения этих отрезков — искомая.

Полный алгоритм решения.

Продолжим прямую GF до пересечения с BD в точке C. Найдем центр тяжести прямоугольника CDEF — точку O_1. Найдем центр тяжести прямоугольника ABCG — точку O_2 (см. рис. 13). Соединим точки O_1 и O_2 прямой O_1O_2.

Продолжим прямую EF до пересечения с AB в точке K. Найдем центр тяжести прямоугольника KBDE — точку O_3. Найдем центр тяжести прямоугольника AKFG — точку O_4 (см. рис. 14). Соединим точки O_3 и O_4 прямой O_3O_4. Найдем точку пересечения Q прямых O_1O_2 и O_3O_4.

13 14



Задача 5 относится к типу задач «Что произойдет в результате выполнения алгоритма?», а задачи 6 и 7 — к типу задач «Найти ошибку в алгоритме». При этом в задачах 5 и 6 использованы допустимые действия Вычислителя, и ошибка кроется в самом алгоритме, а в задаче 7 ошибка состоит как раз в том, что использовано действие, привычное для школьника, но недопустимое для Вычислителя. Их решение оформляется в той же форме, что и решения других задач соответствующих типов (см. § 2.1).

Задача 5. Какие задачи решит Вычислитель, выполнив следующие алгоритмы?

5а). Запросить a, d, n. Присвоить S значение $(a+d\,(n-1))\,n/2$. Сообщить S.

5б). Запросить b, q, n. Присвоить S значение $\frac{b\,(q^n-1)}{q-1}$. Сообщить S.

Решение

5а). Выполнив этот алгоритм, Вычислитель найдет сумму первых n членов арифметической прогрессии, разность которой d, первый член a.

5б). Выполнив этот алгоритм, Вычислитель найдет сумму первых n членов геометрической прогрессии, знаменатель которой q, а первый член b.

Задача 6. Злоумышленник поменял местами действия в алгоритме вычисления среднего арифметического квадратов трех чисел.

Присвоить a значение $(a^2+b^2+c^2)/3$. Запросить a, b, c. Сообщить «Среднее арифметическое квадратов равно». Сообщить a.

Восстановите правильный порядок действий.

Решение

Выполняя первое действие предложенного алгоритма «Присвоить a значение $(a^2+b^2+c^2)/3$», исполнитель столкнется с тем, что значения переменных a, b и c не определены, поэтому на первое место нужно поставить команду:

Запросить a, b, c.

А теперь нужно записать команду присваивания:

Присвоить a значение $(a^2+b^2+c^2)/3$.

Третью и четвертую команды оставляем на месте:

Сообщить «Среднее арифметическое квадратов равно».

Сообщить a.

Задача 7. Исправьте следующий алгоритм решения уравнения $x^2-2x-14=0$.

Присвоить d значение 2^2+4-14. Присвоить p значение $\sqrt{d}$. Присвоить x значение $(2+p)/2$. Сообщить «Корни уравнения равны». Сообщить первое значение x. Сообщить второе значение x.

Решение

В третьей команде сделана попытка присвоить переменной x сразу два различных значения: $(2\pm p)/2$ и $(2-p)/2$. Это так же невозможно, как, скажем, находиться одновременно в Москве и Вашингтоне. Переменная в каждый момент времени может принимать лишь одно значение.

Исправленный алгоритм выглядит так.

Присвоить d значение 2^2+4-14. Присвоить p значение $\sqrt{d}$. Присвоить x значение $(2+p)/2$. Присвоить y значение $(2-p)/2$. Сообщить «Корни уравнения равны». Сообщить x. Сообщить y.

Задачи 8 и 9 — одни из первых на самостоятельное составление алгоритмов для исполнителя Вычислитель. Сначала, как обычно, желательно записать алгоритм в нечеткой форме, а затем составить алгоритм решения, использующий только допустимые действия Вычислителя.

Задачи 9б) и 9в) — на отработку понятий переприсваивания значения переменной. Учащиеся должны усвоить, что одна и та же переменная в процессе выполнения алгоритма может принимать разные (даже по смыслу) значения. Например, в первой команде алгоритма решения задачи 9в) a, b, c — длины сторон треугольника, а в последней команде a — полупериметр, b — площадь, а c — радиус вписанной окружности. Кроме того, в этой задаче впервые возникает тема оптимальности алгоритма в том или ином смысле (в данном случае минимизируется количество используемых переменных).

Задача 8. Автомобиль проехал три участка пути разной длины с разными скоростями. Составьте для Вычислителя алгоритм нахождения средней скорости автомобиля по заданным скоростям и длинам участков пути.

Решение

Обозначим через l, m, n длины трех участков пути; через V, F, G — скорости автомобиля на этих участках соответственно. Вот один из возможных способов нечеткой записи алгоритма.

Найдем время, за которое автомобиль проехал каждый из трех участков дороги (по формуле: время равно пути, деленному на скорость). Затем найдем время, затраченное автомобилем на весь путь, и длину пройденного пути. Разделив весь путь на все время движения, найдем среднюю скорость автомобиля.

Запишем алгоритм для Вычислителя:

Сообщить «Введите длины участков пути и скорости автомобиля на этих участках». Запросить p, q, r, V, F, G. Присвоить x значение p/V. Присвоить y значение q/F. Присвоить z значение r/G. Присвоить t значение $x+y+z$. Присвоить S значение $p+q+r$. Присвоить W значение S/t. Сообщить «Средняя скорость автомобиля равна». Сообщить W.

Можно предложить учащимся составить алгоритм, в котором участвует меньшее количество переменных.

Задача 9. Даны длины сторон треугольника: *a, b, c*.

9а). Составьте для Вычислителя алгоритм, выполняя который он сообщит полупериметр треугольника, его площадь и радиус вписанной окружности.

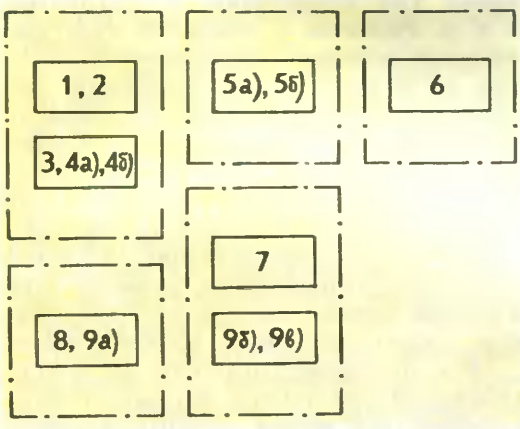
9б). Составьте алгоритм решения той же задачи, в котором используются только четыре переменные.

9в). Составьте алгоритм решения той же задачи, в котором используются только три переменные.

К первому уровню относятся задачи 1а), 2а), 4а), 5а), 5б), 6, 7, 8, 9а). Ко второму уровню относятся задачи 2б), 3, 9б). К третьему уровню относятся задачи 4б) и 9в). К четвертому уровню относится задача 1б).

Разбиение на блоки и группы показано на рис. 15:

15



§ 6. Достижимые цели

В предшествующих параграфах внимание было сосредоточено на алгоритмах и исполнителях вне зависимости от тех задач, для решения которых, собственно говоря, они и предназначаются. Реально же выбор исполнителя и последующее составление для него алгоритмов диктуется классом задач, которые требуется решать. Фактически соотношение исполнителя и алгоритмов при решении задач из некоторого класса представляет собой учебный аналог соотношения аппаратуры (*hardware*) и программного обеспечения (*software*) в «большой» информатике. При этом на «учебную плоскость» проецируется и важная проблема оптимального выбора этих компонент вычислительной системы. Эта проекция выглядит примерно так: «Существуют ли алгоритмы, позволяющие решать с помощью данного исполнителя задачи данного класса?» Иными словами, достижимы ли для данного исполнителя указанные заранее цели? Вопрос о том, какие цели для исполнителя достижимы, а какие — нет, обычно очень труден. Исполнители, применяемые в учебниках [2] и [3], не позволяют решить этот вопрос сколько-нибудь полно и доступно для учащихся. Поэтому в [2] и [3] он даже не обсуждается. А ведь вопрос о достижимых целях важен не только с содержательной, но и с методической точки зрения — он позволяет показать диалектику взаимоотношений трех важнейших понятий: класса решаемых задач, исполнителя и алгоритма. В [1] вопрос о достижимых целях обсуждается (и полностью решается) применительно к исполнителю Чертежник.

Типичные задачи к § 6.

Задача 1. Какие цели достижимы для Чертежника, умеющего выполнять лишь следующие действия:

1а) СДЕЛАТЬ ШАГ и ПРЫГНУТЬ;

2б) ПРЫГНУТЬ и ПОВЕРНУТЬ НАЛЕВО;

1в) ПОВЕРНУТЬ НАЛЕВО?

Задача 2. Какими допустимыми действиями вы снабдили бы автомат, заменяющий:

1а) кассира в магазине;

1б) дворника;

1в) вахтера;

1г) директора школы?

Задача 3. Чертежник-30 и Чертежник-45 очень похожи на обычного Чертежника. Единственное отличие состоит в том, как они понимают команду ПОВЕРНУТЬ НАЛЕВО. По этой команде Чертежник-30 поворачивается на 30°, а Чертежник-45 — на 45° против часовой стрелки. Какого из этих двух

Чертежников вы предпочтете для рисования следующих фигур:

3а) квадрат со стороной 2 см;

3б) правильный шестиугольник со стороной 1 см;

3в) правильный восьмиугольник со стороной 1 см;

3г) отрезок длиной 5 см?

Задача 4. Для каждого натурального N создан Чертежник-N, который по команде ПОВЕРНУТЬ НАЛЕВО поворачивается на угол N° против часовой стрелки. Найти такое N, что для Чертежника-N достижима каждая из двух целей: «нарисовать правильный десятиугольник» и «нарисовать квадрат». Для какого N суммарное число действий в алгоритмах рисования этих фигур наименьшее?

§ 6 завершается первая, вводная часть учебника.

Часть 2
Способы организации действий в алгоритмах

Изучению способов организации действий (ветвления, циклы, вспомогательные алгоритмы) во всех учебниках информатики отводится большое, если не сказать — центральное место. При этом в основу кладется изучение школьного алгоритмического языка или какого-либо языка программирования. Как известно (см. [4], с. 39—40), школьный алгоритмический язык выполняет две основные функции: позволяет стандартизировать форму записи алгоритмов и демонстрирует общие свойства языков программирования. Однако стремление полностью реализовать эти функции приводит к возникновению ряда методических трудностей. Причины этого, во-первых, в ограниченности словаря допустимых слов алгоритмического языка; во-вторых, в наличии жестких правил оформления алгоритмов. Это приводит к тому, что учащиеся не столько думают над составлением алгоритма, сколько над тем, как бы не ошибиться в его оформлении. Иными словами, учащиеся вынуждены осваивать одновременно два навыка: овладевать методами алгоритмизации (семантика) и средствами формальной записи алгоритмов (синтаксис). Ясно, что методически целесообразно развести освоение этих навыков в разные учебные плоскости.

Стандартизация предложений языка и ограниченность его словаря неизбежно приводят к тому, что алгоритм лишается внешне выраженной семантики. Работа с таким алгоритмом напоминает разгадывание шифрованного сообщения, ибо определить назначение той или иной команды в отсутствие комментариев практически невозможно. Например, «Присвоить *a* значение *b*» в разных алгоритмах, и даже в одном, может наполняться различным смысловым содержанием. Необходимость анализировать алгоритмы возникает при отладке, при сравнении различных алгоритмов решения одной и той же задачи, для модификации уже существующего алгоритма и т. д. Развитие у учащихся культуры комментирования — совершенно отдельная (и не так уж легко решаемая) дидактическая задача. Правда, возможно, привнесение в алгоритмы определенной доли семантики за счет мнемоники в обозначении переменных, заголовков подпрограмм (процедур) и т. п. Однако ограничения, существующие и здесь (например, ограниченность числа букв в именах переменных), делают эти средства далеко не всегда эффективными.

Перечисленные трудности лишь усугубляются при замене школьного алгоритмического языка другими алгоритмическими языками и языками программирования.

Все это привело нас к мысли отказаться от записи алгоритмов на школьном алгоритмическом языке, т. е. от фиксации строго определенного набора ключевых слов, обозначений и других строгих правил записи алгоритмов. Фактически мы лишь требуем, чтобы запись алгоритма была понятна не только автору, но и любому человеку, который мысленно имитирует того или иного исполнителя. Мы считаем, что алгоритм в тетради ученика или на классной доске должен быть записью обычной человече-

ской речи текстом, построенным по правилам естественного языка.

В частности, мы считаем допустимым и полезным использовать ключевые слова-синонимы (вместо «ветвление» — «развилка», вместо «цикл» — «повтор», вместо «присвоить a значение b» — «обозначить число b буквой a» и т. д.). Например, вместо команды Вычислителя:

Присвоить a значение 0

допустимо писать:

Присвоить a начальное значение, равное 0.

Обе эти записи свидетельствуют об исполнении одного и того же действия, однако вторая из них сообщает человеку, имеющему дело с данным алгоритмом, дополнительные неформальные сведения. Как видите, вольная запись команд заменяет и мнемоники, и комментарии. При этом, в частности, можно ограничиться привычными для учащихся однобуквенными обозначениями переменных.

Итак, мы считаем допустимой вольную запись алгоритма, требуя от учеников лишь:

а) чтобы отсутствовала двусмысленность в записи действий (например, можно использовать лишь те действия, которые допустимы исполнителем);

б) чтобы из записи алгоритма однозначно восстанавливался порядок выполнения действий.

Правда — может заметить приверженец школьного алгоритмического языка — заучивание правил школьного алгоритмического языка и неукоснительное следование этим правилам не позволяют учащимся писать синтаксически неправильные программы. И что очень важно, не позволяет им использовать в алгоритмах недопустимые действия исполнителей. Ведь именно в этом и состояла одна из целей введения школьного алгоритмического языка. Например, если позволить школьникам записывать действие Вычислителя «Присвоить c значение $a+b$» так: «Обозначить сумму чисел a и b через c» или просто «$c = a+b$», то где гарантия, что они не будут писать: «сложить два числа» (такое действие недоступно Вычислителю, поскольку он может работать только с переменными и константами). Гарантий, конечно, нет. Но если учащийся пишет подобную команду, то это вскрывает его непонимание существа действия присваивания. Формализм школьного алгоритмического языка не даст проявиться этому незнанию и объективно будет способствовать закреплению неправильно сформировавшегося знания. Так жесткие и узкие ботинки (или, скажем, латы), надетые в раннем возрасте, могут способствовать закреплению неправильной походки. За воспитанной школьным алгоритмическим языком синтаксической правильностью может скрываться полная алгоритмическая беспомощность.

Одно из неудобств использования школьного алгоритмического языка — обязательное требование указывать в начале алгоритма все используемые в нем переменные и их типы. Дело в том, что изучение типов данных является самостоятельной и весьма непростой методической задачей. Темы «Организация действий» и «Организация данных» связаны не настолько сильно, чтобы оправдать их одновременное изучение. В теме «Организация действий» вполне можно ограничиться исполнителями, каждый из которых работает с данными одного типа. Таковы, например, исполнители Чертежник, Вычислитель, Робот-манипулятор, фигурирующие в учебнике [1].

Кроме того, требование описания переменных — это не требование алгоритмизации и даже не метод воспитания дисциплины мышления (как это хотели бы представить многие профессиональные программисты), а всего лишь особенность некоторых трансляторов, «заботящихся» об экономии памяти ЭВМ и скорости трансляции. И потому это требование (как и другие чисто программистские условности) нецелесообразно включать в правила записи алгоритмов в тетрадях, которые пока не доступны трансляторам.

Сказанное выше вовсе не означает, что мы сторонники полного произвола в оформлении алгоритмов. Упомянутое требование отсутствия двусмысленности накладывает определенные ограничения на форму записи алгоритмов. Например, необходимо указывать границы

тела цикла и границы ветвлений. Однако мы считаем, что недостаточно просто объявить правила записи алгоритмической конструкции. Каждое ограничение свободы записи должно быть мотивировано. Мотивации каждого из ограничений (как и мотивациям вообще) уделяется большое внимание в нашем учебнике (мы подробно расскажем об этом в следующей статье).

Позволяя учащимся записывать алгоритмы в вольной форме, мы сосредоточили усилия учеников на освоении алгоритмизации, не осложняя процесс обучения ненужными на этом этапе проблемами синтаксиса. После содержательного освоения той или иной алгоритмической конструкции наступает этап выполнения алгоритмов исполнителями, имитируемыми на ЭВМ. Тогда-то и наступает время синтаксических ограничений. Конечно, ученику необходима «акклиматизация» в среде жесткого синтаксиса, навязанного компьютером. Такая «акклиматизация» основывается на двух методических идеях. Во-первых, выходу на ЭВМ предшествует изучение соответствующей алгоритмической конструкции и ее использование в ходе «безмашинного» решения задач. Во-вторых, имитации исполнителей на ЭВМ построены так, что избавляют учащихся от необходимости заучивания ключевых слов. Все ключевые слова естественным образом разбиты на несколько наборов. В специальной строке экрана приведены ключевые слова одного из наборов — текущие значения соответствующих функциональных клавиш. Особая клавиша служит для «перелистывания» наборов слов в этой специальной строке. Ее нажатие вызывает изменение значений функциональных клавиш. Таким образом, учащиеся не только избавлены от запоминания ключевых слов, но и затрачивают гораздо меньше времени на ввод программы. Строка функциональных клавиш выполняет еще и роль «ненавязчивой» подсказки, поскольку содержит, как правило, полный набор ключевых слов той или иной алгоритмической конструкции.

Продолжение следует.

Литература

1. *Гейн А. Г., Житомирский В. Г., Линецкий Е. В. и др.* Основы информатики и вычислительной техники. Свердловск: УрГУ, 1989.

2. Основы информатики и вычислительной техники: Пробное учебное пособие. Ч. I / Под ред. А. П. Ершова, В. М. Монахова. М.: Просвещение, 1985.

3. Основы информатики и вычислительной техники: Пробный учебник для средних учебных заведений / Под ред. А. П. Ершова. М.: Просвещение, 1988.

4. *Ершов А. П., Монахов В. М. и др.* Изучение основ информатики и вычислительной техники. Ч. I. М.: Просвещение, 1985.

5. *Гейн А. Г., Житомирский В. Г., Линецкий Е. В., Сапир М. В., Шолохович В. Ф.* Программно-методический комплекс по курсу школьной информатики // Информатика и образование. 1988. № 3.

В. КАЙМИН, С. ГРИГОРЬЕВ, Н. УГРИНОВИЧ,
А. ЩЕГОЛЕВ

Элементы логики и начала языка Пролог

В первой статье данного цикла мы говорили, что в цели школьного курса информатики должно входить не только овладение учащимися компьютерной грамотностью, но и формирование у них логического мышления как основы информационной культуры: умений общаться друг с другом, умений собирать, систематизировать, перерабатывать и передавать информацию людям. Возможности оперативного доступа и получения информации с использованием сетей ЭВМ и баз данных широко используются за рубежом, что делает неотложным обучение и у нас умениям использовать ЭВМ в этим целях и умениям создавать на них базы знаний.

Наиболее перспективным средством создания баз знаний на ЭВМ считается язык программирования Пролог. Главным достоинством этого языка является простота описания логики решения задач и обработки информации при поиске ответов на вопросы, необходимых для создания машин, способных к воспроизведению рассуждений. Именно такими станут машины пятого поколения, которые будут созданы и запущены в производство в 90-х гг. [1, 2, 3, 4].

Особенностью новейших языков при изучении является простота освоения и детьми, и взрослыми. Причина этого в том, что в их основе лежат простейшие и понятные всем слова — *факты, вопросы, правила, определения, условия* и *выводы*. При этом практическое изучение элементов логики и средств программирования на ЭВМ связывается друг с другом через весь курс информатики. В первом полугодии IX класса элементы логики связываются с работой с базами данных и изучением начал Пролога; во втором полугодии IX класса — через логическое осмысление средств алгоритмического языка и логических средств языка Бейсик.

В первом полугодии X класса в разделе «Решение задач на ЭВМ» появляется возможность решения логических и информационно-логических задач с использованием Бейсика. Во втором полугодии в разделе «Возможности и применение ЭВМ» знание учащимися логических функций помогает рассмотреть базовые логические элементы «или», «и», «не» и работу на их основе некоторых устройств ЭВМ (сумматор, триггер).

Формирование логического мышления целесообразно начинать с изучения элементов Пролога с параллельным изучением элементов логики. В настоящее время Пролог имеется на «Ямахе», а на остальные типы школьных ЭВМ интерпретатор Пролога разрабатывается. Поэтому в ближайший учебный год изучение Пролога может быть в основном теоретическим, что вызовет дополнительные методические трудности.

В логике ключевыми являются слова:

Продолжение. Начало см.: Информатика и образование. 1989. № 2, 3.

«понятие», «суждение», «рассуждение», «доказательство». В Прологе базовыми являются понятия: «факт», «вопрос», «правило». В основе принципов работы вычислительных машин, использующих Пролог, лежат процедуры логического вывода. В частности, быстродействие ЭВМ нового поколения будет измеряться в количестве логических выводов, осуществляемых машинами за одну секунду. Соотнесение понятий логики и Пролога представлено в таблице.

Основные понятия

Пролог	Логика
ФАКТ ВОПРОС ПРАВИЛО	ПОНЯТИЕ СУЖДЕНИЕ РАССУЖДЕНИЕ
вывод	доказательство

Начнем разбирать основные элементы Пролога — факты, вопросы и правила на примере диалога с ЭВМ на тему «Семья». Рассмотрим семью из 3 человек. В ней есть девочка, которую зовут Маша. Ее маму зовут Наташа, а бабушку — Даша (см. рис. 1).

Сведения о семье можно зафиксировать фразами «Наташа — мама Маши» и «Даша — мама Наташи». На Прологе эти факты записываются в форме:

Пролог	Смысл
мама (Наташа, Маша);	«Наташа — мама Маши»
мама (Даша, Наташа);	«Даша — мама Наташи»

Факты в Прологе выражают некоторые сведения о ком-то или чем-то. Ключевое слово соответствующей фразы записывается на первом месте; далее в скобках через запятую перечисляются основные характеристики объектов (субъектов). В конце записи факта ставится точка с запятой.

Приведенные факты могут непосредственно вводиться в ЭВМ, имеющую интерпретатор Пролога. В этом случае сразу же после ввода машине можно задавать вопросы, например:

«Кто мама у Маши?»

На Прологе запись этого вопроса должна иметь вид:

```
? мама (x, Маша);
```

В ответ машина даст ответ:

```
x = Наташа
```

Отличие вопроса от факта состоит не только в том, что его запись начинается со знака ?, но и в том, что в фактах можно записывать только конкретные данные — слова, числа или словосочетания, а в вопросах можно указывать переменные. В Прологе переменные обозначаются либо отдельными буквами, либо словами, начинающимися или оканчивающимися знаком ' (апостроф).

```
? мама (x, Наташа); «кто мама у Наташи?»
```

Ответ машины:

```
x = Даша
```

Принципы вывода ответов в Прологе состоят в том, что машина просматривает все факты, начинающиеся с имени «мама», и сравнивает их на совпадение с данными, указанными в вопросах. В первом случае машина ищет все факты о «мамах», в которых на втором месте стоит имя «Маша», а во втором — факты, в которых на втором месте фигурирует имя «Наташа».

Полный перечень всех фактов данного типа от машины можно получить, указав и на первом, и на втором месте переменные:

```
? мама (x, y)
```

Ответом будет перечень данных о всех фактах (о «мамах»), введенных в ЭВМ.

```
x = Маша
y = Наташа
x = Наташа
y = Даша
```

В отличие от фактов вопросы в Прологе могут быть составными. Для рассматриваемой пары фактов в качестве сложносоставного примера машине можно задать вопрос:

```
«Кто мама у мамы Маши?»
? мама (x, Маша), мама (y, x);
```

Этот вопрос состоит из двух подвопросов: «кто мама «x» у Маши?» и «кто мама «y» у мамы Маши?». Ответом машины будет:

x = Наташа
y = Даша

Ответы на сложные вопросы машина ищет по частям. Вначале ЭВМ выделяет первый подвопрос

? мама (x, Маша)

и перебирает ответы на него. Для каждого из частичных ответов машина рассматривает второй подвопрос

? мама (x, y)

и для найденного значения x (в примере x = Наташа) ищет ответ на этот подвопрос. Если оба подвопроса получили положительные ответы, то ЭВМ выводит окончательный ответ и переходит к просмотру оставшихся фактов. И так до тех пор, пока все факты не будут исчерпаны.

Третье основное понятие языка Пролог — это понятие «правила». В логике ему соответствуют два понятия — «правило вывода» и «правило определения». В рассматриваемом примере (см. рис. 2) можно ввести следующее правило:

бабушка (y, z) < — мама (x, z),
мама (y, x);

2

Это правило читается так: «некто y — это бабушка для z, если есть некто x, кто является мамой z, а y — мама для x». После ввода этого правила в ЭВМ можно задать такой вопрос:

? бабушка (y, Маша); «кто бабушка y Маши?»

И несмотря на то что в машине в явном виде не хранятся факты о бабушках, на этот вопрос ЭВМ даст ответ:

y = Даша «Даша — бабушка Маши»

Особенность данного вывода заключается в том, что при отсутствии фактов, начинающихся с заданного имени, машина начинает просмотр правил, имеющих в заголовке данное имя. Если такое правило обнаружено, то машина исходный вопрос замещает на подвопросы из определения (первой части) правила и начинает искать на них ответы. В данном случае вопрос

? бабушка (x, Маша)

замещается на сложносоставный вопрос

? мама (x, Маша), мама (x, z).

Перечисленные примеры фактов, вопросов и правил полностью раскрывают основные средства языка Пролог. Разбор этих параметров на занятиях может проводиться в форме синхронной работы на ЭВМ либо в форме разбора примера на доске с записью фактов, вопросов и правил в тетрадях. Во втором случае на ЭВМ ученики либо прорабатывают эти примеры с какими-то модификациями, либо составляют описание своей семьи и самостоятельно подбирают вопросы. В этом случае в «модель» понадобится ввести «папу», «дедушку» и, возможно, каких-то других родственников. Но для этого необходимо знать технику составления формальных описаний на Прологе.

17

Составление новых формальных описаний всегда следует начинать со структурно-логического анализа понятий, используемых для интересующих нас суждений. В модели «семьи» базовым понятием являются слова «мама» и «папа» (см. рис. 3). С каждым из этих слов ассоциируется два вопроса: «Кто мама (папа)?» и «Чья (чей) мама (папа)?».

3

В структурно-логическом анализе мы выделяем признаки и характеристики объектов и явлений, интересующих нас с определенной точки зрения, и фиксируем их в некотором порядке. Этот порядок и становится порядком записи значений в фактах и вопросах на Прологе.

Признаки в моделях могут выражаться не только определенным набором знаний, но и некоторыми свойствами, которые описываются правилами определения. Примером может служить рассмотренное определение «бабушки». Однако оно неполно, поскольку бабушки — это

OldPC.su
7002

и мамы мам, и мамы пап. Полное определение получает такой вид:

бабушка (x, z) < — мама (x, y), мама (y, z);
бабушка (x, z) < — мама (x, y), папа (y, z);

Аналогично вводятся определения «дедушки» (рекомендуется сделать самостоятельно). Рассматриваемую модель «семьи» можно расширить, введя признак «пола» в форме фактов (пол (Маша, жен), пол (Петр, муж)); задав определения понятий «сын», «дочь», «брат», «сестра», «дядя», «тетя» и т. д. Рекомендуем проделать это самостоятельно на факультативных занятиях.

Разбор примеров следует вести так, чтобы разъяснять основные элементы логики, обращая внимание учеников на понятия простых и сложносоставных суждений. Напомним, что суждения — это некоторые высказывания, выражающие определенные факты. Сложносоставные суждения строятся из простых с помощью логических связок. Среди них следует выделить связки «и», «или», «не» и разъяснить их на разобранных примерах фактов, вопросов и правил Пролога.

Связка «и» в Прологе используется явно в построении сложносоставных вопросов, а связка «или» — в альтернативных определениях понятий. Связка «не» может быть указана явно в сложносоставных вопросах и определениях правил. Рассмотрим вопрос: «Кто в семье является папой, но еще не является дедушкой?»

В ответ машина выделит всех мужчин, имеющих детей, но не имеющих внуков. Следует обратить внимание на то, что машина умеет искать ответы только на позитивные вопросы.

Вопрос называется негативным, если он начинается со связки «не». В ответ на негативный вопрос машина непременно даст ответ «не знаю», равно как и на другой вопрос, на который ЭВМ не может подобрать соответствующих фактов и правил. Таким образом, все определения понятий должны иметь позитивные формулировки; в противном случае и машина, и люди ставятся в затруднительное положение при поиске ответов.

Демонстрацию примеров вывода позитивных формулировок из отрицаний легче всего проводить на простейших алгебраических соотношениях и неравенствах.

Отрицание соотношений

не (x=0) ≡ (x≠0)
не (x=1) ≡ (x≠1)
не (x≠1) ≡ (x=1)

Отрицание «и»

не [(0<x) и (x<1)] ≡
≡ не (0<x) или не (x<1) ≡
≡ (x<0) или (x>0) ≡
≡ (x≤0) или (1≤x)

Отрицание «или»

не [(x=1) или (x=2)] ≡
≡ (не (x=1)) и (не (x=2)) ≡
≡ (x≠1) и (x≠2)

Разбор этих примеров позволяет обсудить следующие основные законы логики.

Закон двойного отрицания:

не (не A) ≡ A

отрицание отрицания равносильному исходному утверждению.

Закон отрицания совместности (1-й закон де Моргана):

не (A и B) ≡ (не A) или (не B)

отрицание совместности утверждений равносильно отрицанию составляющих суждений.

Закон отрицания альтернатив (2-й закон де Моргана):

не (A или B) ≡ (не A) и (не B)

отрицание альтернативных суждений равносильно отрицанию всех альтернатив.

Таким образом, Пролог может быть использован как пропедевтика логики в вопросах формулирования понятий, суждений, в вопросах вывода умозаключений и их обоснования. Связь между базовыми понятиями логики и языка Пролог на самом деле еще глубже, поскольку одно из его назначений — создание на ЭВМ баз знаний.

В простейшем толковании **базы знаний** — наборы фактов и правил вывода умозаключений по отношению к некоторой предметной области. Создание на ЭВМ баз знаний предполагает проведение структурно-логического анализа предметной области, отбора системы понятий, определений, выбора признаков и характеристик объектов, т. е. создания определенных информационно-логических моделей и подбора фактических данных.

4 школа

Вторая тема, которую необходимо разобрать на уроках с проработкой примеров на ЭВМ — создание базы знаний средствами Пролога. Пример модели семьи является простой базой знаний. Для построения другой базы знаний рекомендуется тема «Школа» (см. рис. 4). В число фактов по теме «Школа» кроме оценок учащихся по различным предметам можно включить сведения об их интересах (см. рис. 5).

5 Модель:

Структурно-логический анализ «оценок» предполагает фиксацию: фамилии ученика, названия предмета и отметки. Примеры записи фактов.

Факты:

оценка (Ларина, литература, 5)
оценка (Петров, физика, 4)
оценка (Иванов, литература, 3)

База данных «Оценки»		
ФИО	Предмет	Балл
Ларина	Литература	5
Петров	Физика	4
Иванов	Литература	3

Модель:

Собирает (Иванов, марки)
Собирает (Ларина, открытки)

База данных «Коллекция»	
Фамилия	Предмет
Ларина	Открытки
Иванов	Марки

Наборы фактов предварительно могут быть сведены в таблицы и в принципе могут размещаться в табличных базах данных. Именно такая форма накопления данных в машинах нового поколения называется базами знаний. Она предполагает размещение на магнитных носителях не только фактов, но и правил вывода умозаключений, составляющих ядро экспертных систем.

Табличное представление данных дает наглядную форму записи фактов, которую можно использовать для разъяснения принципов логического вывода и поиска ответов в базах знаний. Эту форму представления данных рекомендуется использовать на занятиях с целью отработки у учащихся навыков проведения рассуждений, подбора фактов, определений, формулировки и обоснования выводов. Такие задания вполне можно провести в рамках рассматриваемой модели.

Ученикам следует рекомендовать самостоятельно подобрать данные об интересах своих товарищей и их оценках по школьным предметам, ввести ряд определений и провести анализ этой информации на ЭВМ. В число определений можно ввести понятия «филателист», «книголюб», «меломан» и т. п. (см. рис. 6).

филателист (x) <— собирает (x, марки);
книголюб (x) <— собирает (x, книги);

6

По успеваемости можно ввести такие определения:

математик (x) <— оценка (x, математика, 5);
физик (x) <— оценка (x, физика, 5), оценка (x, математика, 5);
лирик (x) <— оценка (x, литература, 5), оценка (x, история, 5);

Из приведенных определений видно, что их можно формулировать по-разному и что между определениями возникают некоторые взаимосвязи, которые можно выявить путем умозрительного анализа, либо путем экспериментов на ЭВМ. Например, из приведенных определений следует, что всякий «физик» является «математиком», но не всякий «математик» оказывается «физи-

ком». Действительность и факты могут показать несколько иное соотношение этих понятий.

С этой целью полезно ввести и разъяснить понятие общих суждений с использованием слов «все», «каждый», «существует», «найдется». В рассматриваемом случае можно высказать общее суждение «всякий физик является математиком», которое не вполне верно. Представляет интерес для учащихся выяснение взаимоотношений между «книголюбами» и «лириками», «книголюбами» и «математиками», «физиками» и «лириками» и т. п.

Формирование общих суждений отчасти возможно и в Прологе, в котором вместо конкретных значений данных разрешается записывать знак подчеркивания «_». Это означает, что при поиске информации от машины требуется найти факт с любым конкретным значением в данной позиции. Пример: понятие «двоечника» и «успевающего»

```
двоечник (x)<— оценка (x, _, 2)
```

которое читается так: «двоечник» — это ученик, у которого есть двойки по какому-нибудь предмету;

```
успевающий (x)<— оценка (x, _, _), НЕ (оценка (x, _, 2))
```

означающее: «успевающий» — это тот, у кого есть оценки, но нет ни одной двойки.

При необходимости сравнения числовых значений в Прологе предусмотрены стандартные встроенные предикаты: БОЛЬШЕ (x, y), МЕНЬШЕ (x, y) и РАВНО (x, y). Например, понятие «отличник» можно ввести с помощью определений:

```
неотличник (x)<— оценка (x,_,y), МЕНЬШЕ (y, 5)
отличник (x)<— оценка (x, _, 5), НЕ (неотличник (x))
```

Первое из них объявляет «неотличником» учащегося, если у него найдется оценка меньше 5. Второе определение выделяет учащихся, имеющих хотя бы одну 5, а затем проверяет: не является ли он «неотличником», что по первому определению означало бы наличие у него оценок меньше 5. В силу закона двойного отрицания это и будет условие того, что он является отличником.

По отношению к такого рода общим суждениям необходимо явно сформулировать и разобрать на примерах логические законы отрицания общих суждений.

А. Отрицание общности:
Неверно (для всех x: P (x)) ≡ найдется x: неверно P(x).

Б. Отрицание существования:
Неверно (найдется x: P (x)) ≡ для всех x: неверно P(x).

Разъяснение этих законов можно проводить с помощью наглядных иллюстраций. Но еще лучше пояснить, что опровержение неверных общих утверждений проводится конкретными противоречащими примерами, а опровержение суждений о существовании объектов с определенными свойствами требует доказательства, что ни один объект из данного класса не обладает данным свойством.

Кроме общих законов отрицания необходимо сформулировать и разъяснить на примерах законы логического вывода (дедукции) и опровержения (контрпозиции).

1. Правило вывода (дедукции):
Если A влечет B
и A выполнено,
то истинно и B.

2. Правило опровержения (контрпозиции):
Если A влечет B,
но B не выполнено,
то не могло иметь место и A.

Приведенные законы можно продемонстрировать не только на примерах логического вывода в Прологе, но и на простейших житейских и математических рассуждениях.

Обсуждение начал языка Пролог и элементов логики стоит завершить обсуждением взаимоотношений и взаимосвязей между формальной, математической и диалектической логикой.

Исторически первой возникла формальная логика, которая изучает наши обычные рассуждения, отвлекаясь от их содержания.

Математическая логика изучает способы строго обоснованных рассуждений и использует математический язык для записи математических утверждений. Язык математической логики служит базой для описания принципов работы любых вычислительных машин и устройств. Именно этот язык лежит в основе наиболее перспективных языков и

методов программирования.

Диалектическая логика лежит в первооснове современной философии и применяется при изучении законов развития живых организмов, природы, человеческого общества, наук, мышления человека и т. д.

Сложность взаимоотношений между этими направлениями связана со степенью точности суждений, допускаемых в житейских, математических и научных рассуждениях, и с учетом временных факторов.

Математика и вычислительная техника требуют абсолютной однозначности исходной информации и окончательных выводов, которые в основном строятся на применении принципа дедукции. Житейская практика и научная деятельность сталкиваются прежде всего с различными толкованиями понятий, появлением новых объектов и соответственно новых понятий, расширением их содержания, связанным с течением времени и изменением обстоятельств.

Одним из наглядных примеров диалектического развития является развитие информационной и вычислительной техники как средств накопления, обработки и передачи информации и развитие соответственно информатики — науки о законах и методах накопления, обработки и передачи информации, единых в принципе в жизни живых организмов и в работе вычислительных и информационных устройств, в общении людей и в жизни общества.

Развитие мысли в ходе познания природы или создания новых средств решения задач можно проиллюстрировать на рис. 7.

Познание действительности всегда начинается с конкретных наблюдений, накапливая которые мы переходим к обобщениям. По мере накопления обобщений мы пытаемся делать выводы и обосновывать их путем рассуждений (дедукция). Проверка наших общих выводов на практике отсеивает ложные гипотезы и заставляет на основе новых наблюдений делать новые обобщения (индукция). И так далее — по спирали, которая отражает процесс накопления достоверных знаний и известна под названием *диалектической спирали*.

Подведем итоги. Разбор примеров вопросов, фактов и определений на языке Пролог, проверка их на ЭВМ и обсуждение законов логики направлены на формирование у учеников следующих знаний, навыков и умений. Учащиеся должны:

знать:

основные логические связки;
правила записи фактов, вопросов и определений на языке Пролог;
примеры простейших информационно-логических моделей;
основные законы логики;

уметь:

выражать отрицания составных суждений;
записывать факты, вопросы и определения на языке Пролог;
составлять простейшие информационно-логические модели на языке Пролог;
выражать отрицания для общих суждений.

Продолжение следует.

Приложение

Грамматика языка Пролог

базазнаний	::={<предложение>}*
программа	::={<предложение>}*<вопрос>
предложение	::=<факт>|<правило>
факт	::=<предикат>
правило	::=<предикат><—<предикат>[,предикат]*
вопрос	::=?<предикат>[,предикат]*
предикат	::=<имя>|<имя>(<аргумент>[,<аргумент>]*)
аргумент	::=<фраза>|<число>|<функция>
фраза	::=<буква>[<символ>]*
число	::=<целое>
целое	::=[+|—][<цифра>]*
цифра	::={0|1|2|3|4|5|6|7|8|9}

Литература

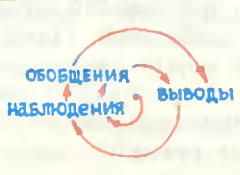
1. *Клоксин У., Меллиш К.* Программирование на языке Пролог. М.: Мир, 1987.

2. *Кларк К., Маккейб Ф.* Введение в логическое программирование на микро-Прологе. М.: Радио и связь, 1987.

3. *Робинсон Д.* Логическое программирование — прошлое, настоящее и будущее // Логическое программирование. М.: Мир, 1988.

4. Язык Пролог в пятом поколении ЭВМ. М.: Мир, 1988.

Диалектика познания:



А. КУШНИРЕНКО

О новом учебнике информатики

Предисловие

Предлагаемые ниже материалы требуют пояснений[1].

1. Подробно будет рассмотрена только глава 1 «Алгоритмический язык». Именно в ней наиболее ярко проявляются методологические особенности книги. Кроме того, на разбор остальных двух глав просто не хватит места в журнале. Ограничимся лишь названиями глав, которые в достаточной мере отражают их содержание:

Глава 2. «Устройство и работа ЭВМ».

Глава 3. «Применение ЭВМ».

2. Предлагаемые материалы делятся на две отдельные части. Первая — *краткий конспект книги* — ориентирована в основном на читателей, не имеющих учебника, но она может быть полезна и любому учителю как обобщение. Вторая часть — *методические рекомендации* — рассчитана на каждого преподавателя, который решит использовать нетрадиционную методику изложения основ алгоритмизации и программирования, причем не только на уроках, но и в факультативной или кружковой работе.

3. Для того чтобы к началу 1989/90 учебного года учитель имел на руках логически завершенный фрагмент материала, конспект учебника и методические рекомендации будут чередоваться. В этом номере предлагается примерное распределение материала § 1—8 по урокам (см. табл. 1) и конспект § 1—8. В следующем номере будут помещены методические рекомендации к этим параграфам. Читая конспект, надо учесть, что при значительном сокращении текста пришлось, к сожалению, пожертвовать стилем изложения.

Авторский коллектив учебника и методического пособия с благодарностью примет любые советы, замечания и рекомендации читателей по адресу: 125130 Москва, 3-й Новоподмосковный пер., д. 7а. Методический кабинет Ленинградского района. Авербуху А. В.

Краткое содержание и основные методические идеи учебника

Введение

§ 1. *Предмет информатики*

Информатика изучает методы представления, накопления, передачи и обработки информации. С понятием **информации** вы будете знакомиться постепенно в течение всего курса, потому что дать краткое определение этому понятию нельзя.

При передаче информации всегда есть **получатель** и **источник.** Информация передается с помощью последовательности сигналов, называемой сообщением.

Для записи, хранения и воспроизведения информации служат различные устройства под общим названием **память.** Основу памяти составляет **носитель** — физическая среда, хранящая образы сообщений. Одно и то же сообщение можно задавать разными способами. Этот процесс называется **кодированием** информации. Кодирование с помощью сигналов двух видов назы-

[1] Предисловие и примерное распределение материала подготовлено учителем А. В. Авербухом.— *Прим. ред.*

Продолжение. Начало см.: Информатика и образование. 1989. № 2.

Таблица 1

№ п/п	Содержание урока
	Тема 1. Предмет информатики
1	Введение. Предмет информатики. Понятие алгоритма (§ 1)
2	ЭВМ — универсальная машина для обработки информации. Работа пользователя на ЭВМ (§ 2)
	Тема 2. Простые команды
3	Понятие исполнителя. Система команд. Среда. Исполнитель. Робот. Автоматическое управление. Алгоритмический язык. Служебные слова: алг, дано, надо, нач, кон. Тело и заголовок алгоритма. Отказы (§ 3)
4	Редактирование алгоритмов на ЭВМ в Е-практикуме. Решение задач по управлению Роботом
5	Чертежник. Команды с аргументами. Формы записи чисел в информатике. Черепашка (§ 4)
6	Решение задач с Чертежником. Обобщение пройденного
	Тема 3. Вспомогательные алгоритмы
7	Вспомогательные алгоритмы. Вызов алгоритма без аргументов (§ 5)
8	Метод последовательного уточнения (§ 5)
9	Понятие величины в информатике. Модель памяти ЭВМ. Алгоритмы с аргументами, их вызов. Вид и тип величины. Слова: арг, вещ, цел (§ 5)
10	Решение задач с помощью вспомогательных алгоритмов
11	Подготовка к контрольной работе
12	Контрольная работа № 1
	Тема 4. Составные команды. Циклы с известным числом повторений
13	Арифметические операции и выражения. Стандартные функции. Линейная запись (§ 6)
14	Команда повторения «*n* раз». Составные команды. Тело и заголовок цикла (§ 7)
15	Решение задач с использованием команды цикла «*n* раз»
16	Циклы с параметром. Промежуточные величины (§ 8)
17	Составление алгоритомов, содержащих циклы с известным числом повторений («*n* раз» и «для»). Обобщение материала

вается **двоичным.** Информационный объем сообщения, закодированного в двоичной форме, измеряется в **битах** или **байтах.**

Важную роль в информатике играет понятие **алгоритма.** С алгоритмами вы встретитесь при решении задач, помещенных после этого параграфа.

Задачи

1. На берегу стоит крестьянин с лодкой, а рядом с ним находятся волк, коза и капуста. Крестьянин должен переправиться сам и перевезти волка, козу и капусту на другой берег. Кроме крестьянина в лодку помещается либо только волк, либо только коза, либо только капуста. Оставшись без присмотра, волк может съесть козу, а коза — капусту. Как должен вести себя крестьянин?

2. Автоматическое устройство имеет две кнопки и экран, на котором при включении горит число 0. При нажатии на первую кнопку число на экране удваивается. При нажатии на вторую — увеличивается на единицу. Как, нажимая на кнопки не более 10 раз, получить на экране число 99?

§ 2. *ЭВМ — универсальная машина для обработки информации*

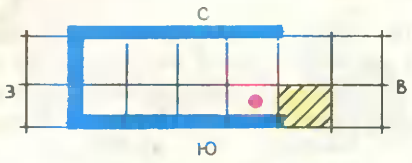
В параграфе рассмотрены состав персональной ЭВМ, названия и назначение основных устройств. Объясняется универсальность ЭВМ как машины для обработки информации по сменяемым алгоритмам.

§ 3. *Исполнитель Робот и его команды*

Робот может перемещаться по плоскому полю, разбитому на клетки одинакового размера. Между клетками поля могут быть расположены стены, некоторые клетки могут быть закрашены. Перемещаясь, Робот может закрашивать клетки. В отличие от Черепашки Робот имеет команды обратной связи: может узнать, закрашена клетка или нет, измерить уровень радиации и температуру в клетке, определить, загорожен путь в соседнюю клетку стеной или нет. Это позволяет на простых наглядных примерах ввести все основные управляющие конструкции: **«если — то — иначе»**, цикл **«пока»** и др. Робота легко изобразить на экране компьютера, сделав процесс разработки алгоритма естественным и увлекательным. На

рис. 1[2] изображен Робот, стоящий на западе от закрашенной клетки:

1



Команды Робота вводятся поэтапно, по мере усложнения решаемых задач. Сначала изучаются команды-приказы (слева), затем к ним добавляются команды-вопросы (справа):

шаг на север	на севере свободно
шаг на юг	на юге свободно
шаг на запад	на западе свободно
шаг на восток	на востоке свободно
закрасить	клетка закрашена

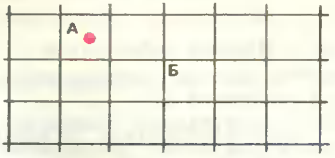
При выполнении команды-вопроса Робот анализирует обстановку на поле и отвечает «да» или «нет».

Использование четырех простейших команд перемещения Робота позволяет просто и естественно объяснить различия между непосредственным управлением Роботом и программированием процесса управления. При непосредственном управлении человек думает, дает команды Роботу, анализирует результаты, дает другие команды и т. д. Программируя процесс управления, человек составляет план будущей деятельности — алгоритм управления Роботом, а затем ЭВМ уже без участия человека командует Роботом по этому алгоритму. При таком противопоставлении «ручного» и автоматического управления естественно возникает понятие алгоритма (программы), а необходимость записи алгоритма по строгим правилам (на алгоритмическом языке) становится очевидной, так как выполнять алгоритм будет машина.

В учебнике алгоритмы записываются на специальном школьном алгоритмическом языке. Никакие другие алгоритмические языки (например, Бейсик, Рапира) в учебнике не рассматриваются.

Формальная запись алгоритма впервые появляется на примере решения простейшей задачи — перевести Робота из клетки А в клетку Б (рис. 2).

2



Один из возможных ответов:

```
алг  ход конем
  дано ! Робот в клетке А, стен на
       ! поле нет
  надо ! Робот в клетке Б
нач
. шаг на восток
. шаг на восток
. шаг на юг
кон
```

Общий вид алгоритма на школьном алгоритмическом языке:

```
алг  имя алгоритма (аргументы и
     результаты)
  дано ! условия применимости алго-
       ! ритма
  надо ! цель исполнения алгоритма
нач

. серия команд
кон
```

После служебных слов дано и надо пишутся условия, которые проверяются до начала исполнения алгоритма (строка дано) и по окончании исполнения алгоритма (строка надо). Если в процессе исполнения алгоритма выясняется, что условие не соблюдается, то ЭВМ аварийно прекращает исполнение алгоритма. В этих же строках после знака «!» можно помещать комментарии, поясняющие смысл алгоритма.

Задачи

3. Дан алгоритм, в котором опущены комментарии и название:

а)

```
алг
  дано !
  надо !
нач
. шаг на восток
. закрасить
. шаг на запад
. шаг на запад
. закрасить
. шаг на восток
```

б)

```
алг
  дано !
  надо !
нач
. шаг на север
. шаг на восток
. закрасить
. шаг на юг
. шаг на юг
. закрасить
```

[2] Во всех рисунках, связанных с Роботом, расположение сторон света подразумевается, как на рис. 1.— *Прим. ред.*

. шаг на юг
. закрасить
. шаг на север
. шаг на север
. закрасить
. шаг на юг
кон

. шаг на запад
. шаг на запад
. закрасить
. шаг на север
. шаг на север
. закрасить
. шаг на восток
. шаг на юг
кон

Опишите движение Робота в процессе исполнения алгоритма. Нарисуйте конечное положение Робота и закрашенные в результате исполнения клетки. Придумайте подходящее название алгоритма и впишите после слов **дано** и **надо** подходящие комментарии.

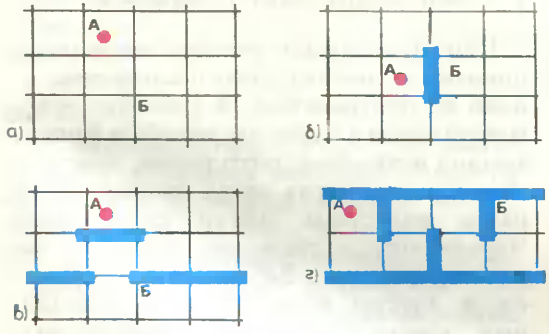
4. Измените алгоритм задачи 3, б так, чтобы при его исполнении Робот:

а) прошел тем же маршрутом, но ничего не закрашивал;

б) закрасил все клетки, в которых он побывал.

5. Составьте алгоритм, при исполнении которого Робот переместится из клетки А в клетку Б (рис. 3).

3

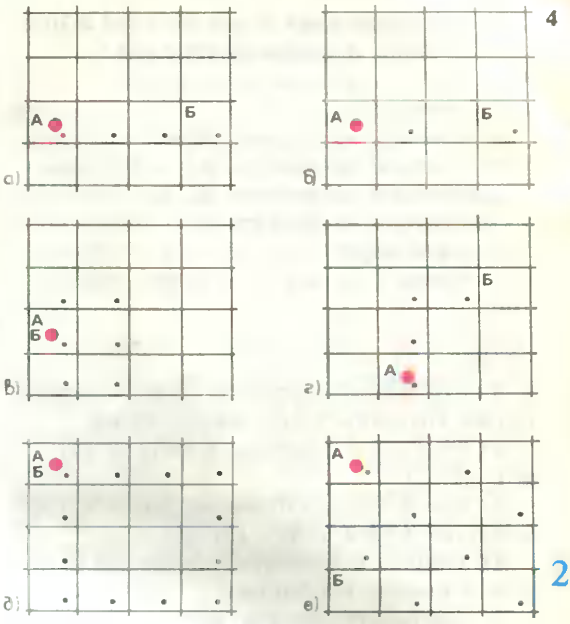


6. Составьте алгоритм, при исполнении которого Робот переместится из клетки А в клетку Б и закрасит клетки, отмеченные точками (рис. 4).

7. Петя составил алгоритм, переводящий Робота из клетки А в клетку Б с закрашиванием каких-то клеток. Что должен сделать Коля с этим алгоритмом, чтобы получить из него алгоритм, переводящий Робота из клетки Б в клетку А и закрашивающий те же клетки?

§ 4. *Исполнитель Чертежник*

Чертежник предназначен для построения рисунков, чертежей, графиков и т. д.



4

25

на листе бумаги и умеет выполнять 4 команды:

поднять перо
опустить перо
сместиться в точку (x, y)
сместиться на вектор (dx, dy)

При перемещении опущенного пера за ним остается след. Поднимая, опуская и перемещая перо Чертежника, можно рисовать разные картинки. В начале работы Чертежника лист чист, перо поднято и находится в начале координат, т. е. в точке (0,0).

