ISSN 0234—0453

ИНФО

ИНФОРМАТИКА И ОБРАЗОВАНИЕ

4 1987

```
АЛГ  MOD ( ВЕЩ X, ВЕЩ У)
     АРГ X
     РЕЗ У
НАЧ
  ЕСЛИ  X>0
     ТО  У:=X
     ИНАЧЕ  У:=-X
  ВСЕ
КОН
```

	1	2	3
АРГ X	−2	−2	−2
РЕЗ У			2
УСЛОВИЕ ВЕТВЛЕНИЯ		−2 ⩾ 0 НЕТ	−2 ⩾ 0 НЕТ
У		↓	↓
У		2	2
КОНЕЦ ВЕТВЛЕНИЯ		X	X
КОНЕЦ АЛГОРИТМА		X	X

АРГ X	−2	3	0
РЕЗ У	2	3	0
УСЛОВИЕ ВЕТВЛЕНИЯ	−2 ⩾ 0 НЕТ	3 ⩾ 0 ДА	0 ⩾ 0 ДА
У	↓	3	0
У	2	↓	↓
КОНЕЦ ВЕТВЛЕНИЯ	X	X	X
КОНЕЦ АЛГОРИТМА	X	X	X

```
АЛГ   УПОРЯДОЧЕНИЕ ( ЦЕЛ N, M; ВЕЩ ТАБ C [N:M] )
      АРГ C, N, M
      РЕЗ C
НАЧ   ЦЕЛ i, L  ВЕЩ R
        i:=1
      ПОКА  i<M
      НЦ
          МИНЭЛЕМЕНТ ( i, M, C, L )
          R=c[i]
          c[i]:=c[L]
          c[L]:=R
          i:=i+1
      КЦ
КОН
```

N=1 M=5 C	16	10	7	8	14
РЕЗ C	7	8	10	14	16
i	1	–	–	–	–
УСЛОВИЕ ЦИКЛА L<M	1<5 ДА	2<5 ДА	3<5 ДА	4<5 ДА	5<5 НЕТ
НАЧАЛО ЦИКЛА					
L	3	4	4	5	↓
R	16	10	16	16	↓
c[i]	7	8	10	14	↓
c[L]	16	10	16	16	↓
i	2	3	4	5	↓
КОНЕЦ ЦИКЛА	X	X	X	X	X
КОНЕЦ АЛГОРИТМА					X

Научно-технический прогресс в XXI веке будут определять сегодняшние школьники. Конечно, грядущий век кажется очень далеким, если тебе только пятнадцать, но годы — увы! — летят очень быстро, и, чтобы не сплоховать, оказавшись на передовой линии НТР, нужно начинать готовиться уже сейчас.

Максим Альт времени не теряет (первая страница обложки). Его статью читайте в молодежной рубрике. Знакомству с компьютером не препятствует и более юный возраст (четвертая страница обложки).

Научно-методический журнал
Министерства просвещения СССР,
Государственного комитета СССР
по профессионально-техническому образованию,
Министерства высшего
и среднего специального образования СССР

Выходит шесть раз в год
Издается с 1986 г.

Издательство «Педагогика»
Москва

ИНФ 4 '87 ИНФОРМАТИКА И ОБРАЗОВАНИЕ

Июль—
август

Содержание

Педагогический опыт

Внеклассная работа

Зарубежный опыт

Нам пишут

Информация

Веселый урок

Обложка *Э. Бажилина*

В оформлении номера принимали участие *А. Архутик, Э. Бажилин, С. Кустарева, А. Лобашинский, А. Пономарев, С. Расторгуев.*

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

Почтовый адрес: 107847, Москва, Лефортовский пер., 8
Телефон редакции 249-97-77
Сдано в набор 27.05.87. Подписано в печать 06.07.87. А 07358.
Формат 70×100/16. Печать офсетная. Усл. печ. л. 10,40. Уч.-изд. л. 12,75. Усл. кр.-отт. 42,88. Тираж 90210 экз. Заказ 1518. Цена 60 коп.

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

М. ГОТЛИБ
НИИ педагогических наук Минпроса АзССР

Компьютеру — дидактическое обеспечение

Превратить ЭВМ в мощное современное технологическое средство обучения невозможно без создания компьютерно-ориентированных предметных методик. Значит ли это, что предстоит разрабатывать совершенно новые области знаний, ранее неизвестные педагогике? Думается, нет. И дидактика, и частные методики располагают многочисленными и разнообразными разработками, которые с успехом могут быть использованы в автоматизированных обучающих системах (АОС). Безусловно, они потребуют серьезного переосмысления и определенных изменений, формирования нового подхода, но не стоит жалеть трудов для обеспечения успешности компьютеризации образования.

Особое место в списке этих методов занимает программированное обучение, которое в зависимости от ситуации может быть наполнено эвристическим, проблемным, продуктивным, иллюстративным, игровым и т. д. содержанием; оно является не только самостоятельным методом, но и формой организации многих других. Если же учесть его логическую строгость, формализуемость, алгоритмическую стройность, становится ясно: оно может быть использовано как методическое обеспечение компьютерного обучения.

Для прогнозирования содержания и возможностей программированного обучения в будущем необходимо объективно и критически оценить историю его развития. Она начинает свой отсчет с начала 50-х гг., когда американский психолог Б. Скиннер предложил разделить учебный материал на небольшие части — шаги, формируемые в виде вопроса (или в другой форме), предполагающего однозначный ответ. После того как обучаемому давалась возможность ответить (или хотя бы задуматься над вопросом), ему предлагался контрольный ответ, играющий роль подкрепления. Затем, независимо от правильности ответа, а значит, и прочности усвоения данного шага, он переходил к следующему. Такие программы получили название линейных. Достоинство их — в возможности изучать материал в посильном для учащегося темпе, а слабость — в отсутствии обратной связи, в невозможности реализации индивидуального обучения с учетом качества усвоения материала.

В некоторой степени эту проблему разрешил метод разветвленного программирования, разработанный в США Н. Краудером. Среди его отличительных особенностей можно назвать следующие: длина шага несколько больше, а сами задания сложнее, нежели в линейных программах; не осуществляется подкрепление в виде указания правильного ответа, вместо этого предлагается выбор ответа из некоторого множества, в котором один правильный, остальные правдоподобные, но неверные. Создатели разветвленных программ хотели отразить в них стереотипные ошибки учащихся. Анализируя выбор обучаемого, программа, а точнее — ее создатели пытались диагностировать его знания. В зависимости от типа ошибки выбирался последующий ход обучения: перейти с основной ветви программы на одну из дополнительных, боковых ветвей, где даются разъяснение, необходимая информация, дополнительные задания. Так можно изучать материал не только в посильном темпе, но и на доступном уровне. Слабой стороной метода является отражение в нем логики предмета, но не логики его усвоения учащимися. Сомнительной представляется диагностика знаний по допущенным ошибкам; большинство исследователей

считают, что без дополнительного исследования ничего нельзя сказать о причине их возникновения. Ведь одна ошибка может иметь разные причины, а внешне малопохожие — одну.

Были предприняты попытки повышения эффективности программированного обучения. Наиболее ярким примером служит теория поэтапного формирования умственных действий (А. Н. Леонтьев, П. Я. Гальперин, Н. Ф. Талызина). Одно из ее положений гласит: освоение предметных знаний основано на ранее сформированных внутренних психологических действиях, которые сами выступают как предмет усвоения. Задача обучения заключается в том, чтобы выделить наиболее рациональную систему приемов интеллектуальной деятельности, управлять ходом их формирования, получать заданное качество. Необходимо отметить, что предметные знания и психические действия находятся в сложной динамической индивидуальной взаимоформирующей связи, для выявления которой, а тем более использования в обучении, необходимо проводить сложные психолого-педагогические исследования, технологически не реализуемые в школьных условиях на современном этапе.

Даже в таком далеком от совершенства виде программированное обучение привело к значительному увеличению продуктивного рабочего времени учащихся, их активности, удельного веса самостоятельной работы, большей логической строгости в изложении предмета, ликвидации заучивания как вида учебной деятельности. Этим, казалось бы, и ограничены его возможности.

Однако ситуация меняется, как только начинает использоваться компьютер. ЭВМ и программированное обучение обладают высоким потенциалом взаимного влияния. Программированное обучение может стать «методическим интерпретатором» психолого-педагогического опыта для ЭВМ, а она — реализовать все возможности этого метода, которые никакими иными средствами достигнуты быть не могут. Какой же должна быть структура программированного обучения, реализуемого в ЭВМ?

Как известно, управление научением осуществляется за счет внутренних и внешних стимулов. Внешние создаются действиями обучающей системы, внутренние — всем предыдущим личным опытом индивида, его эмоциональным состоянием, сосредоточенным вниманием. Внешние стимулы в значительной степени инвариантны для учащихся, находящихся в идентичной обучающей среде, внутренние сугубо индивидуальны. На начальных этапах процесс усвоения понятия управляется внешними стимулами. Это значит, что научение как психическим действиям, так и предметным знаниям имеет в группе обучаемых большую «кучность», т. е. результаты приблизительно одинаковы у всех. С этим сталкивается на практике каждый педагог: привлечь внимание всего класса и удерживать его намного проще в начале изучения нового раздела. Такая однородная среда очень удобна для диагностики психологических особенностей.

По мере усвоения понятия внешние стимулы меняются незначительно (прежде всего из-за ограниченности сенсомоторных каналов). Напротив, как снежная лавина, растет объем внутренних стимулов. Процесс научения становится все более индивидуализированным, а значит, более скрытыми, менее поддающимися контролю становятся его особенности. Следовательно, индивидуальная модель (стратегия) обучения, основанная на психологических особенностях учащегося, должна выбираться на начальных этапах усвоения понятия. С другой стороны, одна качественная модель может иметь различные конкретные воплощения — в зависимости от результатов усвоения, формирования умений и навыков оперирования понятием. Если психологические особенности легче поддаются контролю в начальной стадии, то конкретные знания контролируются тем точнее, чем прочнее усваиваются, т. е. по мере усиления внутренних стимулов. Таким образом, контроль конкретных знаний может стать основой корректировки качественно выбранной модели (тактики). Можно высказать гипотезу о необходимости использования в компьютерно ориентированном программированном обучении *прогностически-корректирующей* структуры.

Для ее демонстрации рассмотрим формирование понятия «электрический заряд». Если у индивида хорошо развиты психические процессы обобщения, разумно сначала изложить свойства элементарных частиц: наличие у некоторых из них особого свойства — заряда, характер взаимодействия между заряженными частицами, его виды, два рода зарядов, невозможность существования заряда без частицы. Затем на основе уже усвоенных конкретных представлений можно создать обобщенное.

Если же у индивида более развиты аналитические способности, целесообразно познакомить его с явлениями, требующими для своего осмысления именно этих способностей. В приведенном ниже сценарии диалога «ученик — ЭВМ» стимул — информация от ЭВМ для анализа; ответ — результат анализа, формулируемый учащимся; дополнительная информация (ДИ) — элементы знаний, выдаваемые ЭВМ, которые не

могут быть получены в результате анализа (во всяком случае, учеником средней школы), но которые необходимы для последующего обучения.

Электростатика.
Тема: электрический заряд

1. Стимул. (Демонстрация притяжения наэлектризованного шарика к одному телу (1), отталкивания от другого (2), индифферентности по отношению к третьему (3).) Что можно сказать о телах 1 и 2?

2. Ответ. Тела 1 и 2 обладают некоторым свойством, по-разному проявляющимся у них и отсутствующим у третьего тела.

3. ДИ. Причиной действия на пробный шарик является электрический заряд тел 1 и 2.

4. Стимул. (Демонстрация зависимости от расстояния взаимодействия шарика с телами 1 и 2.) Сравним действие на шарик тел 1 и 2.

5. Ответ. Характер действия тел на шарик очень схож, отличается лишь направление. Значит, заряды тел чем-то различаются между собой.

6. ДИ. Принято говорить, что заряды различаются знаками.

7. Стимул. В природе существует два вида действия одного электрического заряда на другой, следовательно...

8. Ответ. ...в природе существует два рода зарядов.

9.* Стимул. Могут ли два вида взаимодействия создаваться более чем двумя родами зарядов?

10. Ответ. Заряды в природе можно разделить на две группы, различающиеся направлением действия на определенным образом заряженное тело. Каждая из групп соответствует определенному роду зарядов.

11. ДИ. В природе существует два рода зарядов, называемых «положительный» и «отрицательный».

12. Стимул. От чего зависит характер действия заряженного тела на шарик?

13. Ответ. От знаков зарядов обоих тел. Во-первых, любое действие в природе — обязательно взаимодействие; если бы действие определялось лишь одним зарядом, то данное заряженное тело одинаково взаимодействовало бы со всеми другими телами. Шарик не смог бы притягиваться к одному телу и отталкиваться от другого.

14. Стимул. Как определить характер взаимодействия разноименных зарядов?

15. Ответ. Исследовать взаимодействие тел 1 и 2, как нам известно, имеющих заряды противоположных знаков.

16. ДИ. (Демонстрация взаимодействия тел 1 и 2.) Разноименные заряды притягиваются.

17.* Стимул. Мог бы существовать мир, в котором разноименные заряды отталкиваются, а одноименные притягиваются один к другому?

18. Ответ. Если бы такой мир существовал, он разделился бы на две части, «положительную» и «отрицательную», разлетающиеся в противоположные стороны. Поэтому такой мир существовать бы не мог.

(О том, какими возможностями должна располагать АОС для осуществления такого диалога, поговорим позже.)

Итак, нужно обучать аналитическими методами тех, у кого *развиты* аналитические способности, и оберегать тех, у кого они *ущербны*? Отнюдь. Следует связать внешние стимулы с сильными сторонами личности, т. е. сеять новое понятие на благодатную почву, а затем уже подключать весь арсенал средств обучения. Тогда, как показывают наблюдения, происходит благотворное взаимодействие психологических особенностей — менее развитые способности получают значительный толчок к своему развитию.

Реализовать ту или иную качественно выбранную модель можно только с помощью конкретных методов и определенных соотношений между ними: репродуктивных и продуктивных, образных и вербальных, абстрактных и конкретных. Приведенный сценарий соответствует начальному этапу усвоения понятия, поэтому он лишь в незначительной степени поддается индивидуализации, но и в нем следует учитывать результаты научения, пусть даже небольшого. Введем для определенности три уровня усвоения учебного материала. Первый соответствует наилучшим показателям («знаток»). Третий — самым незначительным успехам («новичок»). Показывающего промежуточные результаты будем называть «мастер». Сценарий может быть предложен «знатоку» целиком. При обучении «новичка» нецелесообразно использовать фрагменты 9* и 17*, рассчитанные на достаточно высокий уровень творческого мышления. Более того, учащимся этого уровня подобные элементы не должны предлагаться даже в режиме пассивного ознакомления, так как они, продолжая оставаться стимулами, становятся отрицательными. Пассивная форма восприятия таких элементов оправдана для «мастера». Но как перейти из активного режима работы с данным учебным метериалом в пассивный? Есть несколько способов (они будут продемонстрированы в последующих примерах), самый простой — выдавать фрагменты на эк-

ран монитора с таким расчетом, чтобы интервал между появлением стимула и ответа был достаточным для осознания вопроса, но недостаточным для формулирования ответа. Когда «знатоку» предлагается выбрать вариант ответа, «мастеру» можно предложить проанализировать готовый, например между фрагментами 17* и 18 сделать вставку:

17.1. Стимул. Оцените истинность следующего ответа.

На последующих этапах обучения, когда знакомство с понятием уступает место отработке умений и навыков, гамма приемов расширяется. Например, АОС зафиксировала, что минимум времени при минимальном количестве ошибок учащийся затрачивает на выполнение заданий, связанных с опорным элементом знаний (ОЭЗ)

20.3. Электризация проводников происходит благодаря перемещению в них свободных носителей электрического заряда.

Напротив, задания, связанные с элементом

20.4. Электризация полярных диэлектриков происходит в результате ориентации диполей

требуют от него долгих раздумий, а их выполнение чаще обычного приводит к неправильным результатам. В этом случае можно осуществить адаптацию учебного материала с помощью:

а) увеличения шага между уровнями освоения ОЭЗ 20.3 и сокращения его в отношении элемента 20.4;

б) увеличения потока обучения через задания, связанные с элементом 20.4, за счет сокращения его через элемент 20.3 (это можно осуществить, меняя приоритеты соответствующих заданий).

В результате реальный процесс обучения будет, колеблясь, асимптотически приближаться к некоторому «идеальному», при котором равномерно и гармонично отрабатываются все ОЭЗ, формируются умения и навыки.

АОС должна иметь модульную структуру — это так же естественно, как то, что структурным должно быть программирование. Кроме того, использование модулей позволит сблизить теоретические рассуждения о возможностях АОС с ее практической реализацией в ближайшее время. Если мы, к примеру, не можем пока реализовать эффективную адаптацию учебного материала по психологическим показателям индивида, то это не помешает создать АОС без такого элемента, но с расчетом на то, что рано или поздно, введя соответствующий программный модуль, мы решим данную задачу.

Среди таких модулей, которые можно подготовить уже сегодня, учитывая при этом будущее АОС, идентификатор (набор «паспортов») заданий. В нем должна быть заключена полная информация о задании.

Это может быть, например, двухмерный числовой массив ИДЗ (М, К), где М — номер задания, К — позиция той или иной его характеристики.

Первый столбец (К=1) отражает уровень сложности, принимающий три значения:

$$ИДЗ(М, 1)=1, 2, 3.$$

Второй (К=2) указывает тип задания и является ключом для управления его обработкой:

$$ИДЗ(М, 2)=1, ..., 10.$$

Можно ограничиться этим количеством, так как исследованные нами методические разработки без ущерба для их разнообразия укладываются в десять оригинальных форм. При появлении новых вариантов достаточно будет создать соответствующие модули обработки заданий, расширить каталог, а в только что приведенном выражении заменить последнее число. В третьем столбце указывается степень формализованности ответа (о ней мы будем говорить отдельно), в четвертом — шифр контрольного ответа, в пятом — количество ОЭЗ, связанных с данным заданием. В случае высокой комплексности $ИДЗ(М, 5)=6$. Это значение, кроме того, указывает, сколько последовательных элементов строки, начиная с седьмого, отводится для регистрации шифров указанных ОЭЗ. Шестой столбец — количество психологических особенностей, лежащих в основе выполнения задания, и количество элементов строки с шифрами этих особенностей (они расположены сразу же после шифров ОЭЗ); указатель первой психологической особенности находится в столбце

$$К=6+ИДЗ(М, 5)+1,$$

а последней — в столбце

$$К=6+ИДЗ(М, 5)+ИДЗ(М, 6).$$

Конкретный вариант строки идентификатора приведен в таблице.

К	1	2	3	4	5	6	7	8	9
ИДЗ (4, К)	3	5	1	8	2	1	9	7	4

Из нее следует, что задание 4 имеет низкий коэффициент сложности (3), рассчитанный на «новичка». Обработку задания — адаптивные изменения, выдачу дополнительной информации, распознавание ответа и комментарии к нему — осуществляет программа, имеющая шифр 5. Она, кстати, расшифровывает четвертый элемент строки (в нашем случае 8) — то ли это сам ответ, то ли место его нахождения в специальном текстовом массиве. ИДЗ (4, 5) указывает, что задание отрабатывает два ОЭЗ — 9 и 7 (их шифры — в 7-м и 8-м столбцах).

ИДЗ (4, 6) = 1 означает, что выполнение задания основано на изолированной психологической особенности, код которой (4) занесен в столбец 9.

Мы ограничим содержание идентификатора задания перечисленными данными. Они не исчерпывают всей информации о задании, но позволяют продемонстрировать структуру организации данных. Будем помнить, что идентификатор сможет принять любую дополнительную информацию, если только зарезервировать в нем несколько запасных столбцов.

Характеристики программированных заданий

Желая оценить качество программированного задания (даже вне связи с ЭВМ), мы прежде всего обнаружим, что критерии такой оценки не сформулированы — во всяком случае, достаточно конкретно и обоснованно. Существуют работы, где этот вопрос поставлен, однако мимоходом, в русле исследования иных проблем. А ведь важность таких критериев для компьютерно-ориентированного обучения трудно переоценить. Потребность в них определяется даже не столько необходимостью оценки качества готового продукта, сколько их ролью в планировании создания АОС. Попытаемся сформулировать критерии качества компьютерно-ориентированных прогностически-корректирующих программированных заданий.

Структурность относится к числу наиболее важных. Ценность задания при прочих равных условиях тем выше, чем более расчленено оно на отдельные стадии (этапы, шаги). Чем больше их число, тем разнообразнее подгруппы, составляемые из них с адаптивной целью, тем гибче тактика обучения. Критерий этот характерен для линейных программ.

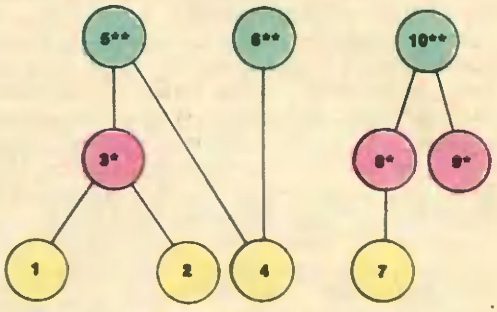
Рассмотрим следующее обучающее задание.

Механика. Тема: прямолинейное равномерное движение

Ответьте на вопросы.

Координатная ось направлена вправо. Тележка начинает движение из точки, находящейся левее начала координат на 5 м, и за каждую секунду совершает одинаковые перемещения. Через 20 с она находится правее начала координат на 45 м. Чему равны ее:

1) начальная координата?
2) конечная координата?
3)* перемещение за 20 с?
4) перемещение за 1 с?
5)** скорость?
6)** перемещение за 10 с?
7) координата в момент 1 с?
8)* координата в момент 5 с?
9)* время прохождения начала координат?
10)** время перемещения на 5 м?



i

Его граф приведен на рис. 1. Каждый из десяти фрагментов принадлежит одному из трех уровней обучающей значимости. Фрагменты первого уровня (5**, 6**, 10**) являются основными. Ответы на них и есть цель работы над заданием, посильны же они разве что «знатоку». Исследования показывают, что у «мастера» результаты повышаются, если данные фрагменты сочетаются с элементами второго уровня — подчиненными (3*, 8*, 9*). И лишь «новичкам» оптимально предлагать выполнить все фрагменты. Если «знаток» все же ошибется при выполнении задания первого уровня, ему в определенной последовательности, задаваемой ребрами графа, предлагаются подчиненные и даже соподчиненные (1, 2, 4, 7) фрагменты. Таким образом, программный модуль, обрабатывающий подобные задания, комбинирует последовательно-подготовительные и последовательно-корректирующие алгоритмы [2].

При таком подходе возникает кажущееся несоответствие: «новичок» вынужден пройти все фрагменты, что в три раза превышает объем работы «знатока». Однако то, что перед «знатоком» не ставятся соответствующие задачи, не значит, что он их не решает. Для ответа на основные вопросы он должен не только ответить на подчиненные и соподчиненные, но и самостоятельно поставить их перед собой. Кроме того, обучение осуществляется с контролем времени. Если у «новичка» обнаруживается его дефицит, работа приостанавливается, скажем, на шестом фрагменте. И еще. Среди модулей АОС должна быть программа оптимизации процесса обучения, в функции которой входит выбор тех заданий, которые создают максимальный обучающий эффект для индивида. Это позволит «знатокам»

потратить сэкономленное время, если таковое у них окажется, на те задания, которые не будут предложены «новичку».

Системная полнота является критерием информационно-технологического характера. Задание должно включать в себя весь набор ОЭЗ, обеспечивающих завершенность процесса формирования понятия. Это не означает, что активизации подлежат все некогда усвоенные знания, имеющие отношение к изучаемой теме. Активизироваться должны только те, которые в свое время не были усвоены достаточно прочно. Это позволяет осуществить «глубоко эшелонированное» обучение. Усвоение понятия происходит не на одном-двух уроках, а на протяжении ряда тем, прямо или косвенно с ним связанных. Необходимые для этого данные хранятся в идентификаторе ОЭЗ — массиве ИДО (М, К), структура которого аналогична ИДЗ (М, К). Один из его столбцов формируется в процессе выполнения учащимся того или иного фрагмента задания и отражает качество усвоения того ОЭЗ, который при этом отрабатывается. Алгоритм действия управляющего программного модуля при изучении нового понятия таков:

считывание из массива ИДО (М, К) изучаемого понятия тех элементов строки, которые указывают ОЭЗ, лежащие в основе усвоения этого понятия;

считывание из ИДО (М, К) информации о качестве усвоения выбранных на предыдущем этапе ОЭЗ;

вывод на экран монитора ОЭЗ, степень усвоения которых не соответствует контрольному уровню, либо задания, отрабатывающего соответствующие ОЭЗ.

Но это лишь репродуктивная часть деятельности АОС, отражающая повторение некогда усвоенных навыков. Продуктивная же заключается в том, чтобы помочь отработать ряд новых умений, специфичных для новой темы. Вот пример такой ситуации.

Прежде чем приступать к решению задач по теме «Кинетическая энергия», целесообразно провести развитие навыков формализованной записи. Приведенные ниже фрагменты являются специфическими опорными элементами при решении задач по данной теме (в скобках даны ответы).

Механика.
Тема: кинетическая энергия

Сделать формализованные записи следующих величин и высказываний.

1) Кинетическая энергия тела.

$$(E_k = mv^2/2)$$

2) Кинетическая энергия.

$$(E_k = 0)$$

3) Изменение кинетической энергии тела.

$$(\Delta E_k = mv_2^2/2 - mv_1^2/2)$$

4) Изменение кинетической энергии тела, действия на которое скомпенсированы.

$$(\Delta E_k = 0)$$

5) Работа равнодействующей сил, приложенных к телу.

$$(A = m v_2^2/2 - m v_1^2/2)$$

6) Кинетическая энергия системы двух тел.

$$(E_k = m_1 v_1^2/2 + m_2 v_2^2/2)$$

7) Кинетические энергии двух тел равны.

$$(m_1 v_1^2/2 = m_2 v_2^2/2)$$

8) Кинетическая энергия изолированной системы двух взаимодействующих тел.

$$(E_k = m_1 v_1^2/2 + m_2 v_2^2/2 = \text{const})$$

Предварительная отработка этого простого навыка значительно облегчит последующее обучение решению задач, даже простые варианты которых требуют комплексных навыков.

Комплексность, т. е. количество опорных элементов знаний, необходимых обучаемому для успешного выполнения данного задания, может служить следующим критерием. Он является логико-ассоциативным показателем. С заданий низкой комплексности, позволяющих осуществлять контроль элементарных знаний, начинается изучение понятия, по мере усвоения которого комплексность, очевидно, должна возрастать, формируя с каждым разом все более ассоциативное мышление. Качество задания тем выше, чем больше вариантов с разной степенью комплексности содержится в нем.

Пусть, к примеру, в некотором тексте есть слова «Работа по перемещению тела...». Им соответствует ОЭЗ $A = FS\cos\alpha$. Заменив их на «Работа электростатического поля по перемещению заряженного тела...», мы тем самым добавим еще один ОЭЗ: модуль силы, действующей в электростатическом поле на заряженное тело, равен $F = qE$.

При необходимости (и по возможности) комплексность фрагмента может быть еще более увеличена, в данном случае за счет дальнейшей конкретизации условия: «Работа электростатического поля по перемещению заряженного тела из точки x_1 в x_2...». При этом добавится элемент «модуль перемещения равен $S = |x_2 - x_1|$». А заменив текст фрагмента на «Работа электростатического поля по перемещению отрицательно заряженного тела вдоль силовой линии из точки x_1 и x_2...», мы увеличим комплексность еще на два пункта: модуль отрицательной величины $a < 0$ равен этой величине, взятой с противоположным знаком:

$|a| = -a$; сила, действующая на отрицательный заряд, направлена противоположно силовой линии.

Организованная таким образом АОС располагает большим диапазоном возможностей: относительно простое задание, стоит лишь заменить в нем один фрагмент на другой или же незначительно изменить текст, становится емким и сложным. Значение адаптивного изменения комплексности фрагментов — будь то набор ОЭЗ, комментарий или текст информационного характера — трудно переоценить. Фактически это означает, что компьютер находит язык индивидуального общения с каждым обучаемым — понятный тому, но и требующий умственного напряжения.

Энтропия (степень неопределенности) множества правдоподобных ответов. Чем она больше, тем сложнее нахождение верного ответа. Минимальное ее значение — при альтернативных ответах типа «да — нет», «истинно — ложно», «увеличивается — уменьшается» и т. п. Для каждого учащегося на данной стадии обучения существует некоторая оптимальная величина этой характеристики, создающая максимальный эффект научения. АОС должна быть организована таким образом, чтобы существовала возможность варьировать энтропию, например, за счет трансформации текста. Вопрос «Какое взаимодействие наблюдается в системе электрически заряженных тел?» имеет низкую энтропию, так как содержит термины, входящие в ответ на него (электростатическое взаимодействие). Энтропия возрастает, если заменить «взаимодействие» на «явление»; еще сильнее — если исключить «электрически». Хронометрирование работы учащихся показывает, что время обдумывания вопроса «Какое явление наблюдается в системе заряженных тел?» в среднем на 15 % больше, чем при первоначальной его формулировке.

Наполненность изучаемого понятия — критерий дидактического характера. Она зависит от того, насколько многогранна деятельность учащегося относительно усваиваемого содержания, разнообразны создаваемые системой ситуации. Перечислим сначала обязательные действия АОС:

активизация элементов знаний, подготавливающих восприятие данного понятия;

создание ситуаций, указывающих на необходимость введения нового понятия;

введение понятия, использование его в различных ситуациях;

исследование понятия, его трансформация;

синтез введенного понятия с ранее изученными.

Учащиеся при этом проходят через стадии узнавания понятия, запоминания его, репродуктивного, а затем продуктивного действия над ним. Изучая, например, радиоактивный распад, обучаемый отвечает на двадцать вопросов, вынуждающих его всесторонне исследовать это явление.

Ядерная физика.
Тема: радиоактивный распад

О каком или каких радиоактивных распадах идет речь, если известно, что он (они) или его (их) продукты:

1) представля—т[1]) собой поток электронов?
2) представля—т собой поток ядер атомов некоторого элемента?
3) представля—т собой электромагнитную волну?
4) представля—т собой поток положительно заряженных частиц?
5) представля—т собой поток отрицательно заряженных частиц?
6) отклоня—тся в электрических и магнитных полях?
7)* сильно отклоня—тся в электрических и магнитных полях?
8)* слабо отклоня—тся в электрических и магнитных полях?
9) не отклоня—тся в электрических и магнитных полях?
10)* возника—т при распаде ядер с большим атомным числом?
11)* испуска—тся ядрами с относительным избытком нейтронов?
12)* сопутству—т другим видам радиоактивного излучения?
13) уменьша—т массовое число ядра на 4?
14)* не изменя—т массовое число ядра?
15)* не изменя—т массовое число и заряд ядра?
16) не изменя—т заряд ядра?
17) увеличива—т заряд ядра на единицу?
18) уменьша—т заряд ядра на 2?
19)* име—т наибольшую среди всех других проникающую способность?
20)* использу—тся для бомбардировки ядер других элементов?

А теперь проанализируем действия АОС, обусловленные тактикой работы с конкретным учеником. Все ли приведенные фрагменты следует предлагать обучаемому, а если нет, то по какому принципу необходимо проводить отбор? Частично ответ вытекает из протокола взаимодействия обучаемого с

[1]) Пропущенная в окончании буква увеличивает энтропию задания. Если бы в тексте было слово «представляет», то ответом мог быть только один их трех вариантов: α-, β- или γ-распад. Используемый прием рассогласовывает вопрос и ответ.

АОС, содержащего сведения о рациональной степени «погружения» в подобного рода объемные задания. Другие указания содержатся в ИДЗ (сложность и обучающая значимость). Сигналом о необходимости прекращения «погружения» может быть ухудшение эргономических показателей (тут очень поможет учет времени работы, контроль усталости и умственной работоспособности). Уместно также вспомнить о существовании важного педагогического приема: реализации свободы выбора обучаемого. После выполнения основной части задания можно предложить ему решить — прекратить или же продолжить выполнение задания. Это создает благоприятную для работы психологическую атмосферу. Школьник осознает, что он не пассивный объект обучения, а активный руководитель собственным научением.

Степень формализации ответа. Ответ может быть отнесен к вполне формализуемым, слабо- и неформализуемым. Если задание имеет вычислительный характер и ответом является число, если задание предполагает тождественные преобразования или решение уравнения и ответ — формула или математическое выражение, в других случаях, когда можно быть уверенным, что не существует альтернативы заложенному в память ЭВМ ответу, он относится к вполне формализуемым. Даже в таких простейших случаях возникают сложности (наличие в ответе пробелов, тождественная перестановка его элементов), требующие различных ухищрений при его распознавании. Однако эти сложности непринципиального характера, программисты уже разработали стандартные приемы их решения.

К слабоформализуемым можно отнести односложные ответы. Программу их распознавания рационально построить следующим образом. В идентификаторе задания, как уже говорилось, указывается степень формализованности его ответа. Код «2» (слабоформализуемый ответ) требует вызова подпрограммы «Ответ», последовательность действий которой должна быть следующей.

1. Запоминание введенного ответа.

2. Распознавание ответа с помощью посимвольного сравнения с массивом словаря ЭВМ, например, по коренному буквосочетанию:

Введенный ответ	Словарь
растет, возрастает	раст
велико, увеличивается	вел

3. Замена распознанного ответа классифицирующим синонимом:

раст

увеличивается

вел

4. Сравнение классифицирующего синонима с правильным ответом.

5. Вывод результата выполнения задания.

Если же этап 2 невыполним и подпрограмма не может распознать введенный ответ, система выдает соответствующее сообщение на монитор и переключается в резервный режим; введенный ответ сохраняется для анализа с участием человека. Если окажется, что он представляет интерес, то его включают в состав словаря. Так осуществляется обучение самой обучающей системы.

В случае неформализуемости в тексте задания уже имеются варианты ответов. Учащемуся остается «только» выбрать тот из них, который он считает верным. Слово «только» не случайно заключено в кавычки. Автор принадлежит к исследователям, убежденным в том, что педагогически качественно разработанное программированное задание не становится ущербным, если включить в него альтернативные ответы. На это указывают и результаты экспериментов. Успех в выполнении задания увеличивается лишь незначительно, если нахождение ответа сменяется выбором его из предъявляемого набора. Прием этот помогает не столько найти решение, сколько правильно его сформулировать. В 5—6 ответах может быть заключен такой объем информации, что угадывание становится статистически невероятным событием, а если на задание дают значительное число правильных ответов даже «новички», то виноват в этом его создатель, не сумевший сделать задачу обучаемого сопряженной серьезным мыслительным процессам. И еще. Предлагая варианты ответов, мы действительно помогаем учащемуся. Но ведь обучение и есть помощь, задача в том, чтобы она была разумной. Сложность задания должна соответствовать уровню развития учащегося, не актуальному (возможностям его самостоятельной деятельности), а зоне ближайшего развития (то, что он может сделать, используя незначительную помощь) [1]. Этой помощью могут стать альтернативные ответы. Раз уж мы вынуждены их предлагать (в случае невозможности формализации), то превратим эту слабую сторону программированного обучения в сильную, для чего необходимо составлять задания таким образом, чтобы только вместе с ответами они были посильны учащемуся и, следовательно, в действительности соответствовали бы зоне его ближайшего развития. Сложность выбора правильных ответов при этом можно регулировать и изменением их числа, используя в качестве ориентира уровень под-

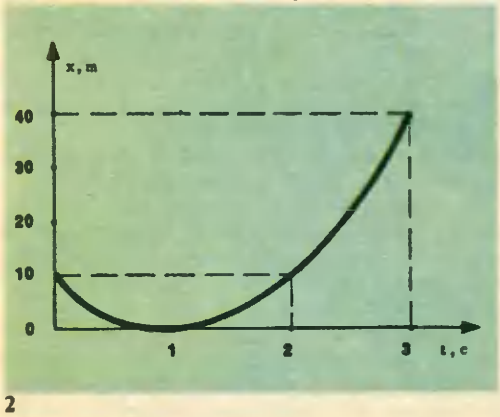
готовленности и еще один важный показатель — скорость чтения, которая измеряется при работе учащегося с информационными текстами и также содержится в его протоколе.

Наличие алгоритма — необходимое условие качественности прогностически-корректирующих программ. Предлагаемое ниже задание демонстрирует различные типы алгоритмов, варианты их сочетания, целесообразность использования в различных ситуациях, возможность перестраивания задания из одного типа алгоритма в другой.

Механика. Тема: график равноускоренного движения

Вставить пропущенные в тексте значения, используя рисунок (рис. 2).

На рисунке изображен график движения тела. Координата тела в начальный момент времени x_0=...м (1), к концу движения стала x=...м (2). Перемещение тела за последнюю, третью, секунду движения S=...м (3)*. Перемещение тела за 3 секунды движения S′=...м (4). Средняя скорость движения за рассматриваемый промежуток времени $V_{ср}$= =...м/с (5)*. Средняя скорость движения на участке 0 м — 40 м V=...м/с (6)*. Используя кинематические соотношения, можно определить ускорение тела: a=...м/с^2 (7)**. Тело двигалось на участке 0 м — 40 м в течение t=...с (8). Мгновенная скорость в момент t=2 с V′=...м/с (9)**. Координату x=40 м тело прошло со скоростью V″= =...м/с (10)**.



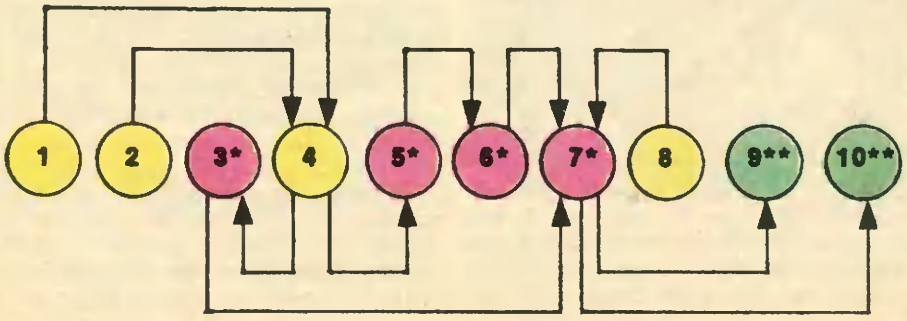
2

Заключительные фрагменты 9** и 10** соответствуют сложности олимпиадного задания и трудновыполнимы даже для «знатока». Однако предварительная комплексная работа со взаимосвязанными фрагментами 1—8, комбинирующая подготовительный и корректирующий режимы, делает эти задания посильными и для «мастера».

Граф алгоритма этого задания приведен на рис. 3. Из него видно, что 1-й и 2-й фрагменты совместно подготавливают 4-й, акцентируя внимание на начальных и конечных координатах и моментах движения, и поэтому являются параллельно-подготовительными; 4-й фрагмент подготавливает 5-й, тот — 6-й, а последний, в свою очередь, 7-й. Это последовательно-подготовительный ряд. С другой стороны, 4-й фрагмент является последовательно-корректирующим для 3-го, где речь идет о перемещении за одну только третью секунду. Зачастую учащиеся ошибочно воспринимают третью секунду как промежуток в три секунды (приблизительно в 40 % случаев). Отрабатывать такого рода стереотипные ошибки лучше всего с помощью корректирующего алгоритма, т. е. предварительно дать возможность ошибиться, а затем прокомментировать ответ таким, к примеру, образом:

«Ответ ошибочен. Убедитесь в этом, выполнив следующее задание».

Такой подход имеет определенное психологическое обоснование. Поменяв местами 3-й и 4-й фрагменты, следовательно, превратив корректирующий алгоритм в подготовительный, мы тем самым защитим учащегося от возможной ошибки сейчас, но не предупредим его на будущее, не зафиксируем внимание на различии этих двух условий. И это сразу скажется заметным увеличением количества неудачных попыток в ответе на 7-й фрагмент, подготовительным для которого является 3-й. Центральным, связующим фрагментом всего задания яв-



3

ляется 7-й. Высокий уровень сложности (1) сочетается с большим порядком (5) его вершины — столько ребер соединяет ее с соседними вершинами. Для успешного выполнения этого фрагмента необходимо сочетание двух эвристических приемов: а) использование не всего графика, а лишь части его, начиная с момента остановки ($t=1$ с) (намек на этот прием содержат 3-й и 6-й фрагменты, а 8-й — прямое на то указание); б) использование при анализе графика формулы перемещения при равноускоренном движении $S=at^2/2$ (подготовка этого приема не осуществляется в данном задании — предполагается самостоятельный поиск). Это пример «большого шага». В случае неудачной попытки или просьбы о помощи АОС должна вывести соответствующий ОЭЗ, после чего выполнение всего задания в группе «мастеров», например, в 60 % случаев приводит к успеху. Изменится ли этот итог и каким образом, если указанные ОЭЗ выводить не после выполнения 7-го фрагмента, а до, укоротить «шаг»? Первый результат вполне ожидаемый: успех в выполнении 7-го фрагмента увеличивается. Второй заставляет задуматься: выполнение 9-го и 10-го фрагментов, где также необходимо воспользоваться формулой, правда, иной, становится затруднительным. Этот результат можно интерпретировать так: 7-й фрагмент формирует некоторое умение, перенос которого на 9-й и 10-й происходит эффективно в том случае, если предварительно предоставить учащемуся возможность самостоятельного поиска познавательного приема и уж затем оказать необходимую помощь.

Приведенный анализ имеет своей целью показать, что отличительной особенностью программированного обучения является не выбор ответа учащимися, а выбор самим педагогом (или обучающей системой на основе тщательнейшим образом составленной психолого-дидактической инструкции) модели и, значит, той разумной последовательности внешних стимулов, которая с учетом уровня подготовленности учащегося и его психологических особенностей создаст наибольший эффект научения.

Придерживающиеся довольно распространенного в педагогике взгляда на программированные задания как на тренажер для отстающих, вероятно, изменят свою точку зрения, если проведут небольшое исследование с помощью приведенного задания, предложив его, варьируя количество, сложность и последовательность фрагментов, различным группам учащихся.

Возможность трансформации. Как видно из предыдущего примера, одним из возможных методов трансформации типа алгоритма является перестановка фрагментов. Следует учесть, что не любая перестановка приводит к желаемому результату; в иных случаях мы не получим новый тип алгоритма, а лишь потеряем старый. Но есть и другие способы достижения поставленной цели, в частности изменение установки задания. Рассмотрим его на примере анализирующего и синтезирующего алгоритмов [2].

Электродинамика.
Тема: физические величины

А. Какие из приведенных в пункте Б величин характеризуют:
1) проводник?
2) электрическое поле?
3) заряженное тело в электрическом поле?

Б
а) емкость C;
б) потенциал φ;
в) напряжение U;
г) сила тока J;
д) сопротивление R;
е) напряженность E;
ж) потенциальная энергия W;
з) сила F, действующая на заряженное тело;
и) работа A по перемещению заряженного тела.

Ответами на первый вопрос являются емкость и сопротивление, на второй — напряженность, потенциал, напряжение. Наконец, сила, работа, потенциальная энергия зависят и от величины заряда тела, и от свойств поля. Задание это наполняется совершенно иным содержанием, приобретает иную направленность, если его «перекроить», изменив установку. Для этого необходимо сгруппировать часть перечисленных величин и предложить учащемуся определить, по какому принципу объединяются они друг с другом, т. е. превратить ответ в условие; пункт А при этом становится ответом. Новое задание приобретает логико-синтезирующую направленность, высокий уровень ассоциативности, требует нестадартных методов мышления и вследствие этого посильно лишь для небольшой группы обучаемых. Оно оказывает большое влияние на формирование интеллектуальных приемов действий, но — что очень важно — лишь у тех, кто с ним справляется. Для остальных оно является отрицательным стимулом. Уместно поэтому задаться вопросом: какие показатели должны учитываться при осуществлении выбора между различными алгоритмами? Эксперименты показывают, что успех выполнения модифицированного задания сильно коррелирует с уровнем развития у индивида процессов памяти — элемента психопредметного про-

филя. Это позволяет сделать такие выводы:

адаптивная замена анализирующего алгоритма на синтезирующий в рассматриваемом случае целесообразна при хорошо развитых у индивида психических процессах памяти;

данный синтезирующий алгоритм может быть использован для диагностики уровня развития психических процессов памяти;

сочетание выполнения данного синтезирующего алгоритма с выполнением подготовительных заданий, понижающих порог его посильности, оказывает влияние на развитие психических процессов памяти.

На этом и многих других примерах можно убедиться в том, что выяснение корреляций между конкретными заданиями, алгоритмами, ОЭЗ, психологическими особенностями является одним из основных источников оптимизации процессов обучения.

Диагностическая избирательность. Как уже говорилось, научение в значительной степени зависит от уровня развития у индивида внутренних познавательных действий. Знание их и соотношений между их компонентами, пусть даже приблизительное, позволяет повысить эффективность любого обучения, в том числе и компьютерного. Средством проведения подобной диагностики могут являться сами обучающие задания. Любое из них в той или иной степени связано с комплексом психологических особенностей, однако можно выбрать и такие, которые полностью или в значительной степени опираются на некоторую изолированную психологическую особенность. Количественный контроль выполнения таких заданий (фиксация времени, процента верных результатов) может служить основой оценки предметно-психологических показателей.

Рассмотрим задание, с помощью которого можно было бы определить такую важную и в то же время труднодиагностируемую психологическую особенность, как лабильность (гибкость) мышления. Ряд представлений о физических величинах, введенных в механике (скорости, силе, работе, энергии), находят затем применение и в других областях физики, в том числе термодинамике. Воспринятые на механических примерах, они подвергаются сложным изменениям в психике обучаемого. Контроль проявлений этих внутренних процессов может многое сказать о лабильности мышления.

Молекулярная физика.
Тема: термодинамические параметры

Установить истинность следующих высказываний.

Термодинамика изучает скорость движения...

1) молекул относительно центра масс системы;

2) центра масс системы относительно окружающих тел.

В термодинамике под силой, совершающей работу, понимают...

3) силу взаимодействия молекул между собой;

4) силу взаимодействия между элементами системы и окружающими телами;

5)* силу действия на систему со стороны стенок сосуда.

Определяя работу термодинамической системы, под перемещением понимают...

6)* среднее перемещение молекул системы относительно ее центра масс;

7) суммарное перемещение молекул системы относительно ее центра масс;

8)* среднее направленное перемещение молекул системы относительно ее центра масс.

Внутренней энергией системы...

9) называется полная механическая энергия элементов системы относительно самой системы;

10) называют чисто термодинамическое понятие, которое нельзя определить с помощью механических понятий.

В каждом случае возможен один из трех ответов: истинно; ложно; для установления истинности недостаточно данных.

Это задание может быть диагностирующим благодаря тому, что количество вариантов выбора одного из трех ответов на каждый из десяти вопросов равно 3^{10}, т. е. более 50 тыс., и только один из них вполне правилен. Безусловно, это достаточное количество для получения статистически надежных результатов. Несколько сложнее доказать его валидность, т. е. показать, что успех выполнения задания действительно определяется лабильностью. Залогом этого может служить педагогический опыт автора и ряда других преподавателей и психологов, участвовавших в обсуждении данного вопроса.

Вариативность программированных заданий. Разнообразие деятельности учащегося приводит к повышению мотивации его обучения. Очень важно поэтому иметь в арсенале АОС возможно больше разнообразных форм и комбинировать их так, чтобы они как можно реже повторялись. Автор столкнулся в своей практике более чем с десятью оригинальными конструкциями программированных заданий, и, вероятно, это еще не все. Кроме того, не следует забывать, что форма не существует сама по себе, а является средством выражения процесса обучения. Замечено, что каждая фор-

ма несет определенный психологический настрой. Работу, например, с фрагментами текста или математических высказываний предпочитают «новички». На это указывают не только протоколы обучения, но и результаты опроса. Редактирование текстов захватывает «знатоков». А установлением истинности высказываний предлагают завершать цикл обучения 75 % опрошенных. Наибольшую ценность имеют гибкие формы — они позволяют менять структуру, содержание, энтропию, алгоритм, уровень сложности задания.

Разнообразными должны быть и интересы, затрагиваемые выполнением того или иного задания. Иллюстрацией может служить следующий пример.

Механика. Тема: мгновенная скорость

Какой из величин следовало бы характеризовать каждое из явлений в отрывках из пункта А? Для ответов использовать варианты из пункта Б.

А

1) «На горизонте показалось едва заметное серое пятно, но охотники знали: не позже чем через час все небо будет обложено тучами».

2) «Всадники рванули все разом, окунувшись в клубы пыли, где уже ничего невозможно было разобрать».

3) «...взлетная полоса кончилась, но скорость самолета была все еще недостаточна, чтобы оторваться от земли».

4)* «Ничто уже не могло остановить или хотя бы замедлить движение этой огромной глыбы льда, равномерно и неотвратимо надвигающейся на судно».

Б. Речь идет о:

а) средней скорости тела;

б) мгновенной скорости тела в некоторый момент времени;

в) мгновенной скорости тела в некоторый точке пространства;

г) прямолинейном и равномерном движении, при котором значения мгновенной скорости и средней скорости совпадают.

Учет упомянутых и многих других дидактических разработок, уже созданных и еще только создаваемых, позволит превратить автоматизированную обучающую систему в надежного, объективного, внимательного, очень много умеющего помощника учителя. И очень важно уже сейчас стремиться к тому, чтобы в АОС отразилось не только то, что педагоги, психологи, программисты и компьютеры могут сегодня, а и то, что они смогут завтра.

Литература

1. *Выготский Л. С.* Избранные психологические исследования. М., 1956.