Рассмотрение аргументов команд Чертежника и составление алгоритмов с использованием этих команд подготавливают введение вспомогательных алгоритмов с аргументами.

С помощью Чертежника можно рисовать любые фигуры, составленные из отрезков, например буквы:

алг нарисовать М
дано | перо поднято и находится в
| точке *a* (рис. 5)

5

надо ! нарисована буква М, перо в точ-
! ке *б* и поднято (рис. 5)
нач
. опустить перо
. сместиться на вектор (0, 4)
. сместиться на вектор (1, —2)
. сместиться на вектор (1, 2)
. сместиться на вектор (0, —4)
. поднять перо
. сместиться на вектор (1, 0)
кон

Задачи

8. Составьте алгоритм, после исполнения которого будут нарисованы:

а) отрезок с концами в точках (1, 2) и (—1, 1);

б) какой-нибудь отрезок длины 3, проходящий через точку (2, 2);

в) квадрат с длиной стороны 2 и центром в начале координат.

9. Составьте десять алгоритмов для рисования десяти цифр почтового индекса так, чтобы при их последовательном исполнении цифры рисовались друг за другом.

10. Измените алгоритмы рисования букв М, И, Р так, чтобы при последовательном исполнении этих алгоритмов слово МИР оказалось написанным:

а) с удвоенным расстоянием между буквами;

б) буквами удвоенного размера;

в) сверху вниз (так выглядят вывески некоторых магазинов);

г) сверху вниз буквами удвоенного размера.

11. Задан алгоритм:

алг фигура
дано ! перо в начале координат и
! поднято
надо !
нач
. сместиться в точку (2, 1)
. опустить перо
. сместиться на вектор (0, 3)
. сместиться на вектор (1, 0)
. сместиться на вектор (0, —1)
. сместиться на вектор (1, 0)
. сместиться на вектор (0, —1)
. сместиться на вектор (1, 0)
. сместиться на вектор (0, —1)
. сместиться в точку (2, 1)
. поднять перо
кон

а) не исполняя алгоритма и не рисуя получившейся фигуры, определите, где будет расположено перо после исполнения алгоритма, будет ли оно поднято или опущено;

б) исполните алгоритм, нарисуйте получившуюся фигуру;

в) впишите комментарий после слова надо;

г) переделайте алгоритм так, чтобы он рисовал где-нибудь на плоскости фигуру вдвое большего размера;

д) переделайте алгоритм так, чтобы он рисовал фигуру, симметричную первой, относительно оси *у*;

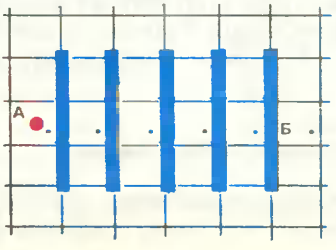
е) определите, что будет нарисовано, если в алгоритме изменить знаки всех аргументов на противоположные.

12. Придумайте автоматическое устройство, которое умеет рисовать разными цветами. Составьте несколько алгоритмов для управления этим устройством и нарисуйте результаты их исполнения.

§ 5. *Вспомогательные алгоритмы*

При изучении алгоритмического языка понятие вспомогательного алгоритма — одно из центральных. В учебнике вспомогательные алгоритмы вводятся раньше команд ветвления, повторения, присваивания на примерах задач рисования заранее известных фигур с помощью Чертежника, а также на примерах задач перемещения Робота из одной клетки в другую и закрашивания заданных клеток в заданной обстановке. Задачи (обстановки) подобраны так, чтобы решение с использованием вспомогательных алгоритмов было красивым и компактным, а решение без использования вспомогательных алгоритмов — некрасивым и громоздким.

Пусть, например, требуется написать



алгоритм, который проводит Робота из клетки А в клетку Б и закрашивает клетки, отмеченные точками (рис. 6).

Ясно, что обход каждой стены состоит в исполнении одной и той же последовательности команд, а именно:

шаг на север
шаг на север
шаг на восток
шаг на юг
шаг на юг

Для исполнения этой последовательности команд и закрашивания конечной клетки можно ввести вспомогательный алгоритм (назовем его «обход с закрашиванием»):

```
алг обход с закрашиванием
  дано ! Робот западнее средней клетки
       ! стены из 3 клеток
  надо ! Робот на клетку восточнее
       ! исходного положения,
       ! эта клетка закрашена
нач
. шаг на север
. шаг на север
. шаг на восток
. шаг на юг
. шаг на юг
. закрасить
кон
```

С использованием этого вспомогательного алгоритма основной алгоритм «из А в Б с закрашиванием» можно записать так:

```
алг из А в Б с закрашиванием
  дано ! Робот в точке А (рис. 6)
  надо ! Робот в точке Б, нужные клетки
       ! закрашены (рис. 6)
нач
. закрасить
. обход с закрашиванием
. обход с закрашиванием
. обход с закрашиванием
. обход с закрашиванием
. обход с закрашиванием
кон
```

Далее вводятся вспомогательные алгоритмы с аргументами, например алгоритм рисования произвольного квадрата:

```
алг квадрат (арг вещ a)
  дано ! перо Чертежника в левом нижнем
       ! углу будущего квадрата и поднято
  надо ! нарисован квадрат с длиной стороны a
       ! и сторонами, параллельными
       ! координатным осям;
       ! перо Чертежника в левом нижнем
       ! углу и поднято
нач
. опустить перо
. сместиться на вектор (a, 0)
. сместиться на вектор (0, a)
. сместиться на вектор (—a, 0)
. сместиться на вектор (0, —a)
. поднять перо
кон
```

Служебные слова **арг, рез, цел, вещ** в заголовках вспомогательных алгоритмов имеют тот же смысл, что и в старом учебном пособии. Служебное слово **нат** в новом учебнике не используется, т. е. если даже величина и принимает только натуральные значения, то описывается она как **цел**.

Задачи

13. Дан алгоритм «домик»:

```
алг домик
нач
. опустить перо
. сместиться на вектор (4, 0)
. сместиться на вектор (0, 4)
. сместиться на вектор (—4, 0)
. сместиться на вектор (0, —4)
. поднять перо
. сместиться на вектор (0, 4)
. опустить перо
. сместиться на вектор (2, 2)
. сместиться на вектор (2, —2)
. поднять перо
. сместиться на вектор (—4, —4)
кон
```

Составьте алгоритм «улица из трех домиков», используя алгоритм «домик» как вспомогательный. Нарисуйте результат исполнения этого алгоритма (т. е. полученную картинку и положение пера Чертежника).

14. Составьте алгоритм, рисующий улицу из:

а) шести домиков;
б) девяти домиков.

15. Нарисуйте результат исполнения алгоритма:

```
алг тоннель
нач
. квадрат (10)
. сместиться на вектор (1, 1)
. квадрат (7)
. сместиться на вектор (1, 1)
```

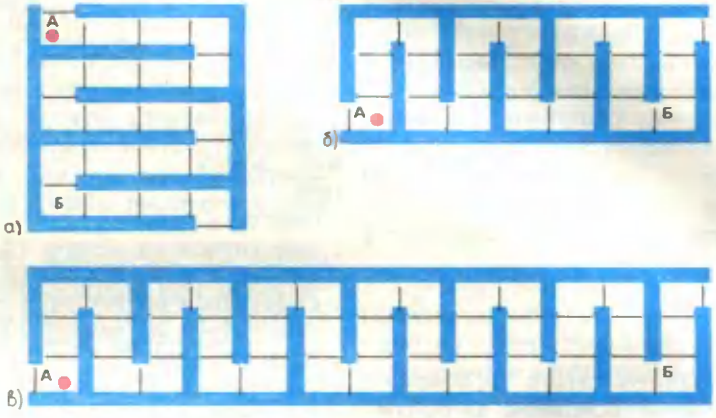
. квадрат (4)
. сместиться на вектор (1, 1)
. квадрат (1)
кон

16. Составьте алгоритм рисования прямоугольника с произвольными длинами сторон.

17. Придумайте картинку, состоящую из прямоугольников разного размера. Составьте алгоритм для рисования этой картинки, используя альгоритм задачи 5 как вспомогательный.

18. Составьте алгоритм, при исполнении которого Робот переместится из клетки А в клетку Б (рис. 7).

7



§ 6. *Линейная запись арифметических выражений*

В отличие от старых пробных учебных пособий в новом учебнике алгоритмический язык рассматривается как обычный формальный язык программирования. Поэтому арифметические выражения в новом учебнике записываются так же, как в других языках программирования (в одну строку, без индексов, радикалов, дробей). Например:

$$x1:=(-b-sqrt(d))/(2\times a)$$

После § 6 в учебнике приведено большое количество упражнений.

§ 7. *Команда повторения (цикл) n* раз

Изучение управляющих конструкций алгоритмического языка начинается с простейшей конструкции — цикла *n* раз :

нц число повторений **раз**
. серия команд
кц

Здесь число повторений — это произвольное целое число или целочисленное выражение. Серия команд — это произвольная последовательность команд алгоритмического языка. При исполнении алгоритма эта последовательность команд циклически повторяется столько раз, каково значение арифметического выражения, записанного между служебными словами **нц** и **раз** . Правила алгоритмического языка допускают задание нулевого и даже отрицательного числа повторений. Эти случаи не считаются ошибочными — просто указанная между строками **нц** и **кц** последовательность команд не будет исполнена ни разу.

На первых порах цикл *n* **раз** воспринимается учащимися просто как удобное обозначение, сокращающее запись алгоритма. Например:

алг на север десять шагов
  дано ! на поле Робота стен нет
  надо ! Робот сместился на 10 шагов
        ! на север
нач
. **нц** 10 **раз**
. . шаг на север
. **кц**
кон

В этом примере можно обойтись без команды повторения, написав 10 раз «шаг на север». Однако далее эта команда используется в ситуациях, где без нее решить задачу уже нельзя, например:

```
алг на север (арг цел k)
  дано ! на поле Робота нет стен
  надо ! Робот сместился на k шагов на
       ! север
нач
. нц k раз
.  . шаг на север
. кц
кон
алг закрасить полосу (арг цел m)
  дано ! на поле Робота стен нет
  надо ! закрашены m клеток восточнее
       ! исходного положения
       ! Робот в исходном положении
нач
. нц m раз
. . шаг на восток
. . закрасить
. кц
. нц m раз
. . шаг на запад
. кц
кон
```

Уже при введении цикла *n* раз учащиеся начинают сталкиваться с тем, что короткие алгоритмы могут описывать очень длинные последовательности действий. Например, при вызове «закрасить полосу (1024)» ЭВМ выдаст Роботу 3072 команды. А ведь в записи алгоритма «закрасить полосу» всего 13 строк!

Задачи

19. Используя алгоритм «закрасить полосу (арг цел *m*)» как вспомогательный, составьте алгоритм со следующим заголовком:

```
алг закрасить прямоугольник (арг
цел m, n)
  дано ! на поле Робота стен нет, Робот
       ! западнее северо-западного угла
       ! будущего прямоугольника
  надо ! закрашен прямоугольник разме-
       ! ром m×n
       ! Робот в исходном положении
```

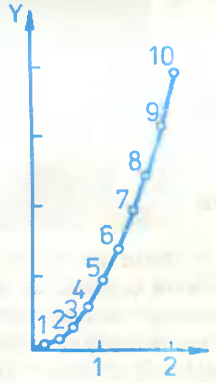
20. Известно, что алгоритм «картинка» рисует некоторую картинку в квадрате 1×1 и возвращает перо в начальное положение — левый нижний угол квадрата. Составьте алгоритм, рисующий картинку в 100 экземплярах в квадрате 10×10.

§ 8. *Команда повторения (цикл)* для

В учебнике команда повторения для вводится при решении такой задачи: изобразить с помощью Чертежника график функции $y=x^2$ на участке от $x=0$ до $x=2$ в виде ломаной из 10 звеньев. Сначала решение записывается без использования циклов:

```
алг параболa из десяти звеньев
  дано ! перо Чертежника в начале ко-
       ! ординат и поднято
  надо ! нарисован график функции y=
       ! =x^2 на участке от 0 до 2 в виде
       ! ломаной из 10 звеньев (рис. 8);
       ! перо в точке (2, 4) и поднято
```

8




```
нач
. опустить перо
. сместиться в точку (1*0.2, (1*0.2)
                                **2)
. сместиться в точку (2*0.2, (2*0.2)
                                **2)
. сместиться в точку (3*0.2, (3*0.2)
                                **2)
. сместиться в точку (4*0.2, (4*0.2)
                                **2)
. сместиться в точку (5*0.2, (5*0.2)
                                **2)
. сместиться в точку (6*0.2, (6*0.2)
                                **2)
. сместиться в точку (7*0.2, (7*0.2)
                                **2)
. сместиться в точку (8*0.2, (8*0.2)
                                **2)
. сместиться в точку (9*0.2, (9*0.2)
                                **2)
. сместиться в точку (10*0.2, (10*0.2)
                                **2)
. поднять перо
кон
```

Далее вводится цикл для. Как и

цикл *n* раз, поначалу цикл для воспринимается как удобное обозначение, способ сокращения записи алгоритма:

```
алг  парабола из десяти звеньев
  дано ! перо Чертежника в начале
       ! координат и поднято
  надо ! нарисован график функции
       ! y=x^2 на участке от 0 до 2 в
       ! виде ломаной из 10 звеньев
       ! (рис. 8); перо в точке (2, 4)
       ! и поднято
нач  цел
. опустить перо
. нц для i от 1 до 10
.   . сместиться в точку (i×0.2, (i×0.2)
    ××2)
. кц
. поднять перо
кон
```


Общий вид команды повторения для таков:

```
нц для i от i1 до i2
. серия команд
кц
```

Здесь $i1$, $i2$ — произвольные целые числа или выражения с целыми значениями. При исполнении алгоритма серия команд циклически повторяется для $i=i1, i=i1+1, i=i1+2, \ldots, i=i2$. Правила алгоритмического языка допускают задание любых целых $i1$, $i2$. В частности, $i2$ может быть меньше $i1$. Этот случай не считается ошибочным — просто указанная между строками нц и кц серия команд не будет исполнена ни разу, а ЭВМ сразу перейдет к исполнению команд, записанных после кц.

Далее, как и при введении цикла *n* раз, цикл для применяется в ситуации, когда без него обойтись нельзя:

```
алг  график (арг вещ a, b арг цел n)
  дано ! n>0, перо Чертежника поднято
  надо ! нарисован график функции
       ! y=f(x) на участке от a до b
       ! в виде ломаной из n звеньев;
       ! перо в точке (b, f(b)) и поднято
нач  цел i
. сместиться в точку (a, f(a))
. опустить перо
. нц для i от 1 до n
.   .сместиться в точку (a+i×(b—a)/n,
    f(a+i×(b—a)/n))
. кц
. поднять перо
кон
```

Задачи

21. Определите, что будет нарисовано в результате исполнения следующих команд:

а)
```
сместиться в точку (0, 0)
опустить перо
нц для k от 1 до 3
. сместиться на вектор (k, 0)
. сместиться на вектор (0, —k)
. сместиться на вектор (—k—1, 0)
. сместиться на вектор (0,k+1)
кц
```

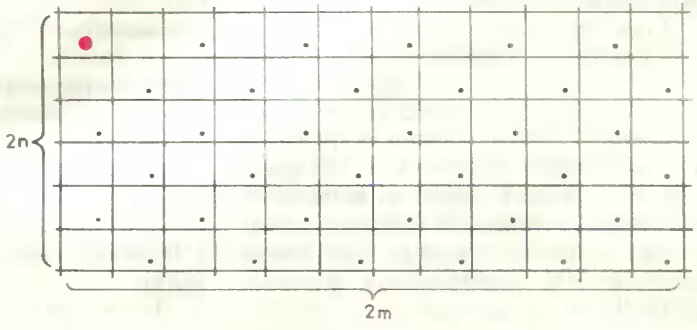
б)
```
сместиться в точку (0, 0)
опустить перо
нц для k от 2 до 5
. сместиться на вектор (k, 0)
. сместиться на вектор (0, —k—1)
. сместиться на вектор (—k—2, 0)
. сместиться на вектор (0, k+3)
кц
```

22. Составьте алгоритм для Чертежника, рисующий приближенно график функции $y=x^2-1$ на отрезке $[-1,1]$.

23. Составьте алгоритм с целочисленными аргументами m и n, который с помощью Робота закрашивает клетки, отмеченные точками (рис. 9).

Продолжение следует.

9



Н. УГРИНОВИЧ

Об одном методическом подходе

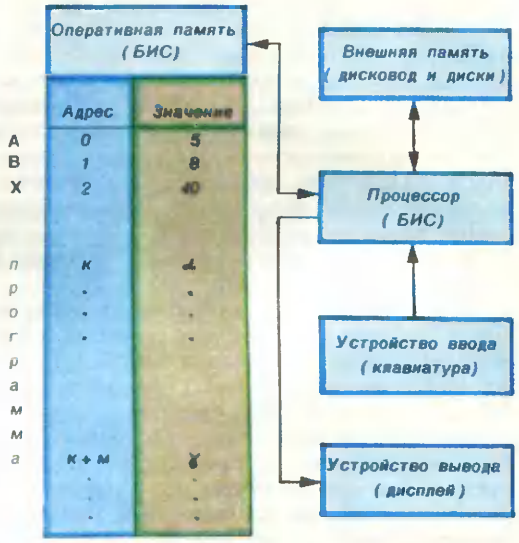
В настоящее время многие учителя и методисты считают нецелесообразным изучение темы «Устройство ЭВМ» в курсе ОИВТ. Действительно, пользователь-непрофессионал может работать на компьютере, рассматривая его как «черный ящик». Однако, если мы формируем у учащихся алгоритмическое мышление, учим основам программирования, без знания структуры ЭВМ им не обойтись. Так, например, практически невозможно объяснить учащимся сущность команды присваивания и ее принципиальное отличие от алгебраического равенства без обращения к структуре памяти ЭВМ.

Опыт преподавания курса ОИВТ показал, что в целях лучшего понимания конструкций языка программирования целесообразно в процессе изучения темы «Алгоритмизация и программирование» опираться на общее представление учащихся об устройстве ЭВМ, в частности на представление о структуре оперативной памяти.

Таким образом, уже в начале изучения курса полезно дать учащимся представление об ЭВМ, основных устройствах ЭВМ, их функциях и взаимосвязи в процессе работы. Основным методическим принципом изложения этого материала должен быть переход от функций тех или иных устройств к кратким сведениям о них, в том числе и об их физической реализации с помощью конкретных технических устройств. Параллельно необходимо показывать учащимся, где эти устройства располагаются в реальном компьютере. В процессе объяснения целесообразно рисовать структурную схему ЭВМ (рис. 1).

Логично начать с основной функции ЭВМ, заключающейся в обработке информации. Любая информация в ЭВМ кодируется на машинном языке (двоичным кодом из нулей и единиц), а физически реализуется в виде электрических импульсов (нет напряжения — 0, есть напряжение — 1). Центральным блоком ЭВМ, где производится обработка информации, является процессор, аппаратно реализующийся на БИС.

Однако человек не понимает информацию, представленную в виде электрических импульсов. Поэтому необходимы специальные устройства ввода-вывода, которые переводят информацию с языка человека на язык ЭВМ и обратно. В качестве устройства ввода используется обычно клавиатура. Человек нажимает на определенную клавишу с каким-либо символом, в ЭВМ поступает определенная последовательность электрических импульсов. Как человек различает символы по их начертаниям, так ЭВМ различает их по различным последовательностям электрических импульсов. Каждому символу соответствует своя последовательность электрических импульсов.



В качестве устройства вывода информации используется дисплей (черно-белый или цветной), на экране которого высвечивается числовая, текстовая или графическая информация. Необходимо обратить внимание учащихся на то, что в качестве дисплея желательно использовать не бытовой телевизор, а специальный монитор, качество изображения которого выше (разрешающая способность экрана, стабильность изображения, цветовая гамма). Для сохранения информации в виде «твердой копии» на бумаге используется принтер. Для вывода на бумагу сложных чертежей и схем используется графопостроитель.

Важным моментом является программный принцип работы ЭВМ. Для хранения программы и данных, с которыми программа работает, необходимо специальное устройство — внутренняя, или оперативная, память. Она состоит из отдельных адресуемых ячеек, в которых может храниться, записываться и стираться информация. Количество таких ячеек велико (в БК-0010, например,

32768). Аппаратно оперативная память реализуется на БИСах.

Однако при выключении ЭВМ из сети вся информация в памяти стирается. Для долговременного хранения большого количества программ или данных имеется еще одно устройство — внешняя, или постоянная, память. Это либо бытовой кассетный магнитофон, либо специальное устройство — дисковод. Оба они используют один и тот же магнитный принцип записи: на магнитной ленте или диске записывается информация (нет намагниченности — 0, есть намагниченность —1). Объем такой памяти достаточно велик, на одном диске может быть записана книга, например учебник ОИВТ.

Желательно кратко рассказать учащимся о процессе работы на ЭВМ.

Важно обратить их внимание на возможность соединения ЭВМ между собой в информационную локальную сеть. Это позволяет иметь внешнюю память не на каждой ЭВМ, а только на ЭВМ учителя.

32

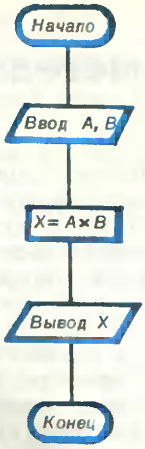
Начинать обучение языку программирования целесообразно с простейших, с точки зрения их математического содержания, алгоритмов, чтобы все внимание учащихся концентрировалось на вопросах собственно программирования, а не на сложностях построения математической модели и алгоритма.

Введение отдельных операторов языка производится по мере необходимости в процессе кодирования все более сложных алгоритмов на Бейсике (Фокале). В процессе изучения основных алгоритмических структур (линейной, разветвляющейся, циклической) важную роль играют блок-схемы, которые позволяют представить их в наглядной форме и проследить выполнение в динамике.

В качестве первого примера можно рассмотреть алгоритм умножения двух чисел (рис. 2). Этот пример позволяет ввести четыре оператора Бейсика (Фокала), рассмотреть их функции в программе, проследить, как ЭВМ выполняет программу, как это отражается на устройстве вывода (дисплее) и какие действия должен произвести пользователь при работе с программой на устройстве ввода (клавиатуре).

К моменту начала изучения языка программирования учащиеся знакомы с алгоритмическим языком и умеют записывать на нем алгоритмы, поэтому сначала целесообразно записать этот алгоритм умножения двух чисел (УДЧ) на алгоритмическом языке.

```
алг УДЧ (вещ A, B, X)
    арг A, B
    рез X
```




```
нач
    X:=A*B
кон
```

Затем следует закодировать этот алгоритм на Бейсике (Фокале), последовательно вводя необходимые операторы.

Бейсик

```
10 INPUT A, B
20 LET X=A*B
30 PRINT X
40 END
```

Фокал

```
1.10 ASK A, B
1.20 SET X=A*B
1.30 TYPE X
1.40 QUIT
```

Работа пользователя на ЭВМ с программой может быть описана в виде последовательности следующих действий:

1) пользователь вводит программу в память ЭВМ. После ввода она займет в памяти некоторую область (например, ячейки от K до K+M);

2) пользователь запускает программу на выполнение. Процессор последовательно считывает из памяти операторы и их выполняет;

3) в случае ошибки в тексте программы ЭВМ выдаст на дисплей сообщение об этой ошибке. В таком случае необходимо произвести редактирование программы. Этот процесс называется отладкой.

Средства редактирования и отладки программ в различных языках и на различных типах ЭВМ существенно различаются. Для удаления строк программы в Бейсике используется оператор DELETE, а в Фокале ERASE. Для частичной корректировки уже введенной строки в Фокале используется оператор MODIFY. Учащихся необходимо познакомить с этими средствами

уже при работе с первыми программами.

Теперь необходимо «проиграть» выполнение программы на ЭВМ и рассмотреть, что происходит в различных устройствах ЭВМ при выполнении того или иного оператора, и пояснить таким образом учащимся функцию, которую выполняет каждый оператор в программе.

INPUT (ASK) — оператор ввода значений переменных. ЭВМ отводит в оперативной памяти области (некоторое количество ячеек) и «называет» их именами переменных из списка ввода, запрашивает у пользователя их значения (на экране дисплея появляется знак ? или :). Пользователь вводит значения переменных с клавиатуры; ЭВМ записывает эти значения в отведенные области памяти. После этих объяснений уточняется формат оператора.

LET (SET) — оператор присваивания. ЭВМ считывает из памяти значения переменных, составляющих арифметическое выражение, вычисляет значение арифметического выражения, отводит в памяти область под определяемую переменную, записывает вычисленное значение арифметического выражения в эту область.

PRINT (TYPE) — оператор вывода значений переменных на экран. ЭВМ считывает значения переменных из памяти и высвечивает эти значения на экране дисплея.

END (QUIT) — оператор окончания программы. ЭВМ заканчивает выполнение программы и переходит в режим ожидания. На экране дисплея появляется соответствующее сообщение (Ok, ready, жду и т. д.).

Предлагаемые объяснения содержат некоторые упрощения, опущены детали, которые будут изложены в дальнейшем. Важно донести до учащихся главное, существенное, в частности раскрыть содержание понятия переменной величины в программировании. Аналогично можно рассмотреть семантический смысл других операторов в процессе рассмотрения разветвляющихся и циклических алгоритмов.

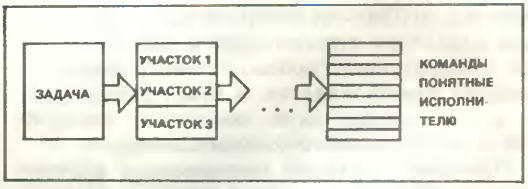
С. СОЧНЕВ

Путь от простого к сложному, или покорение алгоритма

В учебном пособии по ОИВТ [1] содержится немало задач, требующих нестандартного подхода. Однако зачастую, будь то на уроке или на занятиях кружка, преподаватель сталкивается с тем, что учащиеся в лучшем случае могут разобраться только в готовом решении задачи, провести же самостоятельные рассуждения они не могут. Учащиеся уподобляются пассажирам авиалайнера. Движимый энергией учителя, он поднимается на большую высоту, с которой видны вершины горной страны Алгоритм. Однако от этого они не становятся более доступными.

Один из возможных выходов из этой ситуации следующий: необходимо постепенное усложнение заданий, переход от простого к сложному. Занятия надо строить таким образом, чтобы решение одной задачи указывало путь к решению другой, более сложной задачи, чтобы учащиеся пусть медленно, шаг за шагом, но самостоятельно покоряли вершины Алгоритма.

Важную роль в таком продвижении может играть метод последовательного уточнения алгоритма (см. [1], с. 48—51). Метод будет более понятен учащимся, если его проиллюстрировать с помощью схемы, изображенной на рис. 1. Заметим, что эта схема не требует пояснений, глядя на нее даже первый раз, учащиеся сами в состоянии сформулировать суть метода. В дальнейшем метод последовательного уточнения алгоритма следует рассматривать не только в контексте вспомогательных алгоритмов, а как один из принципов программирования.



1

Итак, опираясь на этот метод, мы предлагаем вам совершить восхождение на некоторые вершины необъятной горной страны Алгоритм.

Восхождение 1. «Пила»

Задача 1.1. Подсчитайте наибольшее количество расположенных последовательно эле-

ментов таблицы A [1:20], образующих «пилу» зубьями вверх, т. е. удовлетворяющих следующим условиям:
A[i]<A[i+1]>A[i+2]<A[i+3]>...<
<A[i+K]

Прежде чем начинать решение этой задачи, необходимо вспомнить алгоритмы «количество» и «максимум» [2] (задачи 8, 10). Затем попробуем решить следующую достаточно простую задачу:

Задача 1.2. Выясните количество расположенных последовательно положительных элементов, которыми заканчивается таблица A [1:20]. Если последний элемент таблицы — число неположительное, то ответ считать равным нулю.

Опираясь на алгоритм «количество», получаем следующее решение задачи:

```
алг КОЛИЧЕСТВО-1 (вещ таб A [1:20],
цел K)
  арг A
  рез K
нач нат i
  K:=0
  для i от 1 до 20
  нц
    если A[i]>0
      то K:=K+1
      иначе K:=0
    все
  кц
кон
```


По команде K:=0 после слова **иначе** производится сброс накопленного количества идущих подряд положительных элементов при появлении отрицательного числа или нуля; со следующего положительного элемента подсчет возобновляется. K будет равно нулю и в том случае, если последний элемент таблицы — число неположительное.

Отметим, что сброс накопленного результата — прием довольно распространенный. Его целесообразно использовать при решении ряда задач. Попробуйте, например, использовать этот прием при решении такой задачи: «в таблице A [1:100] подсчитайте количество элементов, имеющих максимальное значение».

Однако продолжим наше восхождение.

Задача 1.3. В таблице A [1:20] подсчитайте наибольшее количество расположенных последовательно положительных элементов.

Для решения этой задачи используем метод последовательного уточнения. При этом будем опираться на решение задачи 1.2 и алгоритм «максимум». Решение сведется к вычислению наибольшего из всех значений K, полученных в задаче 1.2 перед сбрасыванием старого результата и возобновлением подсчета с K, равного нулю (т. е. в момент появления неположительного элемента). Наибольшее среди всех K обозначим KMAX. Тогда схема алгоритма будет следующая:

1. K:=0; KMAX:=0
2. Для i от 1 до 20
нц
  если A[i]>0
    то K:=K+1
    иначе сравнить K и KMAX, большее из их значений присвоить KMAX;
      K:=0
  все
кц
3. Сравнить K и KMAX, большее из их значений присвоить KMAX. Появление 3-го шага в алгоритме связано с тем, что таблица может заканчиваться положительным числом, и при отсутствии этого шага последнее значение K не сравнивалось бы с KMAX.

Произведем теперь уточнение шагов 2 и 3.

```
алг КОЛИЧЕСТВО-2 (вещ таб A[1:20],
цел KMAX)
  арг A
  рез KMAX
нач нат i, цел K
  K:=0; KMAX:=0
  для i от 1 до 20
  нц
    если A[i]>0
      то K:=K+1
      иначе если K>KMAX
        то KMAX:=K
      все
      K:=0
    все
  кц
  если K>KMAX
    то KMAX:=K
  все
кон
```

У составителя алгоритма наверняка вызовет неудовлетворение повторение участка сравнения K и KMAX дважды. Этого можно избежать, производя сравнение по ветви **то**, т. е. при каждом увеличении K на единицу. При этом необходимость последнего (3-го) шага алгоритма отпадает.

```
алг КОЛИЧЕСТВО-3 (вещ таб A[1:20],
цел KMAX)
  арг A
  рез KMAX
нач нат i, цел K
  K:=0; KMAX:=0
  для i от 1 до 20
  нц
    если A[i]>0
      то K:=K+1
        если K>KMAX
          то KMAX:=K
```

```
        все
        иначе K:=0
    все
  кц
кон
```

На первый взгляд может показаться, что выполнение этого алгоритма на ЭВМ займет значительно больше времени, чем предыдущего. Действительно, если в таблице А 17 положительных элементов, то условие K>KMAX придется проверять 17 раз, в то время как в алгоритме КОЛИЧЕСТВО-2 оно будет проверяться не более 4 раз. Однако в обратной ситуации, когда в таблице А 17 отрицательных элементов, алгоритм КОЛИЧЕСТВО-3 выполняется значительно быстрее, чем КОЛИЧЕСТВО-2.

Теперь перейдем к решению задачи 1.1. Очевидно, что алгоритм ее решения будет во многом похож на только что рассмотренную задачу. Внесем необходимые изменения:

1. К будем считать не с нуля, а с единицы, так как любой элемент таблицы всегда можно рассматривать в качестве начала «пилы».

2. Попробуем определить условие, которое надо поставить вместо $A[i]>0$. Между элементами, образующими «пилу», должны выполняться следующие соотношения:

при K=1 $A[1]<A[2]$ или $-A[1]>-A[2]$;
при K=2 $A[2]>A[3]$;
при K=3 $A[3]<A[4]$ или $-A[3]>-A[4]$;
при K=4 $A[4]>A[5]$;

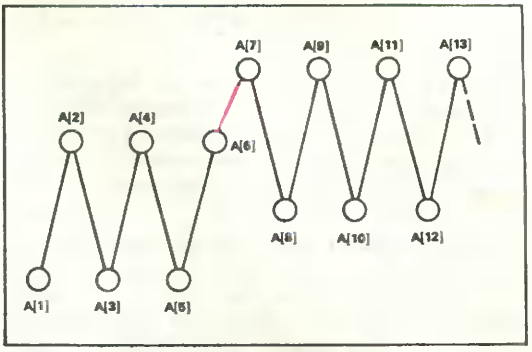
в общем случае $(-1)^k A[i]>(-1)^k A[i+1]$.

3. Так как элементы берутся парами ($A[i]$ и $A[i+1]$), перебирать элементы следует с 1-го номера по 19-й.

Произведя эти изменения, получим алгоритм ПИЛА:

```
алг ПИЛА (вещ таб A[1:20], нат KMAX)
    арг A
    рез KMAX
нач нат i, K
    K:=1; KMAX:=1
    для i от 1 до 19
    нц
      если (-1)^k·A[i]>(-1)^k·A[i+1]
          то K:=K+1
            если K>KMAX
                то KMAX:=K
            все
          иначе K:=1
      все
    кц
кон
```

К сожалению, наше восхождение еще не закончилось. Дело в том, что полученный алгоритм не учитывает ситуацию, изображенную на рис. 2. Подсчет элементов второй



2

«пилы» по нашему алгоритму будет начинаться не с 6-го, а только с 8-го элемента, так как нарушение условия $(-1)^6\cdot A[6]>(-1)^6\cdot A[7]$ фиксирует лишь окончание первой «пилы», но не начало второй. Чтобы устранить этот недостаток, необходимо последнее небольшое усложнение алгоритма.

```
алг ПИЛА (вещ таб A[1:20], нат KMAX)
    арг A
    рез KMAX
нач нат i, K
    K:=1; KMAX:=1
    для i от 1 до 19
    нц
      если (-1)^k·A[i]>(-1)^k·A[i+1]
          то K:=K+1
            если K>KMAX
                то KMAX:=K
            все
          иначе если A[i]<A[i+1]
                то K:=2
                иначе K:=1
                все
      все
    кц
кон
```

35

В этом случае при сбрасывании К проверяется, не являются ли элементы $A[i]$ и $A[i+1]$ началом новой «пилы». Последний алгоритм и является окончательным решением задачи.

Восхождение 2. Перестановка

Задача 2.1. Проверьте, является ли последовательность натуральных чисел A[1:40] перестановкой чисел 1, 2, 3, ..., 40.

Для решения этой задачи обратимся вначале к знакомому по учебному пособию алгоритму «ключ».

Задача 2.2. В таблице натуральных чисел A[1:40], не содержащей одинаковых элементов, найдите номер элемента, равного заданному числу L. Если такой элемент в таблице отсутствует, то считать ответ равным нулю.

Обычно при решении этой задачи учащиеся без труда записывают следующий алгоритм

```
алг КЛЮЧ-1 (нат таб A[1:40], нат L,
цел K)
  арг A, L
  рез K
нач нат i
  K:=0; i:=1
  пока i⩽40
  нц
    если A[i]=L
       то K:=i
    все
    i:=i+1
  кц
кон
```

Здесь следует рассмотреть вопрос о целесообразности перебора всех элементов таблицы. Очевидно, если нам попался элемент, равный L (т. е. K перестало быть равным нулю), дальнейший просмотр элементов таблицы не нужен. С учетом этого условие повторения примет вид

```
пока i⩽40 и K=0,
```


и мы получим алгоритм, аналогичный записанному в учебном пособии [1] (с. 57). Несмотря на более длинную запись условия, исполнение алгоритма упрощается, и его выполнение теперь потребует меньше времени.

Дальнейшее развитие этот алгоритм получил в работе [3] (алгоритм ПОИСК). С незначительными изменениями он выглядит так:

```
алг КЛЮЧ-2 (нат таб A[1:40], нат L,
цел K)
  арг A, L
  рез K
нач нат i
  K:=0; i:=1
  пока i⩽40
  нц
    если A[i]=L
       то K:=i; i:=100
       иначе i:=i+1
    все
  кц
кон
```

Если в таблице встречается элемент, равный L, то i становится равным 100, и по нарушению условия повторения $i \leqslant 40$ осуществляется выход из цикла. Этот алгоритм более простой по сравнению с алгоритмом, записанным в [1], так как при каждом увеличении на единицу вместо составного условия $i \leqslant 40$ и K=0 проверяется простое условие $i \leqslant 40$, тем самым алгоритм будет выполняться быстрее. Заметим также, что если в таблице есть элемент, равный L, то по окончании алгоритма i будет равно 100, в противном случае — 41.

Рассмотренный прием изменения значения переменной, управляющей циклом, для выхода из этого цикла можно с успехом применить для решения многих задач, и мы советуем его также взять «на вооружение». Попробуйте, например, использовать этот прием при решении такой задачи: «Проверьте, совпадают ли таблицы чисел A[1:100] и B[1:100] (будем называть таблицы совпадающими, если A[1]=B[1], A[2]=B[2], ..., A[100]=B[100])».

А теперь рассмотрим следующую задачу.

Задача 2.3. Выясните, имеется ли в таблице натуральных чисел A[1:40] элемент, равный L.

Для решения этой задачи используем алгоритм КЛЮЧ-2. Так как номер элемента K нас уже не интересует, все команды, связанные с K, из алгоритма исключаем. О результате нам будет сигнализировать значение переменной i: если такой элемент есть, то i после выполнения алгоритма станет равным 100, иначе — 41 (см. выше).

```
алг ПРОВЕРКА (нат таб A[1:40], нат L,
i)
  арг A, L
  рез i
нач
  i:=1
  пока i⩽40
  нц
    если A[i]=L
       то i:=100
       иначе i:=i+1
    все
  кц
кон
```

Для решения задачи 2.1 алгоритм ПРОВЕРКА надо выполнить для всех L от 1 до 40. Причем, если для какого-либо значения L в результате выполнения алгоритма ПРОВЕРКА i не стало равным 100, процесс можно прекращать, так как такого числа в таблице нет и, следовательно, таблица не является перестановкой чисел 1, 2, 3, ..., 40. Таким образом, условие повторения будет иметь вид

```
пока L⩽40 и i=100.
```

О результате работы алгоритма нам будет сигнализировать переменная i. Если и при последнем, сороковом выполнении алгоритма ПРОВЕРКА i окажется равным 100 (заметим, что до сорокового раза мы дойдем только в том случае, если и до этого i каждый раз оказывалось равным 100), тогда таблица A является перестановкой чисел от 1 до 40. Если же в результате работы алгоритма i окажется равным 41, ответ будет отрицательным.

Суммируя все сказанное, схему алгоритма можно представить в следующем виде:

1. L:=1; i:=100

```
2. пока L≤40 и i=100
   нц
     выполнять алгоритм ПРОВЕРКА
     L:=L+1
   кц
```

Произведя уточнение алгоритма, имеем:

```
алг ПЕРЕСТАНОВКА (нат таб A[1:40], нат i)
   арг A
   рез i
нач нат L
   L:=1; i:=100
   пока L≤40 и i=100
   нц
     i:=1
     пока i≤40
     нц
       если A[i]=L
         то i:=100
         иначе i:=i+1
       все
     кц
     L:=L+1
   кц
кон
```

Восхождение 3. Положительные, вперед!

Задача 3.1. Дана таблица A[1:50], элементы которой отличны от нуля. Расположите их в таком порядке, чтобы первыми были все положительные элементы, а затем — все отрицательные, причем порядок следования как положительных, так и отрицательных элементов должен сохраниться. При решении задачи новую таблицу не заводить.

Для этого восхождения нам потребуется вспомнить алгоритм «обмен» [2] (задача 7). Одной из разновидностей этого алгоритма является алгоритм СДВИГ.

Задача 3.2. Переставьте элементы таблицы A[1:50] таким образом, чтобы они шли в следующем порядке: A[2], A[3], A[4], ..., A[50], A[1], т. е. в таблице произведен сдвиг на один элемент влево.

Учащиеся, усвоившие алгоритм «обмен», легко справятся с этой задачей. Ее решение будет иметь вид:

```
алг СДВИГ (вещ таб A[1:50])
   арг A
   рез A
нач нат i, вещ C
   C:=A[1]
   для i от 2 до 50
   нц
     A[i—1]:=A[i]
   кц
   A[50]:=C
кон
```

Решать задачу 3.1 начнем с того, что пренебрежем всеми ограничениями, имеющимися в условии. Тогда получим следующее:

Задача 3.3. Дана таблица A[1:50], элементы которой отличны от нуля. Постройте таблицу B[1:50], в которой первыми расположены все положительные элементы таблицы A, а далее — все отрицательные элементы таблицы A.

Эту задачу можно решить за два просмотра таблицы A. При первом просмотре мы будем заполнять таблицу B положительными элементами, при втором — отрицательными.

```
алг ТАБЛИЦА-1 (вещ таб A[1:50], B[1:50])
   арг A
   рез B
нач нат i, k
   i:=1
   для k от 1 до 50
   нц
     если A[k]>0
       то B[i]:=A[k]; i:=i+1
     все
   кц
   для k от 1 до 50
   нц
     если A[k]<0
       то B[i]:=A[k]; i:=i+1
     все
   кц
кон
```

Теперь попробуем решить эту задачу за один просмотр, заполняя таблицу B с двух сторон: положительными элементами, начиная с первого номера и увеличивая значение номера на 1, и отрицательными элементами, начиная с последнего номера и уменьшая его значение на 1.

```
алг ТАБЛИЦА-2 (вещ таб A[1:50], B[1:50])
   арг A
   рез B
нач нат i, j, k
   i:=1; j:=50
   для k от 1 до 50
   нц
     если A[k]>0
       то B[i]:=A[k]; i:=i+1
       иначе B[j]:=A[k]; j:=j—1
     все
   кц
кон
```

Теперь к условию задачи 3.3 добавим ограничение: вторую таблицу заводить нельзя.

Задача 3.4. Переставьте элементы таблицы A[1:50], не содержащей нулей, таким образом, чтобы вначале шли все положительные элементы, а затем — все отрицательные.

При решении задачи новую таблицу не заводить.

Для решения этой задачи будем перебирать элементы таблицы с 1-го по 50-й. В случае A[*i*]>0 никаких изменений в таблице не производим, увеличиваем *i* на единицу и переходим к следующему элементу. Если же элемент A[*i*] отрицателен, поменяем его местами с элементом A[*j*] (где *j* первоначально равно 50) и уменьшим *j* на единицу. Если и этот элемент отрицателен, вновь меняем его местами с A[*j*] (теперь это уже предпоследний элемент, так как *j* уменьшилось на единицу) — и так далее, пока не попадется положительный элемент. Алгоритм будет иметь вид

1. *i*:=1; *j*:=50
2. выполнить 50 раз
   нц
   если A[*i*]>0
   то *i*:=*i*+1
   иначе обменять местами A[*i*] и A[*j*].
   j:=*j*—1
   все
   кц

Произведем уточнение шагов алгоритма:

```
алг ТАБЛИЦА-3 (вещ таб A[1:50])
  арг A
  рез A
нач нат i, j, k, вещ C
  i:=1; j:=50
  для k от 1 до 50
  нц
    если A[i]>0
      то i:=i+1
      иначе C:=A[i]; A[i]:=A[j]; A[j]:=C
      j:=j—1
    все
  кц
кон
```

Такой алгоритм изменяет порядок следования элементов таблицы. Чтобы порядок не нарушался (в этом случае мы имеем задачу 3.1), по ветви **иначе** предыдущего алгоритма вместо обмена элементов A[*i*] и A[*j*] надо, прежде чем поместить элемент A[*i*] в конец таблицы, произвести сдвиг элементов от A[i+1] до A[50] на один номер влево, т. е. по ветви **иначе** выполнить алгоритм СДВИГ для элементов с номерами от *i* до 50.

```
алг ТАБЛИЦА-4 (вещ таб A[1:50])
  арг A
  рез A
нач нат i, j, k, вещ C
  i:=1
  для k от 1 до 50
  нц
    если A[i]>0
      то i:=i+1
      иначе C:=A[i]
           для j от i+1 до 50
           нц
             A[j—1]:=A[j]
           кц
           A[50]:=C
    все
  кц
кон
```

Это и будет окончательное решение задачи. Первый отрицательный элемент при выполнении алгоритма займет вначале последнее место в таблице, а в дальнейшем при появлении каждого нового отрицательного элемента будет смещаться на один номер влево. Все остальные отрицательные элементы будут располагаться правее его, причем в том же порядке, что и в первоначально заданной таблице.

Итак, третья вершина покорена. Нам остается пожелать читателю не останавливаться на этом, а двигаться и дальше, к новым вершинам Алгоритма.

Литература

1. Основы информатики и вычислительной техники: Пробное учебное пособие. Ч. 1/Под ред. А. П. Ершова, В. М. Монахова. М.: Просвещение, 1985.

2. *Сочнев С.* Изучение алгоритмического языка в курсе информатики // Информатика и образование. 1988. № 1. С. 38—42.

3. *Фукс Л.* Алгоритмы поиска и сортировки // Информатика и образование. 1988. № 2. С. 6—21.

Н. ГЛАГОЛЕВА, Н. ЮНЕРМАН
г. Новосибирск

Основные понятия языка Рапира

Кортежи

Рассмотрим следующую задачу. Дан произвольный текст. Нужно определить, сколько раз в этом тексте встречается каждая буква алфавита, и записать полученные результаты в таблицу, т. е. построить частотный словарь для этого текста. Можно, конечно, воспользоваться написанной раньше процедурой АЛФАВИТ_3. Но при этом полученные результаты придется записывать в таблицу вручную.

Нельзя ли формирование, хранение и вывод таблицы тоже «переложить» на машину? С помощью известных нам объектов языка (чисел, текстов и т. п.) трудно записать таблицу как единое целое. Для записи таких составных объектов в Рапире используются кортежи.

Кортеж — это упорядоченная последовательность элементов. Элементом кортежа может быть число, текст, кортеж и т. п., т. е. все, что может быть значением имени в Рапире. Кортеж обозначают треугольными скобками ◀ и ▶, внутри которых через запятую записывают по порядку его элементы. Порядковый номер элемента в кортеже называется *индексом* этого элемента.

Примеры кортежей:

1) ◀ «ПН», «ВТ», «СР», «ЧТ», «ПТ», «СБ», «ВС»▶
2) ◀ «ИВАНОВ», «ХИМИЯ», 5, 4, 4, 5 ▶

Здесь: ИВАНОВ — фамилия ученика; ХИМИЯ — название предмета; числа — оценки ученика.

3) От каждой школы запросили сведения: номер, названия факультативов, проводимых в школе, наличие кабинета информатики. Эту информацию можно записать, например, в виде такого кортежа:

◀7, ◀ «ФИЗИКА», «ХИМИЯ» ▶,«Н» ▶

Здесь: 7 — номер школы; ◀ «ФИЗИКА», «ХИМИЯ» ▶ — кортеж из названий факультативов, которые есть в школе; Н — нет кабинета информатики.

Над кортежами можно выполнять такие же операции, как и над текстами. В таблице приведены сведения об этих операциях.

Название	Обозначение	Что служит результатом
Длина	#	Натуральное число или 0 (число элементов)
Выборка		Любой объект (элемент кортежа)
Вырезка	[:]	Кортеж
Соединение	+	Кортеж
Умножение	*	Кортеж

Обратите внимание, что если КОР — кортеж, а НОМ — натуральное число, то значения выражений КОР[НОМ] и КОР[НОМ:НОМ] не совпадают (в отличие от текстов)! Значение КОР[НОМ] — элемент кортежа (т. е. любой объект языка — число, текст, кортеж и т. п.), а значение выражения КОР[НОМ:НОМ] — кортеж, состоящий из этого элемента.

Рассмотрите внимательно, как выполняются эти операции.

```
Операции над кортежами                Результаты

#◀"ПН","ВТ","СР","ЧТ","ПТ"▶           5                          число
◀7,◀"ФИЗИКА","ХИМИЯ"▶,"Н"▶[1]         7                          число
◀7,◀"ФИЗИКА","ХИМИЯ"▶,"Н"▶[2]         ◀"ФИЗИКА","ХИМИЯ"▶         кортеж
◀7,◀"ФИЗИКА","ХИМИЯ"▶,"Н"▶[3]         Н                          текст
```

Окончание. Начало см.: Информатика и образование. 1989. № 1—3.

```
⟨7,⟨"ФИЗИКА","ХИМИЯ"⟩,"Н"⟩[1:1]  ⟨7⟩                    кортеж
⟨"КОТОВ","ХИМИЯ",5,4,4,5⟩[3:]   ⟨5,4,4,5⟩              кортеж
⟨"КОТОВ","ХИМИЯ",5,4,4,5⟩+⟨3⟩   ⟨"КОТОВ","ХИМИЯ",5,4,4,5,3⟩ кортеж
4ж⟨0⟩                           ⟨0,0,0,0⟩              кортеж
```

Примеры использования кортежей в предписаниях.

```
НЕДЕЛЯ:=⟨"ПН","ВТ","СР","ЧТ","ПТ","СБ","ВС"⟩
ВЫВОД: "УЧЕБНЫЕ ДНИ:"
ВЫВОД: НЕДЕЛЯ[:6]
ВЫВОД: "ВЫХОДНОЙ ДЕНЬ: ", НЕДЕЛЯ[7]
НЕДЕЛЯ:=7ж⟨"ПТ"⟩ \ семь пятниц на неделе
ВЫВОД:НЕДЕЛЯ
ШК:=⟨7, ⟨"ФИЗИКА","ХИМИЯ"⟩,"Н"⟩
ШК[3]:="Д"    \ в школе оборудован кабинет информатики
ШК[2]:=ШК[2]+⟨"ИНФОРМАТИКА"⟩
   \ добавлен факультатив по информатике
ВЫВОД:ШК
ВЫВОД:"В ШКОЛЕ ФАКУЛЬТАТИВОВ:", #ШК[2]
```

После исполнения этих предписаний будет напечатано:

```
УЧЕБНЫЕ ДНИ:
⟨"ПН","ВТ","СР","ЧТ","ПТ","СБ"⟩
ВЫХОДНОЙ ДЕНЬ: ВС
⟨"ПТ","ПТ","ПТ","ПТ","ПТ","ПТ","ПТ"⟩
⟨7, ⟨"ФИЗИКА","ХИМИЯ","ИНФОРМАТИКА"⟩,"Д"⟩
В ШКОЛЕ ФАКУЛЬТАТИВОВ:3
```


Все сведения о школах района собрали вместе. Представим эту информацию в виде кортежа ШКОЛЫ.

```
⟨⟨130,⟨"ИСТОРИЯ","ФИЗИКА"⟩,      "Н"⟩,
 ⟨7,  ⟨"ФИЗИКА","ХИМИЯ","ИНФОРМАТИКА"⟩,"Д"⟩,
 ⟨105,⟨"ИНФОРМАТИКА","БИОЛОГИЯ"⟩,"Н"⟩,
  ...                                  ⟩
```

Значением выражения ШКОЛЫ [1] является кортеж ◀ 130, ◀ «ИСТОРИЯ», «ФИЗИКА» ▶, «Н» ▶ , а выражения ШКОЛЫ [1] [2] — кортеж ◀ «ИСТОРИЯ», «ФИЗИКА» ▶, а выражения ШКОЛЫ [1] [2] [1] — текст ИСТОРИЯ. При записи подряд нескольких операций выборки пара рядом стоящих квадратных скобок] [может быть заменена запятой. Например, значением выражения ШКОЛЫ [2, 2, 3] является текст ИНФОРМАТИКА.

Решим несколько задач.

1) Напечатать номера всех школ района.

```
ВЫВОД: "В РАЙОНЕ ЕСТЬ ШКОЛЫ:"
ДЛЯ НОМ ОТ 1 ДО #ШКОЛЫ
ЦИКЛ
  ВЫВОД: ШКОЛЫ [НОМ, 1]
КЦ
```

2) Определить число школ района, в которых есть кабинеты информатики.

```
СУММА:=0
ДЛЯ НОМ ОТ 1 ДО #ШКОЛЫ
ЦИКЛ
  ЕСЛИ ШКОЛЫ[НОМ,3]="Д"
   ТО СУММА:=СУММА+1
  ВСЕ
КЦ
ВЫВОД:"В РАЙОНЕ КАБИНЕТ ИНФОРМАТИКИ"
ВЫВОД:"ИМЕЮТ",СУММА," ШКОЛ"
```

3) Напечатать номера школ, имеющих факультатив по информатике.

Для решения этой задачи оказывается полезной стандартная функция ИНДЕКС(ЧТО, ГДЕ), предназначенная для работы с кортежем или текстом.

Для кортежа это поиск объекта ЧТО в кортеже ГДЕ. Результат функции — индекс первого из элементов кортежа, равных значению ЧТО, или 0, если таких элементов нет. Например:

ИНДЕКС(5, ◀ 3, 4, 5, 7, 5 ▶)=3

ИНДЕКС(2*2, ◀ 3, 5, 8 ▶)=0

Для текста функция ИНДЕКС — это поиск вхождения текста ЧТО в текст ГДЕ. Результат функции — индекс первого вхождения текста ЧТО в текст ГДЕ, или 0, если текст ЧТО не содержится в тексте ГДЕ. Например:

ИНДЕКС(«КУ», «КУКАРЕКУ»)=1

ИНДЕКС(«КУКУ», «КУКАРЕКУ»)=0

Теперь решение задачи можно записать так:

```
ВЫВОД:"ИМЕЮТ ФАКУЛЬТАТИВ ПО ИНФОРМАТИКЕ"
ДЛЯ НОМ ОТ 1 ДО #ШКОЛЫ
ЦИКЛ
 ЕСЛИ
   ИНДЕКС("ИНФОРМАТИКА",ШКОЛЫ[НОМ,2])/=0
  ТО ВЫВОД: "ШКОЛА N", ШКОЛЫ[НОМ,1]
 ВСЕ
КЦ
```

4) Определить число разных факультативов в школах района.

```
ФАК := ◀ ▶
ДЛЯ НОМ ОТ 1 ДО #ШКОЛЫ
ЦИКЛ
 ФАК_ШК:= ШКОЛЫ[НОМ,2]
 ДЛЯ Н ОТ 1 ДО #ФАК_ШК
 ЦИКЛ
  ЕСЛИ ИНДЕКС(ФАК_ШК[Н], ФАК)=0
   ТО ФАК := ФАК + ◀ ФАК_ШК[Н] ▶
  ВСЕ
 КЦ
КЦ
ВЫВОД:"В ШКОЛАХ", #ФАК, " ФАКУЛЬТАТИВОВ"
```


5) Во всех школах района, где есть кабинет информатики, ввели факультатив по информатике. Внести соответствующие изменения в кортеж ШКОЛЫ.

```
ДЛЯ НОМ ОТ 1 ДО #ШКОЛЫ
ЦИКЛ
 ЕСЛИ ШКОЛЫ[НОМ,3]="Д"  И
   (ИНДЕКС("ИНФОРМАТИКА",ШКОЛЫ[НОМ,2])=0)
   ТО ШКОЛЫ[НОМ,2]:=
          ШКОЛЫ[НОМ,2] + ◀ "ИНФОРМАТИКА"▶
 ВСЕ
КЦ
```

До сих пор мы рассматривали условия, состоящие из двух выражений, соединенных знаком сравнения. Сравнение — это операция с результатом «условие истинно» или «условие ложно». В языке для этих результатов есть специальные обозначения: ДА («условие истинно») и НЕТ («условие ложно»). Таким образом, условие — это частный случай выражения, у которого может быть только значение ДА или НЕТ.

Примеры.

Условие	Значение
5>2	ДА
«А»=«Б»	НЕТ
ИНДЕКС(«А», «ШКОЛА»)=1	НЕТ
◀1,2,3▶/=◀3,2,1▶	ДА

А	В	А И В	А ИЛИ В	НЕ А
ДА	ДА	ДА	ДА	НЕТ
ДА	НЕТ	НЕТ	ДА	НЕТ
НЕТ	ДА	НЕТ	ДА	ДА
НЕТ	НЕТ	НЕТ	НЕТ	ДА

Значения ДА и НЕТ называются *логическими*. Над ними можно выполнять следующие операции: И (логическое умножение), ИЛИ (логическое сложение) и НЕ (логическое отрицание). Результаты этих операций в зависимости от значений А и В приведены в таблице.

Эти же операции можно использовать для записи более сложных условий. Вот примеры таких условий и их значения.

Условие	Значение
(5>2) и (3=3) НЕ ИНДЕКС («А», «ШКОЛА»)=5	ДА НЕТ

Порядок выполнения операций в выражениях следующий: сначала выполняются действия в скобках, а затем все остальные операции в следующем порядке:

#
**
* / // /%
— +
> < >= <=
= /=
НЕ
И
ИЛИ

Операции, записанные выше в одной строке, имеют равные приоритеты и выполняются в том порядке, в каком они встречаются в выражении.

Значения ДА и НЕТ, как и другие объекты языка, можно присваивать именам, они могут быть элементами кортежей, результатами функций и т. п.

В следующем примере будем считать, что во всех элементах кортежа ШКОЛЫ информация о наличии кабинета информатики в школе хранится не в виде текста Д или Н, а с помощью логических значений, например:

◄ 7, ◄ «ФИЗИКА», «ХИМИЯ» «ИНФОРМАТИКА» ►, ДА ►

6) Описать функцию, дающую ответ на вопрос: «Есть ли в районе хотя бы одна школа с кабинетом информатики?»

```
ФУНК ЕСТЬ_КАБИНЕТ (ШКОЛЫ)
   ДЛЯ НОМ ОТ 1 ДО #ШКОЛЫ
   ЦИКЛ
     ЕСЛИ ШКОЛЫ [НОМ,3]
      ТО ВОЗВРАТ ДА
     ВСЕ
   КЦ
   ВОЗВРАТ НЕТ
КОНЕЦ
ВЫВОД: "В ШКОЛАХ РАЙОНА"
ЕСЛИ ЕСТЬ_КАБИНЕТ(ШКОЛЫ)
  ТО ВЫВОД: "ЕСТЬ КАБИНЕТ ИНФОРМАТИКИ"
  ИНАЧЕ ВЫВОД: "НЕТ КАБИНЕТА ИНФОРМАТИКИ"
ВСЕ
```

Можно ли решить эту задачу, не пользуясь логическими значениями? Конечно, можно. Для этого нужно ввести признак, показывающий, есть или нет кабинет информатики. Одни выберут в качестве признака какой-нибудь текст («Д», «ДА», «ЕСТЬ» и т. д.), другие будут считать, что число 1 — это ДА, а 0 — это НЕТ. Однако это неудобно, когда приходится использовать процедуры и функции, написанные независимо друг от друга, например, разными людьми. Ведь тогда приходится помнить, какое из обозначений принято в данной процедуре (функции) и знать, где 1 и 0 — числа, а где — обозначения для ДА и НЕТ.

При использовании логических значений запись условий становится компактнее, кроме того, для обработки информации можно использовать логические операции.

Задача 19. Информация о животных зоопарка хранится в кортеже ЗООПАРК, каждый элемент которого имеет вид:

◄ животное, часы кормления, рацион ►

где:

животное — текст с названием животного;

часы кормления — кортеж нескольких целых чисел от 9 до 19 в порядке возрастания;

рацион — кортеж текстов — названий пищи.

Например:

◄ «СЛОНОПОТАМ»,◄ 9,12,15,18 ► ,
◄ «ПОРОСЯТА»,«ВИТАМИН С» ► ►

а) В рацион каждого добавили РЫБИЙ ЖИР. Составьте программу для соответствующего изменения кортежа ЗООПАРК.

б) Составьте программу для печати расписания кормления на день в виде:
время — список животных.

в) В летнее время все интервалы кормления увеличили на 1 час (при этом после 19 часов животных не кормят). Составьте программу для соответствующего изменения кортежа ЗООПАРК.

Вернемся к задаче построения частотного словаря для заданного текста. Словарь-таблицу можно представить разными способами. Рассмотрим два из них.

Вариант 1. Таблица записывается в виде кортежа, каждым элементом которого является кортеж вида ◄ символ, число ► , например ◄ «А», 5 ► (символ А встречается в тексте 5 раз). При этом вначале таблица заполняется кортежами вида ◄ символ, 0 ► .

Алгоритм 1.

1. Заполнить таблицу начальными значениями.
2. Для всех символов текста выполнить действия:

а) найти в кортеже-таблице кортеж, первый элемент которого равен рассматриваемому символу;

б) увеличить на 1 значение второго элемента кортежа.

```
ФУНК СЛОВАРЬ_1(Т)
\начальное значение таблицы:
 ТАБ:= ◄◄"А",0►,◄"Б",0►,◄"В",0►,◄"Г",0►,
        ◄"Д",0►,◄"Е",0►,◄"Ж",0►,◄"З",0►,
        ◄"И",0►,◄"Й",0►,◄"К",0►,◄"Л",0►,
        ◄"М",0►,◄"Н",0►,◄"О",0►,◄"П",0►,
        ◄"Р",0►,◄"С",0►,◄"Т",0►,◄"У",0►,
        ◄"Ф",0►,◄"Х",0►,◄"Ц",0►,◄"Ч",0►,
        ◄"Ш",0►,◄"Щ",0►,◄"Ь",0►,◄"Ъ",0►,
        ◄"Ы",0►,◄"Э",0►,◄"Ю",0►,◄"Я",0►►
\ определение частот:
 ДЛЯ НОМ ОТ 1 ДО #Т
  ЦИКЛ
   ДЛЯ ИНД ОТ 1 ДО #ТАБ
   ЦИКЛ
    ЕСЛИ ТАБ[ИНД,1]=Т[НОМ]
     ТО ТАБ[ИНД,2]:=ТАБ[ИНД,2]+1
        ВЫХОД \ досрочный выход из цикла
    ВСЕ
   КЦ
  КЦ
ВОЗВРАТ ТАБ
КОНЕЦ
```

Вариант 2. Таблица записывается в виде кортежа ◄ АЗБУКА, ЧАСТОТЫ ►. Здесь АЗБУКА — это текст, состоящий из символов, для которых надо построить частотный словарь; ЧАСТОТЫ — кортеж из целых чисел — частот символов.

Начальный вид таблицы:
◄«АБ...Я»,◄ 0,0,...,0►►

Алгоритм 2.

1. Сформировать таблицу.
2. Для всех символов текста выполнить действия:

а) найти в АЗБУКА индекс этого символа;

б) в кортеже частот по этому индексу найти соответствующую частоту и увеличить ее на 1.

```
ФУНК СЛОВАРЬ_2(Т)
\начальное значение таблицы:
 ТАБ:=◄"АБВГДЕЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЭЮЯ",◄0►*32►
\определение частот:
 ДЛЯ НОМ ОТ 1 ДО #Т
  ЦИКЛ
    ИНД:=ИНДЕКС(Т[НОМ], ТАБ[1])
    ТАБ[2,ИНД]:=ТАБ[2,ИНД]+1
```

```
КЦ
ВОЗВРАТ ТАБ
КОНЕЦ
```

Задача 20. Напишите функцию, определяющую в заданном тексте букву с наибольшей частотой.

В заключение составим программу для игры «Мыслитель» (она также известна под названиями «Логика» и «Быки и коровы»).

В игре принимают участие двое. Каждый из игроков задумывает четырехзначное число с неповторяющимися цифрами, которое противник должен отгадать. Игрок, пытаясь угадать, называет очередное четырехзначное число без повторяющихся цифр. В ответ ему сообщается, сколько цифр в нем совпадает с цифрами задуманного числа и сколько из них стоит на своих местах. Например, если задумано число 1234, а названо 1356, то ответ такой: угадано цифр — 2, из них на своем месте — 1.

Реализуем «половину игры»: машина будет «загадывать» число, а человек — отгадывать.

Выберем удобный способ представления задуманного числа и чисел-ответов. Так как нас интересуют не сами числа, а их цифры, то будем хранить числа в виде кортежей из цифр.

Попробуем написать программу. Те фрагменты программы, которые сразу написать не удается, вынесем в отдельные процедуры (функции). Их мы опишем позже.


```
ПРОЦ МЫСЛИТЕЛЬ ()
  ЧИСЛО_ЦИФР:= 4
  ВЫВОД: "ИГРА МЫСЛИТЕЛЬ"
  ВЫВОД:"УГАДАЙТЕ ЧИСЛО ИЗ",
        ЧИСЛО_ЦИФР," ЦИФР"
\загадывание числа в виде кортежа
  ЗАГАДКА:= ЗАДУМКА()
\ на отгадывание числа предоставляется
\ 15 попыток
  ПОВТОР 15
  ЦИКЛ
\ввод и преобразование ответа
    ПОПЫТКА:= ВВОД_ПОПЫТКИ ()
    ЕСЛИ ПОПЫТКА = ЗАГАДКА
      ТО ВЫВОД: "МОЛОДЕЦ!"
         ВОЗВРАТ     \ выход из процедуры
    ВСЕ
    АНАЛИЗ_ПОПЫТКИ (ПОПЫТКА, ЗАГАДКА)
  КЦ
  ВЫВОД: "ОЧЕНЬ УЖ ДОЛГО"
  ВЫВОД: "ОТВЕТ ТАКОВ:",
         ВНЕШНИЙ_ВИД (ЗАГАДКА)
\ преобразование кортежа в число
КОНЕЦ
```

Опишем функцию ЗАДУМКА. Ее результат — кортеж из ЧИСЛО__ЦИФР разных цифр. Как сообщить этой функции значение имени ЧИСЛО__ЦИФР? Ведь это имя определено в вызывающей процедуре МЫСЛИТЕЛЬ. Один способ, как это сделать,— с помощью параметров — вы уже знаете. Можно поступить по-другому: описать это имя как чужое для функции ЗАДУМКА.

```
ФУНК ЗАДУМКА ()
 ЧУЖИЕ: ЧИСЛО_ЦИФР
 ЦИКЛ
   ЧИСЛО := ЦСЧ (10 ** ЧИСЛО_ЦИФР) - 1
   ВНУТР_ВИД (ЧИСЛО, <=ЗАГАДКА, <=ВЕРНО)
 КЦ ПО ВЕРНО
 ВОЗВРАТ ЗАГАДКА
КОНЕЦ
```

Процедура ВНУТР__ВИД преобразует число в кортеж, проверяя, нет ли в числе одинаковых цифр. Ее параметры: ЧИСЛО — исходное число, ЦИФРЫ — кортеж из его цифр, ВЕРНО — правильность параметра ЧИСЛО. Эта процедура еще понадобится для анализа ответов человека, поэтому необходим полный контроль параметра ЧИСЛО.

Стандартная функция ТИП__ЦЕЛ(ЧИСЛО) проверяет, является ли ее параметр целым числом. Аналогично функции ТИП__ВЕЩ, ТИП__ТЕКСТ, ТИП_КОРТ проверяют принадлежность параметра данному типу.

```
ПРОЦ ВНУТР_ВИД (ЧИСЛО, <=ЦИФРЫ, <=ВЕРНО)
 ЧУЖИЕ: ЧИСЛО_ЦИФР
 ВЕРНО:= НЕТ
 ЕСЛИ    (ТИП_ЦЕЛ (ЧИСЛО)) И
         (ЧИСЛО > 0)       И
         (ЧИСЛО < 10 ** ЧИСЛО_ЦИФР)
  ТО ЦИФРЫ:= ◄ ►
     ПОВТОР ЧИСЛО_ЦИФР
     ЦИКЛ
       ЦИФ := ЧИСЛО /% 10
       ЕСЛИ ИНДЕКС (ЦИФ, ЦИФРЫ) /= 0
        ТО ВОЗВРАТ  \ выход из процедуры
       ВСЕ
       ЦИФРЫ:= ◄ ЦИФ ► + ЦИФРЫ
       ЧИСЛО:= ЧИСЛО // 10
     КЦ
     ВЕРНО := ДА
 ВСЕ
КОНЕЦ
```

Теперь опишем функцию для обратного перевода. Ее параметр — кортеж из цифр.

```
ФУНК ВНЕШНИЙ_ВИД (ВНУТР)
  ЧУЖИЕ: ЧИСЛО_ЦИФР
  ЧИСЛО:= 0
  ДЛЯ Н ОТ 1 ДО ЧИСЛО_ЦИФР
  ЦИКЛ
    ЧИСЛО:= ЧИСЛО * 10 + ВНУТР [Н]
  КЦ
  ВОЗВРАТ ЧИСЛО
КОНЕЦ
```


Функция ВВОД_ПОПЫТКИ запрашивает число и преобразует его в кортеж. В функции использован новый вариант цикла — ЦИКЛ — КЦ. Его выполнение заканчивается только по предписаниям ВЫХОД или ВОЗВРАТ.

```
ФУНК ВВОД_ПОПЫТКИ ()
  ЧУЖИЕ: ЧИСЛО_ЦИФР
  ЦИКЛ
    ВВОД ПРИГЛ "ЧИСЛО?": ЧИСЛО
    ВНУТР_ВИД (ЧИСЛО, <=ОТВЕТ, <=ВЕРНО)
    ЕСЛИ ВЕРНО
     ТО ВОЗВРАТ ОТВЕТ
    ВСЕ
    ВЫВОД: "ВЫ ОШИБЛИСЬ"
  КЦ
КОНЕЦ
```

Нам осталось написать процедуру для анализа попытки. Ее параметры — кортежи из цифр.

```
ПРОЦ АНАЛИЗ_ПОПЫТКИ (ПОПЫТКА, ЗАГАДКА)
  ЦИФРЫ:= 0
  МЕСТА:= 0
  ДЛЯ Н ОТ 1 ДО #ПОПЫТКА
  ЦИКЛ
    ЕСЛИ ИНДЕКС (ПОПЫТКА[Н],ЗАГАДКА) /=0
     ТО  ЦИФРЫ:= ЦИФРЫ + 1
         ЕСЛИ ПОПЫТКА[Н] = ЗАГАДКА[Н]
          ТО МЕСТА := МЕСТА + 1
         ВСЕ
    ВСЕ
  КЦ
  ВЫВОД: "УГАДАНО ЦИФР:      ", ЦИФРЫ
  ВЫВОД: "НА ТЕХ ЖЕ МЕСТАХ: ", МЕСТА
КОНЕЦ
```

Вот и всё.

Задача 19.

Решения могут быть оформлены в виде программ или процедур.

В решениях подразумевается, что имени ЗООПАРК присвоено нужное значение.

а)

```
ПРОЦ ВИТАМИНЫ (<=ЗООПАРК)
 ДЛЯ НОМ ОТ 1 ДО #ЗООПАРК
 ЦИКЛ
  ЗООПАРК[НОМ,3]:=
            ЗООПАРК[НОМ,3] + ⟨"РЫБИЙ ЖИР"⟩
 КЦ
КОНЕЦ
```

б) *Вариант* 1. Сразу печатаем нужное расписание.

```
ПРОЦ РАСПИСАНИЕ1 (ЗООПАРК)
 ДЛЯ  ВРЕМЯ  ОТ  9  ДО  19
 ЦИКЛ
  ВЫВОД:"ВРЕМЯ -",ВРЕМЯ," ЧАСОВ:"
  ДЛЯ  НОМ  ОТ  1 ДО  #ЗООПАРК
  ЦИКЛ
   ЕСЛИ ИНДЕКС(ВРЕМЯ, ЗООПАРК[НОМ,2])/=0
    ТО ВЫВОД: ЗООПАРК[НОМ,1]
   ВСЕ
  КЦ
 КЦ
КОНЕЦ
```


После выполнения предписаний

```
ЗООПАРК:=⟨⟨"СЛОНОПОТАМ", ⟨9,12,15,18⟩,
          ⟨"ПОРОСЯТА","ВИТАМИН С"⟩⟩,
          ⟨"ВОЛК", ⟨10, 14, 18⟩,
          ⟨"ЯГНЕНОК","РЫБИЙ ЖИР"⟩⟩ ⟩
```

будет выдано:

```
РАСПИСАНИЕ1 (ЗООПАРК)

ВРЕМЯ - 9 ЧАСОВ
СЛОНОПОТАМ
ВРЕМЯ - 10 ЧАСОВ
ВОЛК
ВРЕМЯ - 11 ЧАСОВ
ВРЕМЯ - 12 ЧАСОВ
СЛОНОПОТАМ
ВРЕМЯ - 13 ЧАСОВ
ВРЕМЯ - 14 ЧАСОВ
ВОЛК
ВРЕМЯ - 15 ЧАСОВ
СЛОНОПОТАМ
ВРЕМЯ - 16 ЧАСОВ
ВРЕМЯ - 17 ЧАСОВ
ВРЕМЯ - 18 ЧАСОВ
СЛОНОПОТАМ
ВОЛК
ВРЕМЯ - 19 ЧАСОВ
```

Обратите внимание на то, что если в некоторое время не кормят ни одного животного, то заголовок все равно печатается.

Вариант 2. Расписание собираем в кортеж, каждый элемент которого имеет вид: ⟨ время, ⟨ названия животных ⟩ ⟩ , например: ⟨ 9, ⟨«ПОРОСЕНОК», «ОСЕЛ» ⟩⟩.

Способ 1.

```
ПРОЦ РАСПИСАНИЕ_21 (ЗООПАРК)
 РАСПИСАНИЕ:=⟨ ⟩
 ДЛЯ  НОМ  ОТ  9  ДО  19
 ЦИКЛ
```

```
 РАСПИСАНИЕ:=РАСПИСАНИЕ +<< НОМ, < > > >
КЦ
ДЛЯ  ВРЕМЯ  ОТ  9  ДО  19
ЦИКЛ
 ДЛЯ  НОМ  ОТ  1  ДО  #ЗООПАРК
 ЦИКЛ
  ЕСЛИ ИНДЕКС(ВРЕМЯ, ЗООПАРК[НОМ,2])/=0
   ТО РАСПИСАНИЕ [ВРЕМЯ-8,2]:=
РАСПИСАНИЕ[ВРЕМЯ-8,2] +<ЗООПАРК[НОМ,1]>
  ВСЕ
 КЦ
КЦ
\печать расписания
ДЛЯ  НОМ  ОТ  1  ДО  #РАСПИСАНИЕ
ЦИКЛ
 ЕСЛИ РАСПИСАНИЕ [НОМ,2]/=< >
   ТО ВЫВОД:РАСПИСАНИЕ[НОМ,1], " ЧАСОВ"
      ВЫВОД:РАСПИСАНИЕ[НОМ,2]
 ВСЕ
КЦ
КОНЕЦ
```

После выполнения предписаний

```
ЗООПАРК:=
  <<"СЛОНОПОТАМ", <7,12,15,18>,
                  <"ПОРОСЯТА","ВИТАМИН С">>,
   <"ВОЛК",  <10, 14, 18>,
                  <"ЯГНЕНОК","РЫБИЙ ЖИР">> >

РАСПИСАНИЕ_21 (ЗООПАРК)
```


будет выдано:

```
ВРЕМЯ - 9 ЧАСОВ
<"СЛОНОПОТАМ">
ВРЕМЯ - 10 ЧАСОВ
<"ВОЛК">
ВРЕМЯ - 12 ЧАСОВ
<"СЛОНОПОТАМ">
ВРЕМЯ - 14 ЧАСОВ
<"ВОЛК">
ВРЕМЯ - 15 ЧАСОВ
<"СЛОНОПОТАМ">
ВРЕМЯ - 18 ЧАСОВ
<"СЛОНОПОТАМ","ВОЛК">
```

Обратите внимание, что «пустые» заголовки не печатаются. Это стало возможно, так как печать выполняется в конце.

Способ 2.

```
ПРОЦ РАСПИСАНИЕ_22 (ЗООПАРК)
 РАСПИСАНИЕ:=< >
 ДЛЯ  НОМ  ОТ  9  ДО  19
 ЦИКЛ
  РАСПИСАНИЕ:=РАСПИСАНИЕ +<< НОМ, < >>>
 КЦ
 ДЛЯ  НОМ  ОТ  1  ДО  #ЗООПАРК
 ЦИКЛ
  ЧАСЫ:=ЗООПАРК[НОМ,2]
  ДЛЯ  ИНД  ОТ  1  ДО  #ЧАСЫ
  ЦИКЛ
   ВРЕМЯ:=ЧАСЫ[ИНД]
   РАСПИСАНИЕ [ВРЕМЯ-8,2]:=
  РАСПИСАНИЕ[ВРЕМЯ-8,2]+<ЗООПАРК[НОМ,1]>
  КЦ
```

```
 КЦ
\печать расписания
 ДЛЯ  НОМ  ОТ  1  ДО  #РАСПИСАНИЕ
 ЦИКЛ
  ЕСЛИ РАСПИСАНИЕ [НОМ,2]/=◄►
   ТО  ВЫВОД:РАСПИСАНИЕ[НОМ,1],  " ЧАСОВ"
       ВЫВОД:РАСПИСАНИЕ[НОМ,2]
   ВСЕ
 КЦ
КОНЕЦ
```

Сравним процедуры РАСПИСАНИЕ__21 и РАСПИСАНИЕ__22. Внешне они отличаются друг от друга лишь порядком вложенности циклов. Рассмотрим, как влияет порядок вложенности циклов на работу процедуры. В процедуре РАСПИСАНИЕ__21 для каждого часа просматриваются все элементы кортежа ЗООПАРК и выбираются названия животных, которых надо кормить в это время, а в процедуре РАСПИСАНИЕ__22 просматривается кортеж ЗООПАРК и название каждого животного сразу заносится во все соответствующие графы расписания.

В первом случае кортеж ЗООПАРК просматривается для каждого часа (т. е. 11 раз), а во втором — один раз (а кортеж ЗООПАРК может быть довольно длинным).

Вариант 3. Расписание можно представить в виде кортежа:

◄◄«СЛОНОПОТАМ», «ПОРОСЕНОК»►,
◄«ВОЛК»►,...,◄«КРОЛИК»►►.

где: ◄«СЛОНОПОТАМ», «ПОРОСЕНОК»► — кормят в 9 ч;
◄«ВОЛК»► — кормят в 10 ч;
...
◄«КРОЛИК»► — кормят в 19 ч.

```
ПРОЦ РАСПИСАНИЕ3 (ЗООПАРК)
 РАСПИСАНИЕ:= 11*◄ ◄► ►
 ДЛЯ  ВРЕМЯ  ОТ  9  ДО  19
 ЦИКЛ
  ДЛЯ  НОМ  ОТ  1  ДО  #ЗООПАРК
  ЦИКЛ
   ЕСЛИ  ИНДЕКС(ВРЕМЯ,ЗООПАРК[НОМ,2])/=0
    ТО РАСПИСАНИЕ[ВРЕМЯ-8]:=
    РАСПИСАНИЕ[ВРЕМЯ-8]+◄ЗООПАРК[НОМ,1]►
   ВСЕ
  КЦ
 КЦ
\печать расписания
ДЛЯ  ВРЕМЯ  ОТ  9  ДО  19
ЦИКЛ
 ВЫВОД:ВРЕМЯ,  " ЧАСОВ"
 ВЫВОД:РАСПИСАНИЕ[ВРЕМЯ-8]
КЦ
КОНЕЦ


ПРОЦ НОВ_РАСПИСАНИЕ (<=ЗООПАРК)
  ДЛЯ  НОМ  ОТ  1  ДО  #ЗООПАРК
  ЦИКЛ
   ЧАСЫ:=ЗООПАРК[НОМ,2]
   НОВ_ЧАСЫ:=◄►
   ДЛЯ  ИНД  ОТ  1  ДО  #ЧАСЫ
   ЦИКЛ
    ВРЕМЯ:=ЧАСЫ[ИНД] + (ИНД-1)
    ЕСЛИ  ВРЕМЯ<=19
      ТО НОВ_ЧАСЫ:=НОВ_ЧАСЫ + ◄ВРЕМЯ►
    ВСЕ
   КЦ
   ЗООПАРК[НОМ,2]:=НОВ_ЧАСЫ
  КЦ
КОНЕЦ
```

Задача 20.

Вариант 1. Используем функцию СЛОВАРЬ__1(Т). В этом случае задача сводится к поиску максимального значения среди вторых элементов кортежей вида ◄ символ, число► .

```
ФУНК МАКС_ЧАСТОТА_1(Т)
 ТАБЛ:=СЛОВАРЬ_1(Т)
\ построение таблицы частот
 СИМ:=ТАБЛ[1,1]
\ СИМ - символ с максимальной частотой
 МАКС:=ТАБЛ[1,2]
\ МАКС - максимальная частота
 ДЛЯ  ИНД  ОТ  2  ДО  #ТАБЛ
 ЦИКЛ
  ЕСЛИ  ТАБЛ[ИНД,2]>МАКС
   ТО  МАКС:=ТАБЛ[ИНД,2]
       СИМ:=ТАБЛ[ИНД,1]
   ВСЕ
 КЦ
 ВОЗВРАТ СИМ
КОНЕЦ
```

Вариант 2. Используем функцию СЛОВАРЬ_2(Т). В этом случае задача сводится к поиску максимального элемента в кортеже и соответствующего символа (сначала можно искать его индекс).

```
ФУНК МАКС_ЧАСТОТА_2(Т)
 ТАБЛ:=СЛОВАРЬ_2(Т)
\ построение таблицы частот
 ЧАСТОТЫ:=ТАБЛ[2]
 ИНД:=1
\ индекс максимального элемента
\ в кортеже ЧАСТОТЫ
 МАКС:=ЧАСТОТЫ[1]
\ максимальный элемент в кортеже ЧАСТОТЫ
ДЛЯ  НОМ  ОТ  2  ДО  #ЧАСТОТЫ
ЦИКЛ
 ЕСЛИ ЧАСТОТЫ[НОМ] > МАКС
  ТО  МАКС:=ЧАСТОТЫ[НОМ]
      ИНД:=НОМ
 ВСЕ
КЦ
    \ определение символа по его частоте
СИМ:=ТАБЛ[1, ИНД]
ВОЗВРАТ СИМ
КОНЕЦ
```


Если в тексте несколько символов с максимальной частотой, то результатом в обоих случаях будет один из этих символов.

Для выдачи всех символов, имеющих максимальную частоту, можно результат функции представить в виде текста, состоящего из этих символов.

Поправки к «ИНФО» № 2

На с. 42 неверно указан год изобретения радио (1895).
На с. 102 в операторе 260 нужно заменить 7636 на 763; оператор 400 следует выполнять в непосредственном режиме до пуска программы.
На с. 111: стоимость системы СВД-ВУЗ в комплекте на 6 терминалов ориентировочно равна 40 тыс. рублей.

А. АРХАНГЕЛЬСКИЙ

Мир ЭВМ

11. Экранный редактор текста KED

11.1. Историческая справка

Экранный редактор KED для редактирования текстовых файлов был разработан фирмой DEC (США) для использования на терминалах VT100, имеющих дополнительную клавиатуру со специальными функциональными клавишами, при поддержке операционной системой RT-11 V4.0. Версия этого редактора для терминалов VT52 и VT55 получила название K52. Кроме того, была разработана версия для работы с XM-монитором ОС RT-11, которую назвали KEX.

Редакторы с подобным набором функций и диалогом пользователя вошли в состав операционных систем RSX-11M V3.2, RSX-11M-PLUS, а также VMS-11 для ЭВМ VAX-11.

Так как терминалы VDT52130 фирмы «Видеотон» (Венгрия) и терминалы MERA-7953N (CM7209) имеют набор функций, идентичный терминалу VT52, то при работе с ними можно использовать оригинальный редактор K52. Для терминалов 15ИЭ-00-013 советского производства была разработана версия, получившая название K13 и входящая в операционную систему ФОДОС-II. Отметим, что терминал MERA-7953N более точно копирует VT52 (в том числе и символы псевдографики), чем VDT52130.

Здесь будет приведена краткая инструкция по работе с редактором K52 и указаны отличия для терминалов VDT52130 и 15ИЭ-00-013.

Продолжение. Начало см.: Информатика и образование. 1987. № 1—3; 1988. № 3—6; 1989. № 1.

11.2. Загрузка

Загрузка редактора в ОС RT-11 V4.0 или в РАФОС-II возможна из клавиатурного монитора командой EDIT с одним из ключей
.EDIT/KED спецификация файла
/K52 спецификация файла
/KEX спецификация файла
который указывает на тип используемого редактора.

Кроме того, для уточнения вида работ над файлами могут быть использованы следующие ключи:
.EDIT/CREATE — создать вновь;
/OUTPUT:SPFILE — результат вывести в файл SPFILE;
/INSPECT — только читать файл;
/ALLOCATE:NN — зарезервировать для выходного файла NN блоков.

Если не указан ни один из ключей, создается новый файл с тем же именем и типом, а старый переименовывается в файл с типом .BAK. Если указан ключ /OUTPUT со спецификацией файла, отличной от исходного, то создается новый файл и исходный остается без изменений.

Ключ /ALLOCATE используется, если известно, сколько места может занять файл после редактирования. Если он не указан, то память на носителе распределяется стандартным для системы образом.

С несистемного устройства редактор загружается как программа пользователя командой
.RUN DEV:K52
после чего он вызывает интерпретатор командной строки CSI и ждет ввода спецификации файла. После спецификации файла могут быть указаны ключи:
/C — создать файл вновь;

/I — просмотреть файл (только чтение);
/A:NN — зарезервировать для файла NN блоков.

В этом случае после закрытия редактируемого файла редактор снова «выходит на звездочку» и можно вводить следующую спецификацию файла. Все установленные режимы, а также содержимое всех буферов сохраняются.

Спецификация файла для вывода редактируемого файла на другое устройство (например, если на исходном устройстве не хватает места) задается в формате CSI обычным образом:

спецификация_выходного_файла = спецификация_входного_файла

Для RT-11 V5.0 и выше, а также для ФОДОС-II допустимы команды

.K52 DEV:INPNAM.TYP

если файл существует,

.K52 DEV:NAME.TYP/C/A:NN

если файл необходимо создать вновь,

.K52 DEV:INPNAM.TYP DEV:OUTNAM.TYP/A:NN

если выходной файл должен создаваться на другом устройстве или с другим именем.

11.3. Управляющие клавиши

KED может работать на терминалах различных конструкций, управляющие клавиши на них расположены по-разному. Ниже приведены рисунки с обозначением управляющих (функциональных) клавиш на дополнительных клавиатурах дисплеев VT52, VT100, 15ИЭ-00-013, VDT52130.

На каждой клавише сверху и/или снизу приведено ее функциональное название в соответствии с описанием редактора KED. В середине, по возможности, приведены надписи, фактически сделанные на клавишах.

PAGE 7 COMMAND	FINDNEXT 8 FIND	DELWORD 9 UNDELWORD
ADVANCE 4 BOTTOM	BACKUP 5 TOP	DELCHAR 6 UNDELCHAR
WORD 1 CHNGCASE	EOL 2 DELEOL	CUT 3 PASTE
BLINE 0 OPENLINE		SELECT ' RESET

GOLD BL	HELP RD	DELLINE GY UNDELLIN
PAGE 7 COMMAND	FINDNEXT 8 FIND	DELWORD 9 UNDELWORD
ADVANCE 4 BOTTOM	BACKUP 5 TOP	DELCHAR 6 UNDELCHAR
WORD 1 CHNGCASE	EOL 2 DELEOL	CUT 3 PASTE
BLINE 0 OPENLINE		SELECT , RESET

	↑ REPLACE	ENTER CUBS
← APPEND		→ SPECINSTR
	↓ SECTION	
GOLD ↓	HELP !	DELLINE ↑ UNDELLINE

	↑ REPLACE	ENTER CUBS
← APPEND		→ SPECINS
	↓ SECTION	
GOLD ↓	HELP !	DELLINE ↑ UNDELLINE

	↑ REPLACE	↓ SECTION
← APPEND		→ SPECINS

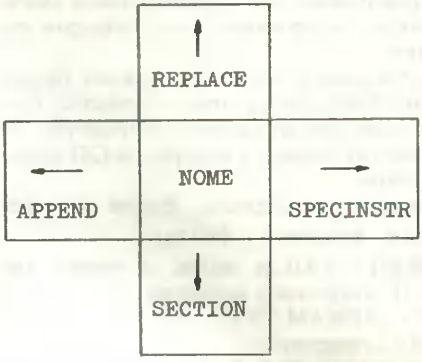
Клавиша HELP (помощь) позволяет пользователю в любое время обратиться к краткому справочнику по функциям редактора, не прерывая работы; кроме того, если редактор не смог выполнить какое-либо задание, то подается звуковой сигнал и при нажатии клавиши HELP выводится сообщение, поясняющее причину ошибки.

При первом нажатии на HELP на экран выводится картинка расположения функциональных клавиш для терминала VT52 (VT100, если используется KED, 15ИЭ-00-013, если используется K13).

При втором нажатии HELP на экран выводится полный список команд редактора,

GOLD F1	HELP F2	DELLINE F3 UNDELLINE

PAGE 7 COMMAND	FINDNEXT 8 FIND	DELWORD 9 UNDELWORD
ADVANCE 4 BOTTOM	BACKUP 5 TOP	DELCHAR 6 UNDELCHAR
WORD 1 CHNGCASE	EOL 2 DELEOL	CUT 3 PASTE
BLINE 0 OPENLINE	SELECT . RESET	ENTER SUBS



используемых в командном режиме. Если при этом перейти в командный режим, то картинка на экране сохраняется, за исключением верхних двух строк. Это позволяет не запоминать точное написание команд, а вызывать его на экран при необходимости.

При третьем нажатии HELP на экран выводится список функций, которые выполняются с помощью клавиши GOLD.

При дальнейших нажатиях процесс повторяется. Выйти из этого режима можно, нажав любую другую клавишу (лучше ENTER, так как она в данном случае не внесет никаких изменений в редактируемый текст).

Все оставшиеся клавиши имеют, как видно из рисунков, двойное назначение: верхнее название соответствует простому нажатию клавиши, нижнее — после нажатия клавиши GOLD.

В дальнейшем функции редактора, выполняемые через GOLD, будут обозначаться так:

GOLD — нижнее название клавиши

Если GOLD нажата неверно или случайно, то ее отмена производится нажатием клавиши RESET.

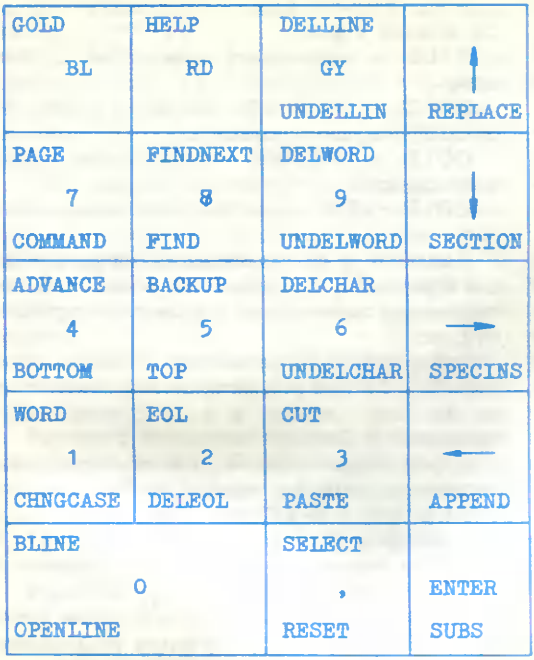
Сообщения или приглашения, выдаваемые редактором, в данном тексте набраны курсивом.

GOLD PF1	HELP PF2	FINDNEXT PF3 FIND	DELLINE PF4 UNDELLIN
PAGE 7 COMMAND	SECTION 8 FIND	APPEND 9 REPLACE	DELWORD . UNDELWORD
ADVANCE 4 BOTTOM	BACKUP 5 TOP	CUT 6 PASTE	DELCHAR — UNDELCHAR
WORD 1 CHNGCASE	EOL 2 DELEOL	CHAR 3 SPECINS	ENTER
BLINE 0 OPENLINE		SELECT 8 RESET	SUBS

11.4. Элементы редактирования

Свободное владение редактором K52 возможно только при усвоении основных понятий об элементах редактирования, приведенных ниже.

Текст — это последовательность байтов, имеющих смысл символов в коде ASCII. Так как среди них имеются символы, управляющие печатающим устройством (такие, как «перевод строки» ПС и «возврат каретки» ВК), то внешне текст имеет вид непрерывной строки. Таким образом, если нужно раздвинуть слова или буквы, то между ними следует вставить символ «пробел»; если нужно раздвинуть строки, то между ними следует вставить символ ВК. Таким же обра-



DELETE ERASE CHAR LEFT
LINEFEED ERASE WORD LEFT
CTRL/U ERASE LINE LEFT

CTRL/C CANCEL
CTRL/Z CANCEL
CTRL/W RESTORE SCREEN
AND KEYPAD
GOLD NNN REPEAT

TO RETURN TO DISPLAY,
PRESS ENTER OR USE
ANY FUNCTION.

FUNCTIONS ARE ACTIVE.

Экран после первого нажатия HELP

SUMMARY OF KED COMMANDS

EXIT
QUIT

[OPEN] INPUT FILESPEC
[OPEN] OUTPUT FILESPEC
CLOSE
PURGE
WRITE SELECT
WRITE OPTION
INCLUDE OPTION
SCIP OPTION
"OPTION" CAN BE: INTEGER PAGES
INTEGER [LINES]
REST
CLEAR PASTE

LEARN

LOCAL [STARTING_VALUE [INCREMENT]]

SET [ENTITY] PAGE "STRING"
SET [ENTITY] PAGE INTEGER [LINES]
SET [ENTITY] SECTION "STRING"
SET [ENTITY] SECTION INTEGER [LINES]

SET [SEARCH] GENERAL
SET [SEARCH] EXACT
SET [SEARCH] BEGIN
SET [SEARCH] END
SET [SEARCH] BOUNDED
SET [SEARCH] UNBOUNDED

SET WRAP [COLUMN_NUMBER]
SET NOWRAP
FILL

SET TABS [INDENT]
SET NOTABS
[TABS] ADJUST [+—] LEVEL

Экран после второго нажатия HELP

GOLD KEYBOARD FUNCTIONS

⟩GOLD⟩ ⟩S⟩ TERMINATE A MACRO WITHOUT EXECUTING IT
⟩GOLD⟩ ⟩X⟩ EXECUTE THE CURRENTLY DEFINED MACRO
⟩GOLD⟩ ⟩A⟩ ALIGN STRUCTURED TABS TO THE CURSOR
⟩GOLD⟩ ⟩E⟩ EXTEND THE LEVEL COUNTER FOR STRUCTURED TABS
⟩GOLD⟩ ⟩D⟩ DECREASE THE LEVEL COUNTER FOR STRUCTURED TABS

Экран после третьего нажатия HELP

зом правая граница текста отодвигается, если перед символами ВК ПС (CR LF) вставляются необходимые символы.

Символ — это байт перед курсором или над ним. Следует учитывать, что символы ВК ПС считаются одним символом, но занимают в буфере два байта.

Пример: 5A

Здесь 5 — символ перед курсором, A — символ над курсором.

Слово — это цепочка байтов от курсора до символов «пробел», ВК ПС, ГТ или ПФ в ту или другую сторону от курсора.

Пример: ПЕТЯ КАТЯ ВК ПС

Здесь КА — слово до курсора, ТЯ — слово после курсора.

Строка — это цепочка байтов от курсора до символов ВК ПС или ПФ в ту или другую сторону от курсора.

Страница — это цепочка байтов от курсора до символа ПФ. Возможно переопределение страницы; тогда это заданное количество строк или цепочка байтов, ограниченная заданной строкой.

Секция — это то же, что и страница, но определенная по умолчанию, как имеющая размер 16 строк. Так же может быть переопределена как имеющая другой размер или имеющая в качестве ограничителя заданную строку. Секцией удобно просматривать текст в любом направлении, так как ее размер по умолчанию равен $^2/_3$ экрана.

Сегмент редактирования — это часть файла, в пределах которой осуществляется редактирование. Предназначен для ограничения области действия таких команд, как контекстный поиск, установка уровня табуляции и т. п. Сегмент редактирования определяется следующим образом:

установить курсор в начало задаваемого сегмента редактирования;

нажать клавишу SELECT;

установить курсор в конец сегмента редактирования;

выполнить действие редактирования.

11.5. Редактирование

Редактирование файла заключается во вставке и удалении вышеописанных элементов редактирования (символов, слов, строк и т. д.). Для этого необходимо уметь передвигаться по файлу и осуществлять контекстный поиск.

Передвижение по файлу. Существует два способа передвижения курсора. Первый — использование автономно работающих клавиш управления курсором, перемещающих его на одну позицию вверх, вниз, влево или вправо. Не следует забывать, что текст — это цепочка байтов и перемещение влево или вправо за границу экранной строки переводит курсор на следующую строку.

Второй способ — использование клавиш установки направления движения курсора и указателей порций передвижения:

ADVANCE — установить направление движения в сторону конца файла;

BACKUP — установить направление движения в сторону начала файла;

BLIN, Begin of LINe — перемещает курсор по началам строк в установленном направлении;

WORD — перемещает курсор по началам слов;

EOL, End Of Line — перемещает курсор по концам строк;

PAGE — перемещает курсор по страницам;

GOLD — SCREEN SECTION — перемещает курсор по секциям;

GOLD — BOTTOM — перемещает курсор в конец файла;

GOLD — TOP — перемещает курсор в начало файла.

Удаление и восстановление элементов редактирования. При удалении элемент редактирования запоминается в соответствующем буфере:

буфер символа размером 2 байта (символы CR LF при редактировании считываются за один символ, а в действительности занимают 2 байта);

буфер слова — 80 байтов, так как длина строки, введенной с терминала без символов SPACE или CR LF, не может быть более 80 символов. При наличии символа SPACE редактор может автоматически перенести слово на следующую строку;

буфер строки — 132 байта, так как длина строки печатающего устройства редко бывает большей;

буфер сегмента редактирования — 512 байтов (один блок в формате RT-11).

Буферы автономны и не зависят друг от друга. Новое удаление заменяет содержимое буфера (удаление символа заменяет содержимое буфера символа и т. п.). Содержимое буфера можно восстановить (вставить) в любом месте файла любое число раз.

DELCHAR удаляет символ над курсором.

DEL удаляет символ перед курсором.

Восстановление происходит после последовательного нажатия клавиш GOLD — UNDELCHAR.

DELWORD удаляет слово вправо от курсора до конца слова вместе со следующими за ним пробелами.

LINEFEED удаляет слово влево от курсора до начала слова вместе с предшествующими ему пробелами.

Восстановление происходит после последовательного нажатия клавиш GOLD — UNDELWORD.

CTRL+U удаляет строку влево от курсора до символов CR LF, не удаляя их.

GOLD — DELEOL удаляет строку вправо от курсора до символов CR LF, не удаляя их.

DELLINE удаляет строку вправо от курсора до символов CR LF, включая и их.

Восстановление происходит после последовательного нажатия клавиш GOLD — UNDELLINE.

После установки границ сегмента редактирования для удаления следует нажать кла-

вишу CUT, а для восстановления — клавиши GOLD — PASTE.

Для добавления выбранного сегмента редактирования к уже заполненному буферу сохранения сегмента необходимо после определения сегмента нажать клавиши GOLD — APPEND.

Для замены выбранного сегмента редактирования на содержимое буфера сохранения сегмента используются клавиши GOLD — REPLACE.

Если курсор установлен в результате контекстного поиска, то заменяется модель поиска.

Освободить заполненный буфер можно в командном режиме командой CLEAR PASTE.

Редактор позволяет производить последовательную замену заданной модели поиска на содержимое буфера сохранения сегмента. При нажатии на клавиши GOLD — SUBS сначала выполнится команда PEPLACE, а затем FINDNEXT.

Редактор представляет возможность задавать число повторений нажатия любой клавиши. Для этого необходимо:

нажать клавишу GOLD;

на основной клавиатуре набрать число повторений от 1 до 65 535 (в десятичной системе счисления);

нажать необходимую клавишу.

Для осуществления контекстного поиска редактору необходимо сообщить текст (модель поиска), который нужно найти. Задается модель поиска (длиной до 72 символов, включая пробелы и управляющие символы) так:

последовательно нажимаются клавиши GOLD, FIND;

в ответ на приглашение «MODEL:» набрать строку символов, которую необходимо искать в тексте;

нажать либо клавишу ENTER для запоминания модели, либо клавишу, указывающую направление поиска (модель при этом будет запомнена автоматически).

В модели учитываются все введенные символы.

По умолчанию и после загрузки редактора поиск ведется в следующем режиме:

GENERAL — общий поиск, без различения символов верхнего и нижнего регистров;

BEGIN — при обнаружении модели курсор устанавливается в начало модели поиска;

UNBOUNDED — поиск ведется по всему файлу, игнорируя границы сегмента редактирования.

В командном режиме можно изменить режимы поиска на противоположные: EXACT — «точный», END — «в конец модели», BOUNDED — «в пределах сегмента редактирования».

Для поиска следующего появления этой же модели достаточно нажать клавишу FINDNEXT. Модель поиска запоминается на все время сеанса редактирования или до следующего определения модели.

11.6. Основной набор команд

Редактор «подчиняется» не только управляющим клавишам; существует еще особый командный режим, позволяющий расширить его функциональные возможности. Для его использования необходимо выполнить следующие действия:

нажать клавиши GOLD — COMMAND для входа в командный режим;

при появлении в верхней строке слова «COMMAND:» набрать на основной клавиатуре текст команды (символьную строку);

нажать клавишу ENTER для ввода набранной команды (до нажатия этой клавиши набранную строку можно редактировать, используя клавишу DEL).

Список возможных команд приведен во второй таблице HELP редактора. Напомним, что таблица не исчезает, если прямо из режима HELP перейти в командный режим, что позволяет не запоминать весь перечень команд и их точное написание.

Завершается редактирование командой EXIT или QUIT.

По команде EXIT результаты сеанса редактирования будут сохранены в новом файле, исходному файлу будет присвоен тип .BAK (запасной).

По команде QUIT произойдет выход из редактора без сохранения результатов сеанса редактирования, исходный файл останется без изменения. Команда используется в основном, если файл был открыт для просмотра (ключ /INSPECT). Если файл открывался для редактирования, то свободная зона может оказаться разделенной на две в соответствии с правилами предоставления дискового пространства в RT-11.

Редактор позволяет задать границу строки, при переходе через которую (при вводе текста) происходит автоматический перенос не уместившегося слова на новую строку. Для этого необходимо ввести команду

COMMAND: SET WRAP NN

где NN — номер колонки, являющейся границей строки, иначе говоря, количество символов в строке.

Отменяется данный режим так:

COMMAND: SET NOWRAP

Если установлена граница строки, то можно отформатировать текст в пределах сегмента редактирования командой FILL:

COMMAND: FILL

Она воспринимает знаки горизонтальной табуляции как пробелы и игнорирует знаки CR LF, так что текст можно форматировать только в пределах абзаца.

Граница для секции может быть установлена в виде некоторой символьной строки или непосредственно числом строк.

Граница в виде символьной строки задается так:

COMMAND: SET SECTION символьная строка

Для задания размера секции целым числом строк необходимо дать команду

COMMAND: SET SECTION NN

где NN — число строк в секции.

Таким же образом можно задать границы страницы, дав команды

COMMAND: SET PAGE символьная строка

и

COMMAND: SET PAGE NN

Режимы поиска устанавливаются в командном режиме и сохраняются до конца сеанса редактирования или до следующей установки.

Общий поиск без различия символов верхнего и нижнего регистров устанавливается командой

COMMAND: SET GENERAL

Точный поиск с различием символов верхнего и нижнего регистров устанавливается командой

COMMAND: SET EXACT

Чтобы при обнаружении модели курсор устанавливался в ее начало, нужно дать команду

COMMAND: SET BEGIN

Чтобы при обнаружении модели курсор устанавливался в ее конец, нужно дать команду

COMMAND: SET END

Поиск может осуществляться в пределах сегмента редактирования, для чего необходимо дать команду

COMMAND: SET BOUNDED

Если поиск должен осуществляться по всему файлу, нужно дать команду

COMMAND: SET UNBOUNDED

11.7. Специальные команды

Команды, перечисленные выше, являются основными. Их достаточно для редактирования. Но существуют и другие команды — они расширяют возможности пользователя.

Редактор можно «обучить» выполнению некоторого набора команд по нажатию одной клавиши. Для этого необходимо выполнить следующие действия:

вести в командном режиме команду LEARN — учись;

вручную набрать команды или действия, которые необходимо запомнить;

последовательно нажать клавиши GOLD, S, если команды нужно только запомнить, и GOLD, X, если команды нужно запомнить и выполнить.