2. *Готлиб М. А.* Структура АОС // Информатика и образование. 1987. № 3.

С. ГРИГОРЬЕВ, М. МОРОЗОВ
Йошкар-Ола

Давайте попробуем Пролог

О традиционных языках и нетрадиционных задачах

Когда рассматривают круг проблем, связанных с изучением и использованием языков программирования в школе, обычно имеют в виду традиционные языки — Бейсик, Рапиру, Паскаль и т. п. В чем главная особенность этих, так называемых процедурных, языков? В ориентации на вычислительную машину. Процедурные языки развивались как средство записи операций, выполняемых вычислительной машиной, и отражают непривычное для человека «машинное мышление». Такой подход предполагает: чтобы ЭВМ решила задачу, необходимо построить алгоритм, определяющий последовательность действий, и запрограммировать его, а машина, точно следуя по указанному пути, сделает то, что было задумано.

Однако подобный образ действий вне физико-математических дисциплин сталкивается с серьезными трудностями.

Ориентированный на математические расчеты Бейсик малопригоден для отражения предметного мира, скажем, географии, истории, иностранного языка. Даже большие возможности символьной обработки, заложенные в Рапире, оказываются недостаточными.

Есть и другая проблема, возникающая при обучении школьников программированию на традиционных языках: поиск конк-

ретных реальных задач, решать которые удобно, используя программирование.

В вузе вопрос решается просто: студентов заставляют работать с численными методами решения математических задач. В младших классах школы проблему пытаются обойти, вводя исполнителя (например, «черепаху»), действующего как ЭВМ. В старших классах все-таки приходится обращаться к вычислению площадей, решению квадратных, нелинейных и других уравнений.

Но школьникам еще предстоит долгий путь до практического применения этих численных методов! Это произойдет, например, когда в вузе будут изучены математические модели реальных физических процессов. А главное, в развитии и использовании современной вычислительной техники наметился явный отход от вычислений. Вчерашнему школьнику предстоит столкнуться в своей практической деятельности с экспертными системами, системами обеспечения принятий решений, базами знаний и всем тем, что принято называть технологией обработки знаний или искусственным интеллектом. А это потребует и других навыков, и другого, «немашинного», мышления.

Возникает необходимость поиска новых, более адекватных средств общения с компьютерами; здесь нельзя не обратиться к активно развивающемуся сейчас логическому программированию и языку программирования Пролог.

Появившись в начале 70-х гг. в лаборатории искусственного интеллекта в университете г. Марселя, Пролог на протяжении почти десяти лет оставался известен лишь узкому кругу специалистов. Но тот факт, что логическое программирование и сам Пролог были положены в основу японского проекта ЭВМ пятого поколения, привлек к языку всеобщее внимание.

В отличие от традиционных языков, которые нужны для изложения способа решения задачи, на Прологе формулируется только ее постановка. Пользователь должен сообщить ЭВМ необходимые факты и несколько правил, описывающих, как эти факты соотносятся друг с другом, а она из полученной информации сформирует ответ на поставленный вопрос.

В основу Пролога положен сложный математический аппарат логики предикатов первого порядка и теории логического вывода. Однако, чтобы изучить этот язык и работать с ним, нет необходимости вникать в тонкости математической логики, как нет необходимости вникать в принципиальную схему телевизора, прежде чем включить его вечером.

Пролог очень легок для изучения и использования, но, несмотря на это, намного мощнее, чем любой другой язык, используемый в настоящее время на микро-ЭВМ. Часто оказывается, что программа на Прологе почти на порядок короче, чем на процедурном языке программирования.

Против Пролога существует предубеждение, что его эффективная реализация возможна только на больших, мощных ЭВМ. Действительно, в недалеком прошлом это было так, и в СССР Пролог действовал только на ЕС ЭВМ и СМ ЭВМ. Теперь ситуация изменилась. Оказалось, что его можно приспособить для быстрой и эффективной работы на небольших машинах и при этом создать развитые средства программной поддержки, ориентированные на пользователя. Сейчас существует реализация Пролога для всех микро-ЭВМ с операционной системой CP/M, в Институте программных систем АН СССР разработана реализация для микро-ЭВМ «Ямаха».

Эти достижения укрепили позицию Пролога как практичного, простого языка, пригодного для использования в школьной информатике. Широкое примерение языка такого типа планируется в болгарских школах для изучения гуманитарных наук; реализуется проект «Логика как язык компьютеров для детей». Аналогичные работы проводятся в СССР.

Программа на Прологе-Д

Опишем процесс программирования на упрощенном логическом языке, реализованном авторами и названном Пролог-Д.

В процедурных языках программа состоит из операторов. В совокупности операторы описывают этапы, необходимые для достижения заданной цели. Программа на Прологе-Д состоит из последовательности предложений (фактов и правил), порядок которых значения не имеет. Совокупность фактов и правил называют базой данных.

Вот пример простой программы:

```
МАМА (НАДЯ, АСЯ);                    (1)
МАМА (АСЯ, ДАША);                    (2)
БАБУШКА (А, В)←МАМА (А, С),
                МАМА (С, В);          (3)
```

В этой программе предложения (1) и (2) — факты, а предложение (3) — правило. В конце каждого предложения ставится точка с запятой. Факты дают представление об объектах и действиях. Предложение «Надя — мама Аси», записанное в виде факта МАМА (НАДЯ, АСЯ), можно было бы записать и как МАМА (АСЯ, НАДЯ), (исправив при этом (2) на МАМА (ДАША, АСЯ), а в (3) записав БАБУШКА (В, А)).

Важно, чтобы заданная последовательность аргументов сохранялась на всем протяжении программы.

Правило (3) определяет зависимость одних объектов или действий от других и записывается в общем виде так:

СЛЕДСТВИЕ←ФАКТ 1, ФАКТ 2;

Его можно прочесть так: «Если ФАКТ 1 верен, ФАКТ 2 верен, то СЛЕДСТВИЕ тоже верно». Левая часть правила — СЛЕДСТВИЕ — называется головой, а составляющие правой части — целями.

В программе (1)—(3) имена и слова БАБУШКА, НАДЯ, АСЯ, МАМА, ДАША называются атомами. Атомы — это константы, они не меняют своего значения при выполнении программы. Записываются атомы в виде последовательности букв, цифр и пробелов и начинаются с буквы. Константой является и целое, представляющее собой последовательность цифр.

В правиле (3) А, В, С — переменные. Область действия переменных в Прологе ограничивается одним предложением. Если переменная в голове правила принимает какое-либо значение, то его же она принимает и в целях. Переменные могут обозначаться одной буквой или последовательностью букв (заканчивающейся апострофом или нижним подчеркиванием, когда значение переменной несущественно).

Выражения вида (1) и (2) (в частности, выражение БАБУШКА (А, В)) называются термами.

Программа должна заканчиваться одним или несколькими вопросами. Вопрос записывается как факт, но впереди должен стоять вопросительный знак. Если, например, к программе (1)—(3) добавить вопрос

? БАБУШКА (НАДЯ, А);

то машина ответит:

А=ДАША

Вопрос должен соответствовать ряду условий:

атом, с которого начинается вопрос, должен стоять в голове хотя бы одного из правил базы данных;

вопрос не должен содержать новых атомов и переменных, в противном случае Пролог-Д ответит «не знаю».

На вопрос, не содержащий переменных, программа ответит «да», если в базе данных есть факт, совпадающий с вопросом, или «нет» в противном случае; если вопрос содержит переменные, то будут напечатаны все возможные их значения. Например, на

? МАМА (А, —);

машина ответит

А = НАДЯ
А = АСЯ

Немного о реализации языка

Поиск ответа на вопрос в Прологе-Д осуществляется доказательством теоремы, основанной на логических предложениях, из которых состоит программа. Доказательство осуществляется методом резолюций и основано на алгоритме, предложенном в работе: *Sammut R. A., Sammut C. A.* Implementation of UNSW—PROLOG // Austr. Comp. Journ. V. 15. P. 57—90 — и отличающемся достаточно высокой эффективностью.

Интерпретатор языка реализован на Бейсике для ДВК-2 и занимает 8К байт. Этот выбор обусловлен тем, что Бейсик широко распространен. Кроме того, хотя реализация интерпретатора Пролога-Д на интерпретаторе Бейсика и уменьшает скорость работы, зато обеспечивается определенная экономия памяти, необходимой для хранения программы на Прологе-Д.

В настоящее время разрабатывается версия Пролога-Д на Бейсике MSX для «Ямахи» и БК-0010. В ней несколько повышена скорость интерпретации.

Простые программы для разных уроков

География. База данных с фактами вида ГРАНИЧИТ (СТРАНА 1, СТРАНА 2) и одним правилом позволит создать информационную систему, дающую сведения о соседях каждой страны. Например, для юго-запада Европы:

```
ГРАНИЧИТ (ПОРТУГАЛИЯ,
          ИСПАНИЯ);
ГРАНИЧИТ (ИСПАНИЯ,
          ФРАНЦИЯ);
ГРАНИЧИТ (ФРАНЦИЯ,
          ШВЕЙЦАРИЯ);
ГРАНИЧИТ  (ФРАНЦИЯ,
          БЕЛЬГИЯ);
ГРАНИЧИТ (ФРАНЦИЯ, ФРГ);
ГРАНИЧИТ (ФРАНЦИЯ, ИТАЛИЯ);
ГРАНИЧИТ (ФРАНЦИЯ,
          ЛЮКСЕМБУРГ);
ГРАНИЧИТ (А, В)←ГРАНИЧИТ
                    (В, А);
```

На вопрос

```
? ГРАНИЧИТ (ИСПАНИЯ,
            ФРАНЦИЯ);
```

машина ответит «да», а на вопрос

```
? ГРАНИЧИТ (ИСПАНИЯ,
            БЕЛЬГИЯ);
```

ответ будет отрицательный.

Можно узнать и всех соседей Испании, задав вопрос

? ГРАНИЧИТ (ИСПАНИЯ, А);

Ответ будет:

А = ПОРТУГАЛИЯ
А = ФРАНЦИЯ

Легко сформулировать базу данных по принадлежности стран частям света, используя факты вида

НАХОДИТСЯ (СТРАНА,
ЧАСТЬ СВЕТА);

Аналогично можно заложить информацию о природных условиях и ресурсах каждой страны, городах, населении, промышленности и т. д.

История. Курс истории содержит большое количество фактов, поэтому историческая информация легко укладывается в конструкции Пролога-Д. Например, база данных с фактами вида СОБЫТИЕ (НАИМЕНОВАНИЕ, СТРАНА, ГОД НАЧАЛА, ГОД КОНЦА) позволит записать основные исторические события, относящиеся к той или иной эпохе. Формулируя различные вопросы, можно узнать, что, когда, где произошло. А добавив в вопросы операции сравнения над целыми числами, можно получить интересные сведения и по хронологии. База данных об исторических деятелях создается с использованием фактов вида

ИСТОРИЧЕСКИЙ ДЕЯТЕЛЬ (ИМЯ,
ГОД РОЖДЕНИЯ,
ГОД СМЕРТИ);

Иностранный язык. Достаточно просто написать программу, которая будет представлять собой англо-русский и русско-английский словари одновременно. С ее помощью отыскиваются эквиваленты русских и английских слов, проверяется словарный запас. База данных имеет вид

ЗНАЧЕНИЕ (МАМА, MAMMY);
ЗНАЧЕНИЕ (ПАПА, DADDY);
ЗНАЧЕНИЕ (БАБУШКА, GRANNY);
ЗНАЧЕНИЕ (МИР, PEACE);
ЗНАЧЕНИЕ (НЕБО, SKY);
и т. д.

Задав несколько вопросов, можно получить ответы:

? ЗНАЧЕНИЕ (МИР, А);
А = PEACE
? ЗНАЧЕНИЕ (А, SKY);
А = НЕБО
? ЗНАЧЕНИЕ (ПЕТЯ, А);
НЕ ЗНАЮ
? ЗНАЧЕНИЕ (МАМА, MAMMY);
ДА

Таким образом, используя минимальные средства логического программирования и создавая простейшие базы данных, можно строить интересные, полезные компьютерные системы для самых разных школьных уроков. Если же привлечь мощные средства Пролога для обработки списков, аппарат встроенных функций, то нетрудно построить простые системы автоматического перевода с языка на язык, аналитических преобразований. И как знать, может быть, скоро, овладев аппаратом логического программирования, школьники будут строить свои маленькие экспертные системы? Давайте попробуем поработать с Прологом!

М. БУРГИН
НИИ педагогики УССР

Проблемы преподавания основ информатики

В настоящее время нет единого взгляда на информатику как науку. Поэтому для объективного и обоснованного обсуждения содержания этого школьного курса необходимо установить, что же представляет собой информатика как научная дисциплина, какие области знаний в нее входят.

Руководствуясь критериями доступности, системности, эффективности, целесообразности и т. д., нужно отобрать соответствующий научный материал и трансформировать его в форму, адекватную задачам обучения и возрасту учащихся.

Существует много определений информатики. Ниже приводится определение, которое мы взяли за основу для выявления предметных областей, входящих в данную научную дисциплину, и для проведения анализа содержания школьного курса основ информатики и вычислительной техники.

Информатика — это наука, предметом которой являются процессы и системы получения, хранения, передачи, распространения, использования и преобразования информации.

В школьном курсе такое определение должно сопровождаться детальными пояснениями, различными примерами. Обязательно следует указать, что главными техническими средствами, выполняющими все

---

* Всесоюзный семинар «Компьютер и образование», март, 1987 г.— Тезисы доклада.

вышеперечисленные действия, являются вычислительные системы. Для понимания этого определения (как и большинства других) школьнику необходимо иметь представление о таких понятиях, как процесс, система, информация.

Среди предметных областей информатики, каждая из которых, в свою очередь, делится на более мелкие, можно выделить следующие:

1. Теория алгоритмов и алгебра логики.
2. Теория структур данных.
3. Теория информации.
4. Теория баз данных.
5. Программное обеспечение вычислительной техники.
6. Аппаратные средства вычислительных систем.
7. Численные методы.
8. Моделирование.
9. Интерактивные системы.
10. Языки программирования.
11. Обучающие системы.
12. Искусственный интеллект.

Выделение предметных областей информатики позволяет провести анализ уже имеющегося курса основ информатики и вычислительной техники и сформулировать некоторые пожелания. Естественно, в достижении основной цели — дать учащимся больше полезных и прочных знаний, добиться более глубокого понимания предмета, привить им новые умения и навыки — нужно избегать дополнительной перегрузки курса и идти путем логичного отбора и адекватного представления учебного материала.

Остановимся на двух первых предметных областях информатики.

Алгоритмы представлены в школьном курсе хорошо, тем более что количество часов, отведенное на весь курс, невелико. Несмотря на отдельные неточности и несогласованности в используемом учебном пособии (которые, можно надеяться, авторы устранят при его переработке), учащиеся могут ознакомиться с различными видами алгоритмов и получить довольно эффективное средство их описания и построения — *алгоритмический язык*. Конечно, не все доступные на школьном уровне вопросы вошли в существующий курс. Примером может служить *эффективность* алгоритмов. Нередко те или иные задачи не решаются только из-за того, что все известные алгоритмы их решения неэффективны и поэтому не могут быть практически реализованы. Однако ограниченность учебного времени является аргументом против включения таких вопросов в курс общеобразовательной школы. Речь может идти только о будущем, когда количество часов в школьной программе будет увеличено, либо о включении небольшого объема подобного материала в курсы специализированных физико-математических школ.

Несмотря на нередко высказываемые мнения о ненужности специального алгоритмического языка и необходимости изучать только конкретные языки программирования, польза от такого языка, на наш взгляд, может быть большая и многосторонняя. Описанный в пособии алгоритмический язык построен по образцу большинства современных языков программирования высокого уровня, которые по своей структуре ориентированы на естественные языки. Это одновременно и достоинство, и недостаток. Отметим, что во второй части учебного пособия «Основы информатики и вычислительной техники» алгоритмический язык определенным образом усовершенствован — в записи алгоритма используются дополнительные линии, выявляющие его структуру.

Педагогическая практика показывает, что эффективным средством усвоения понятия «алгоритм», приемов и методов алгоритмизации, основных алгоритмических конструкций являются *блок-схемы*. Многие из учителей, преподающих основы информатики и вычислительной техники (особенно те, кто прослушал ранее курс программирования), успешно используют их в своей педагогической деятельности. Блок-схемы обладают значительно большей наглядностью, чем обычные языки программирования (как высокого, так и низкого уровня). Следовательно, они более эффективны (по крайней мере как средство обучения), а один из принципов современной дидактики требует представления учебного материала по возможности в наиболее наглядной форме. Поэтому вызывает удивление то, что в существующем учебном пособии блок-схемы почти не используются.

Кроме того, было бы полезно дать учащимся представление о существовании в математике большого и бурно развивающегося раздела — теории алгоритмов и кратко, не вводя формальных определений, коснуться тех его аспектов, которые связаны с практическим использованием алгоритмов.

Со **структурами данных** в существующем и в планируемом курсах информатики дело обстоит значительно хуже, чем с алгоритмами. И в принятой программе, и в имеющемся пособии они представлены недостаточно, хотя их роль в информатике не менее значительна, чем роль алгоритмов. Структуры данных являются (наряду с алгоритмом, информацией и ЭВМ) одним из основных понятий информатики. Необходимо

отметить, что данные — это средство представления реального мира, т. е. носители информации. Поэтому все процессы сбора, хранения, передачи, распространения, использования и преобразования информации осуществляются только посредством работы с данными.

В связи с этим целесообразно, на наш взгляд, ввести в школьный курс информатики специальный раздел, посвященный структурам данных, поместив его в учебном пособии и программе до разделов, относящихся к алгоритмам.

Желательность выделения в курсе основ информатики и вычислительной техники раздела, посвященного данным, подтверждается и уже имеющимся опытом преподавания этого предмета в школах страны. Действительно, если школьники в основном успешно справляются с такими типами (структурами) данных, как целые и вещественные числа, которые они уже изучали на уроках математики и использовали на уроках физики и других предметов, то литерный тип данных (или литерные величины, как они называются в программе и в учебном пособии) вызывает существенные затруднения. Кроме того, в учебном пособии явным образом почти ничего не сказано о том, как с помощью этого типа данных (или величин) представлять реальные объекты, т. е. строить вычислительные модели, и почему данные являются носителями информации.

В связи со структурами данных необходимо сделать и чисто терминологическое замечание. В пособии для обозначения тех объектов, с которыми работают алгоритмы и программы, употребляется термин «величина». Однако в математике (и физике), где понятие величины возникло и широко используется, оно всегда связывается с числами (или по крайней мере с набором чисел). В пособии по информатике это понятие расширяется настолько, что практически любые объекты могут быть величинами. Это вряд ли оправдано и потому, что понятие величины уже сложилось (школьники с ним до определенной степени знакомы), и потому, что в программировании успешно используется термин «данные», а в последнее время — и «структуры данных». Величины же являются некоторым специальным видом данных.

В заключение хотелось бы подчеркнуть, что проведенный анализ и выдвигаемые предложения не должны изменить практическую направленность курса основ информатики и вычислительной техники. Их главная цель заключается в выделении и достоинств существующего курса (которых немало, в чем заслуга его авторов) для их дальнейшего развития, и определенных недостатков, которые в каждом новом деле неизбежны, но которые следует устранять.

Разработанный нами проект программы предполагает совершенствование существующих программ, лучшие компоненты которых он включает.

ОСНОВЫ ИНФОРМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

Проект программы средней общеобразовательной школы

IX, X классы (102 ч)

1. Введение в предмет и первоначальное знакомство с ЭВМ (5 ч).
2. Структуры данных как модели реальных объектов, их свойств и как средство представления информации (12 ч).
3. Алгоритмы как средство решения задач (20 ч).
4. Программы как средство решения задач (включая основы программирования) (12 ч).
5. Основы вычислительных систем, включая математическое и программное обеспечение (12 ч).
6. Применение вычислительных систем для решения задач (35 ч).
7. Вычислительные системы в современном обществе (6 ч).

А. ШЕНЬ

Информатика в IX классе

20 **Глубокоуважаемый коллега-учитель! Предлагаем вашему вниманию методические разработки по курсу «Основы информатики и вычислительной техники». Когда создавалась первая часть пробного учебного пособия и книги для учителей, опыт преподавания основ информатики и вычислительной техники практически отсутствовал. За прошедшее время он появился и в какой-то мере отражен в предлагаемых разработках, но прежде всего мы призываем вас руководствоваться собственным опытом при распределении материала по урокам, выборе разбираемых примеров и т. д.**

Введение

Роль ЭВМ в современном обществе

Цель темы — создать у школьников представление (хотя бы самое приблизительное) о возможностях современных ЭВМ и их приложениях. Помимо приведенных в первой части пробного учебного пособия (далее — УП) материалов рекомендуем воспользоваться сведениями из второй части УП.

Первоначальные сведения об ЭВМ

Если в предыдущей теме объяснялось, что может сделать ЭВМ, то теперь нужно дать хотя бы самое примитивное представление о том, как ЭВМ это делает. Основные моменты:

информация представляется в ЭВМ в виде электрических сигналов;

обработку информации производит *процессор* в соответствии с хранящейся в памяти *программой;* одна и та же ЭВМ может решать совсем разные задачи, в зависимости от того, какая программа в нее заложена;

технические характеристики ЭВМ очень быстро совершенствуются.

Рекомендуем познакомиться с материалами второй части УП, содержащими более подробное изложение сведений об устройстве ЭВМ, истории вычислительной техники и др.; кроме того, советуем не слишком задерживаться на введении и как можно скорее идти дальше: впереди ждет трудный материал, и сэкономленное время очень пригодится.

Упражнения на с. 15 УП предназначены прежде всего для того, чтобы создать представление о технических возможностях современных ЭВМ. Ниже приводятся их решения.

1. На этот подсчет уходит около минуты; всего в УП около 100 страниц, на каждой около 50 строк, на каждую страницу уйдет примерно 5 минут, всего 500 минут, а компьютер тратит 1/6 минуты, т. е. работает в 3000 раз быстрее.

2. 1 рубль = 25 кВт·ч электроэнергии (внимание: киловатты умножаются на часы, а не делятся, как в УП!), т. е. компьютер мощностью в 100 Вт = = 1/10 кВт может работать 250 часов = = 250 · 3600 секунд = 900 000 секунд.

За каждую из них он делает 100 000 операций, так что всего будет 90 000 000 000 (90 млрд.) операций.

3. 24 строки по 80 символов занимают (считая, что символ занимает 1 байт) около 2000 байтов ≈2К байта. Поскольку 1М байт равен 1024К байт, то информация занимает около 2/100 000 части диска, т. е. примерно 0,02 %.

4. При проведении такого эксперимента на чтение одного абзаца из 8 строк ушло 15 секунд (около 2 секунд на строчку из примерно 60 символов). Скорость чтения — около 30 символов в секунду. Скорость письма: чтобы написать это предложение (а в нем примерно 100 символов), понадобилось около 30 секунд. Скорость письма — примерно 3 символа в секунду. Чтобы прочесть вслух то же предложение, требуется около 6 секунд, поэтому скорость чтения вслух — примерно 15 символов в секунду.

5. Если в секунду произносится около 15 символов, символ занимает 1 байт, то на заполнение памяти уйдет около 4 тыс. секунд — больше часа.

6. Будем считать, что объем школьного сочинения — три машинописные страницы (≈6000 символов). Тогда на диске уместится около 90 сочинений.

7. На соответствующей странице УП — около 15 строк, в каждой — около 50 символов. На печать одной строки уходит примерно половина секунды, значит, всего потребуется 8 секунд.

8. Для выполнения каждой операции сигнал должен пройти от процессора к памяти и обратно — это 60 см=0,6 м. За секунду ему пришлось бы пройти путь 600 000 000·0,6 м = 360 000 000 м = =360 000 км, т. е. его скорость должна быть не меньше 360 000 км/с, что больше скорости света (300 000 км/с), являющейся предельной.

Раздел 1. Алгоритмы. Алгоритмический язык

§ 1. Алгоритм и его свойства

Мы дадим рекомендации сразу по двум пунктам (понятие алгоритма и формальное исполнение алгоритма), поскольку их материал тесно связан.

Прежде всего одно общее замечание. Основной принцип изложения в УП — обучение на примерах. Поэтому не следует пытаться добиться от школьников знания «определения» алгоритма или его «свойств» или долго обсуждать словесные формулировки, приведенные в УП.

Основным примером является алгоритм Евклида. Хотя он и весьма древний, его понимание требует некоторого знакомства с математикой. Мы приведем его подробное обоснование, которое предназначено прежде всего для вас. Что окажется возможным рассказать на уроках, зависит от математических знаний ваших учеников. В сомнительных случаях стоит воспользоваться советом из методического письма «О преподавании курса «Основы информатики и вычислительной техники» в 1986/87 учебном году» и обсудить алгоритм Евклида и алгоритм игры Баше (следующий пример) после изучения составных команд, если останется время.