В дальнейшем для выполнения запомненной последовательности необходимо нажимать клавиши GOLD, X. При этом можно использовать функцию повторения нажатия клавиши обычным образом:

GOLD *REPEAT:* **NNN, GOLD, X**

При редактировании текстов программ на макроассемблере MACRO-11 редактор позволяет в пределах заданного сегмента редактирования расставить локальные метки начиная с заданного номера и с заданным шагом, а также ссылки на них в тексте программы. Для этого необходимо определить сегмент редактирования и дать команду

COMMAND: LOCAL NN MM

где NN — начальный номер метки, MM — шаг приращения номера метки. По умолчанию начальный номер метки и шаг приращения равны 10.

Редактор позволяет в пределах сегмента редактирования сдвигать весь фрагмент текста влево или вправо на нужный уровень табуляции. Сдвиг на один уровень табуляции соответствует сдвигу текста на 8 символьных позиций. Для сдвига необходимо:

идентифицировать текущий уровень табуляции командой SET TABS;

определить сегмент редактирования;

изменить уровень табуляции командой

ADJUST+/—NN,

где NN — число уровней табуляции.

Отмена установленного уровня табуляции производится либо установкой нового уровня, либо командой

COMMAND: SET NOTABS

Символы верхнего и нижнего регистров (в ISO это латинские прописные и строчные, в КОИ-7 — латинские и русские прописные буквы) отличаются только значением 5-го бита в коде символа. Редактор предоставляет возможность групповой смены (инвертирование 5-го бита) регистра. Для этого нужно ввести команду

GOLD — CHNGCASE

При этом:

если был определен сегмент редактирования, то смена регистра произойдет по всему тексту в пределах сегмента редактирования;

если курсор был установлен в результате контекстного поиска заданной модели, то замена регистра произойдет в пределах модели поиска;

если не было никаких определений, то смена регистра произойдет у одного сим-

вола над курсором, а курсор переместится на одну позицию в соответствии с установленным направлением движения.

Особым образом производится вставка специального символа, под которым мы будем понимать байт с кодом от 1 до 255. Байт с кодом 000 нельзя вставлять, так как по этому коду редактор определяет конец текста.

Для вставки, например, символа с кодом 007 («звонок») необходимо ввести следующую последовательность команд:

GOLD *REPEAT:*, 7, **GOLD SPECIONS**

Если затем вывести текстовый файл на терминал командой **TYPE**, то в этом месте незримый символ 007 подаст звуковой сигнал.

Следует отметить, что код символа (как и все числа, вводимые через клавишу GOLD) должен быть в десятичной системе счисления.

Часто бывает необходимо вставить пустую строку в место, указанное курсором. Простой способ заключается в том, чтобы ввести символы **ВК ПС**, а затем подняться на одну строку вверх с помощью стрелки. Однако для многократной вставки пустой строки с помощью функции повторения можно использовать клавишу **OPENLINE** (открыть строку), например, так:

GOLD *REPEAT:*, XXX, **GOLD OPENLINE**

Положение курсора при этом не изменится.

Символы **CTRL/C** и **CTRL/Z** при работе редактора не выполняют своих обычных функций, а служат для остановки выполнения продолжительных по времени операций. Курсор при этом будет установлен в той части текста, которая обрабатывалась в это время редактором.

При сбоях аппаратуры или программного обеспечения возможно нарушение соответствия изображения на экране дисплея действительному содержанию текста в памяти. В этом случае следует «перерисовать» экран, используя служебный символ **CTRL W**.

11.8. Работа с файлами

Для понимания принципов работы редактора К52 с файлами необходимо определить названия самих файлов.

Рабочий (входной) файл — тот, который редактируется, физически всегда остается без изменений. Если в командной строке не была указана спецификация выходного файла, по завершении сеанса редактирования файл переименовывается в файл с типом .BAK (запасной), а его имя и тип получает временный файл.

Временный (выходной) файл — тот, в котором происходит редактирование. В этот файл вносятся все изменения. По завершении сеанса редактирования, если результаты должны быть сохранены, получает имя и тип рабочего файла и в следующем сеансе становится рабочим.

Входной вспомогательный файл находится на устройстве с произвольным доступом и может быть открыт для чтения. Информация из него переписывается во временный файл в место, указанное курсором. Курсор устанавливается в конце прочитанного сегмента.

Выходной вспомогательный файл — свободная зона на устройстве с произвольным доступом, которая может быть объявлена пользователем как файл для записи, в который записываются выбранные элементы из временного или рабочего файлов. Информация в выходной вспомогательный файл записывается всегда в конец, после существующей. По окончании работы с этим файлом его можно закрыть, сохранив при этом информацию, или освободить, оставив при этом свободную зону. Чтобы открыть этот файл для чтения, его необходимо сначала закрыть, сделав постоянным.

Для открытия входного и выходного вспомогательных файлов необходимо дать соответственно команды:

COMMAND: **INPUT** спецификация выходного файла

COMMAND: **OUTPUT** спецификация выходного файла

Входной вспомогательный файл закрывается автоматически, если при чтении будет обнаружен конец файла или при выходе из редактора. Вернуться в начало файла нельзя.

Выходной вспомогательный файл может быть закрыт командой

COMMAND: **CLOSE**

если информация в нем должна быть сохранена, или освобожден командой

COMMAND: **PURGE**

если информация в нем не нужна.

Редактор предоставляет возможность записать в выходной вспомогательный файл выбранный сегмент редактирования или отдельный элемент редактирования. Из входного вспомогательного файла можно вставить во временный файл элемент редактирования.

Для записи выбранного сегмента редактирования необходимо определить сегмент редактирования, как это было указано выше, и дать команду

COMMAND: **WRITE SELECT**

Выбранный сегмент редактирования будет записан в конец вспомогательного выходного файла.

Можно записывать и некоторые элементы редактирования, например:

COMMAND: WRITE N PAGES (запись N страниц);
COMMAND: WRITE NN [LINES] (запись NN строк; слово LINES можно не вводить);
COMMAND: WRITE REST (запись оставшейся части файла).

Аналогично можно включить информацию из входного вспомогательного файла, введя команды:
COMMAND: INCLUDE N PAGES (вставка из входного файла N страниц);
COMMAND: INCLUDE NN [LINES] (вставка из входного файла NN строк);
COMMAND: INCLUDE REST (вставка оставшейся части вспомогательного входного файла).

Если какую-либо часть вспомогательного входного файла не нужно включать в редактируемый файл, то ее можно пропустить командой SKIP с такими же элементами редактирования в качестве аргументов.

Следует отметить, что один и тот же файл не может быть одновременно открыт как входной и выходной вспомогательный файлы. Если вспомогательный файл используется как большой буфер, то после записи он должен быть закрыт командой CLOSE, а затем открыт как вспомогательный входной файл.

Если редактор работает в режиме INSPECT, то вспомогательный входной файл открыть нельзя. Однако вспомогательный выходной файл может быть открыт и использован в качестве «записной книжки».

11.9. Некоторые сообщения об ошибках

W-OUTPUT FILES PURGED — выходной файл освобожден. Сообщение появляется после команды QUIT в режиме редактирования файла и говорит о том, что выходной файл с результатами редактирования получил признак состояния «ПУСТОЙ». Запись о входном файле осталась без изменения.

F — RDFWD — NO — SPACE TO READ — нет места для чтения. Мал буфер в памяти.

F — I/O OR DEVICE ERROR — ошибка ввода/вывода или устройства. Сообщение возникает при невозможности прочитать или записать блок на устройстве.

F — WRONG VERSION OF RT-11 — неверная версия RT-11. Попытка работать с редактором в младших версиях RT-11.

F — UNABLE TO ACCESS INPUT DEVICE — невозможный доступ к входному устройству. Не загружен драйвер входного устройства или оно отсутствует.

F — TOO BIG TO EDIT — слишком большой кусок для редактирования. Попытка редактировать файл объемом более 16 384 блоков.

W — FILE NOT FOUND — CREATE IT (Y, N)? — файл не найден — создать? При отсутствии файла с заданной спецификацией редактор предлагает создать его.

F — INVALID DEVICE — неверное устройство. Файл нельзя редактировать на устройстве нефайловой структуры или последовательного доступа.

F — UNABLE TO ACCESS OUTPUT DEVICE — невозможен доступ к выходному устройству. Не загружен драйвер выходного устройства.

F — UNABLE TO OPEN OUTPUT FILE — невозможно открыть выходной файл. Устройство или файл защищены по записи. Нельзя также редактировать, если входной файл защищен по записи.

F — OUTPUT FILE SHORTER THEN INPUT FILE — выходной файл короче входного файла. Нельзя редактировать файл, если свободное место на устройстве меньше входного файла.

W — OUTPUT FILE OR DEVICE PROTECTED — INSPECT (Y, N)? — выходной файл или устройство защищено по записи — смотреть? Защищенные файлы можно смотреть только в режиме INSPECT.

ILLEGAL FUNCTION — неверная функция. Данную функцию нельзя применять в установленном режиме. Например, нельзя вставлять символ в режиме INSPECT.

CTRL/C OR CTRL/Z IGNORED — USE QUIT — CTRL/C или CTRL/Z игнорируются — используйте команду QUIT. Служебные символы CTRL/C и CTRL/Z используются в редакторе для прекращения выполнения продолжительной по времени операции. Соглашения об использовании этих символов в RT-11 в данном случае недействительны. Для выхода в монитор (прекращения работы редактора) используются команды QUIT или EXIT.

INSERT FINDS FILE FULL — при вставке найден полный файл. Зарезервированное место для файла на диске заполнено.

ILLEGAL TO INSERT NULL — неверная вставка нуля. Код 000 считается признаком конца текста и не может быть вставлен в него.

NOT ENOUGH SPACE TO UNDELETE — недостаточно места для восстановления. Попытка восстановить текст из буфера сохранения при заполненном файле.

NO MODEL DEFINED TARGET NOT FOUND — модель, определенная как цель, не найдена. Модель не найдена в пределах файла.

BOUNDED SEARCH REACHED BO-

UND — ограниченный поиск достиг границы. Заданный образ не найден в пределах сегмента редактирования.

NO MODEL DEFINED — не определена модель поиска. Попытка поиска неопределенной модели.

ARROW COMMAND FINDS EXTREMITY OF FILE — стрелочными командами найден край файла. При перемещении курсора клавишами управления курсора (со стрелками) достигнута граница файла.

ILLEGAL TERMINATING KEY TO COMMAND PROMPT — неверный последний ключ после приглашения COMMAND. Редактор не смог выполнить команду из-за недопустимого символа в команде (или после команды не нажата клавиша ENTER).

NO MACRO TO EXECUTE — нет макро для выполнения. Не задана макрокоманда для выполнения.

FILE FULL DURING FILL — файл переполнен в режиме заполнения. При заполнении редактор может добавлять символы CR LF, в результате чего происходит переполнение.

SELECT RANGE TOO LARGE FOR CUT — сегмент редактирования слишком длинный для CUT. Буфер сохранения при удалении сегмента редактирования (операция CUT) равен 512 байтам.

MOVE TO TOP WHEN AT TOP — пересылка вверх, когда вверху. Попытка переслать курсор командой TOP, когда курсор находится в начале файла.

MOVE TO BOTTOM WHEN AT BOTTOM — пересылка вниз, когда внизу. Попытка переслать курсор командой BOTTOM, когда курсор находится в конце текста.

Окончание следует

В. ОЧКОВ, Ю. ПУХНАЧЕВ

Оформление текста программы

Сейчас сложилась ситуация, когда первым языком программирования, с которым сталкивается начинающий, является либо Бейсик, либо язык команд микрокалькулятора. На обоих языках тексты программ обычно записываются в виде колонок из нумерованных строк. Такая запись не способствует пониманию алгоритма, на основе которого составлена программа. По этой и другим причинам оба языка в программистской литературе характеризуются как *неструктурные* — в отличие от *структурных* (Паскаль, Рапира, Си, Ада и др.), чья, так сказать, «орфография» содержит средства, позволяющие видеть в тексте программы ее структуру (выделение отступами вложенных блоков и т. п.).

Первые попытки выделить графическими средствами алгоритм решения задачи в тексте составленной для этого программы, если даже она написана на неструктурном языке, можно найти в работах Л. Брудно [1]. В данной статье демонстрируется использование для этой цели так называемых *структурных диаграмм*. С их помощью можно не только выделять лежащий в основе программы алгоритм, но и делать похожими программы, написанные на непохожих языках, обучать структурному программированию на неструктурных языках и т. д.

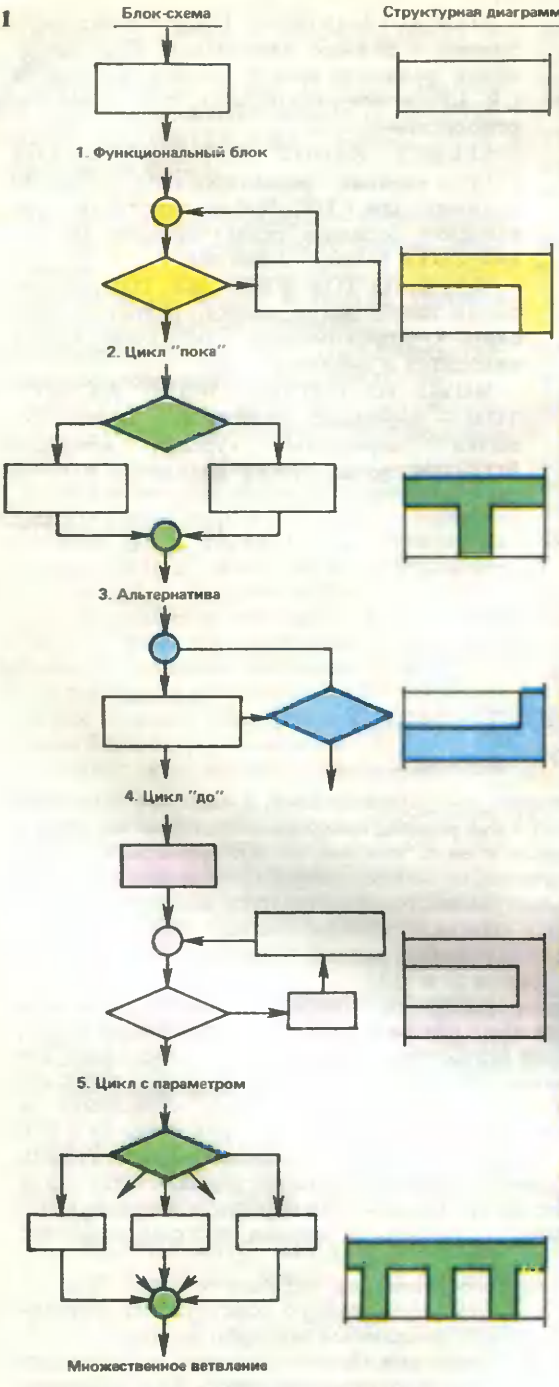
На рис. 1 элементы структурных диаграмм представлены параллельно с аналогичными элементами блок-схем. При разработке структурных диаграмм авторы настоящей статьи взяли за основу диаграммы Несси-Шнейдермана (рис. 2) [2, 3], дополнив их цветом и изменив так, чтобы в них можно было вписывать программы на различных языках программирования.

Обратимся к примерам.

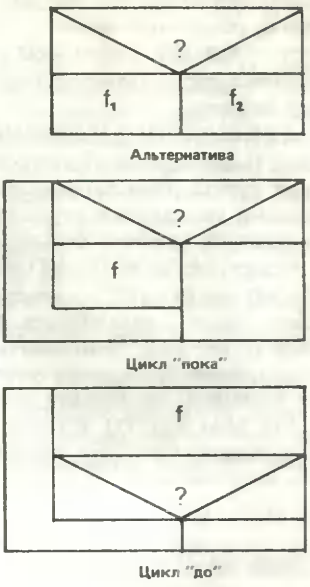
Бейсик-программа поиска корня алгебраического уравнения методом секущих (рис. 3) содержит в себе цикл «пока» — основную структурную управляющую конструкцию программирования. На структурных диаграммах цикл «пока» выделяется желтым цветом.

Считается (хотя это не совсем верно, но об этом ниже), что алгоритм любой сложности можно реализовать, комбинируя и вкладывая друг в друга циклы «пока» и альтернативы (иными словами, любой алгоритм представим при помощи лишь трех конструкций — следование, повторение и выбор). Пример альтернативы, вложенной в цикл «пока», можно увидеть на рис. 4 в программе поиска корня алгебраического уравнения методом половинного деления. Рамки структурной диаграммы сделали похожими программы, написанные на непохожих языках программирования (команды микрокалькулятора, Бейсик и Паскаль). Следует отметить, что число 34 в левом плече альтернативы в программах *а* и *б* для ускорения счета можно заменить на 8 (вероятно, так бы и поступили многие начинающие програм-

1

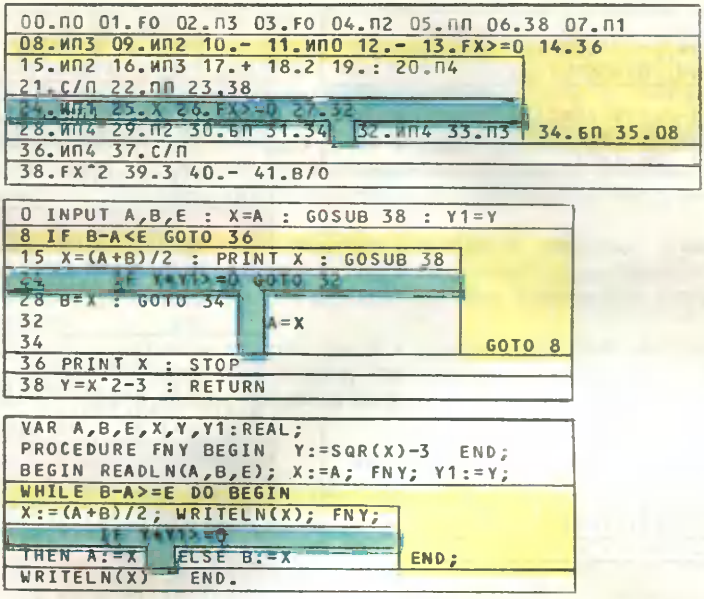


2



3

```
10 DEF FNY(X)=...' ВИД ФУНКЦИИ
20 INPUT "X1,X2,ТОЧНОСТЬ ";X1,X2,E
30 WHILE ABS(X1-X2)>E' НАЧАЛО ЦИКЛА "ПОКА"
40 X3=X2-FNY(X2)*(X2-X1)/(FNY(X2)-FNY(X1))
50 X1=X2:    X2=X3                          :WEND
60 PRINT "Y=0 ПРИ  X=";X2
```



мисты), но в этом случае структура алгоритма «развалится», и его уже нельзя будет реализовать без опоры на метку на таком языке, как Паскаль.

В структурных диаграммах кроме желтого и зеленого можно увидеть еще два цвета — синий и красный.

Бейсик-программа поиска корня алгебраического уравнения методом Ньютона (рис. 5) содержит цикл «до» (синий цвет). Здесь он более удобен, чем цикл «пока», так как булевское выражение, управляющее циклом, приобретает смысл только после выполнения тела цикла. Правда, цикл «до» несложно заменить на цикл «пока», искусственно присвоив булевскому выражению нужное значение при первом входе в цикл. Такой прием инициализации цикла использован в Бейсик-программе, помещенной на рис. 6 и предназначенной для той же цели, что и предыдущая. Вообще-то для поиска корня алгебраического уравнения и методом секущих (см. рис. 3), и методом Ньютона (см. рис. 5 и 6) более подходит цикл с выходом из середины (рис. 7), но эта управляющая конструкция редка в структурном программировании.

Когда заранее известно, сколько раз нужно выполнить тело цикла, то используют управляющую конструкцию, называемую циклом с параметром (красный цвет). На рис. 8 в рамки одной и той же структурной диаграммы удалось вписать составленную на трех разных языках (Фортран, Паскаль и Бейсик) программу сортировки элементов одномерного массива в порядке возрастания методом «пузырька». Рамки структурной диаграммы, как и в предыдущем случае (см. рис. 4), позволяют абстрагироваться от синтаксических особенностей того или иного языка программирования и ясно увидеть алгоритм решения поставленной задачи.

Функциональные блоки на структурных диаграммах (бесцветные прямоугольники) скреплены, как кирпичи раствором, окрашенными структурными конструкциями. В программах на Бейсике (да и на других алгоритмических языках) можно выделить семь управляющих конструкций. Все они присутствуют в Бейсик-программе «Частотный словарь» (рис. 9).

1. Цикл с параметром: ввод исходного анализируемого текста (строка 11); перебор строк при анализе (строки 12—90); вывод частотного словаря текста на дисплей (строка 140).

2. Цикл «до»: поиск в строках анализируемого текста знаков препинания и пробелов, разделяющих слова (строка 20); перебор слов в частотном словаре для сравнения с только что выделенным из текста (строка 50); перебор слов очередной строки текста (строки 20—80).

3. Цикл «пока»: сортировка формируемого словаря в порядке убывания частоты слов (строка 70).

4. Альтернатива: игнорирование промежутков между словами текста с двумя и более знаками препинания (строки 30—70); уточнение, встречалось ли ранее в частотном словаре только что выделенное слово (строки 60—70).

5. Вызов подпрограммы (процедуры) перестановки соседних слов частотного словаря (строка 150) при его сортировке (см. строки 70 и 120).

5

```
10 DEF FNY(X)=... : REM ВИД ФУНКЦИИ
20 DEF FND(X)=... : REM ВИД ПРОИЗВОДНОЙ
30 INPUT"X НАЧ,ТОЧНОСТЬ?";X,E
40 :                          REPEAT
50 X1=X : X=X-FNY(X)/FND(X)
60 UNTIL ABS(X1-X)<E
70 PRINT "Y=0 ПРИ X="X
```

6

```
10 DEF FNY(X)=... : REM ВИД ФУНКЦИИ
20 DEF FND(X)=... : REM ВИД ПРОИЗВОДНОЙ
30 INPUT"X НАЧ,ТОЧНОСТЬ?";X,E : X1=X+10*E
40 WHILE ABS(X1-X)>=E
50 X1=X : X=X-FNY(X)/FND(X)   :WEND
60 PRINT "Y=0 ПРИ X="X
```

62

7

```
10 DEF FNY(X)=...:REM ВИД ФУНКЦИИ
20 DEF FND(X)=...:REM ВИД ПРОИЗВОДНОЙ
30 INPUT'X НАЧ,ТОЧНОСТЬ?';X,E
40 X1=X-FNY(X)/FND(X)
50 IF ABS(X1-X)<E GOTO 70
60 X=X1                    :GOTO 40
70 PRINT 'Y=0 ПРИ X='X1
```

8

```
      DIMENSION A(1000)
      READ (1,1) N,(A(I),I=1,N)
1     FORMAT(I4/10 F4.2)
      DO 3 I=2,N
      DO 2 J=N,I,-1
      IF (A(J-1).LE.A(J)) GO TO 2
      X=A(J-1)
      A(J-1)=A(J)
      A(J)=X
2     CONTINUE
3     CONTINUE
      WRITE(1,4) (A(I),I=1,N)
4     FORMAT(10X,F4.2)
      STOP
      END
```

```
VAR I,J,N:INTEGER;
    X:REAL;A:ARRAY[1000] OF REAL;
BEGIN READLN(N);
FOR I:=1 TO N DO READLN(A[I]);
FOR I:=2 TO N DO
FOR J:=N DOWNTO I DO
IF A[J-1]>A[J]
THEN BEGIN X:=A[J-1];
A[J-1]:=A[J];A[J]:=X END;
(*FOR END*)
(*FOR END*)
FOR I:=1 TO N DO WRITELN(A[I])
END.
```

```
10 INPUT N:DIM A(N)
20 FOR I=1 TO N:INPUT A(I):NEXT
30 FOR I=2 TO N
40 FOR J=N TO I STEP -1
50 IF A(J-1)>A(J)
   THEN SWAP A(J-1),A(J)
70 NEXT
80 NEXT
90 FOR I=1 TO N:PRINT A(I):NEXT
```

9

```
10 INPUT "ЧИСЛО СТРОК ";N:DIM S¤(N),W¤(5*N),W%(5*N)
11 FOR I=1 TO N:LINE INPUT S¤(I):NEXT:L=0
12 FOR I=1 TO N:REM ПЕРЕБОР СТРОК ДЛЯ АНАЛИЗА:
   S¤=S¤(I):S=LEN(S¤):H=0:K=0:REM НАЧАЛО СТРОКИ
20 K=K+1:B¤=MID¤(S¤,K,1):REM ПЕРЕБОР СИМВОЛОВ:
        IF B¤>="A" OR B¤="-" OR B¤="'" GOTO 20
30 IF K=H+1 THEN 80
40 W¤=MID¤(S¤,H+1,K-H-1):REM СЛОВО ВЫДЕЛЕНО:J=0
50 J=J+1:REM ПЕРЕБОР СЛОВ В СЛОВАРЕ:
        IF W¤<>W¤(J) AND J<=L GOTO 50
60 IF J>L THEN
   REM СТАРОЕ СЛОВО:                  L=J:W¤(L)=W¤:
                                      W%(L)=1:GOTO 80
                                      REM НОВОЕ СЛОВО:
70 W%(J)=W%(J)+1:
   WHILE J>1 AND W%(J-1)<W%(J):
   GOSUB 150:J=J-1                    :WEND
80 H=K:REM ПЕРЕХОД К НОВОМУ СЛОВУ НА СТРОКЕ:
      IF K<S GOTO 20:REM СТРОКА НЕ КОНЧИЛАСЬ
90 NEXT:REM КОНЕЦ ЦИКЛА ПЕРЕБОРА СТРОК:
   PRINT "1-ВЫВОД ПО АЛФАВИТУ,2-ПО ЧАСТОТЕ?";
100 B¤=INKEY¤:REM ПРИЕМ БАЙТА БЕЗ ПРЕРЫВАНИЯ:
                  IF B¤<"1" OR B¤>"2" GOTO 100
110 ON VAL(B¤) GOTO 120,140
120 FOR J=2 TO L:REM ЦИКЛ СОРТИРОВКИ ПО АЛФАВИТУ:
        IF W¤(J-1)>W¤(J) THEN GOSUB 150:GOTO 120
130 NEXT
140 FOR J=1 TO L:PRINT J". "W¤(J),W%(J):NEXT:END
150 SWAP W%(J),W%(J-1):SWAP W¤(J),W¤(J-1):RETURN
```

6. Множественное ветвление после ответа на запрос по меню (строки 110—130). Здесь, правда, можно было обойтись простой альтернативой, но режимов вывода частотного словаря

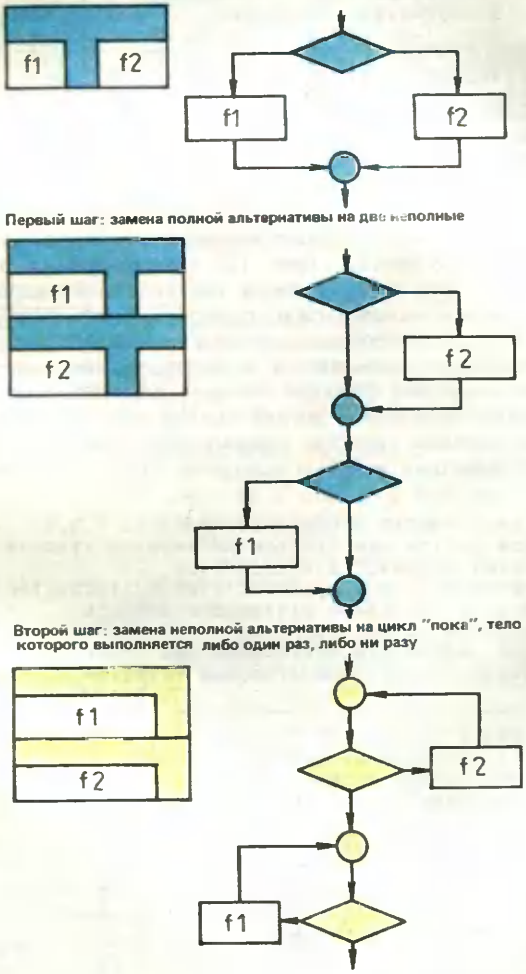
может быть больше чем два: вывод на принтер, «сброс» на диск и т. д.

7. Переход к метке. В чистом виде это делается только при досрочном прерывании цикла с параметром и возвращении к его началу при сортировке по алфавиту (см. строку 120). В остальных случаях переходы к меткам реализуют какую-либо одну из стандартных управляющих конструкций (см. пункты 2, 4, 5 и 6).

Упоминание о метке не случайно оказалось в последнем пункте нашего списка. Ценность этого приема (а он широко используется при программировании на Бейсике) очень низка. Считается, что квалификация программиста обратно пропорциональна числу меток, встречающихся в его программах.

Структурные диаграммы допускают переход к метке в пределах функционального блока — см. строки 120 и 130 программы на рис. 9. Короткие управляющие конструкции можно не выделять структурно и оформлять функциональным блоком (см. строки 11 и 140 с циклами с параметром).

Напомним, что любой алгоритм можно свести к комбинации трех конструкций: следование, цикл «пока» и альтернатива. Любопытно, что можно обойтись и без альтернативы. Схема ее замены на два цикла «пока» показана на рис. 10: вначале полная альтернатива за-



меняется на две неполные (кстати, такая замена рекомендуется в литературе для исключения меток в Фортран-программах), а затем каждая из неполных альтернатив при помощи вводимой для этой цели булевской переменной заменяется на цикл «пока»; при этом тело каждого из циклов должно выполняться либо только раз, либо вообще ни разу. Пример — Паскаль-программа поиска действительных корней алгебраического уравнения (рис. 11, *а*). Описанный прием превращает ее в программу с двумя циклами «пока» (рис. 11, *б*).

```
PROGRAM КОРЕНЬ(INPUT,OUTPUT);
VAR A,B,C,D:REAL;
BEGIN
WRITE('A,B,C'); READLN(A,B,C);
D:=SQR(B)-4*A*C;
                          IF D>=0
THEN WRITELN
('X1=',(-B+SQRT(D))/(2*A),
 'X2=',(-B-SQRT(D))/(2*A))   ELSE WRITELN
                             ('КОРНИ МНИМЫЕ')
END.
```

```
PROGRAM КОРЕНЬ(INPUT,OUTPUT);
VAR A,B,C,D:REAL;  P:BOOLEAN;
BEGIN
WRITE('A,B,C'); READLN(A,B,C);
D:=SQR(B)-4*A*C; P:=TRUE;
WHILE (D>=0) AND (P) DO BEGIN
P:=FALSE; WRITELN
('X1=',(-B+SQRT(D))/(2*A),
 'X2=',(-B-SQRT(D))/(2*A))          END;
WHILE P DO BEGIN
P:=FALSE; WRITELN('КОРНИ МНИМЫЕ')  END
END.
```

64 К структурным управляющим конструкциям следует отнести и вызов процедуры в рекурсивных алгоритмах. Но при этом следует помнить о том, что рекурсия — это способ сделать программу понятной для человека за счет снижения эффективности выполнения ее машиной.

Помимо структурных диаграмм можно предложить еще ряд средств, позволяющих выявить логику работы программы. В Бейсик-программе поиска минимума многомерной функции методом покоординатного спуска (рис. 12) анализируемая функция записана в виде подпрограммы в строке 21. Эта подпрограмма, составляющая ядро расчета, является подпрограммой третьего уровня и вызывается из подпрограмм второго уровня (строки 19 и 20), выполняющих деление интервала неопределенности в золотом соотношении. Подпрограммы в строках 19 и 20 в свою очередь вызываются из подпрограммы первого уровня (строки 13—18) поиска минимума одномерной функции методом золотого сечения. Стрелки, поставленные на рис. 12, позволяют проследить логику вызова подпрограмм различного уровня.

Аппаратные и программные средства современных компьютеров позволяют «озвучить» программу так, чтобы при ее прогонке был «слышен» алгоритм решения задачи. Бейсик-про-

12

```
 1 INPUT "ЧИСЛО АРГУМЕНТОВ,ТОЧНОСТЬ ",N,E
 2 FOR J=1TON:REM ПЕРЕБОР АРГУМЕНТОВ ФУНКЦИИ
 3 PRINT "X НАЧ.";J;:INPUT H(J)
 4 PRINT "X КОН.";J;:INPUT K(J):X(J)=(H(J)+K(J))/2
 5 NEXT J:REM КОНЕЦ НАЧАЛЬНОГО ДИАЛОГА
 6 S=0:REM НАЧАЛО ИТЕРАЦИИ
 7 FOR J=1TON:REM ПОКООРДИНАТНЫЙ СПУСК
 8 X=X(J):A=H(J):B=K(J):GOSUB 13:X(J)=X0:S=S+(X-X0)^2
 9 NEXT J
10 IF SQR(S)>ETHEN6:REM КОНЕЦ ЦИКЛА "ДО"
11 FOR J=1TON:PRINT "X ОПТ.";J;"=";X(J):NEXT J
12 END :REM КОНЕЦ ПРОГРАММЫ, НАЧАЛО ПОДПРОГРАММ
13 GOSUB 19:GOSUB 20:REM МЕТОД ЗОЛОТОГО СЕЧЕНИЯ
14 IF Y1>Y2THEN16:REM ЯДРО РАСЧЕТА
15 B=X2:X2=X1:Y2=Y1:      GOSUB 19:GOTO 17
16 A=X1:X1=X2:Y1=Y2:GOSUB 20
17 IF ABS(A-B)>E/NTHEN14
18 X0=(A+B)/2:                          RETURN
19 X1=.618*A+.382*B:X(J)=X1:GOSUB 21:Y1=Y:RETURN
20 X2=.618*B+.382*A:X(J)=X2:GOSUB 21:Y2=Y:RETURN
21 Y=FNY(X(1),X(2),...,X(N)):RETURN
```

грамма, помещенная на рис. 13, сортирует в порядке убывания массив из пяти элементов, записанный в строке 10. Ремарки, замыкающие программу, поясняют смысл «музыкальных» операторов, задействованных в ней.

13

```
10 PLAY"C1 C2":DATA 5,3,-6,-69,-89,-5
20 READ N:DIM A(N):MAT READ A:FLAG=1:PLAY"C2"
30 PLAY"D2":WHILE FLAG=1
40 PLAY"C3":FLAG=0:PLAY"C3"
50 PLAY"G4":FOR I=2 TO N
60 PLAY"E5":IF A(I-1)<A(I) THEN
     PLAY"C6":SWAP A(I),A(I-1):FLAG=1:PLAY"C6"
70 PLAY"E5"
80 NEXT:PLAY"G4":WEND:PLAY"D2"
90 PLAY"C2":MAT PRINT A:PLAY"C2 C1":END
100 ФУНКЦИОНАЛЬНЫЙ БЛОК - ДО    ("C")
110 ЦИКЛ "ПОКА"          - РЕ    ("D")
120 АЛЬТЕРНАТИВА         - МИ    ("E")
130 ЦИКЛ "ДО"            - ФА    ("F")
140 ЦИКЛ С ПАРАМЕТРОМ    - СОЛЬ  ("G")
150 МНОЖЕСТВ.ВЕТВЛЕНИЕ   - ЛЯ    ("A")
160 ВЫЗОВ ПРОЦЕДУРЫ      - СИ    ("B")
170 ПРОГРАММА            - ПОЛНАЯ НОТА
180 ПЕРВОЕ ВЛОЖЕНИЕ      - 1/2
190 ВТОРОЕ ВЛОЖЕНИЕ      - 1/4
200 И Т.Д.
```


При работе с некоторыми версиями Бейсика возможна «паскалевская» манера выделения структуры программы отступами от левого края листинга. Правда, некоторые трансляторы в целях экономии памяти выкидывают неизбежные при этом пробелы во время трансляции. «Обмануть» машину в ряде случаев удается, поставив сразу после номера строки знак — разделитель операторов (двоеточие). Так и было сделано в Бейсик-программе (рис. 14) поиска максимума многомерной функции методом «Два шага» (модификация метода Хука — Дживса).

14

```
10 : N=...:REM N-ЧИСЛО АРГУМЕНТОВ:DIM X(N):GOTO 30
20 : Y=FNY(X(1),...,X(N)):RETURN
30 : FOR I=1 TO N:PRINT"X"I"НАЧ";:INPUT X(I):NEXT:J=1
40 : GOSUB 20:YMAX=Y:INPUT"НАЧ.ШАГ,ТОЧНОСТЬ?";D,DMIN
50 : WHILE D>DMIN:REM НАЧАЛО ВНЕШНЕЙ ИТЕРАЦИИ
60 :    REPEAT
70 :       P=0:REM ОБНУЛЕНИЕ ПРИЗНАКА
80 :       FOR I=1 TO N:REM ПЕРЕБОР ПЕРЕМЕННЫХ
90 :          FOR X=-D TO D STEP 2*D:REM ДВА ШАГА
100 :            X(I)=X(I)+X:GOSUB 20
110 :            IF Y>YMAX THEN
                    P=X:J=I:YMAX=Y
120 :            X(I)=X(I)-X
130 :         NEXT
140 :      NEXT
150 :      X(J)=X(J)+P:REM ПЕРЕХОД К НОВОЙ ТОЧКЕ
160 :   UNTIL P=0:REM КОНЕЦ ВЛОЖЕННОЙ ИТЕРАЦИИ
170 :   D=D/2
180 :WEND
190 :FOR I=1 TO N:PRINT"X"I"ОПТ="X(I):NEXT:REM ОТВЕТ
```

Несколько слов о русско- и англоязычном программировании.

Проблема лексической основы языков программирования волнует многих преподавателей информатики. Некоторые наивно полагают, что замена английских служебных слов на национальные эквиваленты (русские, литовские, туркменские и др.) может облегчить процесс освоения азов информатики. Но это иллюзия. На уроках музыки преподаватель часто использует итальянские термины, а на занятиях по ритмике — французские. И никому не приходит в голову заменять эти служебные слова на их русские эквиваленты ради того, чтобы ученикам легче было учиться.

С другой стороны, обучаемого нельзя лишать возможности самостоятельно выбрать лексическую основу языка программирования. Целесообразно трансляторы языков программирования организовать так, чтобы можно было в любой момент работы с программой переводить ее на другую лексическую основу. Пусть, например, ученик, сидя за пультом ЭВМ,

наберет служебные слова BASIC:LIST и получит англоязычный листинг программы; затем если он наберет БЕЙСИК:ЛИСТ, то получит русский вариант листинга.

Сформулируем основные выводы.

Наложение рамок структурной диаграммы на программы, записанные на любых языках программирования, позволяет четко выделить заложенный в них алгоритм.

Рамки структурной диаграммы делают похожими программы, написанные на «непохожих» языках программирования.

Вписывание листинга программы в структурную диаграмму помогает освоить законы структурного программирования при работе на «неструктурных» языках.

Знания основ программирования упрочняются при знакомстве с несколькими алгоритмическими языками. Поэтому целесообразно на уроках информатики работать с программами, выражающими один и тот же алгоритм, но написанными на двух или даже трех разных языках.

Программные и аппаратные средства современных ЭВМ позволяют при записи и выполнении алгоритмов задействовать цвет и музыку. Это также облегчает понимание логической структуры программ.

Целесообразно в ряде случаев стрелками отмечать на листингах программ процесс вызова процедур и подпрограмм, выполнение блоков по меню и т. д.

Литература

1. *Брудно А. Л.* Программирование в содержательных обозначениях. М.: Наука, 1968.

2. *Грэхем Р.* Практический курс языка Паскаль для микро-ЭВМ: Пер. с англ. М.: Радио и связь, 1986.

3. *Уолш Б.* Программирование на Бейсике: Пер. с англ. М.: Радио и связь, 1988.

Новинки для системы «Рига-микро»

Кохтла-Ярвеский городской центр НТТМ «Мыте» предлагает учебным заведениям новые программные разработки, расширяющие базу знаний системы «Рига-микро» для КУВТ-86, распространяемой центром на договорных основах (см.: Информатика и образование. 1988. № 6; 1989. № 1).

В комплект дополнительной поставки входят следующие дискеты с файлами базы знаний системы «Рига»:

обучающие программы по русскому языку на четырех дискетах (52 программных файла), содержащие программированные дидактические материалы, адресованные учителю-словеснику, оказывающие практическую помощь в решении проблем орфографической грамотности учащихся. С помощью этого материала можно проводить на КУВТ-86 контрольные и тренировочные работы по основным темам орфографии и морфологии с V по VIII класс, а затем собирать и анализировать результаты проверки знаний учеников. Программированный материал представлен в виде тестов, которые различны по содержанию и уровню сложности. Учитель сам отбирает необходимые программы в зависимости от этапа обучения и состава класса;

обучающе-контролирующие программы по немецкому языку на четырех дискетах (46 программных файлов), предназначенные для организации с помощью КУВТ-86 повторения, систематизации, закрепления и контроля знаний учащихся школ, средних специальных и высших учебных заведений по ряду основных тем учебной программы по немецкому языку;

пакет демонстрационно-обучающих программ по физике для IX—X классов средней школы на одной дискете. Содержит 11 программ по темам «Электричество» и «Ядерная физика». Используется при повторении пройденного материала и в самостоятельной работе учащихся для закрепления полученных знаний.

Конкретный набор файлов для поставки может формироваться по желанию заказчика.

Реквизиты для заключения договора (почтовый адрес, ФИО руководителя, телефон, рассчетный счет организации, МФО банка) просьба направлять по адресу:

202020, Кохтла-Ярве, РУС А/я № 15.
Директору центра НТТМ «Мыте»
Путконен М. А.

Телефон для справок:
26-510, 49-781 (код Кохтла-Ярве: 01433).

НПО «ГОРСИСТЕМОТЕХНИКА»

ЛАБОРАТОРИЯ ЭРГОНОМИКИ

Учебная база данных «Эргономика видеотерминальных систем»

Вы занимаетесь разработкой и проектированием персональных ЭВМ, автоматизированных рабочих мест, технических средств, связанных с видеотерминалами?
Вы хотите, чтобы ваш труд был эффективным, безопасным и комфортным?
Тогда вам не обойтись без эргономики.

С основными законами, положениями и справочными материалами эргономики быстро и наглядно познакомит вас наша программа

«Эргономика видеотерминальных систем».

Вы сможете на экране подобрать для вашего будущего устройства подходящие сочетания цветов, представить графический материал в нужной форме.

Все данные отображаются на экране и при необходимости могут быть отпечатаны.

Программа занимает на диске 97К байт, работает на ПЭВМ «Ямаха», при необходимости может быть переведена на любую 16-разрядную ЭВМ. Документация включает «Руководство пользователя».

Программа будет полезна не только разработчикам и проектировщикам, но и студентам вузов, специализирующимся по автоматизации, инженерной психологии и эргономике.

Программа оценки утомления

Чтобы ваша работа была эффективной, вы не должны переутомляться.
Определить, насколько вы утомлены, помогут вам программы

«Красно-черный квадрат»,
«Кратковременная память»,
«Последовательный контраст».

Они не требуют особого умения, не отнимут много времени. Каждая может использоваться самостоятельно.
Область применения — все сферы деятельности, в которых человеку приходится работать с видеотерминалами (школа, ПТУ, компьютерные клубы, автоматизированные учреждения и предприятия).
Результаты тестирования отображаются на экране и могут быть отпечатаны.
Программы занимают 5—8К байт, работают на ПЭВМ «Ямаха», могут быть переведены на любую 16-разрядную ЭВМ.
Документация включает «Руководство пользователя».

Наши программы помогут вам сберечь свое здоровье и машинное время.

Наш адрес: 252004, Киев-4, ул. Красноармейская, 23-б.
Справки по телефону 221-24-92.

НПО «Горсистемотехника»
Лаборатория эргономики

М. ЛЕБЕДЕВА
канд. пед. наук
ВНИИ профтехобразования

Требования к содержанию и оформлению педагогического сценария

Общие положения

Педагогический сценарий (технический проект педагогических программных средств вычислительной техники) относится к числу программных документов, содержание которого включает описание диалога учащегося и ЭВМ в процессе обучения, а также краткие инструкции по работе с компьютерной обучающей программой.

Педагогический сценарий составляется на основе технического задания и является материалом, с которым работает программист при создании рабочего проекта компьютерной обучающей программы.

Параллельно с педагогическим сценарием всегда разрабатывается алгоритм обучения. В нем фиксируются направления всех переходов, которые должны иметь место в обучающей программе; инструкция программисту, включающая требования к программированию.

Педагогический сценарий и алгоритм обучения целесообразно объединять в единый программный документ, называемый педагогическим сценарием.

Содержание педагогического сценария

Педагогический сценарий как программный документ должен включать следующие разделы:

«Введение»,
«Описание сценария»,
«Алгоритм обучения»,
«Источники, использованные при разработке ПС».

В зависимости от особенностей педагогического сценария допускается объединять отдельные разделы, а также вводить новые.

В разделе «Введение» указывают, на основании какого технического задания разработан сценарий; формулируют главные цели его создания.

Педагогический сценарий (в зависимости от целей) обычно включает следующие типы кадров: начальные (заставка), инструктивные, информационные, справочные, контролирующие, кадры-реплики, кадры-подсказки и др. В зависимости от способа представления информации кадры подразделяются на статические и динамические.

Начальные кадры (заставка) служат для описания названия компьютерной обучающей программы и представления ее разработчиков.

Заставка в общем случае содержит три кадра: титульный, эмблему, предметный.

В состав основных надписей титульного кадра обычно входят: наименование министерства-заказчика, наименование организации-заказчика; наименование организаций-разработчиков, название компьютерной обучающей программы; фамилии разработчиков, место разработки, город и год.

Второй кадр заставки должен соответствовать принятой эмблеме разработчиков. В ряде случаев эмблема может опускаться.

Третий кадр заставки включает наименование профессии, по которой создана компьютерная обучающая программа, названия предмета, раздела, темы.

Инструктивные кадры (ИН) содержат описание основных правил работы с компьютерной обучающей программой.

Наряду с наличием инструктивных кадров в самой компьютерной обучающей программе может готовиться специальный программный документ-инструкция учащемуся, в котором дается более подробное описание правил работы с программой.

Информационные (ИК) и справочные кадры (СК) включают основные теоретические сведения данной компьютерной обучающей программы, справочные сведения из смежных учебных предметов и данного учебного предмета.

Особое место среди информационных и справочных кадров занимают демонстрационные, которые иллюстрируют протекание природных и производственных процессов и явлений, моделируют работу аппаратов и установок, показывают ход опытов.

Контролирующие кадры (КК) представляют собой перечень вопросов и заданий, с помощью которых осуществляется контроль знаний учащихся.

Кадры-реплики (КР) содержат реплики-реакции на различные ответы учащихся.

Кадры-подсказки (КП) включают сведения, помогающие учащемуся работать с программой: отвечать на вопрос, выполнять задания и т. д.

Динамические кадры (ДК) отличаются от статических тем, что в них широко используется динамическая наглядность, например

управляемая мультипликация. По сравнению со статическими они более информативны, и поэтому их использование в автоматизированных обучающих программах предпочтительнее.

В разделе «Алгоритм обучения» помещаются описание алгоритма с указанием направлений всех переходов внутри автоматизированной обучающей программы и описание всех обозначений, использованных в схеме алгоритма.

Раздел «Источники, использованные при разработке ПС» содержит перечень учебников, учебных и методических пособий, научно-популярной литературы, дидактических средств обучения, на которые опирались авторы при разработке педагогического сценария.

Оформление педагогического сценария

Педагогический сценарий оформляется в соответствии с общими требованиями к оформлению программных документов.

В разделе «Описание сценария» могут быть использованы две формы его записи: бланковая и табличная.

При бланковом способе записи для оформления кадров педагогического сценария используются специальные бланки. Мы предлагаем три различные формы бланков для описания разных типов кадров (формы 1—3).

Форма 1 подходит для описания тех типов кадров, которые имеют только один выход, это, например, информационные кадры, справочные кадры, кадры-инструкции.

Формы 2—3 подходят для кадров с двумя и более выходами. Форма 2 наиболее часто может быть использована для описания контролирующих кадров в том случае, когда предусматриваются только две реакции на ответ учащегося: верный, неверный. Форма 3 — если предполагается более тщательный анализ ответов: верный, неверный, неполный, частично верный и т. д.

Форма 1

Тип и номер кадра	
Направление перехода	

Форма 2

Тип и номер кадра		
+	Направление переходов	—

Форма 3

Тип и номер кадра		
—2	Направление переходов	+
—3		—1

Формы 2 и 3 могут быть использованы для описания информационных кадров, если в них содержатся элементы проблемности, и учащиеся в процессе работы с информационным кадром должны отвечать на вопросы, выполнять задания, обращаться к справке, подсказке.

Бланковую форму удобно использовать и в тех случаях, когда описываются кадры, включающие рисунки, графики, схемы.

К числу недостатков бланковой формы записи относится тот, что в большинстве случаев она требует дополнительного составления отдельного программного документа — инструкции программисту, в которой даются пояснения принципов построения отдельных кадров.

Табличная форма представления содержания означает запись сценария в виде таблицы, например, по форме 4.

В графах формы 4 указываются:

№ п/п — номер записи;

участник диалога — учащийся или ЭВМ, к которым относится соответствующая запись;

содержание диалога — запись, которая воспроизводится на экране дисплея или должна вводиться с клавиатуры учащимся;

тип и номер кадра — тип кадра и номер его по порядку следования;

направление перехода — осуществляемый переход после окончания работы с описываемым кадром.

Приведенные рассуждения касались в основном способов оформления педагогических сценариев компьютерных обучающих программ. Для иллюстрации приводим фрагмент подобного педагогического сценария (рис. 1—4).

Форма 4

№ п/п	Участник диалога	Содержание диалога	Тип и номер кадра	Направление перехода

1

Тип и номер кадра	Инструктивный кадр 1	ИН1

При работе с компьютерной обучающей программой нужно соблюдать следующие правила:

1. Переход от кадра к кадру происходит путем нажатия клавиши ВОЗВРАТ КАРЕТКИ.
2. Не следует клавишу ВОЗВРАТ КАРЕТКИ нажимать дважды.
3. При ответах на контрольные вопросы следует внимательно вводить данные, чтобы избежать ошибки ввода.

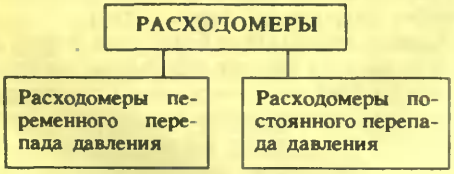
Направление перехода	Информационный кадр 11	ИК11

2

Тип и номер кадра	Информационный кадр 11	ИК11

ИЗМЕРЕНИЕ ОБЪЕМНОГО РАСХОДА

Приборы для измерения объемного расхода — РАСХОДОМЕРЫ



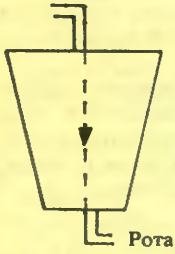
Направление перехода	Информационный кадр 12	ИК12

3

Тип и номер кадра	Информационный кадр 12	ИК12

ИЗМЕРЕНИЕ ОБЪЕМНОГО РАСХОДА

Расходомеры постоянного перепада давления



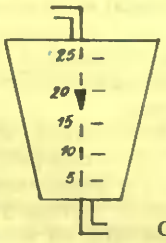
Понятен ли принцип действия ротаметра (Д/Н)?

Направление перехода		
	если «ДА»	Контролирующий кадр 11 КК11
	если «НЕТ»	Справочный кадр 1 СК1

4

Тип и номер кадра	Контролирующий кадр 11	КК11

Определите по показанию ротаметра значение объемного расхода



Очки _____

Ответ _____

Направление перехода		
	Если верно	Контролирующий кадр 12
	Если неверно	Информационный кадр 12

5

Описываемый параметр	Определение процесса

Процесс фильтрования

Процесс разделения суспензий с использованием пористых перегородок, которые задерживают твердую фазу суспензии и пропускают ее жидкую фазу

Переход к параметру	Движущая сила

6

Описываемый параметр	Движущая сила процесса

Процесс фильтрования

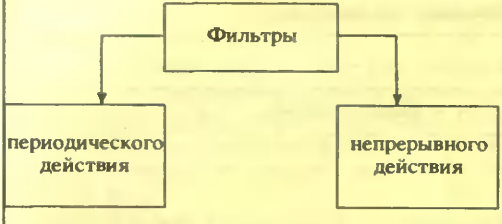
Движущая сила — разность давлений по разные стороны от фильтровальной перегородки

Переход к параметру	Аппаратурное оформление

7

Описываемый параметр	Аппаратурное оформление

Процесс фильтрования



Переход к параметру	Основные закономерности

Помимо компьютерных обучающих программ в учебном процессе могут широко использоваться другие виды педагогических программных средств, например информационно-поисковые системы учебного назначения. Для их описания мы также предлагаем бланковую форму (форма 5).

Форма 5

Описываемый параметр	
Переход к параметру	

Приводим пример описания информационно-поисковой системы учебного назначения (рис. 5—7).

Более подробно правила оформления программной документации ППС ВТ изложены в брошюре «Требования к документам для разработки и применения педагогических программных средств вычислительной техники в профессионально-технических учебных заведениях стран — членов СЭВ».

В ней определены требования к содержанию и оформлению программных документов: технического задания, педагогического сценария, инструкции программисту, текста программы, описания программы, методических указаний преподавателю-пользователю, инструкции учащемуся-пользователю.

Брошюра подготовлена во ВНИИ профтехобразования. Заявки на нее могут быть поданы по адресу: *191119, Ленинград, ул. Черняховского, 2.*

Литература

1. ГОСТ 19.004—80. Единая система программной документации. Термины и определения // Единая система программной документации. М.: Государственный Комитет СССР по стандартам, 1985. С. 28—29.

А. ГАУЕНС, В. СУПЕ
Вычислительный центр при ЛГУ им. П. Стучки

Редактор блок-схем

С наступлением эры персональных компьютеров значительно меняются стиль программирования, принципы взаимодействия программиста и ЭВМ. Развиваются удобства для программиста, появляются новые возможности, значительно повышающие производительность его труда. Большое внимание уделяется психологическим аспектам процесса работы с компьютером, улучшению сервиса и «дружественности настроя» ЭВМ.

Рассмотрим подробнее некоторые особенности работы с персональным компьютером, особенно важные при работе со школьниками.

1. Широко внедряется принцип функциональных клавиш, при котором пользователь минимально использует клавиатуру. Нажатие одной клавиши вызывает отображение целой конструкции. Этот принцип уже внедрен и в программные средства по курсу ОИВТ (примеры — Е-практикум, Рапира).

2. Появление справочных систем (HELP-систем) на больших и мини-ЭВМ было существенным шагом в повышении «дружественности настроя» ЭВМ, но изобразительные возможности при ориентации на алфавитно-цифровые дисплеи значительно уступают красочным и динамичным системам на ПЭВМ. Для некоторых систем (например, различные редакторы, базы данных) пользователь после краткого вводного курса вполне может обойтись без справочной и учебной литературы.

3. Применение меню для выбора действий освобождает пользователя от запоминания синтаксиса команд.

4. Широкие графические возможности ПЭВМ позволяют отказаться от традиционных служебных слов, использовать визуальное изображение операции. В некоторых наиболее развитых системах (например, Apple Macintosh) уже можно говорить о графическом стиле программирования.

5. Применение графических схем алгоритмов не только на стадии проектирования программ, но и в процессе общения с ЭВМ трансформирует программы из традиционных последовательных цепочек операторов в более привычное для непрофессионального пользователя двухмерное представление. Необходимость такого представления уже просматривается при «одномерном» программировании: программисты пытаются разместить операторы по горизонтали так, чтобы при вертикальном просмотре были явно видны ветви операторов

выбора. К сожалению, графическое представление алгоритмов по непонятным причинам исчезло из курса ОИВТ.

6. Использование образного стиля программирования позволяет достичь нового качества при отображении результатов работы программ. Для средств, специализированных в определенной области применения, можно разработать автоматизированные средства динамического отображения хода выполнения программы. Этот так называемый *принцип анимации* широко используется, например, для систем моделирования.

7. И наконец, звуковые возможности ПЭВМ. При индивидуальном использовании ЭВМ, когда это не мешает окружающим, звуковое сопровождение и звуковая информация о работе программы значительно улучшают настроение пользователя, а это тоже немаловажный фактор повышения производительности труда. Даже на примере игровых программ прослеживается закономерность — при включенном звуковом сопровождении играющий часто добивается лучших результатов, особенно в случаях, когда звуки используются в виде предупреждающей информации.

Такой стиль работы с ЭВМ сегодня еще не является повседневным для большинства программистов, но это недалекое будущее. Будущие программисты сегодня сидят за школьной партой. Уже сегодня их следует приобщать к ЭВМ при помощи таких программных средств, которые по возможности шире используют широкий ассортимент средств графики и звука.

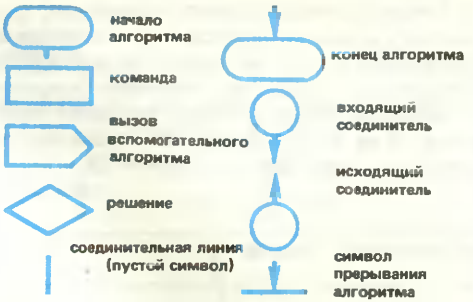
Первым шагом ученика к компьютерной грамотности является освоение понятия алгоритма, приобретение навыков постепенной формализации алгоритма, умение практически взаимодействовать с ЭВМ при помощи клавиатуры. Наиболее подходящим первым программным продуктом, осваиваемым учеником, может быть редактор блок-схем — графического представления алгоритма.

Графический язык

Для курса ОИВТ вполне достаточно графических символов, приведенных на рис. 1.

Внутри символов записываются тексты в свободной форме, что позволяет постепенно формализовать описание до синтаксиса, требуемого алгоритмическим языком (например, Е-практикумом).

Блок-схема позволяет наглядно проследить пути выполнения алгоритма по ветвям или в цикле, определить места вызова подалгоритмов и т. д.



Функции редактора

Редактор схем алгоритмов должен обеспечить ввод схемы, ее исправление, сохранение на магнитном носителе, передачу по локальной сети, вывод на печать, интерпретацию алгоритма при заданном синтаксисе заполнения символов.

В Вычислительном центре при ЛГУ им. П. Стучки разработан редактор блок-схем RED, выполняющий вышеуказанные функции, кроме интерпретации алгоритма (в этом направлении работа продолжается). Разработка выполнена для КУВТ «Ямаха-1».

Учитывая то, что некоторые технические средства (дисковод, печатающее устройство) имеет только ЭВМ учителя, редактор RED предоставляет различные функциональные возможности учителю и ученику.

Учитель может:

вводить алгоритм;

исправлять алгоритм;

сохранять алгоритм на диске или кассете;

считывать алгоритм с внешнего носителя;

распечатывать алгоритм;

передавать по сети алгоритм или вариант редактора для учеников;

получать по сети алгоритмы от обучаемых.

Возможности ученика более ограниченны:

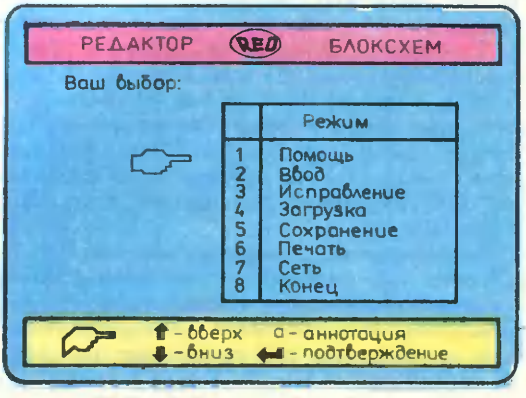
ввод алгоритма;

исправление алгоритма;

прием алгоритма по сети;

передача алгоритма по сети.

На рис. 2 приведено изображение меню

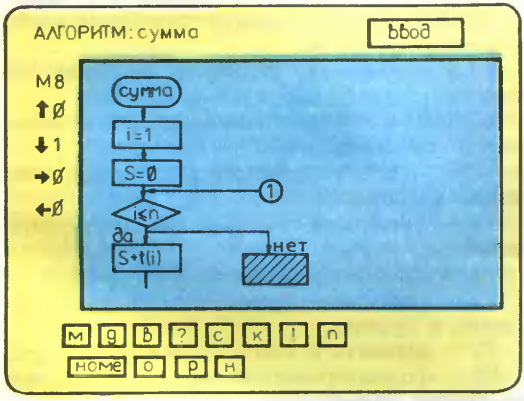


режимов (функций) редактора для учителя.

Работа с редактором

В начале урока учитель запускает свой редактор и при помощи режима «Сеть» передает его на ученические ЭВМ. Затем учитель может передать в том же режиме ранее подготовленные алгоритмы ученикам для доработки или поставить задачи на ввод новых алгоритмов. Выбор режима, параметров и действий происходит по меню при помощи клавиш выбора, всегда указанных в поле информации (нижняя часть экрана). Действия пользователя проверяются редактором, и в случае ошибки выдаются достаточно детальные сообщения.

При вводе или исправлении схемы алгоритма на экран компьютера (рис. 3) выводятся имя алгоритма и режим работы (верхняя часть экрана), данные о размещении схемы «окна» (левая сторона экрана), справочная информация о возможных действиях (набор «клавиш» в нижних строчках экрана).



Основную часть экрана занимает «окно», через которое видим часть «листа» с изображением схемы алгоритма. Данные в левой части экрана указывают, сколько строк или колонок в каждую из сторон от окна мы не видим. Графический курсор — желтый прямоугольник в окне — указывает на то место в алгоритме, где можно ввести следующий символ. В режиме ввода редактор управляет перемещением курсора и таким образом подсказывает место следующего символа. В режиме исправления пользователь сам может подвести курсор к тому месту, где необходимы вставка, удаление или замена символа. Редактор всегда проверяет правильность расположения символов и не допускает ввод неправильной схемы алгоритма. При достижении курсором края окна оно автоматически сдвигается, открывая новую часть схемы.

Ввод алгоритма осуществляется нажатием клавиши, соответствующей нужному символу, и последующим вводом текста в символ.

Исправление включает в себя нахождение места исправления (перемещение графического курсора и диаграммы), выбор действия (вставка, удаление, замена), выбор символа (при вставке или замене) и набор нового текста.

В режимах «Сохранение», «Загрузка», «Сеть» редактор требует указания параметров, например, типа устройства или имени файла. Во время выполнения действия (чтение, запись, передача по сети) на экран выводится рисунок, иллюстрирующий это действие.

Режим «Помощь» предназначен для демонстрации принципов действия редактора.

Мы рассмотрели только один пример системы обучения нового типа. Редактор позволяет приобщить ученика к компьютеру уже на самых ранних этапах обучения. Персональная ЭВМ, являясь средством обучения, дает возможность получить навыки работы, освоить возможности компьютера в широком диапазоне.

Практика показала, что наглядность оформления и визуального изображения повышает интерес, удерживает внимание обучаемого в течение урока и способствует восприятию материала.

Практика также подтвердила необходимость включения в систему визуального интерпретатора алгоритма. Это средство завершает этап обучения, предоставляет возможность решать небольшие практические задачи, когда обучаемый достиг уровня полной формализации алгоритма (включая тексты в символах). В результате освоенное по курсу ОИВТ средство обучения может быть использовано для решения задач и по другим предметам (алгебра, геометрия, физика и т. д.), особенно при создании своего «банка» алгоритмов.

А. ДУВАНОВ

«Микрон»

В бушующем море текстовых редакторов «Микрон» уступает многим по набору примитивов редактирования. Например, в нем нельзя конструировать макрокоманды. Кроме того, он работает лишь в режиме 32 символа в строке, в то время как прочие редакторы поддерживают 40- или даже 80-символьные режимы.

Зачем же создан еще один слабый редактор, когда есть много хороших?

Число символов на экране выбрано малым, так как «Микрон» предназначен для учеников младших классов. Большие буквы на экране читаются с меньшим напряжением глаз.

Что касается возможностей редактирования, то некоторые могут быть добавлены в следующие версии, например поиск по образцу, контекстная замена, а некоторые (например, макрокоманды, ввод в текст символа по его коду), по-видимому, не будут реализованы никогда. Принятие этого решения мотивировано опять же областью использования редактора: курс информатики для младших классов. Управление учебным программным средством должно быть понятно ребенку. В добавлении разных сложных возможностей как факультативных мы не видим большого смысла, ибо они увеличивают объем редактора, удлинняя время пересылки по сети. Когда ученик исчерпает возможности «Микрона», ему можно дать профессиональный редактор.

Основной причиной создания «Микрона» явилось желание контролировать бесчисленные ошибки школьников при наборе учебных текстов. Эта возможность реализована, и именно благодаря ей «Микрон» становится средством, полезным на уроках не только информатики, но и языка (родного и иностранного), и даже на других уроках, таких, как химия, физика.

Возможность «Микрона» контролировать набираемый текст путем сравнения с заранее заданным образцом по сути дела превращает его из редактора в электронного контролера со встроенным редактором.

Инструкция пользователю

«Микрон» — учебный экранный текстовый редактор, предназначенный для использования в сети микро-ЭВМ «Ямаха». Буфер текста редактора имеет объем в 300 строк при длине строки 32 символа. Возможно выполнение следующих операций.

ВПРАВО, ВЛЕВО, ВВЕРХ, ВНИЗ — стрелки управления курсором.

SHIFT+ВПРАВО — установить курсор на конец текста в строке.

SHIFT+ВЛЕВО — установить курсор на начало текста в строке.

HOME — поместить курсор в левый верхний угол экрана.

TAB — двигать курсор по позициям табуляции в строке.

RET — перевести курсор в начало следующей строки.

CTRL+ВВЕРХ — листать вверх по страницам.

CTRL+ВНИЗ — листать вниз по страницам.

BS — стереть символ слева от курсора.

SHIFT+BS — стереть слово слева от курсора.

INS — вставить пробел на место курсора.

SHIFT+INS — вставить строку на место строки с курсором.

DEL — удалить символ на месте курсора.

SHIFT+DEL — удалить строку на месте строки с курсором.

CLS — очистить редактор и экран от текста.

CTRL+E — очистить конец строки начиная с места курсора.

CTRL+N — разрезать строку на две в позиции курсора.

F1 — запомнить символ в позиции курсора в буфере.

F6 — напечатать символы из буфера.

F2 — запомнить строку, отмеченную курсором, в буфере.

F7 — вставить в текст строки из буфера.

F5 — форматировать абзац текста начиная со строки, отмеченной курсором.

Вывод текста на принтер может быть сделан на машине учителя, равно как и на машине ученика, при помощи клавиш CTRL+P либо обычным способом, принятым в ОС MSX.

Справочник «Микрона» вызывается клавишами CTRL+?. В справочнике работают клавиши:

ВВЕРХ — просмотр справочника вверх;

ВНИЗ — просмотр справочника вниз;

ESC — выход из справочника к редактируемому тексту;

CTRL+ESC — выход из справочника с копированием его текста в область редактирования.

По нажатию клавиши ESC на экран выводится следующее меню, обслуживающее работу с сетью, дисковые операции, загрузку и пересылку контрольных текстов:

Прием текста
Передача текста
Диктант
Запись на диск
Чтение с диска

Последние два раздела меню отображаются только на машине учителя (машина с номером 0). Выход из режима меню производится повторным нажатием на клавишу ESC.

Выбор на машине ученика раздела «Диктант» устанавливает компьютер в ожидание приема контрольного текста из сети. Учитель, выбирая этот раздел, загружает контрольный текст с диска и рассылает его в класс. Когда контрольный текст получен (он недоступен ученику для обозрения), можно начинать работу. Сверка набираемого текста с образцом выполняется по нажатию на клавишу SELECT. При этом на экране выдается одно из трех сообщений:

ОШИБОК НЕТ
ТЕКСТ НЕ ВЕСЬ
ОШИБКА

При нажатии на какую-либо клавишу сообщение пропадает и в случае наличия ошибок курсор указывает место первой из них.

«Микрон» на уроке

Для проведения диктанта или набора текста с контролем учитель, используя «Микрон» или любой другой редактор, набирает контрольный текст и записывает его на диск. Во время урока ученики по просьбе учителя входят в режим меню и выбирают раздел «Диктант». Убедившись, что все машины находятся в ожидании приема контрольного текста, учитель сам входит в раздел «Диктант» и осуществляет посылку в сеть. Затем ученики набирают текст с листа или пишут под диктовку учителя, контролируя себя при помощи клавиши SELECT.

Другой способ использования «Микрона» — упражнения на правку заданного текста.

При изучении языка ученику, как правило, предлагается большое число упражнений, в которых требуется вставить пропущенные буквы или слова. Традиционно они выполняются путем переписывания в тетрадь всего текста. «Микрон» позволяет пойти иным путем.

Учитель готовит текст упражнения в редакторе и записывает его на диск в качестве образца. Затем, согласно теме урока, учитель стирает в тексте некоторые слова или буквы, заменяя их, например, знаками вопроса. В таком виде текст рассылается в класс, затем в режиме «Диктант» передается образец и ученики приступают к работе и контролю своих действий при помощи клавиши SELECT.

Аналогично «Микрон» может быть использован для проведения тренировочных упражнений и по другим предметам.

Микрон разработан в ИПС АН СССР под экспериментальный курс информатики для младших классов «Роботландия». Адрес для запросов: 152140, Переславль-Залесский, а/я 46; тел. 08535 20715 (первые 5 цифр — код Переславля в междугородной телефонной сети).

Компьютерный салон «ЛУДУС»
предлагает программное обеспечение
для компьютеров ATARI XL—XE
на кассетах и дискетах

Наши программы помогут вам приятно провести время
с героями популярных компьютерных игр.
Обратившись к нам, вы получите прикладные программы
для сочинения музыки, создания компьютерной графики
и мультфильмов, а также программы
для обработки данных, редактирования текстов,
построения графиков, организации баз данных.
Полный каталог программ и услуг можно запросить по адресу:
200039, Таллинн-39, а/я 5424, «Лудус».

М. АЛЕКСЕЕВ, Т. АЛЕКСЕЕВА
г. Миасс

Доработка КУВТ «Агат» в школьных условиях

Похвальное слово «Агату»

Искреннее сочувствие вызывает душераздирающий рассказ о злоключениях одного учителя и двух комплектов «Агатов», опубликованный в «ИНФО» № 2 за 1988 г. Однако вспоминается почти народная поговорка: хорошо работает не та машина, что из-за океана, а та, что в хороших руках. Машину надо знать и любить! Иначе толку не будет — ни в столице, ни в провинции, ни на IBM PC, ни на программируемом калькуляторе.

«Агат» же на нашем небогатом компьютерном небосклоне уникален уже тем, что на нем с наибольшей полнотой реализована замечательная система «Школьница» с языком высокого уровня Рапира и богатейшими возможностями (подробнее см. в статье А. Ю. Кривцова в том же номере журнала). Характерно, что негативные мнения о «Школьнице» обычно предваряют фразами типа: «Я вообще-то с ней не работал...»

О невысокой надежности отечественной техники независимо от ведомства-производителя известно всем, и бесплодная затея — требовать невозможного при нашем уровне технологии. Естественное решение — в резервировании, дублировании ключевых узлов, распараллеливании узких мест.

И именно в КУВТ «Агат», в отличие от других старых и новых КУВТов, заложены дополнительные возможности для обеспечения надежности и живучести путем резервирования. Главное — это однородная структура комплекта, все ПЭВМ одинаковы, нет разницы между машинами учителя и ученика. Каждый из 13 «Агатов» оснащен дисководом и разъемом для подключения принтера и не зависит от того, готова ли (исправна ли) некая головная машина допустить его к находящимся в ее монопольном ведении единственному дисководу и принтеру. В случае выхода из строя какого-либо узла одной из машин всегда можно привлечь на помощь соседнюю или из двух неисправных сделать одну рабочую.

Справедливости ради необходимо добавить, что знать и любить «Агат» было бы все же легче, будь заводская документация полнее и если бы принципиальные схемы доставались руками из упаковочной коробки, а не теми неисповедимыми путями, на которые только любовь и может подвигнуть.

Как это делается

В базовом комбинате ИВТ школы № 18 г. Миасса Челябинской области, где с 1986 г. эксплуатируются 13 «Агатов», разработан и осуществлен комплекс организационно-технических мероприятий по обеспечению возможно более надежной работы техники путем доработки и резервирования ключевых узлов. Не всё здесь ново, но кое-что пригодится многим.

Несколько дисководов выделены в качестве автономных и помещены в специальные корпуса с тумблерами и индикаторами включения-выключения цепи защиты записи для работы с обратной стороной диска. Платы контроллеров дисководов оснащены вторыми разъемами, окна в задней стенке расширены, и автономные дисководы легко подключаются к любой машине в классе по первому или второму приводу. Копировать с двумя дисководами — сплошное удовольствие!

Все дисководы отрегулированы до хорошего уровня совместимости между собой; однако для оперативного обнаружения неисправностей и минимизации потерь за каждым дисководом на занятиях закреплен определенный рабочий диск, эксплуатируемый преимущественно на данном устройстве. Рабочие диски учащихся унифицированы: на обратной стороне — загрузочный модуль и сервисные программы системы «Школьница», на лицевой — Бейсик, драйверы и около 400 блоков для хранения и отладки разрабатываемых программ.

В кабинете установлены кассетные магнитофоны, используемые как при проверке теоретических знаний школьников, так и для хранения Бейсик-программ на личных кассетах.

Для улучшения теплового режима в «слабые» машины, из которых вынуты дисководы, над блоками питания установлены миниатюрные вентиляторы ВН-2, работающие на вытяжку.

Для фронтальной демонстрации обучающих программ на теоретических занятиях по различным предметам смонтированы цветной и черно-белый мониторы с экранами размером 61 см по диагонали. Эти мониторы являются также резервными на случай выхода из строя штатных дисплеев.

Еще одно узкое место — единственный принтер, который, несмотря на свое заграничное происхождение, тоже иногда ломается. Для снижения нагрузки на штатный принтер СРА-80 у нас задействованы два устройства «Роботрон-1154» (производство ГДР, другое название — «DARO-1154») из комплектов микро-ЭВМ «Искра-1256». Эти принтеры имеют нестандартный интерфейс, для их подключения созданы простые платы сопряжения и соответствующие системные программы — драйверы для Бейсика, «Школьницы» и графической печати.

В дальнейшем планируется осуществить сопряжение «Агатов» с планшетными потенциометрическими графопостроителями Н-306 и ПДП-4 из комплектов «Искры-1256».

Кто это делает

Отнюдь не региональные предприятия ВТИ с их малым опытом, дороговизной и неприемлемыми сроками. Про ставку инженера мы слышали и надеемся до конца века ее добиться. Пока есть лишь полставки лаборанта.

В основном эти работы выполнили на общественных началах инженеры А. Тукачев, А. Осьминин, Е. Федоров, В. Струц, И. Бощенко и школьник М. Булыгин, выполнили в условиях школьного кабинета, как говорится, «на коленке». В результате значительно улучшились возможности использования и работоспособность КУВТ: при загрузке от 30 до 40 ч в неделю, как правило, в нерабочем состоянии находится не более одной машины.

Работая с КУВТ-86

В нашем кабинете установлен КУВТ-86 с Бейсиком, а в соседнем Белгородском районе — с Фокалом. Разные языки — разные ситуации: соседи имеют и систему «Рига», и много программ по предметам школьного курса, а у нас абсолютно ничего, кроме некоторых игр, которые ничему не обучают, разве только использованию двух — шести клавиш.

Поэтому мы с учащимися сразу поддержали идею создания при журнале «Наука и жизнь» клуба пользователей БК-0010 «Контакт», направили туда свои заявки по интересам, получили адреса корреспондентов, с которыми можно обменяться соответствующей информацией. Результат — у всех корреспондентов БК-0010 с Фокалом и единственное, чем мы можем обменяться,— это программы в кодах. Но, как уже упоминалось выше, это в основном игры, причем игры зарубежные, в которых думать не нужно — знай себе стреляй, убивай, бомби, убегай и т. д. Кроме того, увлекаясь игрой, школьники (да и взрослые) начинают с силой бить по клавишам, а ведь клавиатура у БК-0010 и без того очень слабая, легко разбалтывается. Например, клавиша управления курсором «влево» очень часто при попытке отредактировать текст срабатывает как ВВОД, клавиша «вправо» отсекает текст справа от курсора, клавиша ввода срабатывает не менее двух раз в 80 % нажатий. Есть и другие неприятности. Из-за этого при создании программ приходится страховаться — вводить лишние операторы, например так:

```
20 INPUT «КАК ВАС ЗОВУТ»; A$
30 IF A$=« »THEN 20
```

Или еще:

```
40 PRINT «ВВЕДИТЕ ЧИСЛО»
50 INPUT S
60 IF S=0 THEN 50
```

А представьте, что S может быть и нулевым по условию задачи! Все это создает дополнительные трудности при создании программ для БК.

При подготовке программы мы исходим из того, что никакая программа не может заменить учителя, а является для ученика пособием — как учебник, словарь, справочник, только более увлекательным, диалоговым. Сценарии программ мы пишем сами, когда готовим тот или иной урок по информатике. Затем создаем программу.

Наш кабинет информатики является межшкольным, поэтому за неделю мы занимаемся с большим количеством параллельных групп, что дает нам возможность за короткое время основательно обкатать свои программы. Сразу видим, какие есть недостатки, что детям непонятно, а что лишнее и т. д. Сразу стараемся и исправить недостатки.

Из опыта преподавания информатики мы сделали вывод, что удобнее всего создавать программы, включающие в себя и теорию, и небольшой экзамен по ней или ранее изученному материалу. В конце обычно есть задание на программирование, которое ученик должен выполнить в тетради, а затем ввести свою программу в ЭВМ, отладить ее и сдать учителю.

Работы учащихся мы оцениваем по конечному результату, учитывая при этом: время работы ученика с обучающей программой; оценку компьютера за ответы на контрольные вопросы; правильность записи программы в тетради; рациональность решения; исполнение учеником своей программы на ЭВМ как конечный результат. Как видите, критериев довольно много. Если учитель все же сомневается в оценке знаний какого-либо ученика, то приглашает последнего к свободному компьютеру и с помощью программы-теста за 3—4 мин выявляет пробелы в его знаниях.

Уроки стараемся проводить в различной форме:

уроки-лекции, на которых применяются в основном программы демонстрационного типа, да и то не всегда;

комбинированные уроки, на которых теория и практика чередуются и собеседником ученика становятся сначала учитель, а затем ЭВМ;

контрольные работы, на которых, как правило, первое задание — работа с контролирующей программой, а остальные — на карточках;

практические работы, на которых учащиеся в течение всего занятия работают в основном самостоятельно, сначала по программе-тесту, в конце которой после правильно выполненного задания выдается на экран задание практической работы и сценарий програмы. Эту программу ученик должен написать, набрать, отладить и, исполнив, сдать работу учителю;

уроки-семинары, на которых разбираем задачи из контрольных или практических работ или создаем программу по какой-то интересной задаче, и т. п.

Среди учеников есть энтузиасты, благодаря которым учителям становится легче работать. Такие учащиеся глубже постигают язык программирования, на уроках на них всегда можно опереться, задания они получают более сложные, чем другие. А когда есть такая инициативная группа, то можно с ними во внеурочное время поработать над программами по другим предметам.

Мы, конечно, понимаем, что такую работу нужно проводить только в тесном контакте с учителями-предметниками. Так и делаем. Нами созданы программы по математике II класса, по истории VII класса, по физике X класса, по английскому языку и некоторые другие. Трудности вызывает создание программ по немецкому языку — ведь на клавиатуре буквы только латинского и русского алфавитов. Интересно, а как в будущем, когда техники будет достаточно? Неужели учителям немецкого языка будет отказано в применении ЭВМ на своих уроках?

Конечно, методика работы с программами у нас только разрабатывается, но наш кабинет загружен более 60 ч в неделю и, конечно, опыт накапливается быстрее, чем в школах, где 12—16 ч информатики.

Для каждого учащегося у нас напечатан на принтере задачник, содержащий более 100 задач, начиная от самых простых, и справочный материал по Бейсику (типы ошибок, стандартные функции и т. п.).

Текст задачника помогали набирать учащиеся на занятиях кружка. Это делалось так: ученик на БК-0010 набирает текст в режиме 64 символа в строке, используя лишь номера строк, но не применяя операторов Бейсика, например:

```
10
20
30'        ЗАДАЧА 1
40 НАПИШИТЕ ПРОГРАММУ, ВЫВОДЯ-
ЩУЮ НА ЭКРАН
50 СУММУ ДВУХ ЧИСЕЛ, ВВОДИМЫХ С
КЛАВИАТУРЫ,
60 ЕСЛИ ОНА БОЛЬШЕ 100.
70
```

Затем такой файл пересылается по сети в ДВК-2МШ и записывается на диск. Затем он редактируется в редакторе ED, т. е. приводится к форме, которую и должен иметь текст в задачнике, и записывается на диск в таком виде:

```
        ЗАДАЧА 1
НАПИШИТЕ ПРОГРАММУ, ВЫВОДЯ-
ЩУЮ НА ЭКРАН
СУММУ ДВУХ ЧИСЕЛ, ВВОДИМЫХ С
КЛАВИАТУРЫ,
ЕСЛИ ОНА БОЛЬШЕ 100.
```

После редактирования мы объединяем такие файлы в один (по нумерации задач) в редакторе TR81 и записываем окончательный вариант на диск, где хранятся различные тексты. Выводим на печать в редакторе TR81.

Так можно создавать не только тексты, но и программы для БК или ДВК.

Если нужно быстро набрать интересную программу, текст которой есть только на бумаге, то мы поступаем так: на каждом БК набирается определенный фрагмент программы и передается на ДВК. Затем в редакторе мы их объединяем в один файл.

Быть может, это известно многим, но кому-то наш опыт, надеемся, пригодится.

Кроме того, создав простой коммутатор, мы можем с одного магнитофона загружать на все БК-0010 любые программы. Это очень удобно, так как программа одновременно загружается сразу на все компьютеры и ДВК не нужен совсем. На нем можно в это время вести другую работу.

Всем желающим предлагаем безвозмездно

или в обмен на их программы наши разработки по информатике, включая задачи, контрольные работы и пр. Всю информацию можем переписать на высланные дискеты (133 мм), а также на кассеты или бобины.

Наш адрес: 309250, Белгородская обл., г. Шебекино, ул. Ленина, 19. Кабинет информатики.

А. КЛИМАЧКОВ, И. ТУРЯНСКАЯ

В. АНИКЕЕВ, Н. ЦВЕЙМАН
г. Алма-Ата

Бейсик-букварь

Что делать, если в программе на Бейсике есть синтаксические ошибки и вы не можете их обнаружить? Пользователям «Ямахи» в такой ситуации поможет программа **«БЕЙСИК БУКВАРЬ»**, разработанная в КВТ алма-атинского ЦИУУ.

Она работоспособна в локальной сети КУВТа, занимает в ОЗУ менее 1К, сравнительно быстро загружается на рабочее место ученика (за 35—40 с). Основное ее назначение — оказание помощи ученикам при написании программ на Бейсике. Кроме того, она может служить обучающим средством при освоении основных операторов этого языка, в частности при самостоятельном его изучении пользователем, освоившим первоначальные навыки работы на ПЭВМ.

Для входа в «БЕЙСИК БУКВАРЬ» необходимо нажать на клавишу **SELECT.** При этом на экране дисплея появляется оглавление, включающее 40 основных операторов Бейсика.

PRINT	RUN	END	LIST	REM
ATN	SIN	COS	TAN	SQR
LOG	EXP	INT	ABS	INPUT
NEW	DATA	READ	CLS	STOP
CONT	IF—THEN	ELSE	GOTO	LOCATE
SCREEN	COLOR	PSET	TIME	PRESET
LINE	CIRCLE	PAINT	FOR—TO	NEXT
STEP	GOSUB	RETURN	DRAW	RND

В левом верхнем углу оглавления находится курсор, который с помощью клавиш со стрелками может быть установлен на любом из операторов. После нажатия клавиши ПРОБЕЛ на экране дисплея выдается сообщение об операторе, указанном курсором.

В сообщении дается написание оператора, произношение, перевод на русский язык, назначение, синтаксическое описание в общем виде, пример использования в программах и результат исполнения. После нажатия клавиши SELECT или любой другой управляющей клавиши происходит выход из программы «БЕЙСИК БУКВАРЬ» и возвращение к той программе, с которой работал пользователь, причем курсор возвращается в ту же позицию, где он находился до входа в «БЕЙСИК БУКВАРЬ».

Достоинства программы — простота пользования, возможность обращения к ней в любой момент, когда в этом появляется необходимость, столь же простое возвращение к своей программе. «БЕЙСИК БУКВАРЬ» позволяет выдавать сообщение об ошибках в программе пользователя как на английском, так и на русском языке, что очень удобно для начинающих. Положительным моментом является и введение каждого последующего оператора через предыдущие.

Работа над «БЕЙСИК БУКВАРЕМ» продолжается. Во второй части этой программы намечается включение сервисных операторов.

Желающие получить дополнительную информацию и материалы могут обращаться по адресу: 480020, Алма-Ата, ул. Митина, 6, кабинет информатики ЦИУУ; тел. 64-58-55.

Рынок ЭВМ

Электронно-вычислительная техника пока дефицитна, однако по сравнению с 1985 г., когда в школах началось преподавание ОИВТ, ситуация с ее производством изменилась кардинально. Одним из проявлений этого, заметным и важным для многочисленных энтузиастов информатики, стала торговля ПЭВМ за наличный расчет. В первом обзоре этого рынка будет рассказано о товарах, предлагаемых Министерством электронной промышленности СССР. Темами следующих обзоров будут другие министерства и кооперативы *.

В Москве, на Ленинском проспекте, в доме 87, расположен фирменный магазин «Электроника». Здесь продаются (только за наличный расчет) 16-разрядные компьютеры: несколько моделей ДВК, «Электроника-85», «Электроника УКНЦ», БК-0010.01, комплекс КУВТ-86, у которого, кстати, недавно сменилась учительская ЭВМ — вместо ДВК-2М теперь поставляется ДВК-3. Сводные данные по ПЭВМ приведены в таблице. Самое интересное в ней, конечно, цены. Они договорные и поэтому довольно высоки.

Каждая ЭВМ (за исключением БК-0010.01) комплектуется монитором, принтером, дисководами и двумя дискетами. Иногда в комплект входят графопостроитель и жесткий диск (винчестер). К сожалению, состав комплекта магазин варьировать не разрешает. Впрочем, предлагаемые наборы разнообразны, выбрать есть из чего, и спрос достаточно велик: на все нужно записываться в очередь.

Кроме ПЭВМ, в магазине продаются дисководы для гибких дисков 5,25 дюйма «Электроника НГМД-6121А» и МС 5305 (запись с двойной плотностью, 800К байт на дискету) по 950 рублей и мониторы — черно-белые и цветные (цена — около 500 рублей). Есть и программное обеспечение для БК-0010.01: кассета МК-60 с игрой «Биллиард» (9 рублей 50 копеек) и набор из 6 кассет с прикладными программами — Бейсик, система «Рига», ФОКОД и т. п. (195 рублей).

Ожидается поступление БК-0011. Его характеристики: ОЗУ — 128К байт (из них ОЗУ экрана — 32К байт), ПЗУ — 40К байт

* Редакция будет признательна кооперативам, торгующим вычислительной техникой, за предоставление информации об их деятельности.

Название	ОЗУ, К байт	ПЗУ, К байт	НГМД, К байт	НЖМД, М байт	Максимальное число точек экрана	Ориентировочный объем продаж в 1989 г.	Программное обеспечение	Цена, рублей
БК-0010.01	32	24	—	—	512×256	5000	Фокал	650
«Электроника УКНЦ» (МС 0511.01)	192	32	2×400	—	640×288		ФОДОС-2, Бейсик, Фортран, Паскаль, Модула-2, Макро	7 417[1] 9016[2]
КУВТ-86								35 820
«Электроника-85» (МС 0585.01)	512	16	2×800	5	800×240	1500	ОС ПРОС, Бейсик, редактор текстов	19 295[4]
ДВК-3 (МС 0507.01, 0507.02, 0507.03)	64 256 512	8	2×800	— 5—10 5—10	800×240	3400	ОС ДВК, Бейсик, Фортран, Паскаль, редакторы текстов	От 9 744 до 15 212[3]
ДВК-4 (МС 0507.04)	До 1024		2×800	5—10	800×240	100	То же, что у ДВК-3	От 19 716 до 20 244[3] 27 024[4]

Примечание. [1] С черно-белым монитором. [2] С цветным монитором. [3] В зависимости от марки принтера, марки монитора, наличия и марки винчестера. [4] С графопостроителем.

(возможно подключение внешнего ПЗУ на 32К байт), графика от 128×256 точек (16 цветов) до 512×256 точек (черно-белая), контроллеры принтера, НГМД, локальной сети. Ориентировочная цена — 1 200 рублей; по-видимому, продаваться БК-0011 будет только в комплекте с принтером и НГМД, ориентировочная цена комплекта — 9 500 рублей.

Особый интерес вызывает стоящий в витрине экспонат под названием «Электроника МС 1208» (ПК-100) — почти карманный (примерно 3×10×25 см) компьютер с ОЗУ 16К байт, ПЗУ 32К байта, графическим жидкокристаллическим дисплеем примерно 90×140 точек. Однако появление этого перспективного образца в продаже вряд ли можно ожидать в ближайшие годы — слишком мал объем производства.

К. ШЕХОВЦЕВ

За время подготовки журнала к печати цены изменились: УКНЦ стала стоить от 5315 до 8950 рублей, МС 0507.02 — от 14 160 до 18 003 рублей.

Магнитофон или дисковод?

Кто из владельцев БК не хотел бы заменить магнитофон, подключенный к БК в качестве внешнего запоминающего устройства на дисковод! Даже если на диске будет помещаться хотя бы 160К байт, это существенно упростит работу с компьютером. Ну а если это будет дисковод для 3,5 дюймовых дискет, на каждую из которых помещается как минимум 720К байт данных, от такого устройства не откажется никто. Как известно, память на основе жестких дисков, или дисковод типа «Винчестер», существенно увеличивает возможности машины — на что только не был бы способен БК, оборудованный жестким диском хотя бы на 20—30М байт.

Но вот парадокс: оказывается, владельцы машин более высоких классов, оснащенных дисководами, способными хранить по 60М байт информации, стремятся вдобавок приобрести еще и магнитофон для копирования содержимого диска на магнитофонную кассету. Впрочем, парадоксом это может казаться лишь на первый взгляд — этому есть объяснение. Дело в том, что информация, хранящаяся в компьютере, весьма дорога — диск хранит не только важные и собранные по крупицам данные или полученные путем длительных расчетов результаты, но зачастую и концентрированные знания многих специалистов,— короче, некий интеллектуальный продукт, цена которого во много раз может превышать стоимость десятка таких компьютеров со всеми их периферийными устройствами, включая дисководы и магнитофоны. В то же время существует много возможностей потерять столь ценную информацию. Здесь можно вспомнить о надежности техники — магнитные головки, которые читают записи на диске, парят над ним на тонкой воздушной подушке, образующейся за счет быстрого вращения диска. Это означает, что достаточно резкого движения корпуса компьютера, например, если попытаться переставить его на соседний стол, заранее не выключив ЭВМ, чтобы магнитная головка и диск были испорчены из-за мимолетного соприкосновения. Здесь и возможные дефекты изготовления, которые в один момент могут свести на нет многолетний эксперимент; здесь броски напряжения и частоты в нашей электросети — можно вспомнить еще немало причин.

Не реже, а то и чаще, чем технические неполадки, к потерям информации с диска может привести «человеческий фактор». Это распространившиеся и в нашей стране «вирусы» — программы, способные обратить всю информацию на диске в бессмысленный набор нулей и единиц. Подхватить такой вирус очень легко — он может просочиться по компьютерной сети (в том числе и локальной), часто его заносят любители дареных игр. Избавиться от него не так легко, а иногда и невозможно, не потеряв информации.

Сравнительно надежный способ сохранить необходимую информацию — это завести архив. Ежедневно, перед тем как закончить сеанс работы с компьютером, все содержимое диска копируют на дискеты. Теперь в случае поломки дисковода, или потери информации на основном диске, вы потеряете только то, что было сделано за время, прошедшее от последнего копирования. Но на практике, несмотря на кажущиеся достоинства метода, им пользуются реже, обычно раз в неделю, рискуя большим объемом информации. Дело в том, что копирование оказывается медленным процессом, отнимающим слишком много

времени да и дискет. В самом деле, представьте, что нам необходимо скопировать содержимое 25Мбайтного диска. В нашем распоряжении дисковод, позволяющий на одну дискету записывать 720К байт информации. Нетрудно подсчитать, сколько таких дискет понадобится для ежедневного копирования. Да и оператору придется потрудиться...

В то же время выход есть — на помощь вновь приходит магнитофон. Впрочем, поскольку он значительно отличается от обычного своей электронной и механической начинкой, то иное и название — чаще всего его именуют системой бекап или стримером. Вставив в такой магнитофон кассету размерами чуть меньше той, что используется для видеозаписи, вы сможете записывать на каждую такую кассету до 25М байт, причем с высокой скоростью — 1М байт в минуту. Устройство оказалось столь удобным, что его стали встраивать прямо в корпус компьютеров, таких, как IBM, наряду с дисководами. Впрочем, для того чтобы им мог воспользоваться владелец любого IBM-совместимого компьютера, устройство выпускают и в виде отдельного блока. Кстати, тех, кто намучился с БК, безуспешно пытаясь считать без ошибки длинный файл, поспешу успокоить — магнитофон для IBM не только записывает данные на ленту, но одновременно проверяет их верное считывание.

Долго ли будет конкурентоспособным подобное устройство? Может быть, его заменяет перезаписываемые оптические диски или гибкие дискеты сверхвысокой емкости, но пока что производителям ЕС-1841 и разработчикам ЕС-1842, наверное, стоит подумать о таком устройстве, как магнитофон для компьютера.

А. БОРИСОВ

82

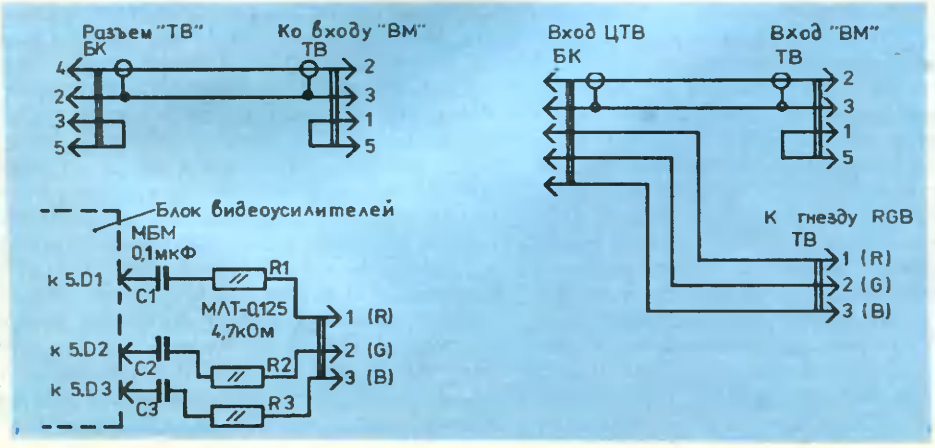
Рекомендую, проверено на практике!

О подключении БК-0010.01 к телевизорам различных марок рассказывалось уже немало. Вспомним хотя бы заметки С. Зильберштейна, В. Кондрашева «Радуга на экране БК» (ИНФО. 1989. № 2.), А. Бойко «Как подключить БК к телевизору» (Наука и жизнь. 1988. № 12.). Владельцы «Электроники-Ц432» смогут подключить к этому телевизору свой БК-0010.01 как в черно-белом, так и в цветном вариантах. Переделка довольно проста, но не следует забывать, что стоимость компьютера достаточно велика, а в телевизионном приемнике можно встретить довольно высокие напряжения, поэтому рекомендуем браться за дело лишь радиолюбителям, которые уверены в своих силах.

Черно-белый вариант. Напряжение +12 В с вывода 5 разъема X1 (все обозначения даны по схеме телевизора), подключите проводником к контакту 5 видеовхода, т. е. к разъему X5 в блоке AS1. Соберите кабель по схеме и подключайте компьютер.

Цветной вариант. Прежде всего нужно сделать в телевизоре отсутствующий там RGB-вход. Эта переделка (кстати, по нашим сведениям, такая работа, выполненная без квитанции в телеателье, стоит 25 рублей, так что есть шанс сэкономить!) выполняется за считанные минуты по схеме, которая изображена чуть ниже. Следующий этап — доработать видеовход, как это делалось для черно-белого варианта. Последний этап — собрать кабель в точном соответствии с третьей схемой.

С. АСТАШЕНОК,
г. Минск



Если надоел звук

Работая с БК на Бейсике, иногда хочется выключить звуковой сигнал, который подтверждает нажатия клавиш. Для того чтобы выключать и включать сигнал всякий раз, когда это требуется, я разработал небольшую программу-драйвер. Для записи драйвера на магнитофон наберите короткую Бейсик-программу из 10 строк. После запуска она сформирует в стековой области подпрограмму в машинных кодах. Чтобы записать ее на магнитофон, воспользуйтесь командой BSAVE "SNDOFF", 256, 348. Теперь всякий раз, чтобы отключить звук, набирайте команды BLOAD "SNDOFF", R и включайте магнитофон в режим воспроизведения.

Считанный драйвер автоматически запускается и передает управление Бейсику. Чтобы снова включить звук, достаточно набрать команду A-USR (A).

Функция была определена ранее при загрузке драйвера. Для тех, кто одновременно с драйвером собирается использовать и другие программы в кодах, важно знать адрес загрузки драйвера & 0400(256) и его длину после загрузки — & 0134(92). После включения драйвера в систему длина уменьшается до & 0100(64).

```
10 DATA 287, 5599, — 22336, 1088, 135, 259, 5623,
                                    —32152, 20, 259, 5623
20 DATA —32004, 12, 2335, —28448, 4454, 2551,
                                    4, 95, 0, 3039, 148, 770
30 DATA 2527, —28322, 5599, —1, 146, 5568, 3000,
                                    32257, 95, —31670
40 DATA 5623, —30716, —70, 5599, 266, 48, 5599,
                                    274, 188, 5599, 256
50 DATA 1088, 95, —24448
60 FOP A %-256 TO 346 STEP 2
70 READ D %
80 POKE A %, D %
90 NEXT A %
100 END
```

О. СЕМЕНОВ,
г. Севастополь

Пишем в служебной строке

Сообщение, появляющееся на телеэкране в служебной строке, придает программе «солидность». Чтобы добиться такого эффекта, обычно пишут короткую программу в кодах, использующую системную команду EMT22 (вывод символа в служебную строку). К такой программе из Бейсика-БК обращаются с помощью функции USR. Есть, однако, иной, более простой способ.

Командная строка POKE112, 15360+ +PEEK(132)+2* M установит курсор в M-тую позицию служебной строки, как это сделал бы оператор LOCATE M, Y, если было бы возможно задать соответствующее значение Y. (в режиме 64 символа в строке умножать M на два не надо). Разумеется, это может быть не только командная строка, но и оператор программы. Теперь можно выводить в служебную строку любые символы в любых режимах командой PRINT.

Программа-пример из двух строк, приведенная ниже, очищает экран и служебную строку, а затем выводит надпись PROGRAM красными (для владельцев БК, работающих в цветном варианте) инверсными символами и устанавливает обычный режим вывода (красные буквы).

```
10 POKE 112, 15360+PEEK(132)
20 ?CHR○ (148) CHR○ (158) CHR○ (145)
CHR○(156) "PROGRAM" CHR○ (156)
CHR○ (12)
```

С. КОМАРОВ,
г. Львов

Л. ШТЕРНБЕРГ,
канд. физ.-мат. наук
Куйбышевский авиационный институт

Уроки с программируемыми микрокалькуляторами

Многие учителя информатики 1 сентября начнут учебный год вооруженными новой вычислительной техникой. Хотя число дисплейных классов растет, для многих школ они пока недоступны. На сегодняшний день наиболее распространенными остаются программируемые микрокалькуляторы. Как же рационально распланировать учителю время уроков с ПМК? Надеемся, что ему помогут статьи этого цикла.

В качестве одного из вариантов можно предложить такой.

IX класс

Уроки 1—6 проводятся так же, как и в безмашинном варианте. Даются основные понятия на примере простых исполнителей (с. 1—29 [1] или соответствующие разделы по [2]). Рекомендуется пропустить материал «Форма представления информации в ЭВМ» — он будет дан позже, так как на 2-м уроке он воспринимается плохо. В качестве простых исполнителей рекомендуется взять «Муравья» [4, 5], графический исполнитель (с. 77—88 [1]), исполнители «Путник» и «Резчик металла» [2].

Уроки 7—9. Понятие переменной величины, линейные алгоритмы работы с величинами (с. 29—34 [1] или соответствующий материал из [2]).

Уроки 10—14. Структура ЭВМ на примере ПМК. ПМК как исполнитель. Структура его команд. Работа в режиме вычислений, запись программ в память и их выполнение. Перевод линейных алгоритмов на «язык» ПМК. Материал к этим урокам изложен в предыдущих статьях цикла. Технические детали подробно рассмотрены в [3] — этот материал может быть использован как учителями, так и учениками. Работа со стеком рассмотрена в [7].

Уроки 15—16. Разветвляющиеся алгоритмы. Теоретический материал и решение задач по учебнику.

Уроки 17—18. Перевод ветвлений на язык ПМК. Методическая сторона этих уроков рассмотрена ранее, техническая изложена в [3].

Уроки 19—20. Контрольная работа и ее анализ.

Уроки 21—28. Циклы. В этой серии уроков пока не затрагиваются вложенные циклы и табличные величины. Методические вопросы рассмотрены в настоящей статье.

Уроки 29—30. Табличные величины (без обработки на ПМК).

Уроки 31—32. Контрольная работа и ее анализ.

Уроки 33—34. Экскурсия на вычислительный центр (эти уроки могут быть проведены не обязательно в конце — их можно дать в любое время).

Остальной материал дается в X классе. Как видим, пока что охвачен далеко не весь материал IX класса. Отметим изменение порядка подачи материала: в уроки 21—28 входит часть разделов «Решение задач из курса математики» и «Решение задач из курса физики» — тех, которые для решения требуют изучаемых технических приемов (циклов определенного вида).

Продолжение. Начало см.: Информатика и образование. 1989. № 2, 3.

* Звездочка означает, что материал этих уроков дается по учебнику.— *Прим. автора.*

X класс

Уроки 1—3. Повторение. Один из уроков отводится для повторения работы на ПМК.

Уроки 4—5*. Задачи на табличные величины. Материал разделов «Решение задач из курса математики» и «...физики», связанный с обработкой таблиц.

Уроки 6—8.** Косвенная адресация, обработка табличных величин на ПМК.

Уроки 9—13. Циклы с параметром. Решение задач с их применением; приведение их к виду, реализуемому на ПМК.

Уроки 14—17. Двумерные таблицы. Вложенные циклы. Реализация вложенных циклов на ПМК.

Уроки 18—19. Контрольная работа и ее анализ.

Уроки 20—22*. Вспомогательные алгоритмы. Параметры. Использование вспомогательных алгоритмов для создания библиотек.

Уроки 23—24.** Подпрограммы на ПМК. Реализация вспомогательных алгоритмов в виде подпрограмм.

Урок 25*. Основные этапы решения задач на ЭВМ [1].

Уроки 25—28. Алгоритмы вычисления значений функции и их реализация на ПМК.

Уроки 29—31. Команда выбора, ее применение в задачах. Реализацию на ПМК следует только упомянуть: программируются вложенные **если**, которым эквивалентен **выбор**.

Уроки 32—33. Контрольная работа и ее анализ.

Уроки 34—36*. Алгоритмы работы с литерными величинами.

Уроки 37—39. Алгоритмы поиска информации.

Уроки 40—47.** Устройство ЭВМ. Основной алгоритм работы процессора. Организация ветвлений и повторений на ЭВМ. Подпрограммы. Кодирование информации в памяти ЭВМ. Решение задач на составление программ для ЭВМ. (Материал в безмашинном варианте достаточно труден, но при наличии ПМК его можно рассказать быстро и понятно, проводя аналогии с ПМК.)

Уроки 48—50*. Логические элементы ЭВМ, переработка информации с их помощью. Физические принципы работы внешних устройств.

Уроки 51—52. Контрольная работа и ее анализ.

Уроки 53—60*. Язык программирования Рапира. Решение задач.

Уроки 61—66*. Роль ЭВМ в современном обществе. Решение задач для повторения. Часть задач можно решить с использованием ПМК.

Уроки 67—68. Контрольная работа (итоговая) и ее анализ.

Обратите внимание, что на контрольные дается два урока. На 2-м уроке важно показать не только правильные решения, но и типичные ошибки, разъяснив, как исполнитель среагирует на ошибку: если ошибка синтаксическая, исполнитель не поймет, а если смысловая — понятное предписание неисполнимо либо дает не тот эффект, который нужен.