Алгоритм Евклида можно сформулировать так.

На доске пишутся два натуральных числа. Затем повторяется одно и то же действие:

пока числа не станут равными, большее число заменяется на разность большего и меньшего чисел.

Пример: 15,39→15,24→15,9→ 6,9→6,3→3,3.

Как видим, алгоритм очень прост: чтобы его выполнить, нужно только уметь вычитать. Замечательно, что такой простой алгоритм позволяет решить довольно сложную задачу — найти наибольший общий делитель двух натуральных чисел. Тут нужны некоторые напоминания:

натуральное число a называется делителем натурального числа b, если b делится на a без остатка, т. е. если b/a — целое число;

общие делители чисел m и n — те числа, которые одновременно являются делителями и m, и n;

наибольшее число, являющееся общим делителем m и n, называется наибольшим общим делителем чисел m и n (обозначение НОД (n, m)).

Так вот, оказывается, что применение к паре чисел (m, n) алгоритма Евклида дает в конце концов пару одинаковых чисел, которые равны НОД (m, n).

Остановимся и удивимся. Ведь в описании алгоритма Евклида нет ни слова о делителях или делении; единственная операция, которая в нем используется,— это вычитание. Почему же в результате его работы получается именно наибольший общий делитель, а не что-нибудь другое?

Отметим для начала, что

при замене большего числа на разность большего числа и меньшего их наибольший общий делитель не меняется: НОД (m, n) = НОД$(m-n, n)$ (при $m>n$).

(В нашем примере было НОД (15,39) = НОД (15,24) = НОД (15,9) = НОД (6,9) = НОД (6,3) = НОД (3,3) = 3.) Это верно потому, что у пары чисел (n, m) и у пары $(m-n, n)$ одни и те же общие делители, и, следовательно, один и тот же наибольший общий делитель. Чтобы доказать это, нужно использовать такое утверждение:

сумма и разность двух чисел, делящихся на некоторое число d, делится на это число d.

Доказывается оно просто: если m/d — целое число и n/d — целое число, то $(m+n)/d = m/d + n/d$ и $(m-n)/d = m/d - n/d$ — тоже целые числа. Отсюда следует, что если d — общий делитель чисел m и n, то он является и общим делителем $m-n$ и n. Если же d является общим делителем чисел $m-n$ и n, то и их сумма $(m-n)+n=m$ делится на d, так что d — общий делитель m и n.

Теперь секрет алгоритма ясен: в процессе его работы наибольший общий делитель не меняется, и в конце, когда числа станут равными, они будут наибольшим общим делителем. Ведь общий делитель двух равных чисел — это любое из них.

Выделим еще раз основные этапы этого рассуждения:

если m и n делятся на d, то $m+n$ и $m-n$ делятся на d;

общие делители у чисел $(m-n)$ и n — те же, что у чисел m и n;

НОД$(m-n, n)$ = НОД(m, n);

при выполнении алгоритма Евклида наибольший общий делитель пары чисел остается неизменным;

алгоритм Евклида кончает работу, когда остаются равные числа; поскольку НОД$(x, x) = x$, то x равен наибольшему общему делителю исходной пары чисел.

Подчеркнем, что приведенные рассуждения нужны лишь для обоснования алгоритма Евклида; для его исполнения ничего этого не нужно — достаточно уметь проверять, равны ли числа, указывать, какое из чисел больше, и вычитать меньшее из большего.

Еще один вопрос, связанный с алгоритмом Евклида: почему его работа рано или поздно закончится? В УП в этом месте ошибка, повторенная в методическом пособии для учителей. Правильное объяснение состоит в том, что с каждым шагом максимум чисел уменьшается, оставаясь положительным, а это не может происходить бесконечно много раз.

Подробный разбор алгоритма Евклида может занять целый урок, а то и больше. Однако это время не будет потрачено зря.

Другой интересный пример, имеющийся в УП,— алгоритм выигрыша в игре Баше с 15 предметами. Советуем, не сообщая алгоритма школьникам, дать возможность им поиграть друг с другом в эту игру (это удобно сделать, нарисовав на доске или бумаге 15 палочек и стирая или зачеркивая их). При этом некоторые из них сами могут изобрести алгоритм выигрыша; во всяком случае они лучше его оценят.

Почему приведенный в УП алгоритм действительно приводит к выигрышу? После первого хода остается 13 предметов. Затем (после ходов обоих противников) остается 13 − 4 = 9, потом 9 − 4 = 5, и наконец остается один предмет. Его вынужден взять последний из играющих.

Интересно также проанализировать игру Баше с другим количеством предметов. Составим таблицу, показывающую, является ли позиция с данным числом предметов выигрышной или проигрышной для того, кто в ней окажется. Позиция с одним предметом — проигрышная по определению. Позиции с 2, 3 и 4 предметами — выигрышные:

находясь в них, можно поставить противника в проигрышное положение, оставив ему один предмет (взяв 1, 2 или 3 предмета). Позиция с 5 предметами является проигрышной: какой ход ни делай, останется 2, 3 или 4 предмета, а эти ситуации выигрышные, и противник сможет выиграть (если будет играть правильно). Раз позиция с 5 предметами — проигрышная, то позиции с 6, 7 и 8 предметами — выигрышные: из них можно, взяв 1, 2 или 3 предмета соответственно, поставить противника в проигрышное положение, оставив ему 5 предметов. Позиция с 9 предметами — снова проигрышная: после любого хода противнику остается выигрышная позиция с 6, 7 или 8 предметами. Позиции с 10, 11 и 12 предметами — снова выигрышные, с 13 — проигрышная, с 14, 15 и 16 — снова выигрышная и т. д.

Предметов	1	2	3	4	5	6	7	8
Позиция	П	В	В	В	П	В	В	В
Предметов	9	10	11	12	13	14	15	
Позиция	П	В	В	В	П	В	В	

Из этого рассуждения видно, что алгоритм выигрыша состоит в том, чтобы оставлять противника в проигрышной позиции — с 13, 9, 5 и, наконец, с 1 предметом.

Рассмотренные примеры показывают, что смысл составления алгоритма в том, чтобы разбить решение интересующей нас задачи на отдельные простые шаги (команды). Например, алгоритм Евклида разбивает одну команду «найти наибольший делитель» на отдельные шаги, каждый из которых требует лишь вычитания. Исполнитель этого алгоритма должен уметь лишь сравнивать числа и вычитать большее число из меньшего. Составляя алгоритм для заданного исполнителя, нужно учитывать его систему команд — перечень доступных ему действий.

Разбиение решения задачи на отдельные шаги иллюстрируется в УП, пример 1.1. Там составляется алгоритм вычисления выражения $y=(Ax+B)/(Cx-D)$ для исполнителя, который может выполнять арифметические действия с парой чисел и запоминать их результат, но не может производить вычисления по более сложным формулам. Аналогичная задача содержится в упражнении 1 на с. 21. Обратная задача (восстановление формулы по алгоритму) поставлена в упражнении 2. Эти упражнения можно использовать на уроке. Можно также предложить части учеников составить по формуле алгоритм вычисления ее значения, а затем передать этот алгоритм другой части, чтобы те восстановили формулу (не зная ее заранее).

В УП есть и другие примеры, но мы не рекомендуем разбирать их все — лучше хорошо разобрать один, чем пройти мимоходом много. Не следует также задерживаться на вопросах формального, словесного характера (типа приведенных на с. 21 УП).

Решения упражнений на с. 21—22 УП.

1. а)

Умножить x на 2, результат обозначить R1 | R1 := $2x$
Сложить R1 с 3, результат обозначить R2 | R2 := R1+3
Умножить x на 7, результат обозначить R3 | R3 := $7x$
Вычесть 5 из R3, результат обозначить R4 | R4 := R3—5
Разделить R2 на R4, результат обозначить y | y := R2/R4

Справа приведена сокращенная запись в том виде, в каком она будет встречаться впоследствии.

б) R1 := x—3; R2 := R1*R1; R3 := =2—R2; R4 := R2+4; y := R3/R4.

2. а) При выполнении этого алгоритма $R1=x^2$, $R2=a\cdot R1=ax^2$, $R3=R2+b=ax^2+b$, $y=R3/c=(ax^2+b)/c$.

Ответ: $y=(ax^2+b)/c$.

б) $A_1=x+1$, $A_2=1/A_1=1/(x+1)$, $A_3=A_2+1=1/(x+1)+1$, $A_4=A_2-1=1/(x+1)-1$, $A_5=A_4/A_3=(1/(x+1)-1)/(1/(x+1)+1)$.

Ответ: $y=A_5-1=(1/(x+1)-1)/(1/(x+1)+1)-1$.

3. Решение этой задачи хорошо известно из курса математики, а также приведено в методическом пособии для учителей на с. 38.

4. В соответствии с алгоритмом первый возьмет два предмета. Если после этого второй возьмет 1 предмет, оставив

первому 17, то поставит первого в проигрышное положение.

Упражнения 5 и 6 использовать не рекомендуем.

7. Шаг 3 (вычитание) будет выполняться 10 раз: 100, 18→82,18→64,18→ →46,18 → 28,18 → 10,18 → 10,8 → 2,8 → →2,6→2,4→2,2. (В методическом пособии дан неверный ответ.)

§ 2. Алгоритмический язык

Алгоритмический язык предназначен для записи алгоритмов. Переход от словесных описаний алгоритмов к их записям на алгоритмическом языке играет в информатике ту же роль, что и введение буквенных обозначений в алгебре.

На его изучение отводится 4 урока, но можно занять и большее время. Не следует торопиться: лучше (временно) отстать от программы, чем излишней спешкой сделать дальнейшие уроки бессмысленными.

Общие правила алгоритмического языка

Правила записи алгоритмов на алгоритмическом языке будут постепенно уточняться. Начнем с внешнего оформления алгоритмов. Это оформление (с. 23 УП) можно проиллюстрировать на примере 3.1 или 3.2. После разбора примера следует предложить учащимся самостоятельно оформить какие-либо известные им алгоритмы в виде текстов на алгоритмическом языке.

1.

алг сделать чай сладким
нач
положить сахар
размешать
кон

Что изменится, если мы переставим команды алгоритма «положить сахар» и «размешать»? (О т в е т. Сахар останется неразмешанным.)

Этот пример показывает, что порядок команд в алгоритме очень важен.

2.

алг кипячение
нач
налить воду в чайник
поставить чайник на плиту
зажечь газ
кон

Здесь некоторые команды можно переставить. Можно зажечь газ сначала, а поставить чайник на плиту потом. При этом получится другой, но тоже правильный алгоритм. Но если мы переставим первую и третью команды и будем действовать в таком порядке: зажечь газ — поставить чайник на плиту — налить воду в чайник, то до выполнения третьей команды чайник, стоя на огне без воды, может испортиться.

Предлагая школьникам записывать алгоритмы на алгоритмическом языке, нужно иметь в виду, что в этот момент они могут записать лишь простейшие алгоритмы, сводящиеся к последовательному выполнению команд. Для записи более сложных алгоритмов (например, алгоритма Евклида) на алгоритмическом языке необходимы средства, пока не известные ученикам (составные команды).

Общие правила записи очень просты: по существу, они сводятся к тому, что перед началом алгоритма нужно написать служебное слово **алг**, название и служебное слово **нач**, а в конце — служебное слово **кон.**

Алгоритмы работы с графической информацией

Хотя эти алгоритмы описаны в конце первой части УП, в соответствии с рекомендациями методического письма материал по ним перенесен в начало курса.

Исполнителя алгоритмов работы с графической информацией можно представить себе как маленькую черепашку, движующуюся по плоскости и оставляющую след. Система команд этого исполнителя такова:

вперед (*n*)
назад (*n*)
налево (А)
направо (А)
рисуй
не рисуй

По команде «вперед (*n*)» черепашка движется на расстояние *n* единиц вперед, по команде «назад (*n*)» — назад. По команде «направо (А)» черепашка

поворачивается на угол в А градусов по часовой стрелке, по команде «налево (А)» — против часовой стрелки. По команде «не рисуй» черепашка перестает оставлять след, по команде «рисуй» — вновь начинает.

Упражнения.

1. Выполнены команды

вперед (2)
назад (2)

Где оказалась черепашка? Что нарисовано? (Ответ. Черепашка находится в исходном положении, видя перед собой нарисованный ею отрезок длины 2.)

2. Есть ли разница между действием серии команд

вперед (2)
вперед (3)

и команды «вперед (5)»? (Ответ. Нет; и в том и в другом случае черепашка сдвинется на 5 единиц вперед и нарисует отрезок длины 5.)

3. Есть ли разница между серией команд

вперед (2)
назад (3)

и командой «назад (1)»? (Ответ. Положение черепашки будет одинаковым, однако нарисовано будет разное: в первом случае будет нарисован отрезок длины 3, а во втором — отрезок длины 1.)

4. Можно ли найти такое А, чтобы команды «налево (А)» и «направо (240)» делали бы одно и то же? (Ответ. Можно, например: А=120.)

5. Есть ли разница между серией команд

налево (180)
вперед (1)

и командой «назад (1)»? (Ответ. Есть. Хотя в обоих случаях черепашка сдвинется одинаково, в первом она повернется на 180 градусов, а во втором сохранит первоначальную ориентацию.)

6. Как составить серию команд, которая производила бы то же самое действие, что и команда назад (1), но не содержала бы команд «назад»? (Ответ. Например, годится такая серия:

налево (180)
вперед (1)
налево (180))

Разобравшись с системой команд исполнителя, можно составлять простейшие алгоритмы. Вот несколько примеров.

1)

```
алг правильный треугольник со стороной 4
нач
    вперед (4)
    направо (120)
    вперед (4)
    направо (120)
    вперед (4)
кон
```

2)

```
алг буква М
нач
    вперед (4)
    налево (30)
    назад (2)
    направо (60)
    вперед (2)
    налево (30)
    назад (4)
кон
```

3)

```
алг буква И
нач
    вперед (4)
    назад (4)
    направо (30)
    вперед (4,8)
    налево (30)
    назад (4)
кон
```

4)

```
алг буква Р
нач
    вперед (4)
    направо (90)
    вперед (2)
    направо (90)
    вперед (2)
    направо (90)
    вперед (2)
кон
```

5)

```
алг шаг вправо
нач
    не рисуй
    направо (90)
    вперед (1)
    налево (90)
    рисуй
кон
```

Теперь можно составить алгоритм рисования слова МИР. Чтобы нарисовать это слово, необходимо последовательно выполнить такие действия:

написать букву М;
сделать шаг вправо;
написать букву И;
сделать шаг вправо;
написать букву Р.

Записав последовательно соответствующие команды, получим алгоритм, приведенный в УП (с. 80, пропущены лишь команды «рисуй» и «не рисуй» при написании буквы И — они лишние).

К этому примеру полезно вернуться при изучении темы «Вспомогательные алгоритмы». Ведь, используя построенные ранее алгоритмы как вспомогательные, можно записать алгоритм рисования слова МИР совсем просто:


```
алг написание слова «МИР»
нач
    буква М
    шаг вправо
    буква И
    шаг вправо
    буква Р
кон
```

Раздел «Алгоритмы работы с графической информацией» содержит еще некоторые примеры, которые пока использовать нельзя, так как они предполагают знакомство с составными командами, величинами и другими еще не изученными понятиями.

Составные команды

Команда ветвления. При составлении алгоритмов да и просто в жизни часто встречаются ситуации, когда необходимо сообразовываться с обстоятельствами, выполняя то или иное действие. Мы одеваемся, выходим из дому. Если на улице холодно, то надеваем пальто, а иначе — куртку. Выйдя на улицу, мы смотрим, идет ли автобус. Если он есть, то садимся и едем на автобусе, а иначе идем пешком. На алгоритмическом языке это запишется так:

```
алг одеться
нач
    если на улице холодно
      то
        надеть пальто
      иначе
          надеть куртку
    все
кон
и
алг маршрут
нач
    если подошел автобус
      то
          ехать на автобусе
      иначе
          идти пешком
    все
кон
```

Смысл команды ветвления (в полной форме, с. 25 УП) часто оказывается непонятным. Нужно обратить внимание в первую очередь на следующее.

1. Слова **если, то, иначе, все** не являются командами и не исполняются. Это служебные слова, указывающие порядок выполнения команд. Их роль можно сравнить с ролью знаков препинания в русском языке или скобок в алгебраических формулах.

2. При выполнении команды ветвления выполняется либо серия команд между **то** и **иначе**, либо серия команд между **иначе** и **все**. Одна из этих серий обязательно будет выполнена. Ни в каком случае не будут выполнены обе серии. Завершив выполнение одной из серий, исполнитель тем самым завершает выполнение команды ветвления (после чего переходит к следующей команде, если она есть).

Употребление слов русского языка (**если, то, иначе**) при записи алгоритмов имеет и преимущества, и недостатки. Преимущество состоит в том, что служебные слова напоминают о смысле соответствующих конструкций и позволяют читать алгоритм почти как текст на русском языке. С другой стороны, аналогия с русским языком может служить источником недоразумений. Вот один из многочисленных примеров ошибочного употребления служебных слов (сообщен Ю. Махлиным):

```
если нога поднята
  то
```

```
    опустить ногу
иначе
    упадешь
все
```

Приведем несколько примеров команд ветвления.

```
если последняя цифра числа равна 0 или 5
то
    сообщить, что число делится на 5
иначе
    сообщить, что число не делится на 5
все
```

```
если первое число меньше второго
то
    вычесть первое число из второго
иначе
    вычесть второе число из первого
все
```

Сокращенная форма команды ветвления. Часто оказывается, что в одном из двух возможных в команде ветвления случаев никаких действий предпринимать не нужно. Тогда применяется сокращенная форма команды ветвления (с. 25 УП). Ее отличие от полной состоит в том, что отсутствуют слово **иначе** и серия идущих за ним команд. Связь сокращенной формы команды ветвления с полной можно пояснить так: команда ветвления в сокращенной форме

```
если условие
то
    серия
все
```

эквивалентна команде ветвления в полной форме

```
если условие
то
    серия
иначе
    ничего не делать
все
```

где «ничего не делать» — команда, не требующая никаких действий от исполнителя.

В качестве примера использования сокращенной формы команды ветвления разберем алгоритм включения электроприбора в сеть 220 В, приведенный на с. 26 УП. При этом можно использовать следующие вопросы.

1. Останется ли алгоритм верным, если электроприбор имеет переключатель на три напряжения — 110, 127 и 220 В? (Ответ. Нет, не останется: если переключатель был установлен на 110 В, то он останется в таком положении и при включении в сеть прибор сгорит.)

2. Как исправить алгоритм, чтобы он оставался верным и для прибора с переключателем на 3 напряжения?

Ответ.

```
алг включение электроприбора
    в сеть 220 В
нач
    если переключатель прибора
        установлен не на 220 В
    то
        установить переключатель
        прибора на 220 В
    все
    включить вилку в розетку
кон
```


Другой пример алгоритма с использованием команд ветвления таков:

```
алг завтрак
нач
    если картошка не очищена
    то
        очистить картошку
        сварить картошку
    все
    съесть картошку
кон
```

Правилен ли этот алгоритм? Нет: если картошка была очищена, то исполнителю придется есть ее сырой. Как его исправить? Например, так:

```
алг завтрак
нач
    если картошка не очищена
    то
        очистить картошку
        сварить картошку
    иначе
        сварить картошку
    все
    съесть картошку
кон
```

Другой вариант исправления, использующий сокращенную форму команды ветвления, таков:

```
алг завтрак
нач
    если картошка не очищена
      то
          очистить картошку
    все
    сварить картошку
    съесть картошку
кон
```

Еще одна задача. Ученик, начав записывать алгоритм определения кислотности раствора (УП, пример 4.3), ошибся в первом условии, написав «если бумажка синяя» вместо «если бумажка красная». Как он должен изменить оставшуюся часть алгоритма, чтобы алгоритм снова стал правильным?

Ответ. Например, так:


```
если бумажка синяя
    то
      ответ: раствор щелочной
    иначе
      если бумажка красная
          то
            ответ: раствор кислотный
          иначе
            ответ: раствор нейтральный
    все
все
```

Команда повторения. Это центральная и, наверно, самая трудная тема курса информатики. К сожалению, программа отводит на нее до смешного мало времени, а в УП мало примеров и почти нет упражнений. Поэтому приведем большое количество примеров, из которых можно выбрать наиболее подходящие для использования на уроках, в качестве домашних заданий и т. п.

Пример 1.

```
пока переключатель каналов не установлен на 10
нц
  повернуть ручку переключателя на одно деление вправо
кц
```

Этот пример иллюстрирует такое свойство команды повторения: после ее выполнения условие не соблюдается, т. е. переключатель установлен на 10. Вопрос: сколько раз будет выполняться команда, входящая в команду повторения, если переключатель установлен на 9? на 1? на 11? на 10? Ответы: 1 раз, 9, 11 (всего каналов 12), 0 раз.

Пример 2. Команда «провести воспитательную работу» может быть описана так:

```
пока нарушитель не раскаялся
нц
  провести воспитательную беседу с нарушителем
кц
```

В результате выполнения этой команды беседа повторяется до тех пор, пока это необходимо. Отметим три важных обстоятельства: если нарушитель раскаялся с самого начала, то беседа не проводится ни разу; команда «провести воспитательную работу» может выполняться бесконечно долго (или, как говорят, зациклиться); но если она кончит работу, то нарушитель окажется раскаявшимся.

Пример 3 (подготовительный к примеру из УП).

```
пока ведро не полно
нц
  долить в ведро литр воды
кц
```

Этот пример иллюстрирует то же свойство: после выполнения команды условие не соблюдается, т. е. ведро полно. Вопрос: сколько раз будет выполняться команда, если в 7-литровом ведре был 1 л? 6 л? 7 л? 3,5 л? Ответы: 6 раз; 1 раз; 0 раз; 4 раза. В последнем случае пол-литра воды проливается. Это происходит потому, что условие в команде повторения проверяется не во время выполнения серии входящих в нее команд, а перед выполнением. Еще вопрос: можно ли придумать такие условия (такую емкость ведра и количество налитой в него воды), чтобы после выполнения этой команды ведро было бы не полным? Ответ: нельзя, после выполнения команды повторения входящее в нее условие не соблюдается.

Пример 4 (4.4 по УП, с. 28) иллюстрирует еще более наглядно, что в середине выполнения серии команд проверка не производится. В качестве упражнения в УП предлагается изменить алгоритм так, чтобы вода не проливалась. Это можно сделать различными способами, один из них таков:

```
пока ведро не полно
нц
  долить 1 л холодной воды
  если ведро не полно
    то
      долить 1 л горячей воды
  все
кц
```

Пример 5.

```
алг завтрак
нач
  пока в тарелке не меньше ложки каши
  нц
    съесть ложку каши
  кц
  выпить чай
кон
```

Сколько каши останется несъеденной в тарелке? Меньше ложки, так как после выполнения команды повторения условие не соблюдается. Что было бы, если бы «не меньше» заменить на «больше»? В этом случае можно было бы утверждать, что в тарелке останется ложка каши или меньше. Что было бы, если мы бы поменяли местами строки «кц» и «выпить чай»? Новый алгоритм требовал бы запивать стаканом чая каждую ложку каши.

Пример 6.

```
пока написанное на доске число меньше 10
нц
  увеличить написанное число на 1
кц
```

Что будет после выполнения команды, если вначале было написано число 7? 10? 11? (Ответ: 10; 10; 11.)
Сколько раз в каждом из этих случаев выполнится команда повторения? (Ответ: 3, 0, 0.)

Пример 7.

```
пока написанное на доске число больше 10
нц
  увеличить написанное число на 1
кц
```

Что будет, если на доске до выполнения команды повторения было написано 7? 10? 11? (Ответ. 7; 10; в третьем случае выполнение команды повторения не закончится никогда.)

Два предыдущих примера легко видоизменить и использовать для упражнений.

Пример 8. Объяснить разницу между командами

```
пока в тарелке не меньше ложки каши
нц
  съесть ложку каши
кц
```

и

```
если в тарелке не меньше ложки каши
  то
    съесть ложку каши
все
```

(Ответ. Во втором случае исполнитель съест только одну ложку каши (если она была) и остановится. В первом случае он будет есть кашу, пока это возможно, и остановится, только когда каши останется меньше ложки.)

Пример 9. На доске написано целое число. Выполняется такая команда:

```
пока число больше или равно 10
нц
  уменьшить число на 10
кц
```

Что останется, если вначале было написано 15? 10? — 15? 123456789? (Ответ: 5; 0; — 15; 9.)

Последний ответ объясняется так: после каждого выполнения команды, входящей в команду повторения, последняя цифра числа будет равна 9 (так как вначале она равна 9 и не меняется при вычитании 10 из числа, большего или равного 10), а в конце число должно быть меньшим 10, т. е. однозначным.