Для учителей, которые работали в IX классе по безмашинному варианту, а к X получили ПМК, можно рекомендовать план уроков в X классе: проводится один урок на повторение линейных алгоритмов, затем серия «калькуляторных» уроков на эту тему (уроки 10—14 для IX класса в описанном выше плане); один урок на повторение разветвляющихся алгоритмов, серия «калькуляторных» уроков по этой теме (уроки 17, 18 для IX класса) и т. д. При такой схеме уроков к концу полугодия будут пройдены все калькуляторные темы, далее можно идти по описанному плану для X класса.

Перейдем к рассмотрению методики проведения уроков 21—28 для IX класса. Тема «Циклы».

В учебнике [1] циклы вводятся сразу вместе с табличными величинами (§ 10) — это методический недостаток: вводятся сразу два новых понятия, поэтому материал воспринимается учениками хуже, чем хотелось бы. Рекомендуется начать циклы с наиболее простой и естественной конструкции (см. рис. 1а):

повторять M **раз** (или **цикл** M **раз**)
тело цикла
кц

С ее помощью гораздо проще записываются, например, алгоритмы *степень* (с. 50 [1]), *площадь* (с. 64 [1]) и др.

Сравним тела алгоритмов:

цел i $y:=1$; $i:=1$; **пока** $i \leqslant N$ **нц** $y:=y \cdot a$; **кц**	$y:=1$; **цикл** N **раз** $y:=y \cdot a$ **кц**
вещ h, x; **нат** i; $h:=(b-a)/n$; $S:=0$; $x:=a$; **пока** $i \leqslant n$ **нц** $S:=S+f(x) \cdot h$; $x:=x+h$; $i:=i+1$ **кц**	**вещ** h, x; $h:=(b-a)/n$; $S:=0$; $x:=a$; **цикл** n **раз** $S:=S+f(x) \cdot h$ $x:=x+h$; **кц**

** Две звездочки означают, что методика изложения этих уроков будет рассмотрена в следующих статьях данного цикла.— *Прим. автора.*

Проведя два урока по изучению конструкции цикла и решению задач (можно рассмотреть задачу колебания шарика на пружине или вычисление значений какой-либо функции в равноотстоящих точках), проводим урок по реализации этой конструкции на ПМК. Объяснение работы команд цикла FL 0, FL 1, FL 2 и FL 3 (общее название — FLi) займет порядка 10 мин, так как они во многом похожи на ранее изученные команды переходов. Далее формулируются строгие правила программирования таких циклов.

1. Если в алгоритме имеется цикл типа **раз**, распределение памяти начинается с отведения регистра из Р0 ÷ Р3 для счетчика. (впоследствии, при изучении вложенных циклов, следует упомянуть, что для вложенных циклов такого типа потребуется несколько счетчиков.)

2. Цифра в **цикл... раз** программируется командами вычисления количества повторений с записью результата в регистр, отведенный для счетчика.

3. Тело цикла программируется как обычно.

4. **кц** для цикла типа **раз** программируется командой "FLi", где i — номер отведенного данному циклу регистра-счетчика, с адресом начала тела цикла.

Правила иллюстрируются на примере какого-либо алгоритма, например алгоритма *степень* (он очень удобен для пошагового исполнения). Программа для ПМК записывается в столбец с параллельной записью алгоритма и отчеркиванием групп команд, соответствующих отдельным конструкциям алгоритмического языка. Пример оформления приведен в табл. 1. Еще раз напомним, что система записи — один из важных методических моментов.

Сразу отметим, что возможны вопросы вундеркиндов, связанные с тем, что программу можно значительно сократить, если разместить n там же, где счетчик, т. е. в Р0, а также накапливать результат непосредственно в РХ, без хранения в Р3. Это действительно так, и такую оптимальную программу можно получить, если использовать более сложные правила распределения регистров (с. 66, 70 [3]). Их можно рассмотреть на кружках или в математических классах, а на уроках в обычных классах ограничиться наиболее простыми правилами, заботясь о простоте понимания, а не об оптимальности программы.

Мотивацией для написания этого алгоритма на ПМК является то, что вычисление a^n с помощью операции "F x^y" дает большую погрешность (вычислите 2^3 по программе и операцией "F x^y"). Для работы с этой программой надо заслать в память значения аргументов: a — в Р1, n — в Р2; после выполнения прочитать результат из Р3 (в силу особенностей программы результат

Таблица 1

Адрес	Код	Команда	А — язык	Комментарий
счетчик — Р0 a — Р1 n — Р2 y — Р3			**алг** степень (**вещ** a, **нат** n, **вещ** y);	Сначала отводим регистр счетчику, затем всем остальным переменным — аргументам и результатам (внутренних переменных, описываемых после **нач**, в данном алгоритме нет).
			нач	
00 01	01 43	1 x — П 3	y:=1	Присваивание программируется обычно.
02 03	62 40	П — x 2 x — П 0	**цикл** n **раз**	Количество повторений цикла засылается в счетчик.
04 05 06 07	63 61 12 43	П — x 3 П — x 1 × x — П 3	y:=y·a;	Тело цикла (в данном случае состоящее из одного присваивания) программируется обычно.
08 09	5Г 04	F L 0 04	**кц**	**кц** программируется в соответствии с п. 4 правил: тело цикла начинается с адреса 04, счетчик — в Р0.
10	50	С/П	**кон**	

05

Таблица 2

Адрес	Команда	А — язык	Комментарий
		алг время прибытия (вещ S, *h*, *a*, T); арг S, *h*, *a* рез T нач вещ путь	Регистры распределены по порядку: S — P0, *h* — P1 и т. д. Здесь S — заданный путь, *h* — шаг по времени, *a* — время старта, T — время прибытия, *путь* — пройденный путь с момента *a*.
00 01 02 03	0 x—П 4 П—x 2 x—П 3	путь:=0; T:=*a*	Присваивания программируются стандартно.
04 05 06	→П—x 4 П—x 0 —	пока S<путь	Программируем цикл: вычисляем условие, приведенное к сравнению с нулем (*путь* — S<0 вместо S<*путь*),
07 08	F x<0 [ ]	нц	ставим команду условного перехода, оставляем пустую ячейку (заполним ее позже).
09 10 11 ...	П—x 4 П—x 3 F x^2 ...	путь:= путь+ $T^2 \times h$; T:=T+*h*	Для вычисления конкретного значения взяли конкретную функцию: $y=x^2$. Присваивания в теле цикла программируем обычно.
20 21	БП 04	кц 22	кц программируем в соответствии с правилами.
22	→С/П	кон	

при останове будет и на индикаторе).

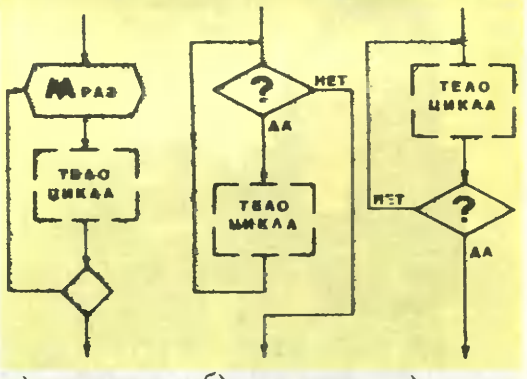
Для иллюстрации работы команд цикла рекомендуется пройти эту программу в пошаговом режиме (класс работает при этом в «командном режиме») с простыми данными, например: $a=2$, $n=3$. После команды с адресом 07 следует заглянуть в счетчик адреса ("F ПРГ", смотрим адрес, "F АВТ") и посмотреть содержимое счетчика цикла («П—x 0»); то же самое нужно сделать после выполнения команды «FL 0».

Домашним заданием может быть перепись в виде программы для ПМК одного из ранее разобранных алгоритмов, использующих цикл типа раз.

На следующем уроке можно ввести цикл пока, отталкиваясь от задачи, в которой такой цикл действительно необходим. Ключевым отличием этого вида цикла от цикла раз является то, что количество повторений к моменту начала цикла неизвестно и определяется только в процессе его выполнения. Пример — небольшая модификация задачи о площади криволинейной трапеции: при заданной функции, значениях *a* и *h*, определить *b*, при котором площадь достигает заданной величины S. Алгоритм вместе с программой приведен в табл. 2.

На схемах цикл пока изображается так, как показано на рис. 1 б. Из схемы нетрудно усмотреть правила его программирования.

1. Программируется условие, приведенное к сравнению с нулем.
2. Пишется соответствующая команда условного перехода, пропускается ячейка под адрес.
3. Программируется тело цикла.
4. кц для цикла пока программируется командой «БП» с адресом начала проверки



а) цикл раз б) цикл пока в) цикл до

2

условия, затем пропущенная на шаге 2 ячейка заполняется очередным свободным адресом.

Отметим, что в этой программе при останове на индикаторе будет не результат. Его следует искать в Р3. С программой можно провести ряд интересных машинных экспериментов, подробности которых описаны на с. 55, 145—146 в [3] и в [8].

Домашним заданием может служить составление программы для одного из ранее разобранных алгоритмов с циклом **пока**. Следующий урок можно провести как лабораторную работу, на которой ученики должны самостоятельно отладить программу домашней задачи и предъявить учителю ПМК с программой, выдающей правильные результаты.

На последних уроках по этой теме целесообразно ввести еще одну конструкцию цикла: цикл **до**, проверяющий условие окончания после тела цикла (см. рис. 1 в), который неявно применен в [1].

3

нц
тело цикла
до условие **кц**

Правила ее программирования легко выводятся из схемы:

1. Программируется тело цикла.
2. Программируется условие, приведенное к сравнению с нулем.
3. Ставится нужная команда условного перехода с адресом начала тела цикла.

Опыт показывает, что гораздо легче объяснить дополнительную конструкцию, чем то, как свести ее к уже имеющимся. Сравните тело алгоритма «уточнение корня»: (с. 95 [1]).

```
вещ c;
пока b–a>2·ε
нц c:=(a+b)/2;
  если f(a)·f(c)<0
    то b:=c
    иначе a:=c
  все
кц
```

```
x:=(a+b)/2;
нц x:=(a+b)/2;
  если f(a)·f(x)<0
    то b:=x
    иначе a:=x
  все
до b–a≤ε;
```

В варианте с циклом **пока** дополнительное присваивание нужно именно потому, что тело цикла **пока** может не выполниться ни разу, и тогда x не получит вообще никакого значения.

Технические подробности, примеры и машинные эксперименты с программой с циклом **до** описаны на с. 55—60 в [3].

В заключение отметим, что кабинет информатики, оснащенный программируемыми микрокалькуляторами, желательно оснастить также и плакатами, иллюстрирующими правила программирования конструкций и связь алгоритмов, схем и программ.

Продолжение следует.

Литература

1. Основы информатики и вычислительной техники: Пробное учебное пособие Ч. I, II / Под ред. А. П. Ершова, В. М. Монахова. М.: Просвещение, 1985.

2. Основы информатики и вычислительной техники / Под ред. А. П. Ершова. М.: Просвещение, 1988.

3. *Штернберг Л. Ф.* Программирование на микрокалькуляторе. М.: Просвещение, 1988.

4. *Звенигородский Г. А.* Первые уроки программирования. М.: Наука, 1985.

5. *Гутман Г. Н., Карпилова О. М.* Азбука программирования // Информатика и образование. 1987. № 6.

6. *Шень А.* Информатика в 9 классе // Информатика и образование. 1987. № 6.

7. *Штернберг Л. Ф.* Зачем микрокалькулятору стек? // Квант. 1986. № 4.

8. *Штернберг Л. Ф.* Умеет ли считать ваш микрокалькулятор? // Информатика и образование. 1988. № 5.

Е. КАПУСТИНА

«Творческая лаборатория» на уроках информатики

Эта статья рассчитана прежде всего на учителя. Для того чтобы правильно расставить акценты в учебном материале, он должен знать ответ на вопросы: чему учить на уроках информатики, каковы должны быть результаты обучения?

Считаю, что нужно стремиться к тому, чтобы каждый человек знал, где и как в работе ему сможет помочь компьютер, чтобы каждый смог построить алгоритм своей деятельности и продумать, какие ее элементы могут быть автоматизированы.

Средство реализации этих целей — методика обучения, которую учитель создает для себя сам. Мне кажется, на уроках необходимо решать содержательные задачи практического характера, которые встают перед биологом или бухгалтером, работником спорткомитета или врачом.

Занятия будут эффективнее, если при решении конкретных задач иметь результаты работы программы для различных начальных данных, а на уроке раздать ученикам эти распечатки. Было бы желание и энергия — учитель самой отдаленной сельской школы может получить распечатки в ближайшем компьютерном центре.

Ниже приводятся несколько таких задач. Они завершают темы курса и позволяют проконтролировать знания учащихся. К некоторым задачам даются решения.

Линейные алгоритмы

Задача 1. В магазине стройматериалов продаются прямоугольные плиты разного размера из различных материалов. Известна стоимость 1 дм^2 материала. Написать программу, определяющую стоимость одной плиты.

Решение

1. Постановка задачи

Математическая запись условия: найти площадь прямоугольника с известными сторонами. Аргументы и результаты: A, B — длина и ширина плиты в дм, вещ, арг; K — стоимость 1 дм^2 в руб., вещ, арг; P — стоимость плиты в руб., вещ, рез. Математическая модель:

$P = K \cdot B \cdot A$

2. Построение алгоритма

```
алг плита (вещ A, B, K, P)
    арг A, B, K, P
    рез P
нач P:=K·A·B
кон
```

3. Программа на Бейсике

```
10 PRINT «ПЛИТА»
20 INPUT «введите длину и ширину плиты»; A, B
30 INPUT «введите стоимость 1 кв. дм плиты»; K
40 LET P=A*B*K
50 PRINT «стоимость плиты»; K
60 END
```

4. Получение результатов для начальных данных: A=5.32; B=6.71; C=1.34; P=47.833148

5. Анализ результатов

Результаты выводятся в неудобной форме — много знаков после запятой. Достаточно два знака. Вносим коррективы в алгоритм и программу.

P:= ЦЕЛЧ (K·A·B·100))):100+0.1

ЦЕЛЧ — функция вычисления целой части от числа

0 01 — для того чтобы магазин не остался «внакладе», округляем каждый раз стоимость в меньшую сторону.

Коррективы на Бейсике:

P= (INT (K*A*B*100)/100+0.01

Задача 2. Написать программу, которая по старому и новому показаниям счетчика определяет стоимость израсходованной электроэнергии.

Задача 3. Написать программу, вычисляющую стоимость масла в бочке цилиндрической формы, если известна стоимость одного кг масла.

Задача «Начисление зарплаты». Этой задачей в X классе завершается тема «Команда выбора». Пусть рабочий заработал x рублей по сдельной оплате. При условии, что план выполнен, была начислена премия 20 % от x (сумму заработка и премии обозначим через P). После этого добавили 15 % от P — коэффициент. Общая сумма начисления — N. Затем вычли подоходный налог исходя из табл. 1:

Таблица 1

Начисления, руб.	70	70—100	100—150	150
Налог с начисления, %	0	7	10	13

Результат после вычисления налога обозначим K. Если у рабочего нет детей, вычитается 6 % от N — налог за бездетность. И наконец, вычитается 1,5 % от N — профсоюзные взносы.

Написать программу (алгоритм), которая бы «сообщила» кассиру, сколько денег он должен выдать рабочему.

Решение

1. Математическую модель этой задачи можно не писать, так как основные формулы — вычисление % от заданного числа достаточно просты. Аргументы и результаты: аргументы — вещественная переменная X и целая переменная D, которая принимает значение равное 0, если у рабочего нет детей. Результат Z — зарплата в рублях, которую кассир должен выдать рабочему.

2. Построение алгоритма.

```
алг зарплата (вещ X, Z, цел D)
  арг X, D
  рез Z
нач вещ P, N, K
  P:=X+0,2·X
  N:=P+0,15·P
    выбор
        при N<70:K:=N
        при N<100 и N⩾70:K:=N—
            —0,07·N
        при N⩾100 и N<150:K:=N—
            —0,1·N
        при N⩾150:K:= N—0,13·N
    все
    если D=∅
        то K:=K—0,06·N
    все
    Z:=K—0,15·N
кон
```


Табличные величины

Задача 1. В таблице хранятся данные о количестве макулатуры, собранной каждым учеником школы. Написать алгоритм, определяющий сколько всего макулатуры собрано.

Задача 2. В таблице A [1:315] хранятся данные о результатах спортсменов, участвовавших в кроссе на 1000 м. Найти номер лучшего бегуна.

Задача 3. Данные о 895 учениках хранятся в таблице P [1:895]. Они закодированы следующим образом: если ученик октябренок — код 1, пионер — код 2, комсомолец — код 3, относится к бессоюзной молодежи — код 0. Сколько в школе комсомольцев и сколько октябрят?

Задача 4. В таблице K [1:125] находятся данные о количестве конфет (*кг*) каждого из 25 сортов, хранящихся на базе. В магазины было отгружено по 2000 *кг* конфет каждого сорта. Написать алгоритм, который вносит соответствующие изменения в таблицу.

Задача 5. В таблице A [1:46] хранятся данные о количестве заготовленных кормов каждым из 36 колхозов района. Найти, какой % составляют колхозы, заготовившие меньше 20 000 *т* кормов. Найти общее количество заготовленных в области кормов.

Биологический эксперимент

Задача. Пусть требуется провести исследование популяции желтых и зеленых бактерий, их влияния друг на друга. Бактерии развиваются в растворе с концентрацией питательных веществ, равной K. Известно что все бактерии гибнут, если концентрация меньше 5 % или больше 40 %. Известно также, что каждую секунду среди зеленых бактерий 2K % делится, а (K+1) % погибает; среди желтых бактерий (2K+2) % делится, а (K+5) % погибает. Если бактерий одного вида становится в 3 раза больше, чем бактерий другого вида, подавленный вид погибает полностью. Так как бактерии и раствор, в котором они развиваются, очень дефицитны, то требуется вначале составить программу — модель, с помощью которой выяснить, какие начальные условия (начальное количество бактерий двух видов и концентрацию раствора) необходимо задать, чтобы популяция просуществовала как можно дольше. Суммарное количество бактерий двух видов не должно быть больше 50 штук.

После того как программа отлажена, можно предложить ученикам ответить на вопрос, например: какова зависимость времени жизни популяции от начального количества желтых бактерий при фиксированных концентрации и начальном количестве зеленых? Для этого необходимо получить результаты работы программы для различных начальных данных.

Решение

1. Постановка задачи

Аргументы и результаты: аргументы — целые переменные Q, Z — начальные количества желтых и зеленых бактерий соответственно, вещественная переменная K — концентрация раствора в %. Результат — вещественная переменная T — время существования популяции. Математическая модель изменения количества зеленых и количества желтых бактерий в зависимости от концентрации раствора:

$$Z=Z\cdot(1+(K-1):100)$$
$$Q=Q\cdot(1+(K-3):100)$$

2. Построение алгоритма

```
алг био (цел Q, Z, вещ K, T)
  арг Q, Z, K
  рез T
нач
```

```
    T:=0
    пока Z:3<Q и Q:3<Z
    нц
      Z:=Z·(1+(K—1):100)
      Q:=Q·(1+(K—3):100)
      T:=T+1
    кц
кон
```

Уроки «Творческая лаборатория»

Постоянно поддерживать интерес к предмету позволяют разнообразные формы проведения уроков. Считаю, что одна из наиболее эффективных форм — «творческая лаборатория», вариант бригадного (группового) метода проведения уроков информатики.

Творческая — потому что призвана учить коллективному творчеству, лаборатория — потому что моделирует отношения, складывающиеся в коллективе заводской или институтской лаборатории.

В школе мы уделяем мало внимания коллективному творчеству. Чаще всего ученик либо полностью сам решает задачу (активный вариант), либо ждет пока ее решат другие (пассивный вариант). К сожалению, пассивных учеников достаточно много. Форма «Творческая лаборатория» не позволяет ученику быть пассивным, так как от его труда зависит результат работы всей группы.

Для проведения такого занятия класс делится на несколько групп по 4—6 человек. Подбор групп и старшего в ней — очень важная задача. Если группа подобрана из средних учеников, то на уроке развиваются все ребята. В группе проходит коллективное обсуждение задач, вклад каждого ученика примерно одинаков. Но если в группе есть сильный ученик, средний и несколько слабых, то отношения складываются иначе. Конечно, основные идеи решения подает сильный ученик, но если он тактично, с желанием помочь относится к своим товарищам и в группе есть взаимное уважение, то слабые проявляют максимальную активность в усвоении материала, чтобы не подвести группу, если им придется рассказывать решение задачи у доски.

Таким образом, у слабых ликвидируются пробелы в знаниях, возникает уверенность в своих силах. Есть примеры, когда троечники после таких уроков проявляли большее желание учиться, и в конце полугодия у них выходила твердая «4». Лидер в такой группе развивает в себе умение объяснять, просто излагать свои мысли, приобретает организаторские навыки.

Если же лидер проявляет пренебрежение к другим участникам, то из группы не складывается коллектива, эффект от уроков сводится на нет. Поэтому учитель должен обязательно провести перед уроком работу с будущими старшими групп, тщательно продумав их назначение. Каждой группе дается свое задание — одна или несколько задач. Старший группы должен распределить задачи и организовать работу так, чтобы за урок группа успела выполнить все задания.

На уроке, спаренном с предыдущим, каждая группа делает отчет о своих задачах. Решения пишутся на пленке и проецируются на доску. Защищать работу группы может любой ее участник. От того, как он расскажет решение задачи, зависит фонд оценок всей группы. Размеры фонда определяет класс. Учитель после проверки задач анализирует работу и выдает каждой группе соответствующий фонд оценок. Каждый ученик получает оценку из этого фонда, утверждает ее — маленький трудовой коллектив. Например, группа из 5 человек получает фонд 20 баллов, она может всем утвердить оценки «4», а может двум активным поставить «5», а двум более пассивным — «3».

Ниже приведен пример урока «Творческая лаборатория», который проводится в X классе после изучения темы «Команда повторения с параметром».

Вводная часть

Мы уже знаем, что информатика изучает методы сбора, хранения и переработки информации. Познакомимся с этими методами на примере создания простейших баз данных для специалистов разных профессий.

Простейшая база данных — это совокупность линейных таблиц, находящихся в памяти ЭВМ. Каждая таблица содержит информацию определенного типа. Например, база данных 1-го цеха для отдела кадров может в одной таблице содержать фамилии всех работников цеха, в другой таблице — информацию о возрасте работников, в третьей таблице — информацию о средней зарплате и т. д.

1. **Сбор информации.** Это первое, что необходимо сделать при подготовке базы данных. Для этого приходится собирать необходимые документы, делать подборку соответствующей литературы или проводить социологический опрос. Мы воспользуемся методом анкетирования.

2. **Хранение информации.** Для хранения будем использовать линейные таблицы. Для того чтобы обрабатывать информацию было удобно, закодируем ее в числовой форме и лишь потом поместим в таблицу.

3. **Обработка информации.** Информация, помещенная в таблице, будет обработана с помощью компьютера. Для этого необхо-

димо написать соответствующие алгоритмы и программы.

Конечно, настоящие базы данных сложнее, чем те, которые мы будем составлять. На уроках «Творческая лаборатория» вы получите возможность применить свои знания по темам «Табличные величины», «Команда выбора», «Команда повторения с параметром», «Этапы решения задач с помощью ЭВМ».

Организация уроков

Для проведения уроков класс делится на 7—8 групп по 4—5 человек. Каждая готовит простейшую базу данных для специалистов. Это могут быть работники спорткомитета, медики, работники школьного буфета, демографы, работники горкома ВЛКСМ и т. д.

Цель первого урока — подготовка базы данных. Учитель объясняет, как выполнять задания, показывает оформление решения на примере задачи для медиков, формирует группы и назначает старших, распределяет задания по группам.

Каждой группе выдается карточка с заданием. Группа должна с помощью анкет собрать нужную информацию, закодировать ее в числовой форме и поместить в линейную таблицу. Количество элементов таблицы равно числу учащихся данного класса. Номер элемента таблицы — номер ученика по школьному журналу. Готовая таблица и будет являться простейшей базой данных (см. карточку № 1). Старший должен обеспечить готовность своей группы к следующему уроку.

Второй урок — использование базы данных. Когда база данных создана, с ее помощью можно решать много различных задач. У нас база данных очень проста. Каждой группе будет предложено две задачи (см. карточку № 2). Старший распределяет их внутри группы. В конце урока группа должна предоставить отчеты по каждой из задач. В начале третьего урока проводится анализ «Творческой лаборатории».

Медики

Карточка № 1

Собрать информацию о зрении учащихся. Нормальное, —1 диоптрий, —1,5 диоптрий и т. д. Закодировать и поместить в таблицу.

Решение

1. **Сбор информации. Составление анкеты**

Фамилия	Зрение
Алферов	—3,5
Аникина	норм.
Баженов	норм.
Ветрова	—1
Герасимов	норм.

2. **Кодирование:** нормальное зрение — код 0, остальная информация не требует кодирования, так как уже представлена в числовой форме.

3. **Заполнение таблицы Z**

Таблица Z

1	—3,5	Алферов
2	0	Аникина
3	0	Баженов
4	—1	Ветрова
5	0	Герасимов
....	.....	.....

Карточка № 2

1. Какой процент учащихся обладает нормальным зрением?

2. Сколько человек имеют близорукость слабой степени (от 0 до 3 диоптрий)?

Решение задачи 2:

1. Постановка задачи

В линейной таблице Z [1:33] найти количество элементов, значения которых находятся в интервале от 0 до 3. Аргументы и результаты:

аргумент: — линейная вещественная таблица Z [1:33];

результат: — целая переменная M — количество элементов, удовлетворяющих условию задачи.

2. Построение алгоритма

```
алг зрение (вещ таб Z [1:33], цел M)
    арг Z
    рез M
нач цел k
    M:=0
    для k от 1 до 33
    нц
      если Z [k]<0 и  Z [k]>—3
          то M:=M+1
      все
    кц
кон
```

3. Программа на Бейсике

```
5   DIM Z (33)
10  PRINT «ЗРЕНИЕ»
20  FOR I=1 TO 33
30  INPUT Z (I)
40  NEXT I
50  LET M=0
60  FOR K=1 TO 33
70  IF Z (K)<0 THEN IF Z (K)>(—3) THEN
                              LET M=M+1
80  NEXT K
90  PRINT «БЛИЗОРУКОСТЬ СЛАБОЙ СТЕ-
                    ПЕНИ ИМЕЕТ»; M; «ЧЕЛ»
100 END
```

С. ФАДЕЕВ
Ленинградский политехнический институт им. М. И. Калинина

Компьютер, цвет, язык

Осуществленный нами анализ компьютерных программ по русскому языку, реализованных на машинах с цветными дисплеями, позволил установить, что цвет в программах довольно часто используется произвольно и в значительной степени зависит от вкуса разработчика программы.

Как известно, первым, кто сделал попытку проанализировать эмоциональное воздействие цвета на человека, был И.-В. Гёте.

«В своих самых общих элементарных проявлениях, независимо от структуры и формы материала, на поверхности которого мы его наблюдаем, цвет оказывает воздействие на чувство зрения, которому он главным образом и предназначен, а через это посредство и на душу»*,— писал И.-В. Гёте в своем «Учении о цвете» (1, с. 103). Многочисленные эксперименты психологов и физиологов доказали, что любой цвет может определенным образом влиять на человека. Но лишь незначительный процент разработчиков компьютерных программ учитывает медицинские данные о цвете.

Создавая компьютерную программу, они уделяют основное внимание ее предметному наполнению и вопросам методической организации предметного материала, считая при этом вопрос о цвете второстепенным.

Однако практика показывает, что в компьютерном обучении, находящемся на стыке многих наук, нет и не может быть второстепенных вопросов.

В работах профессора А. Журавлева [3; 4] приводятся любопытнейшие сведения о звукоцветовых соответствиях. Проведенные им эксперименты с тысячами информаторов показали, что все гласные звуки и буквы в нашем подсознании, как правило, связаны с вполне определенным цветом [3, с. 128].

По мнению А. Журавлева, «цветовое устройство мира отразилось в цветовом устройстве языка. Названия главных цветов встречаются в речи наиболее часто, и звуки О, А, Е, И наиболее частотны из гласных» [3, с. 126—127].

Заинтересовавшись проблемами повышения эффективности компьютерных программ, реализованных на цветных дисплеях, мы пришли к выводу, что уникальные данные А. Журавлева, до сих пор не имевшие практического выхода, могут оказать существенную пользу в компьютерном обучении русскому языку.

Опираясь на цветовую семантику русских гласных, мы разработали несколько типов специальных компьютерных программ по русскому языку для цветных дисплеев.

Известно, что правописание русских безударных гласных представляет определенные трудности как для советских школьников, так и для иностранцев, начинающих изучать русский язык. Для снятия этих трудностей мы предлагаем компьютерные задания типа: «Вставьте пропущенные буквы: ГОР...Д». Подобные задания, конечно, довольно легко реализуются и на монохромных дисплеях. Однако лишь на цветных дисплеях появляется возможность для более успешного решения поставленной задачи использовать цветовую семантику русских гласных. В отличие от монохромного дисплея на цветном место пропущенной буквы окрашивается в соответствующий ей цвет.

Большинство учащихся, как правило, уже на первом же занятии по таким программам самостоятельно устанавливают зависимость между цветом и буквой. А после нескольких занятий эта зависимость перерастает в прочную связь, т. е. цветовое сознание школьников оказывается тесно связано с правописанием. У иностранцев же формирование навыков правописания идет параллельно с формированием русского «буквенно-цветового сознания».

Если же обучаемый все-таки не может самостоятельно установить буквенные и цветовые соответствия, то целесообразно использовать несколько иной вид этого задания, когда окрашиваются в соответствующий цвет не только пропущенные безударные гласные, но и все другие такие же гласные, имеющиеся в данном слове. Например, в слове МОЛ...КО будут окрашены в желтый цвет как пропущенная буква (точнее — место, где она находится), так и все имеющиеся в слове буквы О.

Если же и этот вид подсказки не помогает (что случается крайне редко), то в комментариях к неправильным ответам обучаемого компьютер сообщает о соответствии цвета определенной букве.

Используя метод цветовых пропусков в компьютерных программах, следует всегда

* Перевод с немецкого мой.— *С. Ф.*

помнить и об общем цветовом фоне, на котором находится слово, чтобы не получилось так, что цветовой пропуск сливается с общим фоном экрана. Практика показывает, что наилучший эффект достигается при использовании в качестве фона дополнительных к цветовым пропускам цветов.

К основным дополнительным цветам относятся, как известно, три пары: красный — зеленый, желтый — фиолетовый, синий — оранжевый. Использование взаимодополнительных цветов в качестве общего фона основано на эффекте одновременного цветового контраста, который заключается в том, что при рассматривании какой-либо окрашенной детали на взаимодополнительном фоне она воспринимается более насыщенной по цвету, чем на фоне любого другого цвета. Поэтому, например, красные буквы выглядят более насыщенными на зеленом фоне, а зеленые, наоборот, на красном.

Таким образом, компьютерные задания типа «Вставьте пропущенные буквы» с цветовым пропуском вместо букв следует оформлять на соответствующем взаимодополнительном к цветовому пропуску фоне.

Русский язык относится, как известно, к языкам флективного типа, т. е. большинство слов русского языка имеют соответствующие окончания. Окончание или его основу составляют обычно гласные буквы. Причем, как правило, это так называемые основные, опорные гласные А, О, Е, которым соответствуют и главные цвета: красный, желтый, зеленый.

При создании компьютерных программ на цветных дисплеях мы пошли несколько дальше, предположив, что отмеченная А. Журавлевым связь гласных с тем или иным цветом гораздо глубже, что она проявляется также и в сфере грамматики. В самом деле, традиционно мы определяем принадлежность, скажем, имен существительных к одному из трех грамматических родов по окончаниям именительного падежа единственного числа. Но ведь эти окончания основаны на гласных! В этой связи встает вопрос: а не является ли цвет конечной гласной указанием и на окраску грамматического рода?

Даже беглого взгляда на приведенную выше таблицу достаточно, чтобы увидеть, что женский род «окрашен» в красный цвет, средний — либо в желтый (твердый вариант), либо в зеленый (мягкий вариант), мужской же род чаще всего не имеет цветовой маркировки (точно так же, как и не имеет в именительном падеже единственного числа и флексий). Согласные, по данным А. Журавлева, в нашем подсознании ни с каким определенным цветом не связаны. Лишь существительные мужского рода, оканчивающиеся

«Соотношение грамматических родов и цветов»

Род	Окончание в им. п. ед. ч.	Цвет
мужской	— — — —	— — — —
	-й	синий
женский	*-а/-я*	красный
средний	*-о*	желтый
	-е	зеленый
	-ый	черный/синий; темно-коричневый/синий
мужской	*-ий*	синий
	-ой	желтый/синий
	-ая	темно-красный/светло-красный
женский	*-яя*	светло-красный
	-ое	желтый/зеленый
средний	*-ее*	зеленый
множественное число	*-ые*	черный/зеленый
	-ие	синий/зеленый

на *-й*, имеют синеватую окраску (влияние сходства графической формы *й* с *и*).

За рамками таблицы, конечно, остались многочисленные исключения, например: м. р. на *-а/-я (папа, дядя)*, ср. р. на *-мя (время)* и т. д.

Как же быть с ними? Конечно, можно было бы, как и в традиционной методике, сохранить их статус исключений, но слишком большое количество исключений, как правило, очень раздражает учащихся (особенно иностранных), которые всегда стараются найти в языке универсальное и понятное правило. Думается, что в данном случае роль такого универсального правила может сыграть цвет, т. е. все существительные-исключения следует целиком окрашивать на экране дисплея в соответствующий родовой цвет. При этом цвет будет сигнализировать учащимся о принадлежности к тому или иному грамматическому роду, а одинаковая окраска всего слова — о статусе исключения. Существительные мужского рода без окончаний также целесообразно полностью окрасить в синий цвет.

При объяснении иностранным учащимся темы «Категория рода» компьютерная программа обращает их внимание на то, что в русском языке определенному роду соответствует вполне конкретный цвет (*он* — синий, *она* — красная, *оно* — желтое или зеленое). Затем учащимся предлагаются задания: «Определите по цвету род существительного» или более сложное: «Определите по цвету конечные буквы и род существительных» (дается список слов с цветовыми пропусками вместо окончаний).

Существенную помощь окажут цветовые компьютерные программы иностранцам и при изучении имен прилагательных:

При работе с прилагательными компьютер предлагает задания: «Определите по цвету окончания и род прилагательных».

Аналогичная работа проводится и при изучении местоимений: *мой* (желтый — синий), *моя* (желтый — красный), *моё* (желтый — желто-зеленый, *наш* (синий), *наша* (красный), *наше* (зеленый) и т. д.

Когда же приходит время познакомить учащихся с правилами согласования имен прилагательных с существительными, то с именительным падежом единственного числа у иностранцев не возникает никаких трудностей, так как в действие вступает простое правило: каждый цвет сочетается только с тем же цветом. Прилагательные, имеющие окончания «красного цвета», сочетаются с существительными, обладающими окончаниями «того же красного цвета». Прилагательные с «синими» окончаниями сочетаются либо только с «синими» существительными, либо с существительными, имеющими «синие» окончания. И наконец, нетрудно запомнить и то, что прилагательные, имеющие «зеленые» или «желто-зеленые» окончания, сочетаются с существительными, у которых окончания аналогичного цвета.

Закреплению этого правила способствуют компьютерные задания типа «Образуйте словосочетания из данных слов» (приводится список существительных вперемежку с прилагательными, все окончания соответственно окрашены).

С успехом можно использовать цветовую семантику гласных и при изучении спряжения русских глаголов. В состав глаголов в окончаниях I спряжения входят, как известно, либо гласные *-у/-ю (пишу, пишут; работаю, работают)*, либо *-е- (работаешь, -ет, -ем, -ете)*. Глаголы же II спряжения имеют в составе своих окончаний либо *-у/-ю (сижу, строю)*, либо «синее» *-и- (строишь, -ит, -им, -ите)*, либо «красное» *-я- (строят, сидят)*.

Таким образом, в компьютерном обучении при первичном знакомстве с образцами глаголов I и II спряжений все гласные в окончаниях личных форм глаголов окрашены в соответствующий их семантике цвет. Учащиеся, ранее занимавшиеся по компьютерным программам с цветовой маркировкой рода, очень быстро устанавливают закономерности новых соотношений цветов и букв в личных окончаниях глаголов и без каких-либо дополнительных объяснений компьютера или преподавателя по преобладающему в личных окончаниях глаголов цвету присваивают I спряжению определение «зеленое», а II спряжению — «синее».

С целью закрепить это новое «открытие» учащихся компьютер предлагает им довольно простые программы, например: «Напишите в левой колонке таблицы личные формы глаголов I спряжения, а в правой — II спряжения». При этом дается список личных форм глаголов с маркированными соответствующим цветом гласными в окончаниях. Левая часть таблицы окрашена в зеленый цвет, правая — в синий. В более сложных заданиях таблица цветом не маркирована.

Широко используются также программы на заполнение цветовых пропусков в личных окончаниях глаголов (как в изолированном положении глаголов, так и в контексте). Причем программы решены, как правило, в игровом стиле.

Например, в программе «Цветы» каждый правильный ответ учащегося способствует появлению на экране дисплея какого-либо цветка. Тем самым учащийся, правильно выполнивший всю программу, получает в награду удивительно красивый компьютерный букет. Учащемуся же, который допустил слишком много ошибок, приходится довольствоваться лишь одним-двумя цветками (в зависимости от числа правильных ответов).

Таким образом, мы показали, как можно использовать цветовую семантику русских букв в компьютерном обучении для автоматизации навыков русского правописания и изучения отдельных грамматических тем (род, спряжение глаголов).

В заключение хотим обратить внимание на то, что в подобных программах сознательно используется метод повышенного цветового контраста, который призван максимально воздействовать на зрительные анализаторы с целью пробуждения «буквенно-цветового сознания» у наших школьников и его формирования у иностранцев, изучающих русский язык. Однако не следует забывать, что при этом происходит и одновременная интенсивная загрузка зрительных анализаторов обучаемого, повышается их утомляемость. Поэтому использование программ, реализованных по данной методике, не должно превышать 10—15 мин в течение урока. Это связано с санитарно-гигиеническими требованиями.

Литература

1. *Goete*. Farbenlehre. Goethes Werke in zwölf Bänden. Zwölfter Band. Berlin. Weimar. 1981.

2. *Дерибере М.* Цвет в деятельности человека. М.: Госстройиздат, 1964.

3. *Журавлев А. П.* Диалог с компьютером. М.: Молодая гвардия, 1987.

4. *Журавлев А. П.* Фонетическое значение. Л.: Изд-во Ленингр. ун-та, 1974.

М. БУЛОВАЦКИЙ
г. Могилев

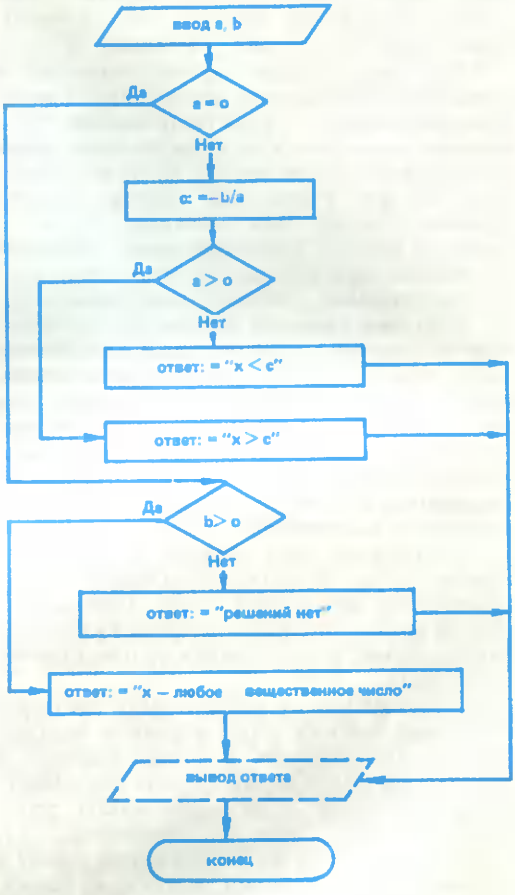
Из практического опыта

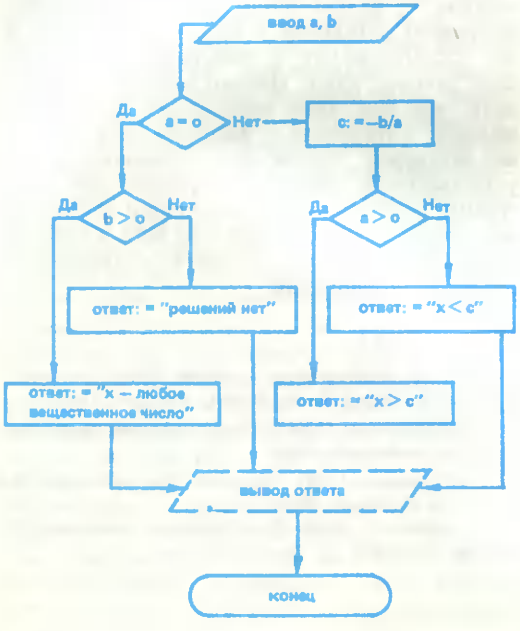
При переходе от алгоритмического языка к языку программирования (например, Бейсику) одним из камней преткновения, как показывает практика, является команда ветвления. Трудность заключается в том, что в алгоритме после указания условия следует серия команд «да», а в соответствующей программе серия «нет». Облегчить этот переход могут помочь блок-схемы. Представление алгоритма в форме блок-схемы в IX классе должно быть хорошо отработано. Такое положение дел естественно, если блок-схемам алгоритмов уделялось внимание на уроках математики начиная с IV класса (с несомненной пользой и для математики).

Составление программы с ветвлением можно начинать теперь с построения блок-схемы. Рассмотрим для примера задачу решения неравенства $ax+b>0$. Ее блок-схема показана на рис. 1.

Блок «вывод ответа» выделен пунктиром потому, что во многих случаях (и здесь тоже) вывод ответа удобнее совместить с получением ответа. О нумерации блоков будет сказано ниже.

Теперь эту блок-схему надо «выпрямить», сохраняя все связи между блоками. При этом приоритет каждый раз отдается ветви





«нет» (т. е. эта ветвь выписывается раньше) (рис. 2).

Слева — стрелки условного перехода по ветви «да» (оператор IF), справа — стрелки безусловного перехода (оператор GO TO).

После этого осуществляется «прямой» перевод содержимого блоков на Бейсик:

```
INPUT A, B
IF A=0 THEN □
C=—B/A
IF A>0 THEN □
PRINT «X<»; C
PRINT «X>»; C
IF B>0 THEN □
PRINT «РЕШЕНИЙ НЕТ»
PRINT «X — ЛЮБОЕ ВЕЩЕСТВЕННОЕ ЧИСЛО»
END
```

Это еще не программа, а лишь каркас. Теперь нужно после каждого оператора

PRINT (кроме последнего) вставить операторы GO TO □, уточнить типы задействованных переменных, дать программе имя, пронумеровать строки, заполнить окошки в соответствии со стрелками блок-схемы и «навести косметику». Окончательно получим:

```
10 PRINT «ПРОГРАММА LINER РЕШАЕТ ЛИНЕЙНЫЕ НЕРАВЕНСТВА ВИДА A*X+B>0»
20 PRINT «ВВЕДИТЕ КОЭФФИЦИЕНТЫ A, B»
30 INPUT A, B
40 IF A=0 THEN 110
50 C=-B/A
60 IF A>0 THEN 90
70 PRINT «X<»; C
80 GO TO 150
90 PRINT «X>»; C
100 GO TO 150
110 IF B>0 THEN 140
120 PRINT «РЕШЕНИЙ НЕТ»
130 GO TO 150
140 PRINT «X — ЛЮБОЕ ВЕЩЕСТВЕННОЕ ЧИСЛО»
150 END
```

После этого можно обсудить варианты более компактной записи программы. Например:

```
60 IF A>0 THEN PRINT «X>»; C\GOTO 150
...
```

И т. д.

Следует заметить, что непосредственное «выпрямление» блок-схемы необходимо лишь на начальном этапе — для осмысления этого процесса. В дальнейшем его можно заменять, например, нумерацией блоков, идущей по порядку якобы «выпрямленной» блок-схемы. Это несколько упрощает работу. Большинство моих учеников так и делали. Однако некоторые из них посчитали более удобным сразу составлять «выпрямленные» блок-схемы.

Обучать применению знаний всегда труднее, чем обучать самим знаниям. Информатика не исключение. И здесь перевод знаний в умения и навыки представляет серьезную проблему для учителей. Облегчить этот процесс можно специальным подбором задач, привлекающих интерес учащихся и обладающих оптимальным уровнем сложности для каждого в отдельности ученика.

Интерес ученика вызывают те задачи, за условиями которых он видит реальное и знакомое ему явление. Например, решение задачи подсчета количества положительных чисел в некоторой числовой последовательности, несомненно, полезно для ученика, но решать ее он будет лишь по своей ученической обязанности. Совсем другое отношение будет у него, например, к такой задаче: составить программу, выписывающую билет в кинотеатр «Юпитер» (с указанием ряда, места, цены и времени начала сеанса), если на вечерние сеансы (начало в 16.00 и позже) билет на первые 4 ряда стоит 25 коп., с пятого по пятнадцатый ряд — 35 коп., с шестнадцатого по двадцать пятый ряд — 45 коп., а на дневные сеансы стоимость всех билетов 25 коп.

Еще один маленький нюанс. Ученик охотнее составляет программу к той задаче, ответа к которой у него нет. Одно дело — выбрать наибольшее из ста заданных чисел, другое — выбрать наибольший из синусов ста заданных чисел. В первом случае ученик может указать искомое число и без помощи машины, тем самым работа над составлением программы теряет смысл и становится неинтересной. В другом случае получение ответа является вознаграждением за труд.

Задача оптимального уровня сложности — это такая задача, решение которой ученику посильно, но и не настолько простая, чтобы ее решение могло быть найдено без труда. Проблема здесь в том, что ученики в классе разные, у каждого свой оптимальный уровень. Но учителю вовсе и не надо заниматься определением этих уровней; эта работа бесконечна, потому что достигнутый учеником уровень освоения знаний не является чем-то постоянным, он меняется. Пусть эту работу выполняет сам ученик. Учитель должен предоставить ему возможность выбора из нескольких задач различных уровней сложности одной — «своей». Можно при этом ограничиться тремя уровнями соответственно трем положительным отметкам действующей системы оценивания знаний.

Вот вариант задания для отработки команды ветвления или выбора в алгоритмическом языке.

Написать и исполнить на таблице в трех различных ситуациях алгоритм:

1) определения стоимости железнодорожного билета от Москвы до Могилева, если билет в общий вагон стоит 8 рублей, в плацкартный — 10 рублей, в купейный — 12 рублей;

2) решения уравнения $a{:}x=|a|$;

3) определения вида треугольника по данным длинам его сторон.

Первая задача не требует никаких знаний, кроме элементарных сведений о записи и исполнении алгоритма. Все возможные ситуации в условии четко просматриваются. Это минимальный уровень, на который должен выйти ученик. Вторая задача требует от него самостоятельно разобраться в возникающих ситуациях ($a<0$, $a=0$, $a>0$); здесь нужен некоторый навык решения уравнений с параметрами. Это усложняет процедуру алгоритмизации, хотя и не очень существенно. Третья же задача, на

вид совсем простая, содержит немало тонкостей. Там возникает неожиданно много условий, к тому же составных, с использованием логических союзов И, ИЛИ. Разобраться во всем этом и алгоритмизировать процедуру распознавания вида треугольника (равносторонний, равнобедренный или разносторонний) под силу лишь ученику, хорошо мыслящему. Поэтому задача отнесена к высшему уровню сложности.

Каждый ученик, получив такой вариант на самостоятельной работе, должен выполнить одно из трех предполагаемых заданий — по своему выбору, зная, что это будет отражено соответствующей оценкой. В случае успешного выполнения задания на следующей самостоятельной работе он получит аналогичный вариант заданий по следующей теме (команда повторения или команда повторения с параметром). Неудачно выполненная работа приводит к тому, что в следующий раз он получит новый вариант из трех заданий по несданной теме (пробелов оставлять нельзя). И так — по всем важнейшим темам.

Интересной разновидностью такого рода заданий являются продолженные задачи, т. е. такие задания, в которых вторая и третья задачи являются продолжением (дополнением) первой.

1) Дана последовательность из ста действительных чисел. Написать программу вычисления суммы синусов неотрицательных ее членов 2) с подсчетом количества 3) и (или) указанием номеров использованных чисел.

Для получения низшей положительной оценки ученику достаточно выполнить первую часть задачи. Выполнение дополнительно каждого следующего пункта повышает оценку на балл. (Впрочем, в данном случае отличная оценка будет поставлена и за выполнение первой и третьей частей.)

Пока еще достаточно полной системой задач такого рода я не располагаю. Но, возможно, эта идея заинтересует других учителей и общими усилиями будет создан хороший сборник задач по информатике, отсутствие которого сегодня сильно ощущается.

Г. АНТИПОВ
канд. психол. наук

Ю. РЫСЕВ
канд. пед. наук, старший научный сотрудник ВНИИ профтехобразования

Опыт использования факторного анализа на ЭВМ

Исследуя профессиональную подготовку учащихся СПТУ, мы анализировали большие объемы (массивы) информации по оценке уровня обучения и воспитания учащихся. В этом нам помогали мастера производственного обучения. Они оценивают работу учащихся по следующим показателям:

усвоение программы производственного обучения;

теоретические знания по предметам;

применение теоретических знаний на практике;

умение организовать рабочее место;

уровень отработки навыков в процессе обучения;

уровень самоконтроля за процессом работы;

уровень самоконтроля результатов работы;

уровень самоконтроля за поведением на уроке производственного обучения;

мобильность поведения и мышления в учебно-производственной ситуации;

активность и желание хорошо отработать тему;

уровень осознания социальной значимости труда;

дисциплина в училище и вне его;

бережливость, отношение к общественной работе;

желание и готовность помочь товарищу.

Лучше мастеров вряд ли кто-то другой в училище знает индивидуальные особенности учащихся. Это, несомненно, увеличило степень точности оценки.

Авторы статьи рекомендовали мастерам ставить оценки по девятибалльной системе. Педагог оценивает качественную работу учащегося в 9, 8, 7 баллов. Более низкое качество оценивается в 6, 5, 4 балла; если работа выполнена некачественно — в 3, 2, 1 балл.

Оценка работы учащихся проводилась в группах, занимающихся слесарным, радиомонтажным и станочным делом в СПТУ № 33, 68 Ленинграда. Основная цель

исследований заключалась в том, чтобы выявить уровень (факторы) обучения на I, II и III курсах СПТУ. Фактором называют источник воздействия на систему (в нашем случае — на систему обучения), который отражается на значениях переменных ее модели.

Предложенные в свое время бывшим Госпрофобром СССР для апробации «Опытные критерии оценки знаний, навыков, умений учащихся средних профессионально-технических училищ» в целом удовлетворяют потребностям практики, так как ориентируют на усвоение знаний и овладение умениями, навыками, предусмотренными профессионально-квалификационными характеристиками. Однако для выполнения функции контроля не только результата, но и процесса учебно-производственных работ они нуждаются в дополнении более детальной системой показателей, отражающей особенности учебно-производственной деятельности по определенной профессии.

Авторы совместно с мастерами производственного обучения ленинградских профтехучилищ № 33, 68 А. Ключковским, Н. Устиновым, Э. Александровым, Т. Кротовой помимо апробации перечисленных в начале статьи показателей оценки обученности, развитости и воспитанности проводили более детальную оценку по опытным критериям, например по критерию «соблюдение требований безопасности труда». Деятельность учащихся детально анализировалась с помощью следующих показателей:

соблюдение правил работы на рабочем месте;

правил ношения спецодежды;

правил подготовки и уборки рабочего места;

правил безопасного использования приспособлений, транспортных средств;

умение в процессе работы, например, правильно поднимать тяжести и др.

Более подробная детальная оценка действий учащихся на различных этапах обучения необходима потому, что качества личности человека поддаются измерению только с помощью оценки результатов его действий.

Обработка данных производилась на ЭВМ «Наири-4» (операционная система ДОС-400, язык Фортран-IV). Комплект программ разработан одним из авторов статьи. Применялся метод главных компонент с последующей ротацией факторных нагрузок по квартимакскритерию.

В итоге удалось выявить скрытые обобщенные факторы, которые объясняют изменения изучаемых показателей.

Характеристика выделяемых факторов определяется наибольшими факторными весами (нагрузками) показателей. Значимые веса свидетельствуют о тесной взаимосвязи избранных показателей в реальном учебном процессе.

Число выделяемых факторов может быть различным. Степень их важности определяется так называемой факторной дисперсией, которую удобно выражать в процентах. Фактор, имеющий дисперсию менее 5 %, в данном исследовании считался малозначимым и не анализировался. Так, для одной из групп I курса суммарная дисперсия первого фактора оказалась равной 68 %. Причем максимальные нагрузки (0,927, 0,882, 0,877, 0,875) имели соответственно следующие показатели: усвоение программы производственного обучения, отработка навыков в процессе упражнений, дисциплина в целом и на уроке производственного обучения. Отметим, что факторный вес такого важного показателя, как производительность учебного труда, при анализе работы первокурсников имеет отрицательный знак (—0,355). Это соответствует реальному учебному процессу, так как на I курсе в основном внимание уделяется усвоению программы обучения и отработке навыков.

При анализе оценок учащихся той же группы на II курсе картина изменилась. Сохранилась корреляция факторов усвоения программы и дисциплины на уроке производственного обучения, но появились и другие: умение планировать работу — 0,913; самоконтроль за процессом работы, мобильность поведения и мышления в учебно-производственной ситуации — 0,779; готовность помочь товарищу — 0,783.

Появление в факторной структуре на II курсе шести показателей вместо четырех соответствует более сложным задачам мастера и второкурсников в процессе обучения.

На III курсе, когда учащиеся работают на базовом предприятии, число показателей увеличивается до восьми. Причем в их числе сохраняются показатели, проявившиеся на I и II курсах, но добавляются новые: отношение к выполнению общественных поручений — 0,939; активность в выполнении учебно-производственных заданий — 0,929, организация рабочего места — 0,924.

Анализируемые факторы отражают педагогический опыт мастера, учебно-производственную работу учащихся. Организация их рабочего места — предмет постоянной работы мастера. Одно дело, когда он постоянно контролирует состояние рабочего места в мастерской, другое — когда учащийся должен проявить свои знания и умения самостоятельно в условиях производственной практики. Увеличение факторных нагрузок показателей, отражающих уровень само-

развития, самоорганизации и воспитания учащихся наблюдается при анализе результатов оценки работы учебных групп III курса.

Сравнение оценок учащихся различных групп позволило выделить четыре фактора эффективности профессионального обучения.

1. Полученные знания, умения, дисциплина.

2. Высокий уровень отработки учебных заданий в сочетании с производительностью учебного труда.

3. Творческая и социальная активность.

4. Прогнозирование результатов обучения в СПТУ, общих и профессиональных качеств личности учащихся для успешной профессиональной деятельности в будущем.

Данные факторы действуют в комплексе. Они едины при обучении на I, II и III курсах, однако у разных мастеров они могут варьироваться в зависимости от их индивидуального подхода к обучению. Некоторые мастера с первых дней занятий больше занимаются воспитательной работой, другие больше внимания уделяют формированию профессиональных умений и навыков, третьи умело сочетают тот и другой подходы к обучению.

В группах, где мастером ведется целенаправленная, воспитательная работа, доминируют корреляция показателей отношения к учебе и таких, как осознание социальной значимости учебного труда, бережливость, желание и готовность помочь товарищу, отношение к общественным поручениям.

Более подробные разъяснения и консультации по практическому использованию разработанных программ можно получить, обратившись письменно по адресу: 197042, Ленинград, пр. Динамо, 2, Научно-исследовательский институт физической культуры. *Антипову Г. Л.*

От редакции.

Читателям, заинтересовавшимся подобными методами исследования, можно посоветовать следующие книги:

Окунь Я. Факторный анализ. М., 1974;

Франселл Ф., Баннистер Д. Новый метод исследования личности М., 1977.

Г. КРУЧИНИНА

Нужна ли школьнику информатика?

В 1987 г. в масштабах всей страны впервые закончен полный цикл изучения школьниками нового учебного предмета «Основы информатики и вычислительной техники». В короткие сроки учеными АН СССР и АПН СССР были разработаны учебные программы, написаны и выпущены в свет учебники и методические пособия к данному курсу. Можно много обсуждать их положительные и отрицательные стороны (что, кстати, делается на страницах печати), возраст учащихся, в котором следует начинать изучение этого предмета. Но главное все же нужно отметить — начата большая работа по обеспечению компьютерной грамотности подрастающего поколения.

Нами была поставлена цель исследовать отношение к новому учебному предмету школьников, изучавших его в машинном и безмашинном вариантах.

Анкетированием было охвачено около 900 учащихся X класса г. Горького и его спутников — г. Бор и Кстово: свыше 600 выпускников, изучавших ОИВТ в машинном варианте в школах, УПК, лабораториях предприятий, вузов, научно-исследовательских институтах (II группа); около 300 учащихся, изучавших новый предмет по безмашинному варианту (III группа). В число школьников II группы входили учащиеся школы № 23 Советского района г. Горького, базовой школы Горьковского университета, имеющей лучшие возможности — как технические (школьный вычислительный центр, включающий 3 терминал-класса, предоставленных шефами), так и педагогические — помощь ученых и преподавателей университета (I группа, свыше 130 чел.).

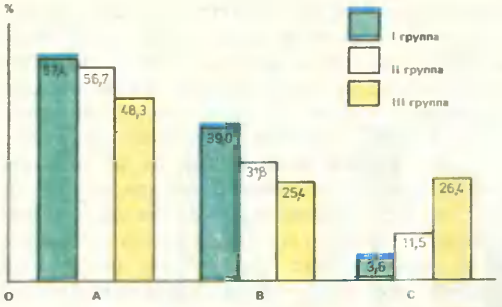
Исследование показало, что учащиеся, изучавшие ОИВТ в машинном и безмашинном вариантах, по-разному оценивают необходимость изучения информатики. Этого можно было ожидать. Нас же интересовали конкретные результаты, и мы предложили учащимся выорать справедливое, по их мнению, утверждение из следующих:

компьютерная грамотность (в основном обеспечиваемая школьным курсом ОИВТ) в настоящее время нужна всем учащимся (А);

информатика нужна лишь как факультативный предмет, для желающих (В);

введение нового курса преждевременно — ЭВМ нет ни в школах, ни на большинстве предприятий (С).

Результаты представлены на рис. 1.



Около 57 % школьников, изучавших ОИВТ в машинном варианте (I и II группы), считают, что знания и умения, которые дает новый учебный предмет, необходимы в настоящее время всем учащимся. Свыше 48 % учащихся III группы также разделяют это мнение.

39 % учащихся I группы, свыше 31 % — II группы и около 25 % школьников III группы полагают, что ОИВТ нужен лишь как факультативный предмет.

И наконец, существует группа учащихся, которые считают, что введение курса ОИВТ преждевременно. Как и следовало ожидать, мнения выпускников резко различались в зависимости от того, организованы ли, и насколько успешно, практические занятия с применением компьютера (I и II группы) или без него. Только около 48 % учащихся I группы считают, что введение нового учебного предмета преждевременно; этой же точки зрения придерживаются 11,5 % учащихся II группы и свыше 26 % выпускников III группы (знакомых с ЭВМ лишь по учебнику, рассказам педагога, экскурсиям и средствам массовой информации).

Для расчета достоверности различий в мнениях учащихся 3 выделенных нами групп был рассчитан интегральный показатель K для каждой группы, характеризующий ее отношение к компьютерному обучению.

$K = \frac{1 \cdot n_1 + 2 \cdot n_2 + 3 \cdot n_3}{n_1 + n_2 + n_3}$ — среднее арифметическое в выборе ответа A — 1 балл; B — 2 балла; C — 3 балла; n_1, n_2, n_3 — число учащихся, выбравших тот или иной ответ.

Данные представлены в табл. 1.

Таблица 1

Оценка значимости компьютерного обучения в школе (в условных баллах)

	K	σ	Доверительный интервал для K
I группа (n=133)	1,35	0,58	1,27 K 1,43
II группа (n=619)	1,54	0,69	1,49 K 1,59
III группа (n=272)	1,77	0,84	1,69 K 1,85

С достоверностью 0,95 можно считать, что оценка зависимости компьютерного обучения в школе зависит от организации процесса обучения.

Значение G — среднего квадратического отклонения (меры разброса результатов) — свидетельствует, что наиболее однородно мнение школьников, имеющих наилучшие условия для работы с компьютерной техникой, наиболее подготовленных преподавателей; наибольший разброс показателей у учащихся, изучавших ОИВТ в безмашинном варианте.

Но группы неоднородны. Мы рассмотрели на примере учащихся II группы следующие варианты дифференциации: юноши и девушки (дифференциация по полу); «гуманитарии» и «технари» — условно выделенные группы школьников (дифференциация по профессиональной направленности); подгруппы учащихся в зависимости от успешности обучения в школе. Результаты представлены в табл. 2.

С достоверностью 0,95 можно считать, что существует различие в отношении к компью-

Таблица 2

Дифференциация значимости компьютерного обучения в школе (машинный вариант обучения ОИВТ)

Дифференциация по признаку подгруппы		K	σ	Доверительный интервал для K		
Пол	юноши (n=247)	1,52	0,70	1,45	K	1,59
	девушки (n=372)	1,56	0,68	1,50	K	1,62
Успешность обучения	4,5 балла и выше (n=126)	1,53	0,69	1,43	K	1,63
	4,0—4,5 балла (n=304)	1,53	0,67	1,47	K	1,59
	до 4,0 баллов (n=189)	1,57	0,68	1,49	K	1,65
Профес. направлен.	«гуманит.» (n=234)	1,65	0,66	1,58	K	1,72
	«технари» (n=385)	1,48	0,69	1,42	K	1,54

терному обучению в школе между подгруппами «гуманитариев» и «технарей» (доверительные интервалы для K не перекрываются). Выпускники, имеющие интерес к технике, физике, математике и предполагающие выбрать специальность в соответствии со своими интересами, более склонны считать, что компьютерное обучение в школе необходимо всем в настоящее время. По другим выделенным нами подгруппам достоверной разницы в результатах не выделено.

Изучая более внимательно отношение к компьютерному обучению в школе у учащихся, изучавших ОИВТ в машинном варианте, необходимо отметить более значительный интерес к практическим занятиям на ЭВМ, чем к теории. Выпускники школы оценивали отношение к теоретическим и практическим занятиям ОИВТ по пятибалльной шкале (от 5 баллов — да, очень нравятся — до 1 балла — не нравятся). Результаты представлены в табл. 3.

Интегральный показатель K, оценивающий отношение к теории, составляет 2,58 балла (ближе к нейтральной оценке); по практике — 4,13 балла (несколько выше, чем «нравится»). Анализируя оценку юношами и девушками теоретических и практических занятий, мы отмечаем примерно одинаковые результаты (разница статистически недостоверна): в оценке теории $K_{ю}=2{,}59$ балла, $K_{д}=2{,}57$ балла; в оценке практики: $K_{ю}=4{,}08$ балла, $K_{д}=$ балла. В зависимости же от направленности интересов личности, от выбора будущей профессиональной деятельности достоверно различимые оценки у «технарей» и «гуманитариев». И та и другая выделенные нами группы выше оценивают практику, чем теорию, но оценки «технарей» выше. Исследование показало, что содержание нынешнего курса ОИВТ более предпочтительно для «технарей», чем для «гуманитариев», причем разница в оценке теории более значительна ($K_{теор.}=0{,}33$ балла), чем в оценке практических занятий на компьютере ($K_{практ.}=0{,}24$ балла).

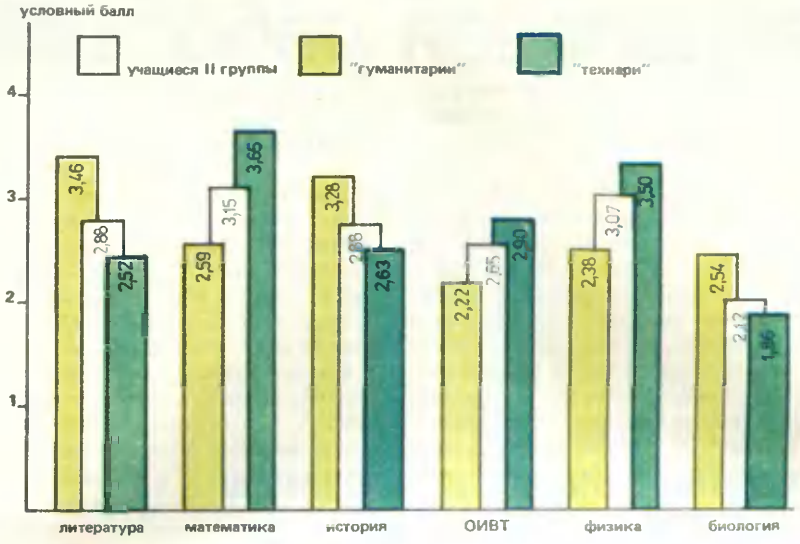
Мы кратко охарактеризовали оценку выпускниками курса ОИВТ с точки зрения его необходимости для всех, для части школьников (как факультатив) или его преждевременности. Интересно также, как оценивают они те знания, умения и навыки, которые получили в течение двух лет, с точки зрения необходимости их в дальнейшей жизни. Оценка была проведена по следующим критериям: 4 балла — очень нужны; 3 балла — нужны; 2 балла — нужны, но не очень; 1 балл — не нужны. Но, как говорится, все познается в сравнении. Оценка знаний и умений, заложенных в содержание нового учебного предмета, интересна не только сама по себе, но и в сравнении с аналогичной оценкой других учебных дисциплин. Поэтому мы просили школьников оценить знания и умения, полученные ими также по литературе, математике, истории СССР, физике, биологии. Результаты представлены на рис. 2.

С вероятностью 0,95 статистически достоверна разница в оценке необходимости знаний и умений по различным предметам (лишь литература и история имеют одинаковый показатель), а также разница в оценке необходимости знаний и умений по каждому учебному предмету между «гуманитариями» и «технарями». Мы не будем подробно анализировать данные, представленные на этом рисунке, каждый читатель при желании сможет это сделать сам. (Здесь очень четко проявляются аморфность понятия «средний ученик». Какие они разные! Говорим и пишем об этом много, а со всех учащихся часто требуем одинаково.) Обратим внимание на ОИВТ. Как этот «юный» учебный предмет выглядит в ряду предметов «солидных» по возрасту? С одной стороны, и физико-математическому, и историко-филологическому циклам пока уступает, но последнему — совсем немного. Да, действительно, на сегодняшний день использование компьютера на производстве, в науке, образовании не является в целом широко распространенным (мы не касаемся отдельных областей деятельности, о которых у учащихся, возможно, мало информации), и поэтому их оценка необходимости знаний и

Таблица 3

Оценка значимости теории и практики в компьютерном обучении
(в условных баллах)

		K	σ	Доверит. интервал для K		
Теория	«гуманит.» (n=220)	2,37	1,18	2,24	K	2,50
	«технари» (n=375)	2,70	1,18	2,60	K	2,80
	всего (n=595)	2,58	1,19	2,50	K	2,64
Практика	«гуманит. (n=232)	3,99	1,04	3,88	K	4,10
	«технари» (n=379)	4,23	0,94	4,16	K	4,30
	всего (n=611)	4,13	0,94	4,07	K	4,19



умений, получаемых в курсе ОИВТ, ниже, чем по физике, математике и несколько уступает гуманитарным дисциплинам. Но если учесть, что это только первые шаги в таком масштабном деле (не говоря уж о трудностях с технической базой, педагогами), то оценка информатики в ряду других предметов не представляется низкой. (Здесь необходимо подчеркнуть, что оценка производилась для школьников, изучавших машинный курс.)

Какие положительные стороны в изучении информатики указывают учащиеся? Не ранжируя их по значимости, отметим наиболее часто встречающиеся ответы.

Первая группа ответов связана с оценкой роли компьютера в ознакомлении с современной техникой, развитием интереса к современной вычислительной технике и умением пользоваться ею, с признанием необходимости компьютера в будущем.

Вторая группа ответов связана с влиянием информатики на развитие личности: расширяет кругозор; развивает логическое мышление (этот мотив указывается очень часто); формирует умение четко поставить перед собой задачу; показывает универсальность решения задач определенных типов; прививает умения анализировать и систематизировать знания; приучает к внимательности, последовательности, аккуратности, точности и т. д.

В то же время учащиеся указывают на недостатки в организации занятий и преподавании ОИВТ, которые нужно устранить в первую очередь: недостаточное количество компьютеров в школе, а в большинстве случаев (на 1987 г.) их отсутствие (терминал-класс — каждой школе); разрыв между теорией и практикой — некоторые учащиеся предлагают изучать новый предмет вообще только практически или, во всяком случае, увеличить на практику количество учебных часов; недостаток высококвалифицированных преподавателей, объясняющих на доступном языке (некоторые учащиеся категорически требуют повысить качество преподавания!).

Учащиеся также ощущают необходимость оснащения школ современными персональными компьютерами.

Н. ШАРОВА, Т. КАРАСЕВА

Школьники пишут программы

Конкурсы программ, созданных школьниками, которые проводятся по инициативе районного методического объединения учителей информатики, стали традиционными в Калининском районе Москвы.

Сравнивая конкурс с олимпиадой, отметим, что олимпиада — это соревнование, в котором участники должны дойти до финиша по заранее «проложенной трассе», за исключением тех редких случаев, когда участник отыскивает более оригинальное, чем предусмотрено организаторами олимпиады, решение. Конкурс — это и праздник реализации задуманной идеи, и творческое удовлетворение от законченной работы, и соревнование идей. В конкурсе могут принять участие все желающие, т. е. каждый, способный реализовать алгоритм и сценарий в виде программного продукта.

В первом конкурсе, который был проведен в 1987 г. на базе кабинета, оборудованного комплектом учебной вычислительной техники «Ямаха», приняли участие учащиеся VII—X классов. Тематика представленных программ была однообразна, в основном они носили демонстрационный характер, однако конкурс собрал много зрителей и оказался очень хорошей рекламой: ребята поняли, что компьютер — это просто и доступно. В 1988 г. количество участников конкурса увеличилось вдвое, да и тематика стала более разнообразной и серьезной. Например, были представлены программа для расчета заработной платы в учебно-производственных мастерских, психологический тест для определения предпочтительной сферы деятельности, обучающие программы, графический редактор для компьютера БК-0010.

Готовясь к конкурсу, учитель организует поисково-творческий режим работы на уроках и во внеурочное время для учащихся, которые наиболее успешно изучают основы информатики и вычислительной техники. Лучшие программы поступают в библиотеку кабинета и могут быть в дальнейшем использованы в учебном процессе.

Встреча на конкурсе как участников, так и учителей способствует обмену опытом, идеями. Конкурс проходит под девизом «Фантазия — умение — цель». Жюри конкурса (а оно состоит из учеников X классов школ — участниц конкурса) выставляет по пятибалльной системе три оценки. Оценки суммируются, и победителем становится участник, набравший наибольшее количество очков. Но как сравнить игровую программу с обучающей или деловой? Не все участники остались довольны судейством в 1988 г., поэтому методическим объединением учителей информатики Калининского района было разработано «Положение о проведении конкурса 1989 г.», по которому представленные программы должны быть отнесены к одному из трех типов: деловая, обучающая, игровая; по каждому типу программ были проведены раздельные конкурсы. До конкурса с каждой программой внимательно знакомится оппонент и при представлении программы высказывает свое мнение.

По аналогии с конкурсом в Калининском районе, 6 января 1989 г. на ВДНХ СССР прошел конкурс игровых и обучающих программ, разработанных школьниками Москвы. В нем участвовали представители пяти школ и учебно-производственных мастерских Калининского района. Конкурс проводился комиссией в составе учителей и учеников, которая оценивала представленные программы по следующим критериям: новизне и оригинальности решения, увлекательности, тактичности диа-

лога и удобства пользования, полноте реализации цели, надежности.

Представленные программы по целевому назначению были разделены на игровые, деловые, учебные. Среди учебных программ были программы, предназначенные для уроков русского языка, математики, информатики, английского языка.

Первое место заняли три программы: «Редактор текстов», автор — Миша Гонодонов, 491-я школа (ПЭВМ БК-0010); «Биржа», авторы — Наташа Хазова, Таня Селянинова, 444-я школа (ПЭВМ «Агат»); «Меню» автор — Олег Блащук, 35-я школа (КУВТ «Ямаха». «Редактор текстов» позволяет добавлять строки в середину текста и в его конец, убирать строки, сохранять набранный текст на внешнем носителе и т. д. «Редактор» имеет подробную инструкцию, к которой можно обращаться из любого режима. В настоящее время используется при обучении школьников в классах, оснащенных БК-0010.

Программа «Биржа» предназначена для проведения контроля знаний учащихся по теме системы счисления. В ней присутствует игровой элемент, и школьники с удовольствием выполняют контрольные задания по переводу и сравнению чисел в 10-, 12-, 16-ричной системах счисления.

Программа «Меню» позволяет составить рациональное по калорийности меню в зависимости от веса, пола, трудовой деятельности места жительства пользователя.

В ходе конкурса и по его завершении участники и зрители (школьники и учителя) имели возможность познакомиться со всеми представленными работами, скопировать заинтересовавшие их программы и дать собственную оценку их качеству. Возникший при этом диалог заинтересованных людей позволил обсудить многие вопросы профессионального характера.

Обсуждение показало, что процесс создания школьниками обучающих программ успешно проходит в тех коллективах, где поставлена общая задача создания библиотеки учебных программ по различным предметам. Качественные программы могут создаваться только при выполнении следующих условий: постановка задачи осуществляется учителем, преподающим соответствующий предмет; разработка сценария ведется учащимися совместно с учителем по данному предмету и учителем информатики; при разработке программ учитываются требования, предъявляемые к программам соответствующего класса.

Одной из наиболее серьезных проблем в организации этой работы является планомерное вовлечение учителей, которые не только не представляют всех возможностей ЭВМ, но и недостаточно четко осознают преимущество использования вычислительной техники в учебном процессе. Организуемые для них кратковременные занятия по основам информатики не дают быстрого и ощутимого эффекта. Это является причиной неготовности, а зачастую и нежелания учителей включаться в процесс разработки учебных программ, что отражается и на отношении школьников к этой работе.

Вместо того чтобы конкретно поставить задачу, совместно с учениками написать сценарий, учителя выдают список литературы, устные пожелания и отдают все на откуп школьнику. Естественно, что качественной программы в этом случае не получается. Зачастую эти программы повторяют учебник, с переносом картинок со страниц на экран с минимальной обратной связью между программой и человеком. Интереса такая программа не представляет ни для ученика, ни для учителя. Полная творческая отдача при работе над программой у школьников может быть лишь тогда, когда он уверен в необходимости выполняемой работы. Совместная целенаправленная деятельность ученика и учителя должна основываться на сотрудничестве. При обсуждении учителя и ученики пришли к выводу, что польза, приносимая процессом разработки учебных программ, несомненна. В этой работе выявляются профессиональные и технологические особенности программиста. Есть заказчик (учитель), есть исполнитель — коллектив программистов (два или три школьника, которые делят работу между собой), есть пользователи (школьники, которые в ходе опытной эксплуатации выявят имеющиеся недостатки, чем помогут довести программу до требуемого качества).

В ходе обсуждения все пришли к единому мнению, что практическая ценность программы зависит и от ее сценария. Имея только знания по программированию, школьники не могут написать сценарий программ, удовлетворяющий требованиям современного урока. Привлечение учителей к этой работе является первоочередной задачей.

Важно, чтобы учителя не только понимали, что ЭВМ может считать, рисовать, «петь», но видели готовые программы, чтобы поверили в необходимость использования современных технических средств на уроке, могли сделать что-то нужное для себя.

Просмотр готовых программ создаст у учителя представление о хороших и плохих программных продуктах, позволит сформировать методические приемы написания сценария. В этой работе помогут библиотеки программ школы, которые могут формироваться из программ, разработанных собст-

венным коллективом, полученных в результате контактов с другими школами, поступающих от организаций народного образования и через конкурсы. Последний вариант дает возможность выбора, так как представленные на конкурс программы не только оцениваются, но демонстрируются и копируются для желающих.

Мы ожидаем появления централизованно разработанных комплектов программ, которые будут поставляться в качестве методического материала всем школам. Однако процесс создания пакетов программ длительный и сложный. Работа школьников над учебными программами может предварять и дополнять его.

Регулярное проведение конкурсов позволит придать этой работе планомерный, массовый и целенаправленный характер, а также стимулировать более качественное выполнение программ.

Анализируя детское творчество и восприятие детьми конкурсных программ, можно изучить психологические особенности отношений ученик-компьютер, возможности использования ЭВМ при работе с различными возрастными группами.

Участники конкурса предлагают и в дальнейшем проводить Московский конкурс школьных программ через отборочные районные туры, с проведением заключительного тура на базе организации, оснащенной всеми видами техники, применяемой в школах Москвы, например в городском ИУУ.

Ю. ПОЛУШКИН

Компьютер и азбука Морзе

В Омском педагогическом институте наряду с внедрением новых информационных технологий в подготовку учителей всех специальностей большое внимание уделяется применению компьютеров во внеклассной работе.

Более двадцати лет в институте работает спортивно-технический радиоклуб, где студенты различных специальностей готовятся стать руководителями школьных радиостанций. Одним из разделов такой подготовки является изучение телеграфной азбуки. Традиционные методы обучения приему на слух требуют много времени и рассчитаны на среднего ученика, в то время как скорость восприятия информации и реакция на сигнал зависят от индивидуальных особенностей учащегося. Именно поэтому мы поставили перед собой задачу создать компьютерный курс «Изучение телеграфной азбуки». В 1986 г. была разработана его первая версия для ПЭВМ «Ямаха». Курс прошел двухлетнюю экспериментальную проверку, в которой участвовали школьники III—X классов и студенты различных факультетов. С учетом выявленных недостатков методического и технического характера была разработана вторая версия курса для ПЭВМ «Агат» (Бейсик, объем 19К байт).

Программа-тренажер позволяет самостоятельно подготовиться к приему на слух сигналов азбуки Морзе с любой необходимой скоростью. Максимальная скорость следования сигналов — 300 зн/мин.

Курс состоит из 12 уроков, из них 11 предназначены для разучивания сигналов (букв, цифр и знаков препинания), а двенадцатый — для наращивания скорости приема. Выбор урока учащийся осуществляет с помощью меню. В каждом уроке — 8 обязательных упражнений, выбираемых по этому же принципу.

В первом упражнении учащемуся 5 раз предъявляется звуковой сигнал («мелодия») и соответствующая ему словоформа, а затем 5 раз другой сигнал, отличающийся от первого последним элементом, и его словоформа. В каждом случае программа сообщает, каким пальцем и рукой надо вводить символ с клавиатуры. Сигналы предъявляются по очереди до тех пор, пока учащийся не решит, что он надежно их усвоил, причем число повторений одного и того же сигнала постепенно уменьшается с пяти до двух.

Второе упражнение — тренировка вновь разученных знаков по мелодиям. Учащемуся предъявляется случайно выбранный звуковой сигнал, один из двух разученных, а от него требуется ввести с клавиатуры соответствующий этому сигналу знак (букву, цифру, знак препинания). В случае верного ответа на экране появляется положительный комментарий — и ЭВМ предъявляет следующий сигнал. Если ответ неверен, то после соответствующего комментария сигнал звучит снова — и ЭВМ предлагает повторить ввод. В случае повторной ошибки программа разъясняет, какой сигнал прозвучал, и предлагает ввести его правильно, после чего предъявляет следующий.

Упражнение построено таким образом, что те сигналы, которые ученик опознает неверно, предъявляются ему чаще остальных, т. е. машина добивается правильного усвоения всех сигналов. Если число ошибок велико, ЭВМ предлагает повторить это упражнение; если менее 5, то программа предлагает разучить еще два знака (3-е упражнение). Упражнение 4 аналогично второму, только теперь учащемуся предъявляются все 4 знака, разученных в данном уроке.

Если действия учащегося удовлетворяют критерию обученности (ошибок менее 5), программа предлагает упражнение 5 — тренировочный текст. Оно отличается от предыдущего отсутствием комментария после каждого знака. Кроме того, ЭВМ предоставляет всего одну попытку для исправления ошибки. Это упражнение адаптивное. Его темп определяется индивидуальными способностями учащегося. В предыдущем упражнении критерием усвоения был верный ввод символа, в этом упражнении два критерия — верный ввод символов и скорость выполнения упражнения. В случае несоответствия результата хотя бы одному из критериев предлагается повторить упражнение.

Упражнение 6 — контрольный текст. В отличие от предыдущего в нем скорость следования звуковых сигналов не зависит от действий учащегося. Его задача — вводить символы в соответствии с предложенным темпом. Критерий усвоения — число ошибок меньше 5. Если результат учащегося ему не удовлетворяет, упражнение повторяется.

Упражнение 7 (тренировочный текст) отличается от пятого (а упражнение 8 от шестого) тем, что в него входят все знаки, разученные к данному моменту.

Упражнение 9 — необязательное, справочно-консультационное. Оно позволяет вызвать звуковой сигнал и получить необходимые пояснения при вводе интересующего вас символа.

Каждое из перечисленных упражнений учащийся может прервать или начать сначала. При желании можно выйти в меню урока или курса и выбрать другое упражнение или урок.

За каждое упражнение начисляется определенное количество очков (игровой элемент) в зависимости от числа ошибок и времени выполнения упражнения. В конце занятия на экране появляются статистические сведения об итогах работы.

В 12-м уроке учащийся сам выбирает сжатость сигналов и паузу между ними. Текст записывается на бумаге, а затем сравнивается с эталоном на экране дисплея. Учащийся может заказать корректировочные упражнения, запретив выдачу некоторых сигналов или, наоборот, увеличив вероятность их появления.

В июле — августе 1988 г. совместно со школой ДОСААФ был проведен производственный эксперимент. Он показал гораздо большую активность курсантов экспериментальной группы по сравнению с контрольной.

Среди главных достоинств компьютерного обучения учащиеся отмечали психологический комфорт, возможность повторять материал без боязни отстать от других, проводить оперативную проверку усвоения материала. Значительно повысился интерес к занятиям и, как следствие, интенсивность обучения — количество знаков, распознаваемых в единицу времени, возросло в 2—3 раза.

Накопленный опыт позволяет сделать следующие выводы относительно построения программ-тренажеров:

1. Для повышения интереса к работе с программой, особенно у школьников IV—V классов, большое значение имеют личностная (обращение по имени) форма комментариев и их разнообразие. В нашей программе — 100 вариантов положительных комментариев и столько же отрицательных. Последние обязательно заканчиваются оптимистическим пожеланием, что внушает учащимся веру в свои силы и стремление добиться успеха.

2. На начальном этапе необходимо предусмотреть возможность прервать упражнение и, проверив то, что сделано, начать его сначала, при желании уменьшив степень сложности.

Более способные учащиеся должны иметь возможность «досрочно» перейти к более сложному заданию. Такой подход к выбору упражнений и их сложности создает ощущение психологического комфорта, чувство, что учащийся управляет программой, а не она им.

Таким образом наш опыт показывает, что при разработке программ-тренажеров нужно стремиться к созданию среды, в которой происходит совершенствование определенных навыков в полном соответствии с индивидуальными особенностями и желанием учащегося.

За справками обращаться к Юрию Викторовичу Полушкину по адресу: 644043, г. Омск, а/я 1742, тел.: 23-20-00, 23-47-81.

Точка зрения

В. ФЕДОТОВ

Нужна ли информатика как самостоятельный предмет в общеобразовательной школе?

Можно лишь удивляться тому, что вынесенный в заголовок вопрос не стал предметом бурной дискуссии еще 3—5 лет тому назад. Негативные высказывания в адрес нового учебного предмета неоднократно звучали, едва только были приняты решения о его повсеместном преподавании. Однако в тот момент многочисленные голоса не слились в дружный хор, а наиболее громкий аргумент тогдашних противников школьной информатики — нельзя достичь компьютерной грамотности, не имея компьютеров,— легко опровергался идиллическими планами повальной компьютеризации на текущую пятилетку. И если невыполнение самих этих планов стало уже почти свершившимся фактом, то нельзя не отметить, что именно на этом направлении сделан качественный шаг вперед, радикально изменивший отношение к компьютеризации со стороны несведущей публики и заставивший смолкнуть первый отряд противников школьного курса ОИВТ.

Когда же были отвергнуты лежавшие на поверхности контраргументы, то выявились более серьезные. Заметно изменился и отряд оппонентов. Если первоначально его основу составляли учителя, имевшие особо острый личный интерес к тому, чтобы избежать ожидавших их трудностей, то теперь стали слышны голоса специалистов, включая крупных ученых — и математиков, и педагогов. В частности, В. Г. Болтянский и В. М. Монахов неоднократно с разных высоких и малых трибун высказывались за то, чтобы вместо единого курса ОИВТ сделать добавления в ранее существовавшие дисциплины, прежде всего математику и физику.

Неявным, но и весьма сильным аргументом против школьной информатики явился и крайне низкий уровень ее преподавания, «достигнутый» почти повсеместно. Конечно, никто не ожидал блестящих результатов от прошедших двухнедельный ликбез в ИУУ учителей, старшие поколения которых ни в вузе, ни где-либо еще не изучали программирования. Однако никто не ожидал и столь удручающей картины. Так, участники областной олимпиады в Иванове допускали в записи алгоритмов на Е-языке грубейшие ошибки (например, то раньше, чем если). А ведь там собрались лучшие из лучших — победители районных олимпиад. Первая Всесоюзная олимпиада показала, что Ивановская область в этом отношении весьма типична: кроме ведущих научных центров — Москвы, Ленинграда, Киева, Новосибирска и Свердловска — относительно благополучным преподавание ОИВТ является только в Прибалтийских республиках.

Четко выделился и сопутствующий вопрос: когда начинать изучение информатики? Сейчас в большинстве школ это делается в IX классе. Однако, согласно уже обнародованным перспективным учебным планам, в следующей пятилетке курс ОИВТ должен спуститься в VII—VIII классы. В Свердловске уже издан соответствующий учебник (правда, пока по нему идет эксперимент в IX—X классах, но авторы не скрывают своих намерений), еще один учебник готовится в НИИ СиМО. Многие экспериментаторы не ждут руководящих указаний. Так, Н. А. Юнерман в Новосибирске уже выпустила несколько классов, изучавших информатику и программирование в течение 7 лет (с IV по X класс). Под названием «Роботландия» в Переславле-Залесском Ю. А. Первин, А. А. Дуванов, Я. Н. Зайдельман и другие разработали курс информатики, первоначально предназначавшийся пятиклассникам, но ока-

завшийся вполне доступным и третьеклассникам. Самые смелые пошли еще дальше: Г. В. Морозова в г. Иванове, А. Н. Николаев в г. Саранске и А. К. Поливанова в Москве ведут уроки с первоклассниками. Наверное, не предел и это: телепередача «Спокойной ночи, малыши» уже вела репортаж из компьютеризованного детского сада.

Наконец, не избежать и главного вопроса: каковы цели преподавания информатики? Отвечая на этот вопрос, мы обязаны учитывать, что, во-первых, информатика является общеобразовательной дисциплиной, а потому цели ее преподавания должны быть обязательными для всех обучающихся. Во-вторых, информатика является одним из многих школьных предметов, а потому цели ее преподавания не могут дублировать целей преподавания ни математики, ни физики, ни языков, ни какого-либо иного предмета. В форме лозунга «достичь компьютерной грамотности» ответ на этот вопрос дал еще А. П. Ершов. Этот тезис пребывал в качестве лозунга до тех пор, пока В. А. Каймин не расшифровал компьютерную грамотность по аналогии с обычной грамотностью, как умение читать, писать, считать, рисовать, но с помощью компьютера. И сразу же стало ясно, что провозглашенный лозунг не соответствует содержанию учебников А. П. Ершова (как первоначальных — в соавторстве с В. М. Монаховым и другими, так и новых — в соавторстве с А. Г. Кушниренко и другими).

Попытаемся, учитывая все контраргументы, прийти к выводу о целях и месте информатики в школьном образовании. Бесспорно, следует согласиться со сторонниками включения разделов информатики в другие дисциплины в той части, что устройство компьютера, наряду с устройством других электроприборов, должно изучаться в курсе физики, а место машинных вычислений — в курсе математики. Однако информатика имеет и сугубо свои цели, а также содержание, не укладывающиеся в рамки других дисциплин. По мнению автора, оно должно быть сгруппировано в три концентра, которые, учитывая возрастные и психологические особенности детей, следует отнести к трем различным этапам обучения в школе.

Первый — вычислительная техника, точнее, первоначальное знакомство с ней. Здесь должны быть только названы основные части компьютера и продемонстрировано их функциональное назначение. Цель — последующее использование компьютеров, пока только в режиме «меню», на других уроках. В соответствии с этой целью этот концентр должен быть отнесен к самому раннему этапу обучения в школе. Повествовательно-назывной характер изложения позволяет сделать его доступным даже шестилеткам. Именно такое положение концентра позволит достичь еще одной цели: интенсивно использовать клавиатуру компьютера для обучения чтению. Так как в начальной школе, как правило, все уроки ведет один учитель, то нет нужды дискутировать на тему, следует ли выделять этот концентр в самостоятельный предмет или включать в состав родного языка.

Второй — программирование. Именно на этом этапе и должна в основном достигаться компьютерная грамотность. Этот концентр следует отнести к середине школьного обучения (V—VI класс). С одной стороны, эксперименты показывают, что школьники в этом возрасте уже способны изучить язык программирования. С другой, своевременно приобретенное умение программировать может активно использоваться в математике, физике, химии, биологии, истории, иностранном языке и т. д., качественно изменяя содержание самих этих дисциплин. Преподавать программирование чаще всего будут учителя математики. Однако включение программирования в состав курса алгебры представляется ошибкой.

Наконец, третий концентр — собственно информатика. Здесь должны быть даны не наивные, а достаточно серьезные определения информации, алгоритма и других сопутствующих понятий. Здесь нужен содержательный разговор о происходящей на наших глазах информатизации общества и ее социальных последствиях. Такой предмет должны преподавать не математики или физики, а, прежде всего, выпускники философских факультетов университетов, а также соответственно подготовленные учителя обществоведения, биологии или литературы.

Я. БАРСУК, Ю. КОГАН, Л. МЕЛЕНТЬЕВА

Компьютерное обучение в системах административного управления

Различные аспекты деятельности управленческих работников настоятельно требуют ныне применения вычислительной техники. От квалификации тех, кто стоит на капитанском мостике той или иной отрасли, зависит, начнется ли оздоровление нашей экономики.

Обучение работников сферы управления основам компьютерной грамотности будет проходить эффективнее с помощью подхода, о котором рассказывается в предлагаемой статье.

Проблема автоматизации интеллектуальной работы чрезвычайно актуальна. Производительность труда в сфере производства растет значительно быстрее, чем в сфере интеллектуального труда. И это существенно сдерживает общее движение вперед.

Сегодня очевидна тенденция к формированию информационно-ориентированного общества, для которого важна адекватная потребностям человеческой деятельности обработка и использование информации. Она достигается с помощью современной вычислительной техники, специальных систем, которые становятся важнейшим структурообразующим фактором современного общества, охватывающим практически все сферы человеческой деятельности: экономику, науку, управление, культуру, искусство, образование, международные отношения и т. д.

Наконец, решается проблема персонализации взаимодействия человека и компьютера. Не человек идет к компьютеру, а компьютер неотвратимо приходит на рабочее место. Несмотря на очевидную тенденцию обеспечить взаимодействие пользователя с ЭВМ в форме, принятой между людьми («дружественный» интерфейс), существует еще много проблем (технических, организационных, социально-психологических и других), затрудняющих освоение современной информационной техники и технологии (именно освоение, а не внедрение, поскольку последний термин имеет «насильственную» окраску).

Передовая информационная технология базируется на применении современных ПЭВМ и оборудования, пакетов программных средств и административных процедур, которые обусловливают взаимосвязи между элементами систем получения, хранения, переработки и передачи информации.

До последнего времени при создании локальных автоматизированных систем в организационном управлении был принят стандартный для АСУ функциональный подход, который характерен тем, что с начала проекта определяется набор автоматизируемых функций и затем под этот набор проводится разработка так называемой обеспечивающей части, включая техническое, программное, информационное, организационное и т. д. обеспечение. Сдача системы в эксплуатацию определяется по соответствию проекта техническому заданию, сформированному исполнителем и заказчиком в начальной стадии работы. Ввиду того что исполнитель и заказчик, особенно в начале работы, общаются на разных «языках», сама система управления, в которой предполагается внедрить средства автоматизации, динамично изменяема, а срок разработки достаточно велик (2—3 года), к началу промышленной эксплуатации технология выполнения автоматизированных функций существенно изменяется по сравнению с техническим заданием и проектом.

При этом обычно пользователь оказывается лишен возможности оперативно изменять систему независимо от разработчика.

Новый подход связан с отказом от ранней автоматизации функций, выполняемой разработчиком. В отличие от функционального подхода он может быть определен как операционный подход.

Суть его заключается в разработке пакета типовых базовых программных модулей автоматизированных рабочих мест (АРМ), обеспечивающих автоматизацию типовых операций по обработке информации, характерных для различных аспектов деятельности управленческих работников; создании системных функционально-ориентированных программных модулей, настройка которых обеспечивает создание системы информационного обслуживания (СИО) на базе АРМ (типа электронной почты, контроля исполнения, доступа к базам данных и т. д.); разработке ряда методических рекомендаций по ведению информационных массивов, взаимодействию персональных и локальных (подразделенческих) баз данных, организации ведения новой информационной технологии и т. д.

При автоматизации конкретных управленческих подразделений основные усилия группы внедрения должны быть сосредоточены на определении начального состава комплекса технических средств и передаче пакетов ПО и методических рекомендаций; обучении пользователя работе на ЭВМ, АРМ, конструированию информационных массивов и алгоритмов обработки информации, настройке системных объектно-ориентированных программ, обеспечивающих организацию СИО на базе отдельных АРМ; организации консультационно-методического руководства созданием СИО и обработкой информации силами пользователя.

Таким образом, суть нового подхода состоит в том, чтобы установить технические средства, передать необходимые настраиваемые пакеты системного уровня и обеспечить обучение пользователя, закончив его совместной пробной разработкой нескольких задач по выбору пользователя. На этом заканчивается внедрение.

В последующем заказчик самостоятельно проводит разработку и развитие системы под консультационно-методическим наблюдением специалистов.

Предполагается, что на период создания и тиражирования систем информационного обслуживания (СИО) основные усилия при внедрении должны быть направлены на обучение пользователей (до 80 % затрат).

Отсюда — важнейшая проблема организации обучения пользователей (привыкших работать с информационными ресурсами традиционно) в «дружественном» режиме, при котором человек, не будучи программистом, за сравнительно короткий промежуток времени осваивает основные возможности современных персональных компьютеров, постоянно соотнося эти возможности с собственными потребностями. В результате обучения пользователь должен осваивать прикладные программы и использовать их в повседневной работе.

В настоящее время в рассматриваемой нами предметной области (системы административного управления — САУ) просматриваются три уровня компьютерного обучения:

1. Обучение пользователей работе с ЭВМ и АРМ. Имеется в виду работа в операционной системе по инициализации и копированию дискет, файлов, работа с оглавлениями, создание и уничтожение файлов, вызов пакетов программ, редактирование текстов, формирование документов, проведение расчетов, создание и ведение персональной базы данных и организация запросов к данным.

2. Обучение пользователей ведению системы информационного обслуживания подразделения. Имеются в виду работы по созданию и ведению локальной (подразделенческой) базы данных; настройке (инстолляции) системных объектно-ориентированных программ, обеспечивающих доступ к данным, контроль исполнения, контроль прохождения документов; определению последовательности автоматизации задач и функций и т. д.

Обучению подлежат только те пользователи, в должностных обязанностях которых будут зафиксированы перечисленные функции.

3. Обучение пользователей организации работы в новых условиях с учетом повышения интеллектуальности труда, появления новых должностных обязанностей, качественного изменения и перераспределения функций, изменения коммуникаций, общей интенсификации управленческой деятельности.

Обучению подлежат все пользователи (первые руководители и организаторы-технологи процессов управления отдельно).

Специально организуемое проведение профессиональной подготовки персонала в САУ должно преследовать следующие основные цели: предупреждение негативного отношения к ЭВМ (отсюда особые требования к методическому обеспечению обучения), обучение практической работе с машиной, обучение деятельности в компьютеризованной системе.

Эксперимент по компьютерному обучению осуществлен в НПО «Горсистемотехника» (Киев) в рамках исследования по созданию

АРМ работника административно-управленческого аппарата. Реализована первая часть практики компьютерного обучения: работа с прикладными программами на автоматизированном рабочем месте.

Есть основания полагать, что необходимо учить не только и не столько тому, как «нажимать кнопки», сколько решению задач с помощью имеющихся средств. Потребность решать задачи возникает у обучаемого тогда, когда он хорошо представляет себе возможности современных компьютерных систем и собственные нужды. Необходимо обучать этим возможностям и активизировать потребности, чтобы в месте их «стыковки» возникал эффект естественного (а не насильственного) внедрения, инициатива в котором принадлежала бы пользователю.

Потребности в компьютерном обучении в системах административного управления формулируются следующим образом:

концептуальное знание, не требующее от обучаемого последующей активности в часы использования компьютеров (ему необходимо быть «в курсе дела»);

активное знание, необходимое для организации широкого внедрения компьютеров;

умения и навыки, позволяющие использовать вычислительную технику в работе, не занимаясь программированием, а применяя готовые ППП (работа в непроцедурном режиме);

умения и навыки программирования, приобретенные неспециалистами в области вычислительной техники, необходимые им для работы;

умения и навыки, приобретенные специалистами в области программирования и ВТ и связанные с вновь появляющимися программными и техническими средствами.

Для указанной категории обучаемых (бухгалтеры, плановики, снабженцы, работники отдела кадров) была разработана 100-часовая программа обучения (в том числе 32 ч — начальный курс), включающая знакомство с ЭВМ, некоторые необходимые действия и команды, а также четыре прикладные программы: «Текст», «Экран», «Таблица» и «СУБД», с помощью которых автоматизируется информационная работа и существенно повышается производительность труда в сфере административной деятельности.

Вообще говоря, в изучаемые прикладные программы включены мощные диалоговые средства («меню-подсказки» и диалоги человека с машиной), позволяющие существенно облегчить понимание программы в сравнении, скажем, с сопровождающей ее технической документацией. Однако, как показывает опыт, эти диалоговые средства недостаточно просты для усвоения прикладных программ пользователем — непрофессионалом в области информатики и вычислительной техники.

Именно поэтому возникла необходимость разработки методических рекомендаций как «посредников» между человеком и ЭВМ (прикладными программами), когда в процессе обучения (за исключением его начальной стадии) отсутствует преподаватель, к которому в любой момент можно было бы обратиться за помощью. Более того, как нам представляется, обучаемый (в нашей ситуации) должен обходиться без «живого» посредника между собой и машиной (прикладной программой). В этом случае как раз и необходимы методические материалы.

Всего подготовлено 16 методических рекомендаций по ознакомлению с работой ЭВМ типа СМ 1800 и «Роботрон-1715». В настоящее время описываемая в этой работе методология и методика обучения используется для разработки компьютерных обучающих систем на базе 16-разрядной техники.

Обучение пользователей включает два этапа: освоение начального и полного курсов. Это позволяет реализовать принцип постепенности в обучении и учесть психологические особенности взрослого, которому с самого начала все должно быть понятно, иначе он не станет применять предлагаемые средства.

Соответствующим образом строятся и методические рекомендации.

Полные курсы предназначены для тех, кто после первоначального ознакомления (уже позволяющего использовать начальные возможности программ), приступит к применению программ с учетом их тонкостей и самых широких возможностей. Результаты эксперимента дают основание полагать, что полные курсы следует реализовывать в процессе создания приложений на автоматизированном рабочем месте.

Полный курс обучения созданию и ведению баз данных — единственный, который связан с обучением программированию. В этом курсе, являющемся наиболее сложным для восприятия конечными пользователями, использованы элементы игры. Ситуации, на фоне которых вводились новые понятия, отбирались так, чтобы трактовать их можно было однозначно, а задачи, при решении которых использовались эти понятия, естественным образом вытекали из этих ситуаций.

При обучении непрофессиональных пользователей программированию в среде СУБД оказалась удобной ситуация под условным наименованием «Дом». Информация о жителях этого «Дома», отражающая происходящие в их жизни события, хранится в

файлах базы данных, а команды языка обработки данных (ЯОД) СУБД являются средством моделирования этих событий. Переход от игровых к реальным ситуациям несложен, так как функции обработки информации там и здесь аналогичны: например, появление нового «члена семьи» в «Доме» влечет за собой такие же действия по обработке файлов, как и принятие на работу нового сотрудника или поступление продукции на склад. Таким образом, обучение становится независимым от области, в которой работает обучающийся, вследствие чего круг обучающихся может быть очень широким.

Последовательность подачи учебного материала диктуется, быть может, несколько неожиданной аналогией между ЭВМ и музыкальным инструментом. Процесс обучения «игре» на этих инструментах и собственно игра, т. е. использование инструмента в производственной деятельности, имеют много общих черт. Обучение работе с ПЭВМ начинается с изучения «нотной грамоты» (знаков языка программирования и его грамматики) и клавиатуры дисплея. После приобретения навыков работы с клавиатурой изучаются готовые «произведения» — последовательности команд, несложные программы. В результате активного освоения «инструмента» обучающийся получает достаточно знаний и навыков для создания собственных композиций (программ), т. е. достигается цель обучения.

Обучение может быть эффективным только при условии интенсивной индивидуальной практической работы, поэтому материал по освоению СУБД разбит на отрезки — уроки, изучение которых происходит в диалоговом режиме. Каждый урок состоит из двух частей — ознакомления с предлагаемым фрагментом материала и самостоятельной работы по распечатанной после ознакомления инструкции. Темп ознакомления с новым материалом задает сам обучаемый, но на этот этап уходит не более 10—15 мин, основная нагрузка падает на самостоятельную работу. Обучающие программы написаны на ЯОД СУБД, поэтому являются дополнительным источником примеров использования команд.

Проведенные нами эксперименты позволяют выделить следующие компоненты технологии компьютерного обучения работников систем административного управления (САУ):

демонстрация возможностей программ, ознакомительное обучение, целью которого является заинтересовать пользователей;

обучение начальным возможностям программ для того, чтобы пользователи как можно скорее почувствовали полезность изучаемого и научились операторской работе за терминалом;

создание простейших приложений (на основе знания пока еще узких возможностей предлагаемых программ);

то, что можно назвать «инкризингом»,— познание новых возможностей программ, более широких по сравнению с начальными курсами, освоение тонкостей с целью научиться полностью использовать возможности программ;

совершенствование простейших приложений и создание новых, в гораздо большей степени использующих широкие и глубокие возможности прикладных программ.

Целью традиционного обучения является получение учащимся достаточного объема знаний, умений и навыков (так называемые ЗУНы) в некоторой области. В современных условиях этого недостаточно. Целью современного обучения является формирование и поддержание в каждом работнике творческого потенциала, воспитание творческой личности. Таким образом, традиционная триада — ЗУН — становится лишь промежуточной целью процесса творческого обучения.

Современная компьютерная техника является мощнейшим средством реализации такого подхода к обучению. Но, как и всяким сильнодействующим средством, им нужно пользоваться очень осторожно.

При построении курса обучения сотрудников административных служб работе с персональными компьтерами учитывалось то, что среди обучаемых могут встретиться лица, уже знакомые с вычислительной техникой, но составившие неверное представление о ее или своих возможностях. Причиной появления негативного отношения к компьютерам, даже при наличии положительной мотивации, может быть отсутствие руководителя либо плохое руководство обучением; попытка самостоятельно, без соответствующей подготовки, решать слишком сложные задачи; использование компьютера в системе с традиционной технологией обработки информации и т. п.

Объективно существует «несовместимость» человека с компьютером, которая вырабатывается в результате неблагоприятного психологического климата в период обучения, или связана с особенностями индивидуального восприятия режима работы с дисплеем (утомление зрения, раздражающе высокий темп работы, боязнь ошибиться).