Пример 10.

```
пока число на доске не меньше 7
нц
  уменьшить написанное число на 7
кц
```

Что останется после выполнения команды, если вначале на доске было написано натуральное число A? Останется остаток от деления A на 7. В самом деле, при уменьшении числа на 7 этот остаток не меняется. Таким образом, число, которое будет после выполнения команды, дает тот же остаток при делении на 7, что и исходное. С другой стороны, оно мень-

ше 7 (условие в команде повторения не соблюдается после выполнения команды) и неотрицательно, так как мы вычитаем 7 только из чисел, не меньших 7. Упражнение: написать по образцу этой команды команду нахождения остатка при делении на 9. (Решение: заменить в двух местах команды 7 на 9.)

Пример 11.

```
пока число на доске содержит больше
      одной цифры
нц
  заменить число на доске суммой его
  цифр
кц
```

На доске было написано положительное число А. Что останется на доске после выполнения команды? Будет написан остаток от деления числа А на 9 (или 9, если этот остаток равен 0). В самом деле, сумма цифр положительного целого числа дает при делении на 9 тот же остаток, что и само это число (на этом свойстве основан известный признак делимости на 9). Поскольку после выполнения команды должно получиться однозначное число (после выполнения команды условие не соблюдается), то оно будет равно остатку от деления А на 9 (или 9, если остаток равен 0).

Пример 12. На доске написано пятьдесят семь единиц и девяносто одна минус единица. Выполняется такая команда:

```
пока на доске написано больше одного
      числа
нц
  стереть какие-то два числа и заменить
  их на их произведение
кц
```

Что будет написано на доске после выполнения команды? Будет написано единственное число, так как после выполнения команды повторения условие не соблюдается. Оно будет равняться —1, так как произведение всех чисел на доске остается неизменным. Что было бы, если вначале на доске было бы 100 чисел, каждое из которых равно 2? Осталось бы одно число, равное 2 в степени 100.

Пример 13. Запишем, наконец, алгоритм Евклида.

```
алг нахождение НОД двух натуральных
      чисел
нач
  пока числа не равны
  нц
    если первое число больше вто-
         рого
    то
      вычесть второе число из пер-
      вого
    иначе
      вычесть первое число из вто-
       рого
    все
  кц
кон
```

Резюмируем коротко основные свойства команды повторения:

после выполнения команды повторения условие не соблюдается;

серия команд, входящих в команду повторения, выполняется как единое целое, выполнение которого не прерывается, даже если условие перестало соблюдаться;

может случиться, что команда повторения не закончит работу («зациклится»), если после любого числа повторений условие не перестает соблюдаться;

если какая-то величина не меняется при каждом выполнении серии команд из команды повторения, то она не меняется и при выполнении команды повторения целиком, что позволяет предсказать результат выполнения команды повторения.

Не страшно, если изучение команды повторения не уложится в отводимые два урока, а, например, потребует месяца или двух. Гораздо хуже, если эта тема останется не понятой учениками. Поэтому настойчиво рекомендуем вам не двигаться дальше, пока с командой повторения (или с командой ветвления, но она проще) не станет всем все ясно.

Продолжение следует

Я. ЗАЙДЕЛЬМАН

Алгоритмический язык в X классе

Эта работа не методические рекомендации по проведению уроков и не комментарии к учебному пособию. Материал предназначен для самих учителей, а не для передачи ученикам, хотя отдельные моменты могут пригодиться и на уроках.

До сих пор у большинства педагогов не было ни времени, ни возможности изучить предмет глубже, чем предполагает школьная программа. А ведь обычно ученикам передается лишь верхушка айсберга знаний учителя, и именно потому, что учитель обладает глубокими и прочными знаниями, выходящими далеко за пределы обязательной программы, он может ответить на любые вопросы и оценить неожиданные ответы, связать в единое целое различные разделы программы и указать на межпредметные связи, сформировать собственное мнение о содержании курса и методе его изучения. К сожалению, все это пока недоступно для многих преподавателей информатики, ибо на ускоренных летних курсах можно было в лучшем случае твердо усвоить программный материал, а углублять свои знания многим негде.

Сделаем попытку приоткрыть «второй слой», осветить некоторые вопросы, непосредственно связанные с программным материалом и вытекающие из него. Пусть это поможет учителю лучше понять содержание курса, придаст ему больше уверенности в своих силах.

Предполагается, что читатель знаком с содержанием пробных учебных пособий (УП) для IX и X классов.

В курсе X класса в алгоритмический язык вводится четыре новых элемента: выбор, цикл «для», функции и литерные величины. Их изучение преследует двоякую цель.

Во-первых, новые элементы увеличивают изобразительные возможности языка, облегчая тем самым составление алгоритмов. Мы сталкиваемся здесь с классической ситуацией: разумное усложнение инструмента упрощает его использование.

Во-вторых, изучение новых элементов алгоритмического языка готовит учеников к последующему знакомству с языками программирования, которые обычно включают эти элементы в той или иной форме.

Сформулированные таким образом цели изучения диктуют следующий порядок рассмотрения новых элементов:

связь с ранее изученными конструкциями алгоритмического языка;

новые возможности, связанные с появлением данного элемента;

особенности применения;

рекомендации по использованию;

реализация данного элемента в наиболее распространенных языках программирования.

Команда выбора

Она появляется в алгоритмическом языке как естественное развитие команды ветвления.

Команда ветвления (**если**—**то**—**иначе**—**все**) обеспечивает выбор одного из двух возможных действий. На практике часто возникает необходимость выбора из трех и более возможностей. Использование в этой ситуации команды ветвления обычно приводит к одной из следующих типовых конструкций:

```
если условие 1                                   (1)
  то серия 1
  иначе если условие 2
        то серия 2
        иначе...
              если условие n
                то серия n
              все
        все
  все
все

если условие 1                                   (2)
  то серия 1
```

```
все
если условие 2
    то серия 2
все
...
если условие n
    то серия n
все
```

Различия в работе этих конструкций вытекают из свойств ветвления. В (1), после того как одно из условий оказалось истинным, исполняется соответствующая этому условию серия. *Остальные условия не проверяются и серии не исполняются.* В (2) все условия проверяются в любом случае.

Результаты работы этих конструкций будут одинаковы, если все условия взаимно не пересекаются и исполнение серий не влияет на истинность условий. В общем случае это не так.

Следующий простой пример иллюстрирует сходство и различие приведенных конструкций.

```
если X>5
    то Y:=X—5
    иначе если X>0 и X≤5
            то Y:=X—3
            иначе если X≤0
                    то Y:=—X
                все
    все
все
если X>5
    то Y:=X—5
все
если X>0 и X≤5
    то Y:=X—3
все
если X≤0
    то Y:=—X
все
```

Эти алгоритмы дают одинаковые результаты, хотя и работают по-разному.

```
если X>5
    то Y:=X—5
    иначе если X>0
            то Y:=X—3
            иначе если X≤0
                    то Y:=—X
                все
            все
```

```
все
если X>5
    то Y:=X—5
все
если X>0
    то Y:=X—3
все
если X≤0
    то Y:=—X
все
```

Эти алгоритмы дают различные результаты, так как первые два условия пересекаются. При X>5 исполнение первого дает Y=X—5, а второго Y=X—3.

```
если X>5                                        (3)
    то X:=X—5
    иначе если X>0 и X≤5
            то X:=X—3
            иначе если X≤0
                    то X:=—X
                все
    все
все
Y:=X
```

```
если X>5                                        (4)
    то X:=X—5
все
если X>0 и X≤5
    то X:=X—3
все
если X≤0
    то X:=—X
все
Y:=X
```

В алгоритме (4) во втором и третьем условиях проверяется значение X, возможно уже измененное предыдущими командами. При начальном значении X=6 в (3) X будет изменен один раз, получится результат Y=1, а в (4) X будет изменен трижды, и результатом будет Y=2.

Итак, в (1) исполняется не более одной из имеющихся *n* серий; серия K исполняется тогда и только тогда, когда условие K истинно, а условия 1...K—1 ложны. В (2) каждая из серий исполняется или нет в зависимости от инстинности единственного условия; может исполниться несколько серий; исполне-

ние одной серии может влиять на другие только косвенно, через изменение значений величин (как в (4)).

Таким образом, выбор в полном смысле этого слова, выбор одной альтернативы из многих, реализует конструкция (1). Она и легла в основу введенной в алгоритмический язык команды выбора:

```
выбор                                          (5)
    при условие 1: серия 1
    при условие 2: серия 2
        ...
    при условие n: серия n
все
```

Конструкции (1) и (5) полностью эквивалентны и взаимозаменяемы. Полная форма команды выбора (с **иначе**) эквивалентна включению **иначе** в самое внутреннее **если**:

```
выбор                                          (6)
    при условие 1 : серия 1
    при условие 2 : серия 2
        ...
    при условие n : серия n
    иначе  серия
все
эквивалентно
если условие 1                                 (7)
    то серия 1
    иначе если условие 2
              то серия 2
              иначе ...
                    если условие n
                       то серия n
                       иначе серия
                    все
              ...
              все
все
```

Эквивалентность выбора и ветвления — ключ к пониманию механизма работы команды выбора. Опыт показывает, что при ее изучении наибольшие трудности у учеников вызывает то, что условия проверяются строго последовательно и только до первого истинного. Анализ эквивалентной формы и преобразование выбора в ветвление и обратно должны помочь в решении этой проблемы. Преобразования рекомендуется выполнять, пользуясь приведенной выше схемой, на живых примерах, а не на абстрактных фрагментах типа упражнения 7 на с. 32 УП. Можно порекомендовать анализ алгоритмов такого типа:

```
алг ТРЕУГОЛЬНИК (вещ A, B, C, лит S)
    арг A, B, C
    рез S
нач
    выбор
        при A=B или A=C или B=C :
              S:=равнобедренный
        при A=B и A=C :
              S:=равносторонний
        иначе S:=разносторонний
    все
кон
```

Этот алгоритм будет рассматривать равносторонние треугольники как равнобедренные, так как его составитель неправильно понял механизм работы команды выбора. Для исправления необходимо поменять местами первые два условия.

Итак, один и тот же алгоритм можно оформлять по-разному — с использованием любой из этих двух команд. Что же предпочесть в каждом конкретном случае? С формальной точки зрения это абсолютно несущественно. Конструкции (1) и (5), а также (6) и (7) полностью эквивалентны, их использование определяется удобством составления алгоритма и его последующего разбора другим человеком. Многое зависит от личных привычек: например, некоторым из моих учеников настолько понравилась команда выбора, что они практически отказались от использования ветвления, однако нашлись и «консерваторы», которые, успешно сдав зачет по команде выбора, продолжали пользоваться исключительно ветвлением.

Сформулируем некоторые советы.

1. Длинная цепочка вложенных **если** (типа (1)) очень громоздка и трудна для понимания. Ее лучше заменить командой выбора.

2. В команде выбора легко разобраться, если входящие в нее условия однородны. Если это не так (например, в разных условиях фигурируют разные величины), лучше воспользоваться ветвлением. Если же однородность есть, то

удобно использовать выбор даже при небольшом числе вариантов.

3. Избегайте вложенных выборов. Конструкция типа

```
выбор
    при условие 1 : серия 1
    при условие 2 : выбор
                при условие 3 : серия 3
                при условие 4 : серия 4
        все
    при условие 5 : серия 5
все
```

сложна для понимания. Ее лучше заменить одной командой выбора, используя составные условия, или использовать вместо одного из выборов (внешнего или внутреннего) ветвление.

Команда выбора в языках программирования. Выбор по условию введен в алгоритмический язык из Рапиры. В большинстве языков высокого уровня этой конструкции в чистом виде нет, она моделируется последовательностью ветвлений типа (1). Зато во многих языках (Рапира, Паскаль, Си, Модула, Ада) есть другая форма выбора (по значению выражения), не вошедшая в алгоритмический язык. Вот как выглядит такой оператор на Паскале:

```
CASE целое выражение OF
список 1 : оператор 1;
список 2 : оператор 2;
...
список n : оператор n
END
```

Здесь «список» — перечисленные через запятую константы, соответствующие возможным значениям выражения.

При работе этой конструкции прежде всего вычисляется значение выражения (целочисленное). Если оно входит в список 1, выполняется оператор 1, если в список 2 — оператор 2 и т. д. В некоторых версиях разрешается задавать оператор, который будет исполняться, если значения не оказалось ни в одном списке.

Таким образом, механизмы работы выбора по условию и по значению совпадают. Отличие лишь в форме условия: в выборе по условию оно может быть любым, а в выборе по значению только проверкой значения выражения.

Отметим, что в некоторых языках, где нет выбора по значению, есть специальные средства для его моделирования. Это переключатели в Алголе, вычисляемый GOTO в Фортране, переход по массиву меток в ПЛ/1, оператор ON в Бейсике. Однако все эти средства потенциально опасны, особенно для начинающих, так как их неаккуратное использование может привести к запутанной, трудной для понимания программе. Программирующим на этих языках необходимо строго следить за тем, чтобы указанные средства не использовались ни в каких других целях, кроме моделирования выбора по значению.

Команда повторения с параметром

Ее смысл легко понять, разобрав алгоритм обработки таблиц. Обобщенный алгоритм поэлементной обработки линейной таблицы таб A[K:N] можно представить в таком виде:

```
I:=K
пока I⩽N
нц
   обработка A [I]
I:=I+1
кц
```

Анализ этого фрагмента показывает, что в общем случае в подобных конструкциях используются:

целая переменная (I), указывающая текущий элемент таблицы;

начальное значение этой переменной (K), присваиваемое ей до начала цикла;

конечное значение (N), по достижении которого происходит выход из цикла;

шаг переменной, т. е. величина изменения ее значения при каждом проходе цикла.

Задачи, для решения которых необходимы табличные величины, встречаются очень часто, поэтому в алгоритмический язык включена команда специальной формы, позволяющая записать перечисленные элементы коротко и наглядно. Это и есть команда повторения с параметром, или цикл для:

```
для X от XMIN до XMAX шаг XШАГ
нц
```

серия
кц

Здесь X — управляющая переменная цикла, а XMIN, XMAX, XШАГ — целые выражения, причем XШАГ>0.

На первый взгляд цикл для значительно проще и понятнее, чем пока, однако в нем есть подводные камни.

1. Когда вычисляются XMAX и XШАГ? Один раз, при входе в цикл, или заново при каждом выполнении серии?

От ответа на этот вопрос зависят изменение управляющей переменной и момент прекращения работы цикла в случае, если серия содержит команды изменения переменных, входящих в выражения XMAX и XШАГ.

В УП (с. 41) внимание читателя обращено на то, что в приведенных примерах таких изменений нет; однако нет и явного запрета использовать их.

Механизм работы цикла для, описанный на с. 34 УП, создает впечатление, что выражения вычисляются один раз. В то же время из эквивалентной схемы преобразования для в пока (книга для учителя, с. 80) следует, что выражения вычисляются многократно.

Где же истина?

А что такое — истина?

В естественных науках истинной считается та гипотеза, которая точнее описывает наблюдаемое явление, та, что ближе к закону природы.

В программировании законов природы нет. Алгоритмы создаются и анализируются людьми. Формальные исполнители тоже создаются людьми, и именно люди закладывают в них ту или иную трактовку алгоритмических конструкций.

Законы программирования создаются не природой, поэтому в них истина — это удобство человека. Но в понятии удобства много субъективного. Представления об удобном и неудобном различны в разные времена, у разных людей. Это один из источников постоянного развития и великого разнообразия языков программирования. Правила каждого языка не истина в последней инстанции, а всего лишь соглашение между разработчиками и пользователями. Важно только, чтобы все одинаково понимали это соглашение и соблюдали его.

Алгоритмический язык еще молод. Его авторы не предусмотрели всех возможных вариантов. Окончательное слово за ними, а пока я хочу предложить свой ответ. Кто-то может не согласиться с ним; это не важно. Главное, чтобы все поняли сущность поставленных вопросов и тот факт, что ответ на них определяется не природными закономерностями, а человеческой логикой.

Итак, когда должны вычисляться выражения в заголовке цикла?

Запрет на изменение в теле цикла входящих в эти выражения переменных представляется неразумным: выражения могут быть сложными, включать много переменных, и уследить за соблюдением запрета будет нелегко.

Вычисление выражений при каждом проходе цикла неэффективно. Вдобавок при таком подходе по заголовку цикла нельзя определить, сколько раз будет выполняться серия, а такая определенность бывает очень полезна.

Таким образом, наиболее удобно *однократное* вычисление выражений при входе в цикл. Оно ведется с теми значениями, которые переменные получили до начала цикла. В теле цикла эти значения могут изменяться, что не влияет на конечное значение и шаг управляющей переменной. Именно такое решение принято в большинстве языков программирования.

2. Что происходит при изменении значения управляющей переменной в теле цикла?

Если такие изменения разрешить, то возникает вопрос: к какому значению управляющей переменной прибавлять шаг? К измененному (вариант 1) или к тому, с которого был начат предыдущий проход цикла (вариант 2)?

Рассмотрим пример.

```
для I от 1 до 3
нц
    ПЕЧАТЬ (I)
    I:=I+1
    ПЕЧАТЬ (I)
кц
```

Что будет напечатано при работе этого

фрагмента? Сколько раз исполнится цикл?

В варианте 1 цикл исполняется 2 раза, печатается последовательность 1 2 3 4.

В варианте 2 цикл исполняется трижды. Печатаются цифры 1 2 2 3 3 4.

Что же выбрать? В первом варианте заголовок цикла фактически теряет свое значение как средство управления циклом. Количество повторений и значение управляющей переменной определяются не заголовком, а скрытыми в теле цикла присваиваниями. Особенно опасно использование управляющей переменной в качестве параметра-результата при вызове вспомогательного алгоритма; в этом случае проследить за ее изменениями практически невозможно, и человек оказывается не в состоянии понять работу цикла.

Не лучшая ситуация возникает и при использовании второго варианта. Количество повторений в этом случае определяется только заголовком, но управляющая переменная изменяется почти мистическим образом. У нее начинается «раздвоение личности»: главное значение, управляющее работой цикла, таинственно скрывается в неизвестности, а в операции участвует «подставное лицо», т. е. значение, полученное при присваиваниях. Ситуация обостряется при использовании вспомогательных алгоритмов.

Итак, самое разумное — запретить изменение значения управляющей переменной (намек на такой запрет есть в УП). Такое правило принято во многих языках программирования, кроме, пожалуй, Бейсика, но это явно не тот язык, с которого следует брать пример.

Окончательно сформулируем: в теле цикла *нельзя* изменять значение управляющей переменной, т. е. ее нельзя использовать в левой части команды присваивания и в качестве параметра-результата при вызове вспомогательных алгоритмов.

3. Какое значение имеет управляющая переменная по окончании работы цикла?

То, при котором серия исполнялась последний раз, или превышающее его значение, указывающее на необходимость завершения цикла? А может быть, XMAX?

Рассмотрим пример.

```
для I от 1 до 4 шаг 2
нц
  ...
кц
ПЕЧАТЬ (I)

для I от 2 до 1
нц
  ...
кц
ПЕЧАТЬ (I)
```

Что напечатает первый из этих фрагментов — 3, 4 или 5? А второй? Рассмотрим возможные варианты.

А. Сохраняется последнее значение, при котором исполняется цикл; первый фрагмент печатает 3. А второй? Ведь здесь цикл вообще не исполняется! Значение, которое управляющая переменная имела до начала цикла, вряд ли можно считать последним из цикла. Этого значения может и вообще не быть, если, например, алгоритм начинается с для. Таким образом, от этого, удобного во многих других отношениях, варианта приходится отказаться.

Б. Сохраняется значение XMAX. Первый фрагмент печатает 4, второй — 1. Это нелогично. Почему, например, во втором фрагменте, где цикл вообще не исполняется, I должно принимать значение 1? Отбрасываем и этот вариант.

В. Сохраняется превышающее XMAX значение, прервавшее цикл. Первый фрагмент печатает 5, второй — 2. Недостаток этого варианта выявляется при машинной реализации. Дело в том, что множество целых чисел в машинной реализации обычно ограничено каким-то наибольшим числом M. Тяжелая ситуация возникает, когда $XMAX \leqslant M < XMAX + XШАГ$. В этом случае цикл исполняется нормально, можно организовать проверку завершения и выход из цикла, а вот присвоить управляющей переменной превышающее XMAX значение уже нельзя.

Что же выбрать? Пожалуй, оптимально решение, принятое в некоторых языках программирования: по окончании цикла значение управляющей переменной становится *неопределенным*. Ею можно пользоваться только после при-

своения нового значения. Действительно, управляющая переменная предназначена только для организации цикла и не может быть использована в других целях. По окончании цикла от нее остается лишь имя, телесная оболочка, которую можно использовать, только наполнив предварительно новым содержанием.

Неопределенность управляющей переменной после выхода из цикла в сочетании с запретом на ее изменение в теле цикла упрощает машинную реализацию для некоторых ЭВМ, делает цикл для более простым и быстрым.

Теперь, когда сформулированы основные правила работы цикла для, можно построить эквивалентную схему преобразования его в цикл пока.

```
для X от XMIN до XMAX шаг ХШАГ
нц
  серия
кц

эквивалентно
R1:=XMAX
R2:=ХШАГ
X:=XMIN
пока X≤R1
нц
  серия
  X:=X+R2
кц
X:=НЕОПР
```

Здесь R1 и R2 — невидимые для программиста рабочие переменные, необходимые для предотвращения повторного вычисления XMAX и ХШАГ, НЕОПР — неопределенное значение.

Приведенная схема не является полностью эквивалентной циклу для, как это было в случае выбора и ветвления. Во-первых, вводятся дополнительные переменные R1 и R2; во-вторых, не отражен запрет на изменение управляющей переменной в теле цикла; в-третьих, использовано странное слово НЕОПР. Тем не менее эта схема точно соответствует предложенному механизму работы для и показывает, как при необходимости его можно преобразовать в пока.

Обратное преобразование возможно далеко не всегда. Цикл пока — значительно более мощное средство.

Цикл для используется, когда количество исполнений цикла известно заранее, например при полной обработке таблицы. Если же число исполнений определить нельзя, если цикл должен быть прерван не по достижении управляющей переменной конечного значения, а по какому-то другому условию, например при обнаружении в таблице элемента, обладающего заданным свойством, необходим цикл пока.

И последнее. Алгоритмический язык не разрешает использовать отрицательный шаг цикла, по-видимому, чтобы упростить конструкцию и облегчить объяснение ее работы. Во многих языках такого запрета нет. При отрицательном шаге цикл выполняется, пока управляющая переменная превышает конечное значение, и не исполняется ни разу, если ХНАЧ<ХКОН. Эквивалентная схема «универсального» цикла:


```
R1:=ХКОН
R2:=ХШАГ
X:=ХНАЧ
пока SIGN(X−R1)≠SIGN(R2)
нц
  серия
  X:=X+R2
кц
X:=НЕОПР
```

Вспомогательные алгоритмы вычисления значений функций

Мы уже видели, что команда выбора — всего лишь более удобная форма записи группы вложенных ветвлений, повторение с параметром — особая форма цикла, которая может быть сведена к циклу пока. Точно так же алгоритмы вычисления значений функций (или просто функции) — особый класс вспомогательных алгоритмов, которые выделяются и оформляются специальным образом в первую очередь *для удобства программирования*.

В виде функции могут быть оформлены алгоритмы, имеющие единственный результат простого (т. е. не табличного) типа.

Функции отличаются своим оформлением и способом использования. Обращение к обычному вспомогательному ал-

горитму происходит с помощью специальной команды вызова; обращение к функции специальной команды не требует. Оно может использоваться в любом выражении в качестве величины, тип которой определяется типом (единственного!) результата функции. При этом происходит обращение к функции, параметры-аргументы передаются по общим правилам, а полученный результат участвует в дальнейшем вычислении выражения.

Обратите внимание — функции могут быть только вспомогательными алгоритмами. Дело в том, что результат функции фактически нигде не хранится, он рассматривается как одно из промежуточных значений, необходимых для вычисления сложного выражения. Чтобы воспользоваться этим результатом, функцию необходимо не просто исполнить, а вызвать из какого-то алгоритма. Это, однако, не означает, что в алгоритме вычисления функции не может быть вызовов вспомогательных алгоритмов и функций. Из рассмотренного механизма работы функции следует и то, что ее результат не может быть табличной величиной: ведь алгоритмический язык не допускает включения в выражения обращений к табличной величине в целом, использоваться могут лишь элементы таблицы.

Особенности оформления функций сводятся к отличиям в описании параметров. Описание величин в алгоритмическом языке включает три элемента: имя, тип и вид (аргумент, результат, промежуточная величина). Для параметров обычных алгоритмов имя и тип задаются в списке параметров в заголовке алгоритма, а вид определяется включением имени в список аргументов или результатов (такая повторность описания не самая лучшая черта алгоритмического языка).

В функциях же в список параметров включаются только аргументы, следовательно, специальный список аргументов уже не нужен; имя и тип аргументов определяются в заголовке обычным образом.

Единственный результат функции описывается следующим образом. Его тип указывается в заголовке перед именем алгоритма, как если бы это был тип алгоритма в целом. Это естественно, так как вызов функции заменяет обращение к переменной соответствующего типа. В практике программирования бытуют выражения «целая функция», «вещественная функция»; следует, однако, ясно понимать, что типом обладает не сама функция, а ее результат, и «целая функция» — сокращение от «функция, возвращающая результат целого типа».

Имя результата, с точки зрения вызывающего алгоритма, совпадает с именем функции. Кажется естественным использование имени функции в качестве имени результата и в самой функции. К сожалению, при этом становится невозможным рекурсивное обращение функции к самой себе, так как это обращение будет восприниматься как использование текущего значения результата. Поэтому в качестве имени результата в алгоритмическом языке используется специальное служебное слово **знач**.

Хотелось бы обратить внимание на связанное с использованием функций слабое место алгоритмического языка. Тип результата функции описывается в ней самой, но он неизвестен в использующем функцию алгоритме — и в выражениях появляются элементы неизвестного типа.

Рекомендации по применению функций очень просты: их не следует считать какими-то неполноценными алгоритмами, а можно и нужно применять везде, где это сокращает запись и упрощает понимание вызывающего алгоритма.

В заключение несколько слов о рекурсии. Рекурсия — мощный инструмент, требующий соответствующих масштабных целей. Примеры из УП напоминают стрельбу из пушки по воробьям, эти алгоритмы значительно проще и понятнее в нерекурсивном варианте. Кроме того, текст УП создает впечатление, что рекурсивными могут быть только функции. На самом деле рекурсивным может быть любой алгоритм. К сожалению, эта тема слишком сложна для такого беглого обзора; будем надеяться, что в журнале появится отдельная статья, посвященная вопросам использования рекурсивных алгоритмов.

В языках программирования функции

применяются очень широко; механизм их работы и правила оформления не отличаются особым разнообразием.

Вызов функции, как и в алгоритмическом языке, вставляется в выражение и записывается в виде имени функции, за которым в скобках указывается список параметров. В отличие от алгоритмического языка тип функции, как правило, указывается не только в ней самой, но и в вызывающем алгоритме (поскольку речь идет о языках программирования, правильнее было бы сказать «вызывающий модуль» или «вызывающая процедура», но сути дела это не меняет, а пользоваться привычной терминологией удобнее).

Еще одно свойство многих языков — богатый набор встроенных функций для выполнения часто встречающихся математических и некоторых других операций.

Некоторые вопросы, связанные с функциями, по-разному решаются в языках программирования. Это в первую очередь передача результата функции и отношение к *побочному эффекту*. Например, в Фортране рекурсивные вызовы запрещены, поэтому имя функции может использоваться внутри нее в качестве переменной. Последнее значение этой переменной становится результатом функции, т. е. имя функции играет ту же роль, что и слово знач в алгоритмическом языке.

Во всех модулях Фортрана, в том числе и функциях, параметры рассматриваются как результаты; иначе говоря, они могут изменяться в результате работы функции; эти изменения могут сказываться на работе вызывающего алгоритма. Такое влияние и называется *побочным эффектом*. Его использование делает программы запутанными, трудными для понимания, поэтому «правила хорошего тона» рекомендуют его избегать. В алгоритмическом языке побочный эффект просто запрещен.

В Паскале имя функции «двусмысленно». В левой части оператора присваивания оно означает результат функции, в любом другом месте — рекурсивный вызов. Таким образом, если F — имя функции, то в записи

F:=F+1

первое F означает результат функции, а второе — ее рекурсивный вызов. На практике такое правило оказывается весьма неудобным, оно провоцирует ошибки. Побочный эффект в функциях на Паскале допускается, но параметры-результаты должны быть заданы явно. В некоторых версиях Паскаля побочный эффект запрещен.

Алгоритмы работы с литерными величинами

Прежде чем анализировать правила работы с литерными величинами, необходимо уяснить, что такое тип величины.

В алгоритмическом языке тип связывается с именем величины. Он задает множество значений, которые может принимать величина, и множество операций, в которых она может участвовать. Можно сказать, что тип есть множество значений плюс множество операций. (Отметим, что это не единственно возможный подход. В Рапире, например, каждая величина может принимать значения любого определенного в языке типа, который в этом случае связывается не с *именем*, а со *значением* величины и определяет только множество операций.)

Типы бывают простые и составные. В алгоритмическом языке определены простые типы вещ (вещественный), цел (целый) и лит (литерный) и единственный способ организации составного типа — таблица.

Множества значений и операций для целого и вещественного типов определяются естественным путем. Множества значений совпадают с математическими множествами целых и действительных чисел (такое совпадение возможно только в теоретическом алгоритмическом языке, при работе с ЭВМ возникают очень сильные ограничения). Множество допустимых операций включает 5 математических операций, 6 операций сравнения и операцию присваивания. Таким образом, множества значений и операций для этих типов хорошо известны из математики, что позволило ввести их курсе IX класса без дополнительных пояснений.

Литерные значения и величины зна-

чительно менее привычны, поэтому им необходимо уделить особое внимание.

Множество значений литерных величин — это множество упорядоченных наборов символов. Упорядоченность означает, что литерные значения различаются не только набором, но и последовательностью символов, например «краб» и «брак» — это разные значения.

Какие символы могут входить в значение литерной величины (далее для краткости будем говорить просто — в литерную величину)? При работе на ЭВМ — те, которые имеются на внешних устройствах (дисплее, АЦПУ). При «бумажном программировании» — любые удобные символы.

Количество символов в литерной величине теоретически не ограничивается. При работе на ЭВМ обычно задается некоторое максимальное количество: например, в Е-практикуме длина литерной величины не может превосходить 16.

Особое место в множестве значений литерных величин занимает пустая строка, не содержащая ни одного символа. Следует отличать пустую строку от пустого символа. Пустой символ, или пробел,— равноправный с остальными символ, используемый для разделения слов в тексте. Пустая строка обозначается «» (кавычки идут подряд) и имеет длину 0. Строка, содержащая пустой символ, обозначается « » (между кавычками есть пробел) и имеет длину 1.

Множество значений литерных величин определяет двойственный характер этого типа. Литерный тип является одновременно простым (строка может рассматриваться как единое целое) и составным (строка состоит из отдельных символов).

В курсе IX класса были введены простейшие операции: присваивание и сравнение на равенство и неравенство. В них литерная величина рассматривается как единое целое, без учета внутренней структуры. Этих операций часто недостаточно: например, задача алфавитного упорядочения требует введения отношений «больше» и «меньше». Многие языки программирования допускают произвольные сравнения литерных величин; при этом создается «алфавит», включающий все допустимые символы, и величины сравниваются на основе этого алфавита по обычным правилам, принятым в словарях (так называемое лексикографическое упорядочение). Та величина, которая идет раньше по алфавиту, считается меньшей. Двойственность литерных величин проявляется и здесь: сравнение ведется посимвольно, а результат относится к величине в целом.

Такой подход имеет свои недостатки. Символы, как правило, упорядочиваются в соответствии с числами, которые кодируют их внутри ЭВМ, поэтому, во-первых, их последовательность часто не совпадает с той, которую хотелось бы видеть программисту (например, русские буквы чаще всего расположены не по алфавиту), а во-вторых, эта последовательность может оказаться разной на разных ЭВМ даже при использовании одного языка программирования.

Из-за этого введение в алгоритмический язык полного набора сравнений литерных величин вряд ли целесообразно, тем более что нетрудно реализовать лексикографическое сравнение с помощью приведенной ниже функции, задающей произвольный «алфавит программиста». В приведенном примере это кириллица, в общем случае величина АЛФАВИТ должна содержать все используемые символы, упорядоченные в соответствии с требованиями конкретной задачи.

```
алг цел НОМЕР (лит Х)
нач лит АЛФАВИТ
    АЛФАВИТ:="абвгдежзийклмнопрс
    туфхцчшщъьыэюя"
    знач:=1
    пока Х≠АЛФАВИТ[знач:знач]
    нц
      знач:=знач+1
    кц
кон
```

В курсе X класса вводятся новые операции: соединение, длина и вырезка.

Операции соединения и длины настолько просты, что вряд ли нуждаются в пояснениях. Отметим, что они хорошо демонстрируют двойственность литерных величин: каждая величина рассматривается как единое целое, но механизм выполнения операции учитывает,

что она состоит из отдельных символов.

Одинаковый знак (+) для обозначения соединения литерных и сложения числовых величин наводит на мысль о родстве этих операций. В действительности они совершенно несхожи. Соединение, в отличие от сложения, некоммутативно («сумма» зависит от порядка «слагаемых»), для него не существует обратной операции, аналогичной вычитанию.

Для лучшего уяснения различия числовых и литерных величин и операций над ними можно рекомендовать следующий ряд простых примеров, в которых должен разобраться каждый ученик:

3	+	2=5	сложение
«3»	+	«2»=«32»	соединение
3	+	длин («2»)=4	длина литерной величины «2» равна 1
3	+	«2»=???	нельзя складывать литерную
«3»	+	длин («2»)=???	величину с числовой
3	+	длин (2)=???	нельзя вычислять длину числовой величины
3	—	2=1	вычитание
«3»	—	«2»=???	для литерных величин такой операции нет

Операция вырезки позволяет «войти внутрь» литерной величины, и тут двойственность этих величин начинает сказываться в полной мере. Можно ли указывать границы вырезки, выходящие за пределы текущей длины переменной (при присваивании последней значения)? Можно ли «врезать» два символа на место одного? А один на место двух? Эта тема требует особого обсуждения.

А как обстоят дела с литерными величинами в языках программирования?

В Фортране они вообще не предусмотрены. С помощью различных ухищрений программист может присвоить литерные значения переменным любого типа и даже обработать их, но никаких специальных операций для этого в языке нет. Зато в ПЛ/1 есть литерные величины двух видов (постоянной и переменной длины), имеются операция соединения и богатый набор встроенных стандартных функций, в число которых входят LENGTH (длина) и SUBSTR (подстрока, аналог вырезки).

Относительно Бейсика ничего определенного сказать нельзя. Во многих его версиях литерных величин нет вообще; в тех версиях, где они допускаются, правила работы с ними могут быть самыми разными. Единый язык Бейсик — миф. Сегодня Бейсиком называют множество практически не совместимых между собой диалектов, объединенных только структурой программы, кажущейся простотой и неудобством практического применения.

В Паскале имеется стандартный тип CHAR, значениями которого могут быть одиночные символы. Строки рассматриваются как массивы (линейные таблицы) таких символов. Единственная возможная операция — доступ к отдельному символу (элементу таблицы). Специальных операций для обработки литерных величин нет, они должны программироваться как функции пользователя.

У такого подхода есть некоторые теоретические достоинства, связанные с отсутствием двойственности литерных величин, но на практике он весьма неудобен.

Аналогичные правила приняты в языке Си, но там имеются более гибкие возможности для обработки строк, длина которых заранее неизвестна.

Продолжение следует

Я. ГОЛЬЦ, А. ДОДОНОВ
НИИ СиМО АПН СССР

Методические рекомендации

В школах, оснащенных кабинетами вычислительной техники, согласно программе курса ОИВТ, утвержденной Минпросом СССР, на изучение информатики в X классе выделяется 68 ч: 34 ч — на изучение теории и 34 ч — на практические занятия.

Теоретическая часть курса соответствует действующим программе и пробному учебному пособию. 34 учебных часа, отведенные для практических занятий, должны быть использованы для решения на ЭВМ различных задач, отработки навыков применения компьютера и его программного обеспечения.

Практическую работу школьников на ЭВМ можно организовать в двух формах. Первая предполагает, что компьютер с соответствующим учебным программным обеспечением используется при изучении информатики на каждом уроке. Ввиду того что вычислительной техники в школах пока недостаточно, широко используется другая форма организации работы на ЭВМ, а именно практикум, который проводится отдельно от теоретических занятий по информатике. Преимущественное использование этой формы ведения практических занятий обусловлено также тем, что для постоянного использования ЭВМ на каждом уроке необходимо развитое программное обеспечение. Преподаватели, ведущие занятия по этому варианту, могут воспользоваться данными методическими рекомендациями. Они написаны в соответствии с программой практикума, изложенной в методическом письме «О преподавании курса информатики (68 ч) в школах, имеющих вычислительную технику» (см.: Информатика и образование. 1987. № 3).

Рекомендации даны в расчете на использование действующего пробного учебного пособия по курсу ОИВТ (далее — УП), к содержанию которого, как показывает практика, большинство учителей в достаточной степени адаптировались.

В данных методических рекомендациях выделено программно независимое ядро. Это сделано в связи с разнородностью парка ЭВМ и их программного обеспечения. Там, где это методически целесообразно, разобраны примеры использования конкретных программ на занятиях по информатике.

Начальные навыки работы с ЭВМ

Первое знакомство учащихся с назначением основных устройств ЭВМ, правилами их эксплуатации, получение навыков работы с клавиатурой.

1-й урок. Назначение основных устройств компьютера, демонстрация их работы. Правила включения, загрузки и выключения ЭВМ. Инструкция по технике безопасности. Начало работы с клавиатурой.

2-й урок. Работа с программой типа «клавиатурный тренажер». Самостоятельная работа с учебной программой демонстрационного типа.

Основные цели.

1. Закрепить знания учащихся о назначении основных устройств и их взаимодействии при работе ЭВМ.

2. Дать возможность учащимся приобрести навыки работы с клавиатурой ЭВМ.

3. Выработать у учащихся умение вести диалог с учебной программой демонстрационного типа.

Требования к знаниям и умениям.

1. Учащиеся должны знать назначение основных устройств ЭВМ, правильно использовать соответствующую терминологию.

2. Уметь пользоваться клавиатурой для ввода информации в ЭВМ.

3. Знать и применять на практике основные правила техники безопасности при работе с ЭВМ.

4. Уметь подготовить конкретную ЭВМ к работе (включить устройства, подключиться к локальной сети, запустить программу), завершить работу в соответствии с установленными правилами.

5. Уметь вести диалог с учебной программой демонстрационного типа.

Методические указания. Основное ядро. Начальное знакомство имеет важное значение для дальнейших взаимоотношений с компьютером. Преподавателю следует особенно тщательно подготовиться к первому уроку в кабинете вычислительной техники. Стоит вспомнить свое первое свидание с компьютером, типичные ошибки, затруднения. Большинство учащихся проявляют интерес к общению с ЭВМ. Важно этот интерес поддержать, не дать ему угаснуть, помочь школьникам преодолеть почти неизбежный «психологический барьер» боязни общения с новой техникой. Следует продемонстрировать возможности компьютера в графике, музыке, текстовой обработке, сложных вычислениях, хранении информации, играх. Для этого можно использовать любые программы, имеющиеся под рукой. Необходимо сделать так, чтобы, впервые нажав на клавиши и получив

отклик на экране дисплея, учащийся захотел освоить новые, более богатые возможности ЭВМ, осознал «дружелюбность» компьютера.

К первому занятию в КВТ учащимся необходимо повторить материал введения первой части УП (с. 3—16). В начале урока предложите им назвать устройства ЭВМ, имеющиеся в кабинете. Необходимо четко называть устройства своими именами. Попросите нескольких учеников повторить по схеме: как называется, для чего служит, как вводится, хранится, изменяется, выводится информация. Затем включите компьютер. Первый раз необходимо медленно показать порядок включения. Обратите внимание учащихся на основные правила техники безопасности.

Клавиатура. Некоторые учителя сразу предлагают учащимся диалог с программой «клавиатурный тренажер»: «Она всему сама научит». Однако целесообразно до начала работы с клавиатурным тренажером разъяснить назначение клавиатуры как устройства ввода информации в ЭВМ, дать классификацию клавишей: цифровые, буквенные, специальные символы, функциональные, управляющие. Особое внимание обратить на клавиши переключения регистров и режимов. Объяснить их необходимость. Предложить несколько простых упражнений следующего типа.

> Наберите свои имя и фамилию. Пользуясь клавишей вставки, напечатайте свое отчество после имени. Добавьте число, месяц и год рождения.

На втором уроке можно предложить школьникам работу с учебной программой «клавиатурный тренажер». Работа с ней проходит в индивидуальном режиме. Преподаватель, переходя от одной машины к другой, должен консультировать школьников, делать замечания и давать советы.

На работу с клавиатурным тренажером целесообразно выделить не менее 25—30 мин. Более прочные навыки владения клавиатурой будут привиты в ходе дальнейшей работы на ЭВМ.

В заключение можно предложить школьникам 1—2 примера игровых учебных программ.

Основы алгоритмизации

Учащиеся приобретают знания и практические навыки по составлению алгоритмов для нового исполнителя — ЭВМ, закрепляют свои знания о представлении решения задачи в виде алгоритма. Занятия проводятся с использованием программной среды «Е-практикум».

1-й урок. Введение в Е-практикум. Исполнение линейных алгоритмов на ЭВМ.

2-й урок. Команды ветвления (если, выбор).

3-й урок. Команды повторения (пока, для).

4-й урок. Работа с табличными величинами.

5-й урок. Использование вспомогательных алгоритмов.

6-й урок. Алгоритмы работы с литерными величинами.

7-й урок. Решение задач из курсов математики и физики.

8-й урок. Самостоятельная работа.

1-й урок. Введение в Е-практикум. Исполнение линейных алгоритмов на ЭВМ.

Теоретические сведения: УП, ч. I, § 1, с. 17; Д. Варсонофьев, А. Дымченко. Е-86 (Информатика и образование. 1986. № 1).

Основная цель. Познакомить учащихся с назначением и основными свойствами Е-практикума, сформировать у них практические навыки записи и редактирования простейших конструкций алгоритмического языка с помощью Е-практикума. Научить школьников правильной записи алгоритмов и исполнению (целиком и по шагам) простых линейных алгоритмов.

Требования к знаниям и умениям. Учащиеся должны уметь пользоваться командами Е-практикума для записи алгоритмов, знать отличия от стандартного использования управляющих и функциональных клавиш, знать назначение различных полей экрана. Учащиеся должны понимать сообщения Е-практикума об ошибках, уметь правильно набрать текст алгоритма, выполнить несложный линейный алгоритм.

Методические указания. После того как учащиеся приобрели некоторый навык в работе с клавиатурой, можно приступить к знакомству с Е-практикумом.

Е-практикум — система программирования для школьных персональных компьютеров, созданная на основе алгоритмического языка. Она предназначена для практического изучения основных понятий информатики и вычислительной техники: алгоритма, его записи на алгоритмическом языке, исполнения, использования ЭВМ для корректной записи алгоритма и его выполнения. Необходимо отметить, что Е-практикум не является самостоятельным, «рабочим», средством программирования, а предназначен для выполнения небольших несложных алгоритмов учебного характера.

Максимальный объем программы в Е-86 равен 60 строкам, максимальное число переменных и элементов массивов — около 360. Скорость исполнения — около 30 операций в секунду. В дальнейшем будет описываться

версия Е-86 для персонального компьютера «Ямаха». Существуют версии Е-практикума для ДВК-2М, СМ-4.

Перед началом работы следует обратить внимание учащихся на то, что использование клавиатуры в Е-86 отличается от стандартного.

Для долговременной смены режима кириллица/латиница нужно нажать клавиши **CTRL/РУС**; аналогично задействована клавиша **CAPS**; при нажатии этих клавиш без клавиши **CTRL** происходит лишь временная (пока не отпустили) смена режимов.

Цифры и символы «точка», «запятая», «двоеточие» и «/» смещены в нижний регистр, т. е. набираются без нажатия клавиши **SHIFT.**

При нажатии клавиши **INS** происходит вставка пробела — «раздвижение» строки на 1 позицию вправо.

Основной режим, устанавливаемый при запуске Е-86, — ввод строчных русских букв. Латинские и русские буквы одинакового начертания между собой не различаются.

Запись и редактирование алгоритмов. Е-86 предоставляет возможность быстрой и точной записи основных конструкций алгоритмического языка.

В начале работы, после запуска Е-практикума, экран дисплея разбит на два поля. Левое предназначено для записи алгоритма, правое — для сообщений компьютера (об ошибках и ходе исполнения алгоритма).

Курсор устанавливается в первой строке. После этого можно начать записывать и редактировать программу на алгоритмическом языке, используя систему команд Е-86. Эти команды выполняются при нажатии на управляющие клавиши (или их комбинации).

Стрелки *вправо, влево, вверх, вниз* позволяют передвигать курсор в левой части экрана;

клавиши **SHIFT+INS** — вставить пустую строку;

клавиши **SHIFT+DEL** — удалить строку (или целую конструкцию);

клавиши **SHIFT+CLS** — очистить экран.

Вставка и удаление основных конструкций алгоритмического языка производятся только целиком, что позволяет исключить произвольное изменение структуры программы и уменьшить вероятность ошибочной записи.

При нажатии клавиши:

ESC+А вставляется «АЛГ—АРГ—РЕЗ—НАЧ—КОН»;

ESC+Е вставляется «ЕСЛИ—ТО—ИНАЧЕ—ВСЕ»;

ESC+В вставляется «ВЫБОР—ВСЕ»;

ESC+И вставляется «ИНАЧЕ» (в «ЕСЛИ»);

ESC+С вставляется «ПРИ» (в «ВЫБОР»);

ESC+П вставляется «ПОКА—НЦ—КЦ»;

ESC+Д вставляется «ДЛЯ—НЦ—КЦ».

Необходимо заметить, что вставка конструкции возможна только тогда (в том месте), когда это разрешено правилами алгоритмического языка; в противном случае раздастся звуковой сигнал, и вставки не произойдет. При появлении конструкции курсор попадает внутрь нее, на ближайшее пустое место.

Для удаления целой конструкции курсор следует установить в первую ее строку и нажать клавиши **SHIFT+DEL**, после чего она (вместе со всеми командами, входящими внутрь) целиком исчезнет с экрана.

Дальнейшие сведения о Е-практикуме можно излагать учащимся по ходу выполнения практических заданий.

После первоначального знакомства с Е-практикумом можно приступить к записи и выполнению простейших алгоритмов на алгоритмическом языке.

Запуск программ осуществляется при нажатии на **SHIFT+F4.**

Для начала следует записать и выполнить линейный алгоритм, состоящий из 1—2 шагов.

Упражнение 1. Ввести и исполнить алгоритм сложения 2 чисел.

```
АЛГ УПР1 (ВЕЩ а, б, в)
. АРГ а, б
. РЕЗ в
НАЧ
   а:=45
   б:=13
   в:=а+б
КОН
```

Упражнение 2. Ввести и исполнить алгоритм

```
АЛГ УПР2 (ЛИТ а, б, с)
.   АРГ а, б
.   РЕЗ с
НАЧ
   а:=«пар»
   б:=«воз»
   с:=а+«о»+б
КОН
```

Рекомендуется выполнить их с 2—3 различными значениями аргументов: в упражнении 1 предложить учащимся заменить значения, присваиваемые аргументам а и б, на отрицательные; в упражнении 2 — значение б на «ход» и т. д.

Следующий этап — знакомство с пошаговым выполнением алгоритма. Для этой цели удобно использовать достаточно длинный линейный алгоритм, включающий промежуточные величины.

Упражнение 3. Ввести и исполнить по шагам алгоритм

```
АЛГ УПР3 (ВЕЩ у, х)
```

```
.   АРГ x:=5.2
.   РЕЗ y
НАЧ ВЕЩ a, б, c
    a:=x—3
    б:=a*a
    c:=2—a
    y:=c/(б+4)
КОН
```

При нажатии на клавишу **F4** происходит выполнение одного шага программы. Это позволяет учащимся проследить последовательность заполнения таблицы значений и уяснить один довольно тонкий момент, на который следует обратить внимание, — разницу между записью алгоритма (в том или ином виде, например на алгоритмическом языке) и его исполнением конкретным исполнителем. В данном случае таким исполнителем является ЭВМ, снабженная Е-практикумом.

В процессе выполнения заданий в правой части экрана могут появляться сообщения об ошибках:

синтаксическая ошибка — неверно записана команда;

имя не описано — имя переменной отсутствует в заголовке или после слова «нач»;

не определено — значение переменной не может быть вычислено, так как не достает информации.

Следует расшифровывать ученикам эти сообщения и подсказывать пути устранения ошибки.

2-й урок. Команды ветвления (ЕСЛИ, ВЫБОР).

Теоретические сведения: УП, ч. I, § 2, п. 4, с. 25; ч. II, § 6, с. 26.

Основная цель. Ознакомить учащихся с практическим использованием команд ветвления в разветвляющихся алгоритмах.

Требования к знаниям и умениям. Учащиеся должны уметь правильно использовать команды ветвления, четко представлять их структуру; знать основные виды условий (отношений между величинами), уметь исполнить (по шагам) и пояснить каждый шаг исполнения разветвляющегося алгоритма; уметь расшифровывать сообщения об ошибках.

Методические указания. Перед началом работы следует напомнить соответствующий теоретический материал, отметить необходимость использования команды ветвления для записи практически любого алгоритма (за исключением довольно узкого класса линейных). Напомнить схемы полной и сокращенной формы команды ветвления. Обратить внимание на обязательность использования в команде ветвления условий типа отношений между величинами (логических условий), т. е. таких, результат проверки которых — «да» или «нет».