Непрофессионалы в области информатики и вычислительной техники плохо представляют себе процесс обработки информации компьютером, вследствие чего у них разви-

вается либо недоверие к результатам работы ЭВМ, либо убежденность в абсолютной непогрешимости однажды проверенных программ.

Таковы причины появления психологического барьера между человеком и ЭВМ.

При разработке технологии компьютерного обучения был взят за основу антропоцентристский, а не машиноцентристский подход, что позволило избежать многих отрицательных моментов.

В процессе обучения реализована так называемая мониторная система (называемая также белл-ланкастерской), заключающаяся в том, что большинство пользователей, обучившихся прикладным программам, либо учат, либо хотят этому научить своих товарищей по работе. Таким образом, они превращаются, по выражению Дж. Мартина в «адаптантов», за которыми следуют либо отставшие в учебе, либо не приступившие к ней.

Теоретическое и практическое «умения» в процессе обучения сливаются на стадии начального обучения, навыки приобретаются в процессе решения задач на АРМ при освоении полного курса.

Как известно, конторская деятельность строится на основе жесткого разделения функций и отдельных операций. Такое разделение труда существует внутри служб и даже «внутри» отдельных технологических цепочек. Внедрение компьютерной технологии дает возможность перейти к групповой деятельности в сфере конторского труда.

Индивидуальное обучение не приведет к индивидуальной работе. Во-первых, а таких системах, как САУ, это принципиально невозможно, а во-вторых, на последующих уровнях компьютерного обучения предполагается с помощью управленческих игр приобщать обучаемых к сотрудничеству в ходе совместного выполнения заданий.

Сегодня все более актуальной становится интеграция человека и компьютера, создание своеобразного «кентавра», в котором функции человека и машины были бы распределены, но не разделены.

Компьютерная дидактика предполагает подстройку целей, субъектов обучения, дидактических и организационных форм. Здесь цели уже не могут формулироваться столь же аморфно и нечетко, как в традиционной системе, они должны быть определенными, чтобы к ним можно было применить введенное В. П. Беспалько понятие диагностичности, т. е. однозначности и проверяемости. Более того, при компьютерном обучении цели зачастую формулируются объектами обучения, и это влечет за собой революционизирующую роль компьютерного обучения.

Организационные и дидактические формы должны быть более свободными и гибкими, чем классно-урочные, квалификация педагогов значительно более высокой, чем в традиционной системе (даже только «высокий класс» субъекта обучения повышает эффективность обучения).

Необходима лучшая и психологическая, и практическая подготовка обучаемых к самостоятельной работе. Содержание обучения должно быть четко структурированным и точно дозированным. Наконец, иными, по сравнению с традиционным обучением, представляются средства обучения.

Но главное — изменяются методы обучения. Мы на новом качественном уровне возвращаемся к сократовскому методу, требующему от обучаемого непрерывной активности, дающему непрерывную оценку ответу, заставляющему исправлять допущенные ошибки, поддерживающему логические связи между отдельными шагами на пути от незнания к знанию, сохраняющему подходящий темп работы, научающему мыслить самостоятельно и критически.

Есть основания полагать, что формирование компьютерной дидактики в дальнейшем будет связано с тремя направлениями компьютерного обучения: обучением компьютерам, обучением с помощью компьютеров, обучением компьютеров (имеется в виду создание и использование баз знаний, экспертных систем).

Три основных вида деятельности сменяют друг друга в онтогенезе: игра, учение, труд. Компьютер дает возможность интегрировать эти три вида деятельности, увязав их в систему.

К существующим hardware и software, формирующемуся brainware, несомненно, скоро присоединятся studyware и leanware. Все это будет объединено orgware и образует костяк компьютерной дидактики.

Адрес для справок: 252004, Киев, ул. Красноармейская, 23-б, НПО «Горсистемотехника», отдел 154, Карапетяну Рафаилу Мкртычевичу.

ЧТО МОЖЕТ ЭВМ?

Олимпийский диск

До Олимпийских игр остается еще три года, но в Барселоне в рамках культурной программы уже готовится Олимпийский банк электронных изображений. Изображения — неподвижные картинки и фрагменты видеозаписей — предоставят Международный Олимпийский комитет, а также испанские телекомпании. В результате получится интерактивная база данных, в которой будет собрана всевозможная информация о рекордах и рекордсменах, о победителях Олимпиады и их спортивных достижениях. Диск будет работать под управлением микрокомпьютерной системы, которая позволит останавливать изображение и рассматривать его произвольные фрагменты с большим увеличением, осуществлять замедленное и покадровое воспроизведение и проделывать с изображением другие манипуляции.

Электронная фототека

Прямое изображение — таков буквальный перевод названия проекта, над осуществлением которого работает сегодня французское совместное предприятие с одноименным названием.

Рассказывая о новом проекте, нельзя не сказать о причинах, которые предопределили его разработку. Французское пресс-фотоагентство «Кипа» ежегодно производит около 2 тыс. фотографий. Лишь 200 из них в итоге становятся достоянием прессы или телевидения. Дело в том, что даже в Париже, для того чтобы фотография совершила путь из фотолаборатории до клиента, требуется 2 ч. Для других городов задержка увеличивается до 24 ч или даже более. Еще одна причина — достаточно высокая стоимость производства фотокопий, которые в конечном итоге могут и не понадобиться заказчику.

Новая система позволит сделать любую фотографию доступной для каждого клиента уже с того момента, как она будет скопирована в банк электронных изображений. Система состоит из трех основных элементов: хранилища фотооригиналов, банка данных и устройства ввода—вывода; сердца системы — «Месседжера»; рабочих станций на базе персональных компьютеров, установленных у клиентов.

Часть системы размещается непосредственно в агентстве. Это, прежде всего, модуль для оцифровки фотоизображений, сжатия цифровой информации и дополнения ее необходимыми справочными данными. Ежедневно полученные снимки направляются в банк изображений, объем которого непрерывно увеличивается. Банк устроен на основе обычных оптических дисков и стандартного программного обеспечения для записи и воспроизведения. Еще один важный момент — управляющий компьютер, который классифицирует изображения и составляет предметные каталоги.

Каждый день данные о вновь сформированных изображениях по цифровым линиям связи направляются в «Месседжер», который является связующим звеном между фототекой и клиентами.

У каждого из клиентов установлена рабочая станция на основе персонального компьютера, в памяти которого могут накапливаться изображения. Чтобы получить фото, достаточно набрать тематический запрос на клавиатуре компьютера. Запрос по линии связи направляется в «Месседжер», который подбирает в архиве подходящие по тематике изображения и пересылает их в память рабочей станции. На экране подобранные изображения высвечиваются порциями одновременно по девять картинок небольшого формата — этого вполне достаточно, чтобы сделать начальный выбор. Любую из картинок можно вывести во весь размер экрана, обрезать ненужную часть и даже увеличить какой-либо фрагмент. С помощью цветного принтера можно немедленно изготовить копию выбранного фотоснимка небольшого размера, с качеством, вполне достаточным для газетной публикации. При необходимости клиент с помощью рабочей станции отправляет заказ на фотокопию. Заказ через «Месседжер» поступает в агентство, откуда немедленно высылают соответствующую фотокопию.

Создание новой системы во многом определяется наличием так называемых цифровых интегрированных сетей связи, которые позволяют передать данные с высоким качеством и очень быстро. При этом стоимость передачи данных оказывается весьма небольшой. Сжатое изображение проходит по таким линиям связи со скоростью 64К бит/с, т. е. на передачу одной картинки требуется всего около 7 с. Загрузка в систему всех 200 фотографий, которые ежедневно предоставляет «Кипа», занимает всего 3 мин.

Автомобиль без руля

Компьютер «Линус» без натяжки можно сравнить с автомобилем, у которого нет руля. У «Линуса» нет клавиатуры. Но это обстоятельство не мешает его разработчикам успешно рекламировать и сбывать свою продукцию.

Выглядит «Линус» несколько необычно: плоская коробка с жидкокристаллическим дисплеем вместо верхней крышки. Поверх дисплея расположен прозрачный дигитайзер, преобразующий движение карандаша по его поверхности в «электронную линию», которая немедленно появляется на поверхности дисплея. Именно таким образом и вводится вся текстовая, числовая и графическая информация в этот компьютер. Затем программа распознавания символов преобразует начертанный вами текст в стандартный текстовый файл и выводит его на дисплей. Теперь этот файл можно окончательно отредактировать и записать на гибкий диск.

Такой компьютер удобен для тех, кто не любит печатать на машинке или желает снабдить поля своего текстового произведения рисунками. Но особенно хорош он для врачей, юристов и менеджеров, использующих стандартные бланки, на которых к тому же необходимо сохранить подписи.

Для тех, кто все-таки не верит в способность «Линуса» разобраться в любом почерке, дополнительно поставляется стандартная клавиатура.

ИНТЕРВЬЮ НОМЕРА

Нужны смелые идеи, инициативные люди

(интервью с проректором по научной работе Московского областного педагогического института им. Н. К. Крупской Я. А. Ваграменко)

Кор.: Ярослав Андреевич, расскажите, пожалуйста, о вашем институте, о том, когда вы начали вводить информатику в учебный процесс.

Я. А. Ваграменко: МОПИ им. Н. К. Крупской недавно отметил свое пятидесятилетие. Он имеет свои давние традиции. В первые годы Советской власти это был педагогический техникум, работу которого курировала Н. К. Крупская. Теперь, я считаю, наш пединститут — один из ведущих по значимости и по численности в РСФСР (на дневном отделении учится 4 тыс. человек, а численность всего коллектива — около 10 тыс. человек). У нас много работы, связанной не только с обучением студентов, но и с координацией деятельности по созданию методического обеспечения в республике.

Скоро часть наших факультетов, а точнее — половина, переедет в новое здание, и это решит многие наши проблемы.

Развивать изучение информатики мы начали еще в 1983 г. На индустриально-педагогическом факультете мы ввели новый учебный план (в то время экспериментальный, теперь по нему учатся и в других педвузах страны) и начали систематическое обучение информатике. После этого включился в новую работу физико-математический факультет, который начал обучение по специальности «Учитель информатики» (в этом году у нас будет первый выпуск). Сегодня в значительной мере компьютеризирован процесс обучения на факультете иностранных языков, на многих кафедрах установлено специальное оборудование.

Таким образом, на факультетах, которые призваны выпускать специалистов в области информатики, мы ведем эту работу уже шесть лет.

Кор.: Какова оснащенность вашего вуза вычислительной техникой?

Я. А. Ваграменко: У нас есть различные типы вычислительной техники (кроме больших машин). Много классов «Агатов», «Ямах», «Корветов», ДВК, БК-0010, «Ириш», «Роботронов», есть «Электроники-85», машины СМ всех поколений. У нас ведется работа по совместной договорной тематике с Институтом космических исследований АН СССР и Институтом высоких температур АН СССР. Для решения задач большой размерности, связанных с математическим моделированием, мы арендуем большие машины типа ЕС-1065. Но это там, где требуются очень большие ресурсы памяти. А в целом в учебном процессе используются персональные компьютеры.

Кор.: Ярослав Андреевич, у вас в кабинете стоит «Электроника-85», это дань моде или вы действительно ею пользуетесь в работе?

Я. А. Ваграменко: Конечно, пользуюсь (включает машину). Она работает в диалоговом режиме. Мне довольно часто приходится обращаться к ней. Но я делюсь машинным временем и с другими пользователями.

Кор.: По каким направлениям, связанным с информатикой, ведется работа в МОПИ?

Я. А. Ваграменко: С самого начала мы развивали два направления. Одно — постановка курса информатики для преподавания в объеме, охватываемом учебником. И второе, более прагматичное направле-

ние — изучение информатики и вычислительной техники с целью профориентации и подготовки молодежи к работе в новых условиях автоматизации производства. Поэтому физико-математический факультет имеет уклон в сторону изучения информатики как таковой, в сторону вычислений, применения обучающих программ в области математики, физики и т. д., а индустриально-педагогический учитывает потребности в подготовке кадров для промышленности.

Студенты этого факультета изучают эксплуатацию вычислительных машин, автоматизированные системы, робототехнику. Если брать пединституты, такая работа началась именно у нас. Информатика здесь имеет уклон в трудовое обучение. Мы создали комплекс программ для подобных факультетов всех педвузов, соответствующим образом оборудовали учебные классы, лаборатории робототехники, в которых можно изучать и элементную базу роботов, и построение автоматизированных производств — все то, что связано с программируемой техникой. В этом своеобразие нашего института.

Когда мы довели студентов, обучающихся по новым специальностям до IV—V курсов, стало ясно, что новое направление надо подкреплять обучающими программами и в школе внедрять во все учебные программы применение компьютера.

Кор.: Я знаю, что у вас есть необычная подшефная школа. Где она находится, в чем ее своеобразие, кто в ней работает?

Я. А. Ваграменко: Мы ведем рассчитанный на много лет, широкомасштабный эксперимент. Хотим с первого до последнего класса опробовать компьютер в обучении и воспитании. Хотим также понять, когда начинать профессиональное обучение и профориентационную работу, искать ребят, которых в дальнейшем можно будет обучать специальностям «оператор ЭВМ», «программист», «наладчик автоматизированных систем» и т. д. С каким контингентом учащихся работать в этом направлении, в какие разделы учебных предметов вкраплять компьютерные фрагменты? Эти задачи мы отрабатываем в школе № 15 г. Химки. Мысль о создании специальной школы возникла еще два года назад. Мы подготовили «десант» своих выпускников, они и составили основу педагогического коллектива новой школы. Организовали школу в новостройке, где в 1988 г. был первый набор учеников. Учительский коллектив (75 % педагогов — начинающие) имеет свое лицо: все здесь ориентированы на компьютеризацию учебного процесса.

Для того чтобы начать задуманную работу, мы укомплектовали школу техникой: для начала ввели два класса «Ямах» и два класса отечественной вычислительной техники. Уже можно работать с применением компьютеров по 4—5 предметам. В перспективе предусмотрено, что 25 % учебного времени по каждому предмету будет отводиться на компьютерное обучение. Представляете, сколько нужно разных программ для осуществления нашего замысла?!

Для того чтобы воплотить в жизнь намеченное, придется проделать огромную работу. Трудностей много. Сейчас каждый день из подшефной школы — звонки: мы «это» уже прошли, давайте новые программы.

Кор.: Получается, что у вас специальная компьютерная школа по типу английских спецшкол?

Я. А. Ваграменко: Я бы так не сказал. Просто вместе с книгой (а не вместо нее), вместе с учебником средством обучения становится компьютер.

Кор.: А кто же координирует всю работу в школе? И кто помогает в этом?

Я. А. Ваграменко: Директор школы — опытный педагог Светлана Михайловна Моисеенко. В этой школе есть заместитель директора по компьютеризации, хороший специалист Тамара Павловна Пудова. Кроме того, десять методистов от наших основных кафедр постоянно работают в подшефной школе. Еще одна особенность: когда идут занятия в компьютерном классе количество учеников делится пополам, на две группы. Значит, нужно больше и преподавателей. Как мы вышли из положения? Мы пригласили студентов IV и V курсов.

Робот «Универсал» — помощник преподавателя

Компьютер беспристрастно выводит отметку... преподавателю

Для них это и педагогическая, и производственная практика. Они работают по школьному расписанию параллельно со штатными учителями, а в институте у них индивидуальный учебный план. «Тыл» обеспечиваем мы, разрабатывая программы и методические указания. Для этого созданы специальные группы: есть коллектив профессиональных программистов и студенческая лаборатория программных разработок. Нам как воздух нужны новые идеи, инициативные специалисты — работы непочатый край.

Каждой кафедре в том или ином объеме мы планируем работу для этой школы. Целевая программа рассчитана до 1995 г.

Для всех предметов, по всем темам и разделам будем создавать пакеты прикладных программ. Конечно, мы отдаем себе отчет в том, что учителя компьютер не заменит, мы к этому и не стремимся.

К концу запланированного срока весь учебный процесс будет вестись по новым информационным технологиям.

А сначала будем продолжать оборудовать классы, кабинеты, эта работа, как я уже говорил, развернута, будем применять компьютеры в кружковой работе. Следующим этапом будет внедрение компьютеров в производительный труд, перевод школьной документации на «программные рельсы».

Сейчас мы используем все лучшее, что есть в программном обеспечении страны применительно к школе: что-то попросили у коллег в Москве, что-то в Переславле-Залесском, Ленинграде, Новосибирске, но половина средств — наши, «мопийские».

Кор.: А что делается в институте по линии более углубленной подготовки кадров?

Я. А. Ваграменко: У нас действует аспирантура по специальностям «Применение вычислительной техники, математическое моделирование и математические методы в научных исследованиях», «Методика преподавания информатики». В институте работает факультет повышения квалификации специалистов, работающих в педвузах в области информатики. Они изучают у нас такие дисциплины, как «Операционные системы», «Сети ЭВМ», «Автоматизированные обучающие системы», «Параллельные вычисления». К нам приезжают преподаватели из педвузов РСФСР. У нас ожидается создание специализированного совета для защиты диссертаций по методике преподавания информатики.

Кор.: Как еще используется компьютер в МОПИ?

Я. А Ваграменко: Интересное направление нашей работы — создание банка психолого-педагогических данных. Информационная система выполнена на ДВК-2 и комплексе «Мера». В чем цель накопления машинно-ориентированной информации в области педагогики? В банке собраны в концентрированном виде сведения об организации перестройки в школе, решении трудных воспитательных задач, опыте педагогов-новаторов и т. д.

Система выдает не только названия, но и краткие научные сообщения о работе, методе. С ней сможет работать даже неподготовленный пользователь. Вся информация в системе состоит из отдельных модулей. Можно по первому требованию получить твердую копию. Пользуясь персональными ЭВМ, можно будет «выуживать» необходимую информацию на дискеты. Можно будет по запросам пользователей высылать необходимые учителям и научным работникам данные.

Кор.: Вы были научным руководителем этой работы. А кто еще создавал эту систему?

Я. А. Ваграменко: В создании системы

На снимке АРМ педагога к банку психолого-педагогических данных

принимали участие коллективы кафедр информатики и педагогики. Разработкой информационной системы руководил профессор В. И. Журавлев. Работали и студенты, и аспиранты, участвовали сотрудники других учебных заведений.

Заключительную оценку проводили с привлечением специалистов со стороны — из Московского городского комитета по народному образованию, с факультетов повышения квалификации, учителей, студентов.

Эта работа закончена и была представлена на недавней выставке к юбилею Н. К. Крупской на ВДНХ СССР.

Еще один пример использования ЭВМ — компьютеризированная система анкетирования студентов. Для нас очень важно знать их мнение о своих преподавателях. Учитываются мнения студентов по 18 параметрам, выводится средняя оценка, которая принимается во внимание при аттестации, назначении на должность. Оценка выставляется по пятибалльной системе. Поскольку этот вопрос достаточно деликатный, твердая копия (листинг) выдается только заведующему кафедрой. Для работы этой системы создано необходимое программное обеспечение. Существует график, и каждая группа в назначенное время проходит анкетирование. Если кого-то заинтересует наша система, можно обращаться по тел.: 261-82-59.

Ну и конечно, автоматизирована обработка информации по экономическим, кадровым, научным, организационным аспектам деятельности института.

На индустриально-педагогическом факультете зачет по информатике закончился содержательной беседой о вычислительной технике

Кор.: Я вижу у вас на столе учебник, о котором, по-моему, наши читатели еще не знают. Не могли бы вы сказать несколько слов о нем?

Я. А. Ваграменко: Этот учебник написал авторский коллектив, в который кроме меня входили И. Антипов, Э. Кузнецов, В. Дрига, О. Дорошкевич, А. Маслов. Он создан для учеников VI класса, поскольку, по нашему мнению, начинать обучение информатике необходимо не в IX—X, а в VI—VII классах. Книга имеет более прагматичный характер, чем существующие учебники. В ней больше собственно информатики, а не программирования.

Кор.: Большое спасибо за беседу.

Интервью провела
Т. ДРАГНЫШ

Меньше ракет — больше компьютеров

ЧТО МОЖЕТ ЭВМ

Одним из существенных обстоятельств, позволивших заключить договор о ликвидации ракет средней и меньшей дальности, стал высокий уровень развития национальных средств контроля, основным элементом которых являются разведывательные спутники. Однако полную ликвидацию ракет определенного класса контролировать гораздо проще, чем некоторое уменьшение их числа, поэтому заключение договора о 50-процентном сокращении стратегических наступательных вооружений резко повысит требования к разведывательным средствам. Уже сейчас сенат США решил добавить 6 новых спутников Lacrosse к существующим.

Но получить информацию — это только полдела. Из всего объема добываемых данных нужно выделить полезные, а это, по выражению одного из специалистов по разработке систем контроля, «не легче, чем напиться из пожарного брандспойта». Тут и должны включиться в работу ЭВМ: для анализа будущего информационного потока разрабатываются специальные системы искусственного интеллекта, а также изучается возможность использования нейрокомпьютеров.

ЧТО МОЖЕТ ЭВМ?

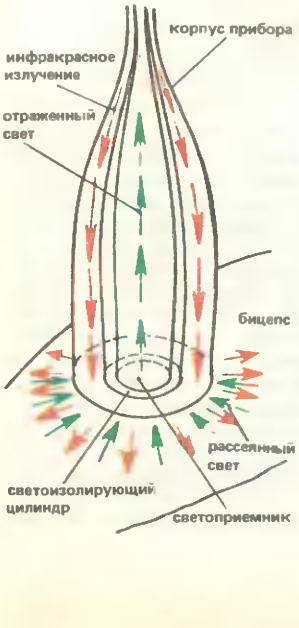
Точно и правдиво

Известно множество формул для расчета нормального веса человека, учитывающих множество параметров и позволяющих сделать быструю оценку состояния пациента. Кроме того, существуют профессионалы — врачи, портные, члены жюри конкурсов красоты,— опыт которых позволяет порой получить и более точные результаты при определении избыточного веса человека.

Однако ни формулы, ни профессионалы, вооруженные ими, не смогут соперничать с прибором, разработанным недавно американскими медиками. В основе его действия лежит тот факт, что подкожный жир наиболее активно поглощает свет с определенной длиной волны.

Прибор устроен следующим образом: от источника инфракрасного излучения свет поступает в цилиндрический корпус, прижатый открытым концом к бицепсу пациента. Отраженный подкожными тканями свет поступает на светоприемник, расположенный в центре корпуса в светоизолирующем цилиндре и далее — в компьютер для анализа спектра.

Помимо отраженного света в ЭВМ вводятся рост, вес и другие антропологические параметры пациента, после чего на основе этих данных компьютер вычисляет и распечатывает процентное отношение веса подкожного жира к общему весу человека, указывает нормальный и максимальный вес и, если это необходимо, предлагает специальную диету и комплекс физических упражнений.



Работа с прибором не требует специальной квалификации. Его разработчики считают, что массовое распространение прибора сделает американцев более подтянутыми и жизнерадостными.

Неутомимый помощник

«Бурное развитие вычислительной техники...» Звучит зуммер...

Примерно так закончится ваша попытка напечатать расхожий штамп на американской пишущей машинке, в которую встроен текстовый процессор.

Помимо предупреждения о попытке напечатать ранее использованную фразу встроенный компьютер предостережет вас от использования противоречивых слов и от синтаксических ошибок. Если же вы захотите более серьезной помощи, ваша пишущая машинка начнет следить за пунктуацией, подбирать определения и подсказывать синонимы из своего словаря для чрезмерно часто употребляемых слов.

Но если и после всего этого вы умудрились наделать ошибок, машинка предоставит вам возможность отредактировать текст в ее памяти. И только потом напечатает его на бумаге.

Дантист и компьютер

У стоматологов иногда возникает необходимость в определении формы прикуса зубов пациента. Для этого обычно применяется кусочек копировальной бумаги или воска. Но получившуюся картину неопытному врачу бывает порой трудно разобрать и совсем невозможно зафиксировать.

Справедливо решив, что всю малоприятную и трудоемкую работу должен делать компьютер, американский дантист В. Манесс изобрел датчик, который не только кодирует информацию о прикусе и записывает ее на диск компьютера, но и демонстрирует на его экране последовательность смыкания зубов и даже усилие их сжатия.

Датчик представляет из себя пластиковую пластину, внутри которой расположена серебряная сеточка. Сжимая датчик зубами, пациент уменьшает расстояние между ее проволочками, а компьютер по соответствующей программе отслеживает и обрабатывает поступающую с него информацию.

Теперь, если дантисту понадобится внимательно рассмотреть или обсудить с коллегами какую-нибудь фазу сжатия зубов пациента, он может обратиться к памяти компьютера. У самого пациента, рот которого свободен, можно уточнить кое-какие детали.

Кредит или дебет?

Новомодные кредитные карточки со встроенным микропроцессором производят отличное впечатление на демонстрационных показах, но в практику внедряются медленно. Причин несколько, например разногласия по поводу стандартов на них, возникшие между крупными кредитными фирмами Visa и MasterCard. Да и старые карточки с магнитной полоской не так уж плохи, хотя и менее защищены от подделок и махинаций; ущерб, наносимый банкам жуликами, невелик — около доллара в год на каждую карточку.

Большая память микропроцессорной карточки (2000—8000 знаков) чрезмерно хороша: неясно, чем ее заполнить. Ведь даже на магнитной карточке, содержащей номер счета, имя, адрес владельца и предпочтительный для него класс авиабилетов, остается резервное место для примерно 100 знаков!

Специалисты полагают, что новые карточки «найдут себя», если выступят в качестве дебетных, которые отличаются от кредитных тем, что за них владелец авансом платит крупную сумму. Она записывается в память «электронного кошелька» и постепенно уменьшается в процессе оплаты товаров. В таком варианте устойчивость против попыток подделки становится существенным достоинством.

Программы для «Корвета» и «Ямахи»

Кооператив «Перспектива»

предлагает пакет «Информатика-89» для КУВТ «Корвет» и «Ямаха-2».
Пакет включает 16 программ (поддерживающих все разделы курса ОИВТ), методическое пособие для учителя по организации занятий и раздаточный материал (описание работ) для учащихся.
Учитель получает максимум удобств при подготовке уроков и свободу творчества.

В составе пакета:

- набор демонстрационных программ по теме «Возможности ЭВМ»;
- клавиатурный тренажер с автоматическим сбором по сети результатов работы;
- конструктор информационной сети, позволяющий моделировать в классе сеть любого типа (звезда, кольцо, линия и др.) и изучать на ней принципы кодирования, передачи и приема информации (в форме телеконференций, учебных игр, экспериментов с сетью).

Имеется система исполнителей, построенная в едином методическом ключе и дающая возможность наглядно и доступно ввести основные понятия курса (задача, исполнитель, алгоритм, структура алгоритма и т. д.):

исполнитель ВЫЧИСЛИТЕЛЬ; исполнитель КОДИРОВЩИК; исполнитель ЧЕРТЕЖНИК; исполнитель ХРАНИТЕЛЬ; исполнитель ЛОГИК; исполнитель ГЕОМЕТР; исполнитель РОБОТ.

Изучать раздел «Устройство ЭВМ» поможет структурная модель ЭВМ, позволяющая демонстрировать и исследовать информационное взаимодействие блоков ЭВМ.
Тема же «Кодирование информации в ЭВМ» может быть раскрыта с помощью исполнителя КОДИРОВЩИК.

Применение ЭВМ иллюстрируют четыре программы:

- учебная справочная система, первоначально загруженная словарем основных понятий курса;
- электронные таблицы, допускающие не только арифметическую, но и логическую обработку содержимого ячеек;
- учебный текстовый редактор — он позволяет ученикам работать в локальной сети, обмениваясь фрагментами текста как с учителем, так и с другими учениками;
- графический редактор, обеспечивающий построение рисунков (передаваемых через сеть и сохраняемых на диске) в режиме графического конструктора.

Любая из программ, входящих в пакет, может быть разослана на ученические компьютеры с помощью монитора сети, результаты работы сохраняются на диске для использования на последующих уроках. Все программы имеют простой унифицированный интерфейс, существенно облегчающий работу учащихся с ними.
Методические материалы, сопровождающие пакет, представляют собой новейшие разработки, еще не нашедшие отражения в периодической печати и широко использующие перспективные формы организации занятий: учебная игра, самостоятельный эксперимент и др.
Пакет прошел практическую проверку в ряде школ Москвы.
В его разработке принимали участие как учителя-практики, так и методисты, были учтены замечания и предложения школьников.
Весь комплект стоит 780 рублей. Для желающих организуются курсы по освоению методики использования пакета на уроках.
Кооператив также продает пакет программ «АСУ-школа», реализованных на ЕС ЭВМ и «Ямахе», включающий: «Систему внутришкольного контроля», «Анализ эффективности урока», «Систему психологического тестирования», «Составление расписания».

Заказы направлять по адресу:
127000, Москва, Б. Переяславская, д. 1. Тел. 280-66-74.
Кооператив «Перспектива»

В творческом поиске

В первом номере за 1989 г. журнал «Народное образование» поместил начало документальной повести А. Вольфа «Всей своей жизнью» — о судьбе бывшей сельской учительницы Анастасии Вшивцевой, легендарной разведчицы, удостоенной многих боевых наград.

Регулярные подписчики «Народного образования» знают, что подобные материалы раньше в нем не печатались. Обращение к новому жанру не единичная попытка преодолеть консервативные традиции. Это лишь один из аспектов коренного обновления журнала, поставившего перед собой задачу быть ближе к читателю, сделать журнал энергичнее, содержательнее, не отступая от его принципиальных позиций.

И первые номера «Народного образования» за этот год — тому подтверждение. Журнал изменился: удобнее стал его формат, более современным художественное и полиграфическое оформление. Наконец-то на его страницах заиграли краски, появились по-настоящему интересные фотоиллюстрации, клишированные заголовки. А сколько новых имен, авторов статей, корреспонденций и очерков...

Дело, конечно, не только в смене «одежек». Главное — появление на страницах журнала новых рубрик, отражающих характер обновления тематики журнала. Среди них — «Демократизация: от школьного класса до министерского кабинета», «Средняя школа — стратегия развития», «Наука и практика», «Грани новаторского поиска» и некоторые другие. Особо хочется отметить рубрику «Информатика — ЭВМ — школа», близкую тематике ИНФО, введение которой свидетельствует о значимости набирающего силу процесса информатизации образования и общества в целом.

Ведущей, приоритетной темой журнала становится тема демократизации народного образования — как принципиальной основы его перестройки, начиная с системы управления и кончая отношениями в главном рабочем звене: учитель — ученик.

Другие важнейшие темы — теоретический и практический поиск новых путей и моделей советской средней и высшей школы, стратегических направлений ее развития, широкий аналитический показ передового педагогического опыта. В распоряжение творчески работающих учителей и воспитателей журнал предоставляет блок новых «технологических рубрик»: «Учитель — учителю: поделимся опытом», «Панорама методических идей», «По ступенькам мастерства», «Факультатив для учителя». Разговор с педагогами поведут специалисты и ученые — психологи, социологи, экологи, представители других наук. Рубрики эти помогут педагогам вести живой обмен мнениями, делиться нестандартными методическими находками, вести коллективный поиск новых путей совершенствования учебно-воспитательного процесса.

Начиная с первых номеров нынешнего года журнал «Народное образование» обстоятельно знакомит читателей с перспективными научно-педагогическими исследованиями, с разнообразной экспериментальной работой, которая разворачивается в прямой связи с обновлением структуры и содержания образования. Почти в каждом номере журнала — публикации о жизни современной зарубежной школы, о новых течениях мировой педагогической науки и практики.

Еще один пласт новых журнальных материалов расскажет о жизни «как она есть» — об учительском быте, житейских проблемах и увлечениях (рубрики «Как живется, коллеги?», «Ничто человеческое...», «Советуем вам», «Подумаем, помечтаем», «Ваше мнение», «Спасибо вам, учителя»).

...Первые номера «Народного образования» в руках у читателей. О них говорят, спорят, их коллективно обсуждают, а порой и ругают. Что ж, это естественно. Новое всегда с трудом пробивает себе дорогу. Но, как говорят, в споре рождается истина. Сегодня очень нужен плюрализм мнений, широкий спектр суждений. Только в этом случае можно выбрать точный курс движения вперед. К этому и стремится коллектив редакции журнала «Народное образование», головного издания педагогической печати, недавно отметившего свое семидесятилетие. Успеха вам, коллеги!

Нам остается добавить, что выписать «Народное образование» можно с любого месяца. Подписка принимается без ограничений всеми отделениями связи и органами «Союзпечати». Подписная цена на 6 месяцев — 3 руб. 60 коп.

Творчество наших читателей

Сотрудники Винницкого пединститута проводят эксперимент в СШ № 1. Здесь в кабинете информатики обучаются по специальной программе шестилетки.
Фото *А. Кузьмичева, Г. Кочеткова*, г. Винница

В селе Яблоневка Стрыйского района Львовской области праздник — в гости приехал львовский клуб «Информатика».
Фото *С. Горевалова*, г. Львов

Экспертные системы в обучении

ШКОЛА-СЕМИНАР «ЭКСПЕРТНЫЕ СИСТЕМЫ И ПРОЛОГ В УЧЕБНОМ ПРОЦЕССЕ»

Внедрение экспертных систем и Пролога в обучение в СССР было рассмотрено на школе-семинаре «Экспертные системы и Пролог в учебном процессе», которая проходила с 30 января по 3 февраля 1989 г. на живописном берегу Волги в окрестностях Йошкар-Олы. Школа-семинар проводилась по инициативе Марийского областного правления Союза научных и инженерных обществ, инициатива была поддержана ГКНО СССР. В работе школы-семинара приняли участие представители АН СССР, АПН СССР, ряда союзных министерств.

Как стало ясно, экспертные системы могут применяться не только для обучения в вузе или школе, но и для профессионального обучения различных категорий специалистов. В качестве примера можно привести интересный доклад группы авторов из Москвы и Воронежа об экспертной системе, обучающей принципам размещения пожарной сигнализации на объектах народного хозяйства.

На наш взгляд, особое значение имеют работы по разработке дружественного интерфейса экспертной системы. В этом смысле нужно отметить доклад сотрудников Ленинградского института точной механики и оптики, которые занимаются проблемой общения с компьютером на естественном языке. Автор этих строк, будучи в Ленинграде, лично убедился, что компьютер понимает человеческие слова, хотя и ограниченные программой вузовского курса общественных наук.

В выступлениях участников затронут широкий спектр применений Пролога для различных форм и уровней обучения. На школе-семинаре демонстрировались программы для учебных ПЭВМ. Показаны разработанные в СССР трансляторы с языка Пролог, обладающие рядом положительных особенностей: высоким быстродействием, дружественным интерфейсом, малыми размерами требуемой для работы оперативной памяти.

Как показали выступления, к настоящему времени накоплен определенный положительный опыт применения Пролога в гуманитарных вузах как средства осуществления межпредметных связей. Актуально и использование Пролога как первого изучаемого языка, в отличие от Бейсика или Паскаля. Этот опыт свидетельствует о развитии у обучаемых навыков независимого, творческого, логического мышления; он достоин самого широкого распространения, в частности, с помощью журнала «Информатика и образование». В Москве, Переславле-Залесском, Йошкар-Оле используются разнообразные версии Пролога. В связи с этим важной проблемой становится унификация синтаксиса и семантики версий языка Пролог для всех типов ПЭВМ, рекомендованных ГКНО СССР для учебных заведений страны с учетом возможности применения национальных языков.

Школа-семинар закончилась. Она была первой. В ее организации были недостатки: недостаточная реклама, совпадение по времени с другими мероприятиями ГКНО СССР, ограниченные сроки организации — два месяца. И все-таки важно, что она состоялась и был полезный обмен мнениями. Хочется верить, что эта встреча не последняя, мы должны помнить — время не ждет!

С. ГРИГОРЬЕВ
доцент МарГУ,
председатель оргкомитета

Семинар в Москве

В конце марта 1989 г. в Московском доме научно-технической пропаганды им. Ф. Э. Дзержинского прошел семинар «Новые информационные технологии в обучении». Впервые в нашей стране состоялась такого уровня встреча специалистов, работающих над созданием обучающих систем нового поколения.

Направление семинара определила в своем вступительном слове научный руководитель семинара доктор технических наук Л. С. Болотова. Для сокращения сроков и повышения качества обучения необходимо использование новых технологий применения ЭВМ в обучении; использование новых аппаратных и программных средств, разработанных в области искусственного интеллекта, для создания нового поколения обучающих (и учебных) систем, переход к которым подразумевает не просто смену аппаратной и программной базы систем, но смену всей идеологии их разработки. Предполагается, например, что системы нового поколения будут существенно ориентированы на человека-преподавателя. Назначение их — не вытеснить преподавателя из процесса обучения, а освободить его от рутинной части работы, дать ему удобный и мощный инструмент, расширяющий его возможности.

В докладах, прозвучавших за два дня работы семинара, были рассмотрены лишь некоторые, наиболее важные с точки зрения организаторов, аспекты разработки обучающих систем нового поколения. Большинство докладчиков основное внимание уделили одному из классов таких систем — экспертно-обучающим системам (ЭОС) (см., например: *Довгялло А. М., Ющенко Е. Л.* Обучающие системы нового поколения // Управляющие системы и машины. 1988. № 1).

ЭОС способны существенно повысить качество обучения на основе используемых в них экспертных знаний об изучаемой проблемной области (ПО), о методах диагностики ошибок обучаемого и управления процессом обучения и т. д. «Знания» системы представляют собой записанные в специальной форме знания, полученные от людей-экспертов разного профиля: предметников, педагогов, психологов и некоторых других. Отличительной особенностью ЭОС является то, что знания в этих системах отделены от конкретного курса обучения. Это позволяет повысить гибкость и адаптивность процесса обучения и организовать многоцелевое использование экспертных знаний.

В составе ЭОС принято выделять три основных модуля: модуль управления обучением, модуль-эксперт в изучаемой ПО, модуль анализа ошибок обучаемого. Каждый из них в идеальном случае представляет собой самостоятельную систему со своей базой знаний, в которой сосредоточены знания экспертов соответствующего профиля. Кроме того, в состав ЭОС включается общая база знаний, которая используется всеми модулями и помогает им работать согласованно. Важную часть общей базы знаний составляет модель обучаемого, в каждый момент обучения отражающая текущее состояние знаний обучаемого и другую информацию о нем. Наличие этой модели позволяет ЭОС очень точно настраиваться на уровень знаний и потребности конкретного обучаемого.

В общем виде ЭОС поддерживает следующий четырехзвенный цикл обучения: 1) модуль управления обучением предлагает обучаемому очередное учебное воздействие, оптимальное с точки зрения целей обучения и модели обучаемого; 2) если это вопрос, задача и т. д., обучаемый готовит ответ и передает его модулю-эксперту в данной ПО; если это информация, пример решения задачи — они демонстрируются с помощью модуля-эксперта; 3) модуль-эксперт проверяет ответ (сравнивая его, например, с ответом эксперта) и в случае неверного ответа определяет допущенные ошибки; 4) по этим ошибкам модуль анализа ошибок пытается сделать заключение о породивших их неверных или неполных знаниях обучаемого. При этом используется и изменяется модель обучаемого, благодаря чему результаты анализа учитываются на новом цикле обучения.

К сожалению, описанное выше лишь умозрительные построения. Пока еще не удалось соединить в одной системе знания экспертов всех необходимых специальностей, однако в реализации отдельных компонентов ЭОС уже достигнуты серьезные успехи. Учеными разных стран ведутся исследования по диагностике ошибок обучаемого, построению моделей обучаемого и предметной области, организации учебного диалога на естественном языке. Все эти исследования составили новое направление в информатике, получившее название «интеллектуальные обучающие системы». В его рамках за рубежом уже разработан целый ряд обучающих систем, включающих одну-две из перечисленных выше экспертных подсистем. За последние пять лет в Англии и США

вышло более десяти серьезных монографий, посвященных интеллектуальным обучающим системам; в 1988 г. в Канаде состоялась первая международная конференция на эту тему. С 1989 г. в США начал выходить специальный журнал «Искусственный интеллект в обучении».

В Советском Союзе разработка интеллектуальных и экспертно-обучающих систем только развертывается. С обзором состояния работ, ведущихся в этом направлении, на семинаре выступили представители признанных центров разработки ЭОС: А. М. Довгялло (ИК АН УССР), Ю. И. Лобанов (НИИВШ), Л. С. Болотова (МИРЭА). Судя по их сообщениям, важное место в работе этих центров занимает создание инструментальных средств для ускорения и упрощения разработки экспертных и экспертно-обучающих систем. Некоторые из этих средств уже доведены до уровня программного продукта и могут быть поставлены заинтересованным организациям.

Помимо обзорных докладов было сделано несколько сообщений о разработанных прикладных системах, реализующих некоторые компоненты ЭОС. Из них хочется отметить сообщение В. А. Ловицкого (Харьковский институт радиоэлектроники) о системе ДЕСТА, позволяющей формировать прототипы Паскаль-программ по описанию на естественном языке. Система ДЕСТА и ряд других инструментальных и прикладных систем были продемонстрированы участникам семинара.

Среди проблем, стоящих перед разработчиками ЭОС, следует выделить проблемы, касающиеся модели обучаемого. Использование в ЭОС хорошей модели обучаемого и средств работы с ней способно существенно поднять адаптивность системы. За рубежом разработан ряд технологий создания и поддержки модели, однако до сих пор большинство систем с моделью обучаемого являются чисто экспериментальными. В Советском Союзе уже в начале 80-х гг. появились две оригинальные практические системы, основанные на модели обучаемого: система АСОЛИЯ (см.: *Растригин Л. А., Эренштейн М. Х.* Адаптивная система обучения с адаптируемой моделью обучаемого // Кибернетика. 1984. № 1) для обучения лексике иностранного языка и система АСОД (см.: *Раатс Ю. Ю., Толмачева А. Ю.* Адаптивная автоматизированная система обучения дифференцированию // Автоматика и вычислительная техника. 1980. № 3) для обучения дифференцированию. Опыт создания этих систем, безусловно, пригодится разработчикам ЭОС.

Реальными шагами были подкреплены призывы организаторов семинара к сотрудничеству ученых разных специальностей. Несколько докладов было сделано представителями «смежных» дисциплин. Участники услышали о пределах совершенствования аппаратной части ЭВМ, об инновационных играх в обучении, об исследованиях понятийной структуры, формируемой в сознании человека.

Семинар получился очень интересным. Его участники узнали много нового, и можно надеяться, что многие из них включатся в работу по созданию нового поколения учебных систем. Остается поблагодарить организаторов семинара, его руководителя Л. С. Болотову и пожалеть лишь о том, что доклады семинара не будут опубликованы. Впрочем, в самом скором времени должны состояться две конференции, специально посвященные этому вопросу, и не менее трех, на которых проблемы ЭОС включены в программу. Материалы всех конференций должны быть опубликованы. Можно надеяться, что и на страницах журнала «Информатика и образование» будет рассказано о некоторых из них.

П. БРУСИЛОВСКИЙ

Возвращаясь к напечатанному

Периодически редакция получает письма с критикой тех или иных материалов, опубликованных в журнале. В четвертом номере за 1988 г. напечатано критическое выступление С. Коршунова «Имитация с целью симуляции», аргументация которого показалась нам убедительной. Однако ответ критикуемой стороны по ряду пунктов содержал весомые контраргументы, в связи с чем мы должны признать, что заметка С. Коршунова требовала более тщательной оценки и редакции следовало воздержаться от ее публикации.

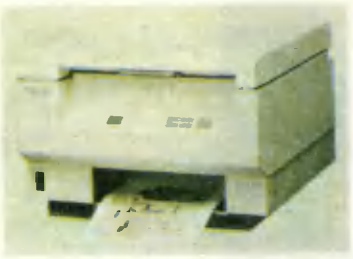
Сейко рисует картинки

Похоже в названии заметки встретились два несовместимых понятия. Группа фирм «Сейко» в нашей стране давно снискала известность своими наручными часами. Но ошибки нет, продукция фирмы «Сейко инструментс» — это и высокопроизводительные цветные принтеры третьего поколения.

Предположим, работая с компьютером, вы создали сложную схему или красивую заставку — цветной экран дисплея и плата графики высокого разрешения сделали свое дело — никого не оставляет равнодушным компьютерная картинка. Если теперь попытаться вывести ее на бумагу, воспользовавшись даже лазерным принтером, схема станет непонятной, а вся привлекательность картинки мгновенно исчезнет. Для того чтобы получать на бумаге точную копию картинки с экрана, создатели новых моделей цветных принтеров воспользовались уже зарекомендовавшим себя методом цветной термопечати с помощью специальной красящей ленты. Изображение, сформированное компьютером, одной командой переносят в оперативную память принтера. Так удается не только сэкономить время высокопроизводительного компьютера. Принтер, работая автономно, позволяет воспроизвести от 1 до 99 копий любого рисунка. Кстати, если снабдить его еще одним блоком — мультиплексором на четыре входных порта, принтер сможет обслуживать сразу четыре компьютера.

Устройство принтера достаточно просто, если говорить о механике. Основной элемент — термоголовка, которая переносит на бумагу красящий состав со специальной ленты, рулон которой заряжен в принтер. Впрочем, если нужно, принтер нанесет изображение и на прозрачную пленку из полиэстера — это удобно, например, для лектора, а также может использоваться в полиграфических целях.

Лист бумаги движется мимо термопечатающей головки параллельно с носителем красок. Крохотные иголочки, из которых набрана термоголовка, нагреваются в соответствии с содержимым оперативного запоминающего устройства принтера и переносят



на бумагу три основных красителя — желтый, красный и синий. Подобно тому как это происходит в дисплее (или на экране телевизора), различные комбинации крохотных цветных точек позволяют получить изображение из семи цветов и 270 000 оттенков. Каждая картинка на листе стандартного машинописного формата (А4) печатается всего за 1 мин. Разрешение, т. е. количество точек на миллиметр, у разных моделей колеблется от 6 до 9,45, а вся картинка состоит, например, из 1871×2343 цветных точек.

Иногда цветное изображение на бумаге не требуется. В этом случае не понадобится другой принтер. Достаточно заменить рулон с красящей лентой.

Если говорить о стоимости, то на выставке «Консумэкспо-89», состоявшейся в Москве в январе 1989 г., принтер можно было приобрести по цене около 8 тыс. долл.

Электронная рулетка

Если вам часто приходится заниматься измерением каких-либо расстояний, площадей или объемов, например обмерами помещений, вашим лучшим помощником станет прибор, созданный на одной из американских фирм. В небольшой коробочке, умещающейся на ладони, размещены ультразвуковой дальномер и калькулятор, ориентированный на обработку результатов измерений и выполняющий несколько арифметических операций. Результаты измерений и вычислений выводятся на жидкокристаллический дисплей в цифровом виде.

Кто кого

Игральные автоматы, прозванные однорукими бандитами, широко распространены на Западе. Суть игры проста — брось мелкую монету и дерни за рычаг. Повезет — получишь крупную сумму, а не повезет — не получишь ничего.

«Везение» назначается генератором случайных чисел — в старых моделях механическим, наподобие рулетки, в современных — электронным, на микропроцессоре.

Если работает микропроцессор, значит, у него есть программа, а если есть программа,— значит, ее можно разгадать. Хакеры занялись этим делом с немалой для себя выгодой — выигрыши посыпались один за другим. Хозяева игорных автоматов заметили это, и началась борьба. Хозяева меняют программы — хакеры их разгадывают.

Впрочем, эти «игры» скоро вышли из сферы информатики. В залах игорных автоматов появились охранники; хакеры ответили введением маскировки и повышением маневренности — частой сменой залов.

Что-то еще будет...

Время — деньги

Все чаще для хранения информации в персональных компьютерах используются компактные герметичные накопители на жестких магнитных дисках — винчестеры. Эти устройства обеспечивают наиболее быстрый доступ к записанным данным и высокую плотность их хранения. Уже сегодня в жесткой коробке объемом не более кубического дециметра умещается до 100М байт информации. И каждые три года плотность ее хранения удваивается, что позволяет постоянно уменьшать размеры накопителя.

В основе прогресса винчестеров лежит постоянное улучшение магнитного покрытия дисков, переход к тонкопленочной композиционной технологии при производстве плавающих головок и смена принципов записи информации на диск. А если добавить к этому улучшение аэродинамики головок, позволившее уменьшить расстояние до поверхности диска, то скорость обмена с компьютером 1,87М байт/с не покажется удивительной.

ЧТО МОЖЕТ ЭВМ?

Веселый урок

Внимание!!!

В данную программу встроен блок с вирусом нового типа. Этот вид вируса не разрушает память, носители, программы и системы, не распространяется и не внедряется в транснациональные компьютерные сети, поэтому владельцы ПЭВМ и пакетов МО могут быть совершенно спокойны, но... данный вирус увеличивает на порядок вредное воздействие компьютера на человека, причем органы чувств его не фиксируют!!! Вирус может быть вставлен в любую программу и будет превращать вас в инвалида теми же средствами, что и обычный компьютер, но только гораздо быстрее.

Автор заявляет о наличии этого вируса в следующих целях:

1) меркантильных — вид авторской защиты (вряд ли кто-нибудь утащит кусок плутония и будет таскать его за пазухой, зная, что он собой представляет). Кстати, вирус можно отключить, задав верный пароль на вход, который вам сообщат при легальной поставке программы. Вероятно, это заинтересует поставщиков ПО;

2) более возвышенных, а именно еще раз обратить внимание общественности на комплекс вредных воздействий, которым постоянно подвергаются профессиональные программисты в процессе своей работы. Необходим коренной пересмотр точки зрения на эту область человеческой деятельности;

3) чтобы подчеркнуть моральную ответственность, которую несут представители этой профессии (ибо если этот вирус соединить с уже известными, распространяющимися и внедряющимися, то можно искалечить половину земного шара, причем она и знать об этом не будет. Это пострашнее, чем вся радиация Чернобыля...).

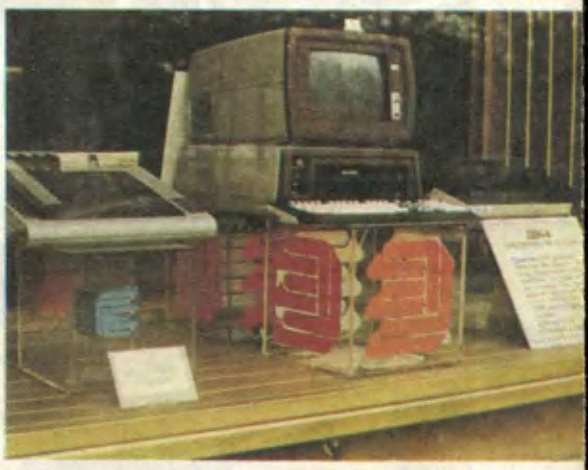
Вышеизложенное представляет собой концепцию архисовременного «компьютерного оружия», и абсолютно непонятно, почему проект Закона об изобретательской деятельности в СССР объявляет алгоритмы и программы объектом, не являющимся изобретательской деятельностью и не подлежащим патентованию...

Персональная ЭВМ не только дает полную свободу слова и возможность быть услышанным, но и, как следствие, в короткое время может сделать любой секрет и тайну достоянием общественности, тем самым ликвидируя массу бюрократов во внутренних аппаратах, призванных контролировать множительные устройства, цензоров в печатных органах и других пастырей идеологии и политики. Причем вы можете сделать это анонимно. Если вещь стоит того — вас прочтет весь мир. Планета становится единой. Автор доказывает это на примере данной программы и еще раз пишет: берегите(сь) программистов!!! Пора придать этой профессии тот статус, которого она заслуживает.

От редакции.
Публикуя это необычное письмо, поступившее без подписи и обратного адреса, предлагаем читателям решить, что это: предостережение, угроза, крик о помощи или шутка.

УВАЖАЕМЫЕ ЧИТАТЕЛИ!

Не забудьте подписаться
на наш журнал на 1990 г.
Индекс журнала «ИНФОРМАТИКА И ОБРАЗОВАНИЕ»
в каталогах «Союзпечати» — 70423.
Стоимость подписки на год — 3 руб. 60 коп.



Снимки сделаны в магазине «Электроника» — все, кроме этого.

Здесь — ассортимент компьютерного аукциона, организованного культурно-досуговым центром «Ритм» Гагаринского РК ВЛКСМ г. Москвы. О компьютерных «инициативах снизу» — в одном из следующих номеров журнала.

Б. № 11-41

60 коп.
Индекс 70423

OldPC.su
7002
музей компьютеров

ИНФО

4 '89

ИНФОРМАТИКА И ОБРАЗОВАНИЕ

ISSN 0234-0453. Информатика и образование, 1989, № 4, 1—128.