В отличие от команды ЕСЛИ команда ВЫБОР позволяет экономно реализовать конструкцию, в которой выражение, используемое в качестве управляющего условия, может принимать несколько значений. Принципиальной разницы между двумя этими командами нет: любая разветвляющаяся конструкция может быть записана с применением одной из них. Команды применяются в зависимости от того, какая из них удобнее пользователю в конкретном случае.

Затем следует коснуться некоторых особенностей реализации этих команд в Е-практикуме:

в строках ЕСЛИ и ПРИ может быть записано только условие;

серии выполняемых команд в ВЫБОРЕ записываются ниже и каждый случай отделяется друг от друга строчкой «———»;

в записи полной формы команды выбора слово ИНАЧЕ не используется, серия команд, соответствующая этому случаю, должна быть записана вслед за последней строчкой «———» перед словом ВСЕ;

вставка и удаление соответствующих конструкций происходит только целиком.

В качестве первого упражнения на использование команды ветвления можно выполнить алгоритм решения квадратного уравнения $ax^2+bx+c=0$, подробно рассматриваемый в первой части УП (пример 7.1, с. 32).

Упражнение 1.

```
АЛГ КВУР (ВЕЩ a, b, c, x1, x2, ЛИТ ответ)
.   АРГ a, b, c
.   РЕЗ ответ, x1, x2
НАЧ ВЕЩ D
.   a:=
.   b:=
.   c:=
.   D:=b**2—4*a*c
.   ЕСЛИ D<0
.   . ТО ответ:=«нет решения»
.   . ИНАЧЕ ответ:=«есть решение»
        x1:=(—b—sqrt(D))/(2*a)
        x2:=(—b+sqrt(D))/(2*a)
.   ВСЕ
КОН
```

Следующее упражнение может быть использовано для наглядной демонстрации преимущества команды выбора при проверке серии условий.

Упражнение 2. Записать и выполнить алгоритм «запуск ракеты» (УП, ч. I, с. 38) с использованием команды ветвления и с использованием команды выбора.

В качестве дополнительного материала можно предложить учащимся выполнить алгоритм выбора типа «меню», отметив при

этом, что такого рода конструкции широко используются во многих как учебных, так и профессиональных программах.

Упражнение 3. Выполнить на ЭВМ упражнение 12 из УП, ч. II, § 6, с. 33.

В процессе выполнения учащимися упражнений могут появляться сообщения об ошибках. Кроме приведенных выше может быть выдано сообщение «нет условия», т. е. после служебного слова ЕСЛИ не записано условие; аналогичная ошибка может возникнуть при использовании команды выбора.

3-й урок. Команды повторения (ПОКА, ДЛЯ).

Теоретические сведения: УП, ч. I, § 2, п. 4, с. 27; ч. II, § 7, с. 33.

Основная цель. Познакомить учащихся с использованием двух вариантов команды повторения (с параметром и с условием) для записи циклически повторяющихся серий команд.

Требования к знаниям и умениям. Учащиеся должны уметь ввести в ЭВМ и исполнить несложный алгоритм, требующий использования команд повторения (организации циклов); знать, что такое условие, параметр и «тело» цикла; уметь выбрать для использования в программе нужный вариант команды повторения (с параметром или с условием).

Методические рекомендации. В начале урока следует вспомнить изученные ранее теоретические положения. Для чего служит команда повторения? Когда нужно использовать команду повторения с условием типа «пока» и команду повторения с параметром? Следует указать на ограниченность применения команды с параметром: для ее использования надо заранее знать число повторений цикла. Между тем существуют алгоритмы решения задач (например, методом последовательных приближений), в которых число повторений (или итераций) неизвестно да и не интересует человека. Важен лишь конечный результат — выполнение некоторого условия. В то же время команда с параметром упрощает запись большого числа алгоритмов. Перед началом и по ходу выполнения упражнений следует обращать внимание учащихся на такие важные понятия, как «условие выполнения цикла», «параметр», «тело цикла». Соответствующие части алгоритмов нужно непосредственно показывать на экране. Следует напомнить, что параметр цикла обязательно должен быть целочисленной величиной, пусть ученики не забудут его описать. То же касается начального и конечного значений параметров.

Особенно важна для этого занятия возможность пошагового исполнения алгоритмов. Ею следует широко пользоваться.

Учащимся нужно рассказать об особенностях реализации циклов в Е-практикуме; напомнить основные команды для вставки и удаления соответствующих конструкций.

Упражнение 1. Ввести и исполнить по шагам алгоритм уточнения корня уравнения $f(x)=0$ (УП, ч. I, с. 95) для функции $f(x)=x^3+2x^2-x-2$ на отрезке [0, 2] с заданной точностью $eps=10^{-4}$.

Упражнение 2. Ввести в ЭВМ и исполнить алгоритм приближенного вычисления площади криволинейной трапеции (УП, ч. I, с. 94) для функции $f(x)=x^2-2x+2$ на отрезке [0, 2], разбив его на 50 частей.

Учитывая, что на следующем занятии при работе с таблицами команда повторения с параметром будет применяться опять, сейчас следует сделать упор на освоение команды с условием. Для этого можно предложить выполнить вариант алгоритма УПР2 — вычисление площади с заданной точностью.

Целесообразно предложить учащимся менять задаваемую точность и следить, как при этом меняется время исполнения алгоритма.

4-й урок. Алгоритмы работы с табличными величинами.

Теоретические сведения: УП, ч. I, § 2, п. 10, с. 40.

Основная цель. Сформировать у учащихся практические навыки работы с алгоритмами решения задач по обработке информации, организованной в форме таблиц (массивов).

Требования к знаниям и умениям. Учащиеся должны уметь вводить в ЭВМ и исполнять алгоритмы, использующие линейные таблицы данных: алгоритм поиска элемента, обладающего заданным свойством, алгоритм сортировки.

Методические указания. Данное занятие фактически является продолжением предыдущего, так как работа с табличными величинами неотделима от использования команд повторения, т. е. организации циклических алгоритмов. На это следует с самого начала обратить внимание учеников.

Нужно отметить, что работа с двумерными, или прямоугольными, таблицами, как правило, требует организации вложенных циклов, так как элемент таблицы пронумерован двумя индексами.

Следует отметить также большую важность и практическую распространенность алгоритмов поиска и сортировки информации, организованной в виде таблиц. Можно указать возможные сферы применения таких алгоритмов (информационно-поисковые системы, работа с большими объемами информации — в экономике, статистике и т. д.).

Учащимся необходимо указать на особенности реализации табличных величин в излагаемой версии Е-практикума: служебные

слова ЦЕЛТАБ и ВЕЩТАБ должны писаться слитно, слова ЛИТТАБ вовсе нет, так как Е-практикум не допускает таблиц из строковых величин.

Упражнение 1. Ввести и исполнить на ЭВМ алгоритм «сумма» (УП, ч. II, с. 34) суммирования элементов линейной таблицы с использованием команды повторения с параметром.

Упражнение 2. Ввести и исполнить алгоритм поиска наименьшего элемента в линейной таблице (УП, ч. I, с. 93).

Упражнение 3. На метеостанции 3 раза в сутки проводятся замеры температуры — утром, днем и ночью. Составьте и исполните алгоритм, который: 1) вычисляет средние за неделю температуры — утреннюю, дневную и ночную; 2) вычисляет среднесуточную температуру для каждого дня недели.

5-й урок. Использование вспомогательных алгоритмов.

Теоретические сведения: УП, ч. I, § 4, с. 46.

Основная цель. Ознакомить учащихся с практическим использованием вспомогательных алгоритмов при решении задач; указать на использование алгоритмов вычисления значений функций в качестве вспомогательных; познакомить с использованием встроенных функций Е-практикума; сформировать представление о нисходящей разработке алгоритма «сверху вниз».

Требования к знаниям и умениям. Учащиеся должны уметь оформить алгоритм для использования его в качестве вспомогательного; знать правила передачи значений величин при вызове вспомогательного алгоритма; уметь правильно организовать вызов вспомогательного алгоритма в своей программе; уметь использовать алгоритм, включающий вызовы вспомогательных алгоритмов; уметь грамотно обращаться к встроенным функциям Е-практикума и использовать их в своей программе.

Методические указания. Перед началом работы следует сказать несколько слов о современной технологии проектирования и разработки сложных алгоритмов и программ «сверху вниз».

Отметить, что этот метод базируется на использовании готовых, ранее написанных, алгоритмов и включении их в состав вновь создаваемого. Ключевым моментом здесь является понятие вспомогательного алгоритма. Следует сказать, что во всех современных языках программирования есть средства для реализации подобного метода (обращение к подпрограммам, вызов процедур, использование библиотек подпрограмм).

Далее следует рассказать, что в Е-практикуме можно составлять алгоритмы, включающие обращения к вспомогательным алгоритмам. Последние могут быть написаны самим учеником для его целей; кроме того, существует целый ряд уже готовых вспомогательных алгоритмов — так называемые встроенные функции Е-практикума. Они вычисляют основные элементарные функции:

вещ sin(вещ x) — вычисление синуса;
вещ cos(вещ x) — —"— косинуса;
вещ exp(вещ x) — —"— экспоненты e^x;
вещ ln(вещ x) — —"— натурального логарифма;
вещ sqrt(вещ x) — извлечение квадратного корня;
вещ abs(вещ x) — абсолютная величина числа;
цел mod(цел a, b) — остаток от деления a на b;
цел int(вещ x) — целая часть числа.

Аргументы тригонометрических функций задаются в радианах. Типы величин, используемых в качестве аргументов, должны соответствовать указанным выше; переменные, которым присваиваются вычисленные значения функций, также должны быть нужного типа — на это следует обратить особое внимание учеников, так как здесь очень велика вероятность ошибки.

Для первого знакомства можно выполнить алгоритм вычисления логарифма по основанию 10 — эта функция в Е-практикум не встроена.

Упражнение 1.

```
АЛГ ВЕЩ LG(ВЕЩ x)
.АРГ x
. РЕЗ
НАЧ
. знач:=ln(x)/ln(10)
КОН
```

Упражнение 2. Найти сумму факториалов всех целых чисел от 1 до 10, используя алгоритм вычисления факториала как вспомогательный (УП, ч. II, с. 49).

Упражнение 3. Исполнить на ЭВМ алгоритм упорядочения по возрастанию элементов линейной таблицы чисел, используя в качестве вспомогательного алгоритм поиска минимального элемента (УП, ч. I, с. 93).

6-й урок. Алгоритмы работы с литерными величинами.

Теоретический материал: УП, ч. II, § 9, с. 52—57, приложение IV, с. 127—138.

Основная цель. Закрепить знания учащихся и привить практические навыки по составлению и исполнению алгоритмов с литерными величинами.

Требования к знаниям и умениям. Учащиеся должны знать правила записи алгоритмов с литерными величинами и операций

над ними; уметь, используя Е-практикум, записать такой алгоритм и исполнить его на ЭВМ.

Методические указания. Это занятие имеет особое значение. На нем учащимся будет продемонстрирована возможность обработки текстовой информации на ЭВМ. Важно, чтобы у них не сложилось впечатление, что компьютер служит в основном для вычислений, является «большим калькулятором». Следует напомнить школьникам о том, что в настоящее время задачи вычислительного характера занимают не более 20 % времени использования ЭВМ; в основном компьютер применяется для обработки текстовой и графической информации.

В начале урока можно предложить записать несколько литерных переменных, присвоить им некоторые значения и исполнить алгоритм.

Упражнение 1.

```
АЛГ УПР 1 (ЛИТ к, л, м, н)
    АРГ к, л=«жена», «муж»
    РЕЗ м, н
НАЧ
    м:=к
    н:=л
КОН
```

Поле вывода результатов в Е-практикуме имеет ограниченные размеры, поэтому не следует задавать литерные переменные длиной более 14 символов.

Затем можно переходить к операциям над литерными величинами.

Операция соединения, или склеивания, обозначается знаком «+». Необходимо сразу продемонстрировать разницу между операцией соединения и арифметическим сложением.

Упражнение 2. Записать и исполнить алгоритм.

```
АЛГ УПР2 (ЦЕЛ а, б, в, г, ЛИТ д, е, ж, з)
    АРГ а, б=654, 876, д, е=«654», «876»
    РЕЗ в, г, ж, з
НАЧ
    в:=а+б
    г:=б+а
    ж:=д+е
    з:=е+д
КОН
```

Операция вычисления длины текста часто используется при поиске нужной текстовой информации. Например: выбрать из списка городов те, названия которых состоят из четырех букв.

В качестве упражнения можно привести соотношение ДЛИН(а+б)=ДЛИН(а)+ +ДЛИН(б).

Упражнение 3. Записать и исполнить алгоритм.

```
АЛГ УПР3 (ЦЕЛ э, ю, я, ь, ЛИТ а, б)
    АРГ а,б=«мясо»,«рубка»
    РЕЗ э, ю, я, ь
НАЧ
    э:=ДЛИН(а)
    ю:=ДЛИН(б)
    я:=ДЛИН(а+б)
    ь:=э+я
КОН
```

Операция вырезки также используется при поиске. Например: выбрать все фамилии жителей Бердичева, оканчивающиеся на «швили».

Упражнения можно взять из УП, § 5, 8, 9.

7-й урок. Решение задач из курсов математики и физики.

Теоретические сведения: УП, ч. I, § 5, с. 53, § 7, с. 62, § 8, с. 68.

Основная цель. Продемонстрировать учащимся возможность решения различного рода задач из курсов физики и математики с помощью ЭВМ; дать представление об использовании приближенных методов вычислений, познакомить с понятием вычислительной модели, используемой при численном решении задач; закрепить приобретенные навыки работы с Е-практикумом.

Требования к знаниям и умениям. Учащиеся должны уметь ввести в ЭВМ и выполнить алгоритмы решения физических и математических задач, приведенные в пробном учебном пособии; уверенно использовать возможности Е-практикума для ввода, редактирования и исполнения достаточно сложных алгоритмов с развитой структурой; уметь использовать различные виды величин (в том числе табличный) в качестве входных, промежуточных и выходных; понимать сообщения системы и адекватно реагировать на них.

Методические указания. На этом занятии учащимся предстоит, пользуясь всеми полученными знаниями и навыками работы с компьютером, составить и исполнить в Е-практикуме несколько содержательных алгоритмов для решения типичных вычислительных задач. Следует сказать несколько слов о характере используемых алгоритмов.

Большинство расчетных задач можно решить различными способами, в том числе некоторые с применением точных аналитических формул. Но инструмент для решения этих задач — ЭВМ, поэтому целесообразно применять алгоритмы, наиболее удобные для реализации именно на машине. Как правило, такими являются универсальные вычислительные алгоритмы, использующие методы приближенных вычислений. Таким образом, при подготовке задачи для решения на машине происходит как бы абстрагирование от ее конкретного содержа-

ния. Особенно это касается решения физических задач. Здесь конкретное физическое содержание несколько отходит на второй план и первостепенное значение приобретает математическая вычислительная модель. Целые группы физических задач объединены именно этим признаком, сильно различаясь по физическому содержанию. Примеры — подсчет линейной регрессии физических величин с помощью метода наименьших квадратов, различные варианты метода конечных элементов (деление физического объекта на мелкие части, в пределах которых его физические характеристики можно считать постоянными).

Таким образом, следует особенно подчеркнуть одно из важнейших свойств применяемых алгоритмов — их массовость.

Далее можно отметить, что реализация таких алгоритмов приближенных вычислений на ЭВМ диктует повышенные требования к их теоретическому обоснованию: вопросам сходимости (существования решения), достигаемой точности вычислений и т. д. Как правило, эти аспекты сложны и выходят за рамки школьной программы. Поэтому очень полезно дать ученикам возможность на практике «увидеть» сходимость алгоритма, посмотреть, как влияет задаваемая точность вычислений на время исполнения (количество итераций).

Перед началом практического выполнения задач следует отметить некоторые особенности Е-практикума, ограничивающие возможности решения сложных вычислительных задач: максимальная точность числа — не более 5 значащих цифр; скорость вычислений довольно низка — около 30 операций в секунду, что не идет ни в какое сравнение с возможностями реальных ЭВМ.

Выбор упражнений для практической работы здесь должен быть максимально свободным. Каждому ученику можно предложить для решения задачу, отвечающую его подготовленности.

Рекомендуется выбрать подходящие примеры и упражнения из соответствующих разделов УП, ч. I, § 7, 8; особенно полезно включить использование метода наименьших квадратов.

Продолжение следует

И. АНТИПОВ, М. СТЕПАНОВ
НИИ школ Минпроса РСФСР

Упражнения по графике ПЭВМ «Агат»

Практика показывает, что учащихся любого возраста увлекает работа на персональных ЭВМ. Особый интерес вызывают игровые программы, а также богатые графические возможности компьютера. При изучении учащимися старших классов основ информатики можно наблюдать творческий подъем, когда они занимаются машинной графикой. Цветовые и динамические возможности графического дисплея ПЭВМ «Агат» позволяют реализовать содержательные задачи по обработке графической информации. Чтобы дисплей не превращался в обычный экран для демонстрации мультфильмов, чтобы учащиеся не были пассивны и не ограничивались примитивными действиями, нажимая на отдельные клавиши, необходимо систематически давать им задания для самостоятельной работы, постепенно знакомить с графическими возможностями компьютера. Важно, чтобы они сами создавали сюжеты изображений, управляли цветом и движением объектов.

В основу статьи положены учебные материалы, которые были использованы в экспериментальной работе в школе № 502 Пролетарского района Москвы. Работа проводилась на ПЭВМ «Агат». Предлагаемая здесь система упражнений может быть использована в курсе информатики в IX—X классах, ее можно рекомендовать для факультативных и внеклассных занятий с учащимися других классов. Эта система (порядок и постановка задач) не связана с конкретной ЭВМ и успешно используется, в частности, при работе с ПЭВМ «Ямаха» в школах Ждановского района Москвы. Вообще, работать с ней можно на любой ЭВМ с графическим (и особенно с цветным) дисплеем.

Экран дисплея содержит 32 строки, в каждой из которых можно разместить по 32 символа. Режим принято называть нормальным, когда на черном фоне экрана высвечиваются белые символы. Если же, наоборот, фон экрана белый, а символы — черные, то режим называют инверсным. Каждый режим задается

соответствующим оператором. Заметим, что клавиша *ввод* (изогнутая стрелка) означает ввод информации или перевод строки. Ее нажатие приводит к исполнению оператора. Так, исполнение оператора NORMAL устанавливает нормальный режим, а исполнение оператора INVERSE — инверсный режим. Для очистки экрана служит оператор HOME.

Пример 1. Выполнение операторов

```
INVERSE
HOME
```

очищает экран (экран белого цвета).

Пример 2. Выполнение операторов

```
NORMAL
HOME
```

очищает экран (экран черного цвета).

Замена белого цвета на другой осуществляется с помощью оператора RIBBON с указанием номера цвета, а именно:

RIBBON =
- 0 черный
- 1 красный
- 2 зеленый
- 3 желтый
- 4 синий
- 5 фиолетовый
- 6 голубой
- 7 белый

Операторы INVERSE и NORMAL отменяют цвет.

Пример 3. Установка красного, а затем желтого цвета экрана:

```
INVERSE
RIBBON = 1
HOME
RIBBON = 3
HOME
```

Далее выполним несколько операторов (они записаны в одной строке)

```
INVERSE : RIBBON = 4 : HOME
```

В результате экран станет синего цвета. Запишем короткую программу, при выполнении которой экран окрасится зеленым цветом.

```
10  INVERSE
20  RIBBON = 2
30  HOME
```

Отметим, что при выводе текста программы на экран командой LIST автоматически устанавливается нормальный режим. Перед началом новой программы, как правило, предыдущую уничтожают с помощью команды NEW.

Упражнение 1. Напечатать (вывести) в центре экрана слово АГАТ разноцветными буквами (крс., злн., жлт., син.).

При составлении программ воспользуемся операторами VTAB X и HTAB X, где X указывает номер строки для VTAB и номер позиции в строке для HTAB при выводе очередного символа.

Программа:

```
 5 HOME: VTAB 16 : HTAB 15
10 RIBBON = 1
15 PRINT «А»;
20 RIBBON = 2
25 PRINT «Г»;
30 RIBBON = 3
35 PRINT «А»;
40 RIBBON = 4
45 PRINT «Т»
```

Видоизмените программу так, чтобы слово АГАТ располагалось в самой верхней строке экрана; буквы были одного цвета; буква А — красного цвета, а другие — синего цвета.

Составьте программу для записи в центре экрана слов ПЭВМ АГАТ, причем ПЭВМ — зеленого цвета, АГАТ — красного цвета.

Упражнение 2. Составить программу, которая запрашивает фамилию и номер цвета. Напомним, что строковые переменные отмечаются символом $. В программе для фамилии воспользуемся переменной F$. Пусть печать будет в строке 16 с позиции 10 (не забудьте очистить экран!). Подсказка выдается в виде текста зеленым цветом.

Программа:	Пояснения:
10 HOME	очистка экрана;
20 RIBBON = 2	включение зеленого цвета;
30 INPUT «ФАМИЛИЯ?»; F$	запрос и ввод фамилии;
40 PRINT	пропуск экранной строки;
50 INPUT «НОМЕР ЦВЕТА»; C	запрос и ввод номера цвета;
60 HOME	очистка экрана перед печатью фамилии;
70 RIBBON = C	включение нужного цвета;
80 VTAB 16	установка курсора в строке 16;
90 HTAB 10	перевод курсора в позицию 10;
100 PRINT F$	печать фамилии.

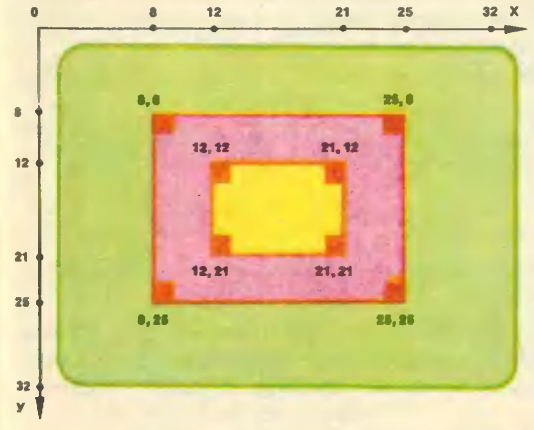
Исполните данную программу, чтобы на экран вывести свою фамилию.

Упражнение 3. Составить программу, которая напечатает слово АГАТ лесенкой:

```
АГАТ
 АГАТ
  АГАТ
   ...
```

Программа:

```
10 HOME: RIBBON = 1
20 FOR I = 1 TO 29
30 HTAB I
40 PRINT «АГАТ»
50 NEXT I
```



Упражнение 4. Закрасив экран зеленым цветом, изобразить на нем вложенных друг в друга два прямоугольника: внешний — красный, внутренний — желтый. Координаты вершин указаны на рис. 1. В программе предусмотрим следующий порядок «рисования»: по зеленому фону с помощью оператора цикла нарисуем красный прямоугольник, а затем внутри его — желтый.

Рис. 1

Программа:

```
10 INVERSE:RIBBON=2:HOME
20 F$="␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣"
30 S$="␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣"
40 VTAB 8:RIBBON=1
50 FOR I=8 TO 25
```

```
60 HTAB 8:? F¤
70 NEXT I
80 VTAB 12:RIBBON=3
90 FOR I=12 TO 21
100 HTAB 12:? S¤
110 NEXT I
```

Пояснения: Пробел — это «пустой символ», прямоугольник, заполненный фоновым цветом. F¤ содержит набор пробелов для закраски внешнего прямоугольника (полоса состоит из 18 символов); S¤ содержит набор из 10 пробелов — полоса для закраски внутреннего прямоугольника. Первый цикл (строки 50—70 программы) закрашивает внешний прямоугольник, второй цикл (строки 90—110 программы) закрашивает внутренний прямоугольник. С помощью HTAB устанавливается начало полосы пробелов.

В качестве дополнительных упражнений могут быть задания различных размеров и цветов окраски указанных прямоугольников, а также различное расположение внутреннего прямоугольника по отношению к внешнему прямоугольнику.

Упражнение 5. По синему фону в диагональных позициях экрана напечатать красные прямоугольники.

Программа:


```
10 INVERSE
20 RIBBON=4
30 HOME
40 RIBBON=1
50 FOR I=1 TO 30
60 HTAB I
70 PRINT«␣»
80 NEXT I
```

Программу можно записать с помощью меньшего числа строк (объединяя несколько операторов в одну строку):

```
10 INVERSE:RIBBON=4:HOME:RIBBON=1
20 FOR I=1 TO 30
30 HTAB I:PRINT «␣»
40 NEXT I
```

Выполнить то же упражнение, но изменить цвет экрана и диагональных прямоугольников. Что надо сделать в программе (какие внести изменения), чтобы получить диагональные прямоугольники синего цвета по красному фону?

Упражнение 6. Получить изображение красного прямоугольника по синему фону. Рассмотрим программу:

```
10 INVERSE:RIBBON=4:HOME
20 RIBBON=1:VTAB 8
30 FOR I=1 TO 18
40 HTAB 8:PRINT SPC(17); "␣"
50 NEXT I
```

SPC — сдвиг курсора от исходной позиции на указанное число и при этом меняется фон.

Нарисовать изображение, получаемое на экране по данной программе (рисунок выполнить на клетчатой бумаге и указать координаты прямоугольника). Как изменить цвет фона или прямоугольника, расположение и размеры прямоугольника?

Работа в графическом режиме

В рассмотренных упражнениях графические объекты складывались из «мозаики» символов (в первую очередь пробелов). Когда изображение строится из пробелов, минимальным его элементом является одна позиция строки (экрана), представляющая собой прямоугольник, в котором может разместиться один символ.

Тем не менее прямоугольники-позиции слишком велики, чтобы из них можно было составлять линии, ограничивающие контуры фигуры. До сих пор выполнялись упражнения в режиме символьного отображения информации. Далее мы перейдем к работе в графическом режиме. В графическом режиме позицию называют точкой. Различают несколько графических режимов. Приведем таблицу, в которой даны различные режимы работы и приведены основные их характеристики (табл. 1).

Точка ГНР получается из позиции символьного режима делением той на 4 части. Таким же образом из точки ГНР получается точка ГСР, а из точки ГСР — точка ГВР (рис. 2).

В символьном режиме нумерация строк и позиций в строчке начинается с единицы, а в графическом режиме — с нуля. Обозначим X — номер позиции в строке, а Y — номер строки. На рис. 3 на экранной графической странице отмечены точки с начальными координатами 0,0, с текущими координатами X,Y и наибольшими координатами X_{max}, Y_{max} (в правом нижнем углу экрана). В табл. 2 приведены значения X_{max} и Y_{max} в зависимости от типа графического режима.

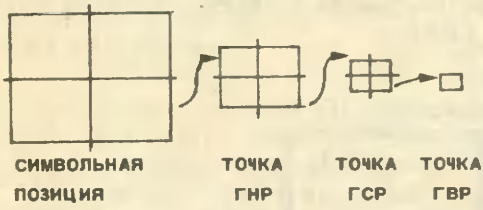
Таблица 1

Название режима	Цветность	Количество позиций	Способ включения
Символьный	Цветной	32×32 прямоугольника	TEXT=0
Графический низкого разрешения (ГНР)	»	64×64 точки	GR=2
Графический среднего разрешения (ГСР)	»	128×128 точек	MGR=1
Графический высокого разрешения (ГВР)	Черно-белый	256×256 точек	HGR=1

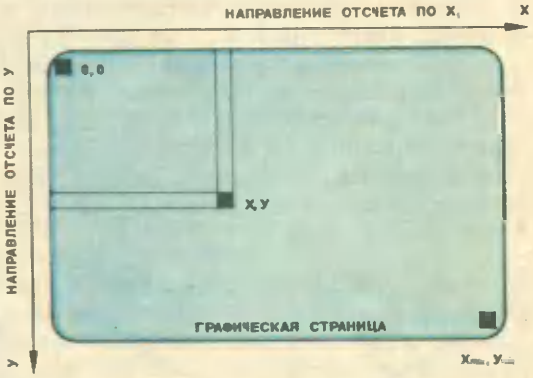
Таблица 2

Режим	ГНР	ГСР	ГВР
X_{max}	63	127	255
Y_{max}	63	127	255

2



3



Работа в графическом режиме предусматривает:

1. Включение режима.
2. Задание цвета. Оператор COLOR (обязателен в графическом режиме).
3. Создание графического объекта (производится оператором PLOT). Оператор PLOT имеет следующие модификации:

1) установка точки

PLOT X,Y

2) проведение отрезка от точки X_1,Y_1 к точке X_2,Y_2

PLOT X_1,Y_1 TO X_2,Y_2

3) проведение отрезка от последней точки, установленной на экране оператором PLOT до точки с координатами X,Y

PLOT TO X,Y

4) проведение ломаной с координатами вершин X_1,Y_1; X_2,Y_2; ... X_n,Y_n

PLOT X_1,Y_1 TO X_2,Y_2 ... TO X_n,Y_n

Координатами точек могут быть константы, переменные или арифметические выражения (дробная часть координаты отбрасывается).

Работать с графикой можно в программном режиме и в прямом диалоге. После включения графического режима набираемые с клавиатуры символы на экран не высвечиваются. По звуковому сигналу об ошибке «бип» надо снова набрать команду. Заметим, что работать «вслепую» достаточно сложно. Выход из графического режима в символьный: TEXT=0, или одновременно нажать клавиши УПР и СБР.

Упражнение 7. Установить точку на экране в заданном месте.

а) Установим точку с координатами X=32 и Y=32 в графическом режиме низкого разрешения ГНР. Работа ведется в прямом диалоге.

GR=2 — установка режима,

COLOR=1 — установка цвета,

54 PLOT 32,32 — установка точки.

Заметим, что две последние команды выполняются «вслепую» (набираемые символы не выдаются на экран). Лучше воспользоваться другим вариантом, когда эти операторы расположены в одной строке. Для их исполнения будет только один раз нажата клавиша *ввод*.

CR=2:COLOR=1:PLOT 32,32

б) Установим точку в режиме среднего разрешения. Какие при этом должны быть значения X и Y, чтобы точка была в той же части экрана, что и для случая а)? После сброса набрать и исполнить:

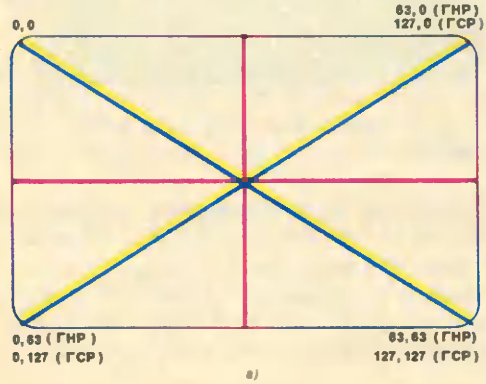
MGR=1:COLOR=1:PLOT 64,64

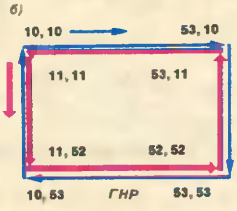
Далее выполните те же задания (а,б), но меняя цвет и расположение точки на экране.

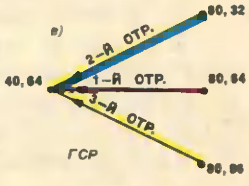
Упражнение 8. Провести диагонали экранного прямоугольника в режиме ГСР. Заметим, что левая диагональ соединяет точки с координатами 0,0 и 127,127, а правая — точки с координатами 0, 127 и 127,0. Пусть диагонали будут зеленого цвета: MGR=1:COLOR=2:PLOT 0,0 TO 127,127:PLOT 0,127 TO 127,0

Выполните то же задание, но в режиме ГНР. Какие при этом будут координаты вершин прямоугольника (через которые надо проводить диагонали)?

Упражнение 9. Разделите экранный прямоугольник на 4 части с помощью горизонтальной и вертикальной прямых, соединяющих середины противоположных сторон экранного прямоугольника. Какие координаты имеют середины сторон прямоугольника для режимов ГНР, ГСР, ГВР?







Упражнение 10. Получите на экране изображение, данное на рис. 4,а. Выполните данное задание в режиме ГНР и ГСР.

Упражнение 11. Выполните следующие действия:

GR=2:COLOR=4:PLOT 10,10 TO 53,10 TO 53,53 TO 10,53 TO 10,10

В результате должен получиться прямоугольник синего цвета (координаты вершин синего прямоугольника и направление его вычерчивания стрелками изображены на рис. 4,б). Запишите последовательность операторов, при которой внутри синего прямоугольника будет нарисован красный, с вершинами и направлением вычерчивания, отмеченным на рис. 4,б. Заметим, что указаны противоположные направления вычерчивания прямоугольников.

Упражнение 12. В режиме диалога «вслепую» выполнить следующие команды:

Команды:	Пояснения:
MGR=1:COLOR=1:PLOT 40,64	режим ГСР, цвет красный, точка 40, 64;
PLOT 80,64	установка точки X=80, Y=64;
PLOT TO 40,64	проведение 1-го отрезка (красный);
COLOR=2	зеленый цвет;
PLOT 80,32	установка точки X=80, Y=32;
PLOT TO 40,64	проведение 2-го отрезка (зеленый);
COLOR=3	цвет желтый;
PLOT 80,96	установка точки X=80, Y=96;
PLOT TO 40,64	проведение 3-го отрезка (желтый);
TEXT 0	выход из графического в символьный режим.

Для получения того же изображения (рис. 4,в) без учета направления вычерчивания отрезков выполните действия с использованием меньшего числа операторов PLOT.

Далее рассмотрим упражнения для построения изображений в программном режиме.

Упражнение 13. Вычерчивание контура параллелепипеда (рис. 5). Нарисуйте на бумаге контур параллелепипеда. Проставьте координаты его вершин (в предположении работы в графическом режиме среднего разрешения ГСР). Продумайте порядок обхода. Рассмотрите разные варианты обхода. Необходимо выбрать рациональный способ (с меньшим числом разрывов и без повтора обвода).

Выполнить следующую программу для получения изображения на рис. 5.

Программа:	Пояснения:
10 MGR=1	режим ГСР;
20 GOLOR=2	цвет зеленый;
30 PLOT 105,15 TO 75,45 TO 75,105 TO 15,105 TO 15,45 TO 75,45	построение ломаной от точки X=105, Y=15 к точке X=75, Y=45;
40 PLOT 15,45 TO 45,15 TO 105,15 TO 105,75 TO 75,105	построение ломаной от 15,45 до 75,105;
50 GET F○:TEXT=0	оператор GET ставится здесь для задержки изображения на экране.

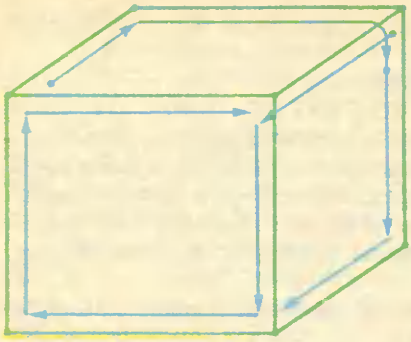
С помощью оператора GET запрашивается значение символьной переменной F○. После нажатия клавиши (введения любого символа) сработает оператор TEXT и произойдет переход в символьный режим (изображение исчезнет с экрана).

Можно видоизменить программу так, чтобы менялась окраска прямоугольника (изображение переливается различными цветами)

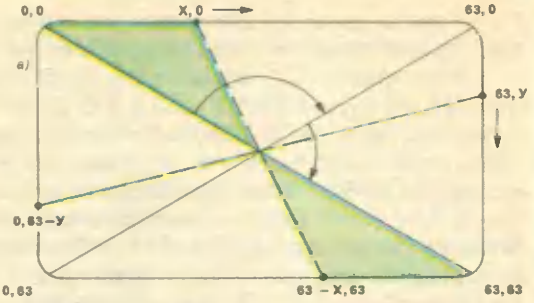
15 FOR I=1 TO 7	начало оператора цикла;
20 COLOR=I	номер цвета — значение I;
50 NEXT I:GOTO 15	конец цикла, переход к строке 15.

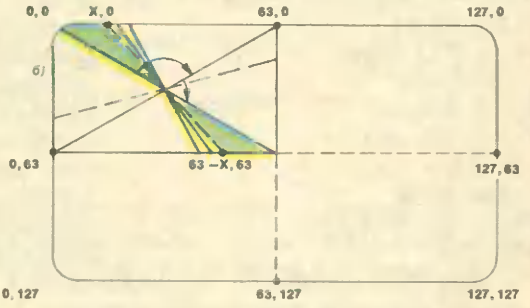
Остальные операторы без изменения. Программа зацикливается (для завершения необходимо сделать сброс). Можно еще усовершенствовать программу (с целью задержки изображения в очередном цвете):

50 NEXT I:GET F ○ :GOTO 15

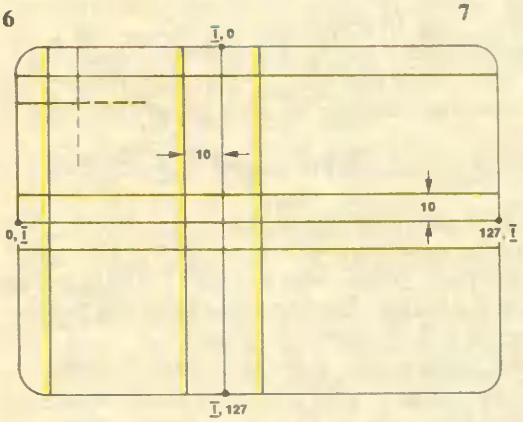


5





6



7

Упражнение 14. Закрасить экран.

```
10 HOME:VTAB 15:?«НОМЕР ЦВЕТА»   печать слов НОМЕР ЦВЕТА;
20 GET X                         запрос значения для цвета;
30 GR=2                          режим ГНР;
40 COLOR=X                       установка цвета (по X);
50 FOR Y=0 TO 63                 цикл с параметром Y;
60 PLOT 0, Y TO 63,Y             проведение горизонтальной прямой (внутри цикла);
70 NEXT Y                        конец цикла.
```

Далее в режиме диалога можно рисовать на цветном экране. Осуществить закраску экрана путем проведения вертикальных линий (как при этом изменится строка 60?). Выполнить то же упражнение в режиме ГСР (как при этом изменятся строки 30, 50, 60?).

Упражнение 15. Провести горизонтальные линии на экране. Рассмотрим режим ГВР и построение линий с шагом 10. В этом режиме оператор COLOR задает не цвет, а плотность точек на экране.

```
10 HGR=1                         режим ГВР;
20 FOR I=0 TO 15                 цикл (от 0 до 15);
30 COLOR=I                       установка плотности точек;
```

```
40 PLOT 0.10*I TO 255,10*I        горизонтальная линия;
50 NEXT I                         конец цикла;
60 GET F ⌘ :TEXT=0                запрос значения.
```

Провести таким же образом вертикальные линии. Выполнить упражнение для режима ГСР и для режима ГНР.

Упражнение 16. В режиме ГСР нарисовать сетку (решетку) с шагом в 10 позиций (рис. 6).

```
10 MGR=1:COLOR=3                  режим ГСР, цвет желтый;
20 FOR I=0 TO 127 STEP 10         цикл от 0 до 127 с шагом 10;
30 PLOT 0,I TO 127,I              горизонтальная линия;
40 PLOT I,0 TO I,127              вертикальная линия;
50 NEXT I                         конец цикла;
60 GET F ⌘ :TEXT=0                запрос значения.
```

Упражнение 17. Закрасить экран или его части, проводя всевозможные прямые, проходящие через центр.

Рассмотрим работу программы (построение в режиме ГНР).

```
10 GR=2                           режим ГНР;
20 COLOR=2                        цвет зеленый;
30 FOR X=0 TO 63                  цикл по X;
40 PLOT X, 0 TO 63—X, 63          концы отрезка «скользят» по горизон-
                                  тали;
50 NEXT X                         конец 1-го цикла;
60 FOR Y=0 TO 63                  цикл по Y;
70 PLOT 63,Y TO 0, 63—Y           концы отрезка «скользят» по вертикали;
80 NEXT Y                         конец 2-го цикла.
```


В результате работы 1-го цикла будут закрашены верхний и нижний треугольники экрана, а 2-й цикл произведет закраску левого и правого треугольников (полученных проведением диагоналей) (рис. 7, а).

Если изменить строку 10, т. е.

```
10 MGR=1
```

то аналогичная работа произойдет с закраской прямоугольника, расположенного в левой верхней четверти экрана (рис. 7, б). Переделать программу, чтобы она закрашивала аналогичным образом каждую четверть экрана. Выполнить следующую программу:

```
10 MGR=1
20 COLOR=2
30 FOR X=0 TO 63
40 PLOT X,0 TO 63—X,63            конец отрезка скользит по горизонтали
                                  в левой верхней четверти;
50 PLOT X+64,0 TO 127—X,63        в правой верхней четверти;
60 PLOT X,64 TO 63—X,127          в левой нижней четверти;
70 PLOT X+64,64 TO 127—X,127      в правой нижней четверти;
80 NEXT X                         конец цикла по X.
...
```

По этой части программы получится рис. 7, в. Далее будет оператор цикла по Y (допишите программу).

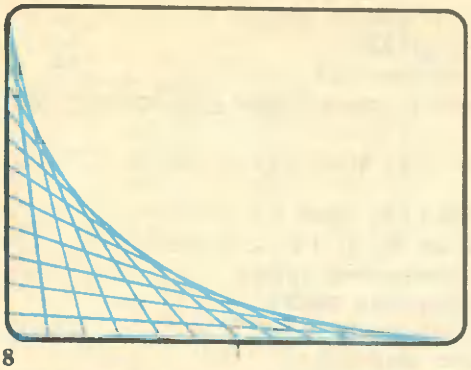
Упражнение 18. Рассмотреть программу, что происходит при ее работе:

```
10 MGR=1:COLOR=4
20 FOR I=0 TO 127
30 PLOT 0,I TO I,127
40 NEXT I
```

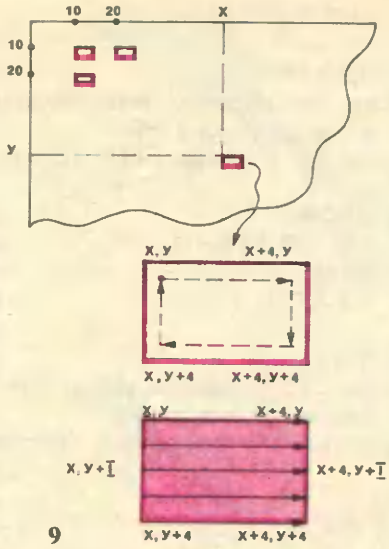
Работа программы будет более наглядной, если в заголовке цикла вставить шаг, например, 10, т. е. записать строку 20 в виде:

```
20 FOR I=0 TO 127 STEP 10
```

На рис. 8 дается изображение, полученное по рассматриваемой программе.



8



9

Какие изображения будут получены на экране для следующих модификаций строки 30:

```
30 PLOT 0,0 TO I,127
30 PLOT 0,0 TO 127,I
30 PLOT I,0 TO 127,I
```

Упражнение 19. Разобрать работу программы, в которой использованы вложенные циклы.

```
10 INVERSE:RIBBON=3:HOME
20 RIBBON=1
30 FOR Y=3 TO 29 STEP 2
40 FOR X=3 TO 29 STEP 2
50 VTAB Y:HTAB X
60 PRINT«␣»
70 NEXT X:NEXT Y
```

На экране прямоугольники бегут по строкам, и поэтому заполнение экрана происходит по строкам сверху вниз. Переставим циклы:

```
30 FOR X=3 TO 29 STEP 2
40 FOR Y=3 TO 29 STEP 2
70 NEXT Y:NEXT X
```

В этом случае прямоугольники бегут по столбцам. Изображение заполняет экран по столбцам слева направо.

Упражнение 20. В режиме ГСР нарисовать прямоугольники размеров 5×5 и шагом 10 по осям X и Y.

```
10 MGR=1:COLOR=1
20 FOR X=10 TO 110 STEP 10          внешний цикл;
30 FOR Y=10 TO 110 STEP 10          внутренний цикл;
40 PLOT X,Y TO X+4,Y TO X+4,Y+4     построение прямоугольника (проведе-
     TO X,Y+4 TO X,Y                ние ломаной);
50 NEXT Y:NEXT X                    окончание циклов.
```

Чтобы целиком закрасить каждый прямоугольник, надо воспользоваться новым циклом и расположить его внутри имеющихся, а именно (вместо построения ломаной — прямоугольника будет его заполнение):

```
40 FOR I=0 TO 4:PLOT X,Y+I
     TO X+4,Y+I:NEXT I
```

На рис. 9 дано изображение экрана, порядок обхода прямоугольника и его

заполнения. Выполните аналогичное упражнение с иными размерами прямоугольников и другими шагами по X и Y.

Дополнительные упражнения

Упражнение 21. Для задержки графической картинки на определенное время можно воспользоваться «пустым» оператором цикла (заголовок и конец цикла). Наибольшее значение для параметра цикла можно запрашивать (им определяется время задержки рисунка на экране).

Программа	Пояснения:
10 INPUT «ВВЕДИТЕ P»; P	запрос значения P;
20 GR=2:COLOR=6	режим ГНР, цвет голубой;
30 FOR X=1 TO 63 STEP 2	внешний цикл;
40 FOR Y=1 TO 63 STEP 2	внутренний цикл;
50 PLOT X,Y	установка точки X,Y;
60 NEXT Y:NEXT X	конец циклов;
70 FOR I=1 TO P:NEXT I	пустой цикл для задержки изображения (по значению P);
80 TEXT=0	выход в символьный режим.

Поработать с программой для разных значений P.

Упражнение 22. Составить программу для изображения на экране шахматной доски. Выполнить построения в режимах ГНР и ГСР. В режиме ГСР построить изображение шахматной доски во весь экран, затем в $^1/_4$ экрана, расположив при этом в центре экрана или в одной из четвертей экрана.

Упражнение 23. Движение символа по диагонали экранного прямоугольника.

10 INVERSE:RIBBON=1:HOME	красный экран;
20 INPUT A,B,C	ввод значений A,B,C;
30 FOR T=A TO B STEP C	цикл по параметру T;
40 RIBBON=4	синий цвет;
50 VTAB T:HTAB T	строка T, позиция T;
60 ? «␣»	печать пробела (синего квадрата);
70 FOR K=1 TO 100	пустой цикл для замедления скорости;
80 NEXT K	конец пустого цикла.

В строке 70 вместо числа 100 можно взять число 20 (работа будет быстрее, задержка в 5 раз меньше), если взять число 200, то задержка увеличится в 2 раза. Вообще, можно запрашивать значение P для интервала задержки изображения, это можно предусмотреть в строке 20. Если убрать строки 70—80, задержки не будет, символ — квадратик быстро движется. Продолжим работу по написанию программы:

90 RIBBON=1	красный цвет;
100 VTAB T:HTAB T	
110 ? «␣»	печать красного квадрата;
120 NEXT T	конец цикла по T.

При этом синий квадрат (см. строки 40—60) становится красным и не виден на красном фоне (закрашивается). Так как предыдущее положение не видно, то наблюдается движение вперед (вниз) по главной диагонали.

Если изъять строку 110, то не будет печать «закраски» и все первоначальные положения видны; будет строится ступенька. В строке 120 завершается цикл, организующий продвижение вперед. Без строк 90—100—110 нет закраски предыдущего квадрата, и поэтому эффект: все синие диагональные элементы видны.

Упражнение 24. Продолжить работу с программой из предыдущего упражнения (23). Организовать движение назад. Для этого заменим границы изменения параметра цикла и шаг (умножим его на —1). Таким образом подготовим к работе цикл для рисования квадратов в обратную сторону — снизу вверх по главной диагонали. Продолжим написание программы:

130 R=A:A=B:B=R	замена значений A и B;
140 C=—C	изменение знака шага C;

```
150 GET F⌑                    остановка перед сменой направления
                              движения, требуется ввод символа;
160 IF F⌑<>"E"THEN 30         проверка на конец работы;
170 END                       конец программы.
```

Условились, что при введенном символе E будет заканчиваться работа (условно считаем E — сокращение слова END). Если вместо строк 150 и 160 оставить лишь строку 160, в которой произойдет переход на строку 30, то после работы циклов остановки не будет; направление движения будет периодически меняться на противоположное.

Упражнение 25. Организация рисования по контурам.

Коды стрелок ←8 ↑25
→21 ↓26

Рассмотреть и выполнить программу:

```
10 INVERSE:RIBBON=1:HOME      цвет красный;
20 INPUT X,Y                  ввод X и Y для установки квадрата;
30 RIBBON=4                   синий цвет;
40 VTAB Y:HTAB X:?«␣»         печать квадрата по X и Y;
50 GET F⌑                     остановка для запроса № кода, надо ввести код стрелки;
60 K=ASC(F⌑)                  переменная K — число — код;
70 IF K=8 THEN X=X-1          одна позиция влево;
80 IF K=21 THEN X=X+1         одна позиция вправо;
90 IF K=26 THEN Y=Y+1         одна строка вниз;
100 IF K=25 THEN Y=Y-1        одна строка вверх;
110 VTAB Y:HTAB X             установка точки по X и Y;
120 ? «␣»                     для печати пробела;
130 GOTO 50                   переход на запрос кода.
```

В строках 70—80—90 можно добавить операторы GOTO 110. Так как работа производится быстро, то нажатие на клавишу со стрелкой приводит к рисованию квадрата в нужном месте (смещенному от предыдущего положения на одну позицию в указанном стрелкой направлении).

На этом можно завершить упражнения для начального этапа изучения графических возможностей ПЭВМ «Агат». Как видно, для выполнения разнообразных графических работ потребовался небольшой набор средств Бейсика в системе «Агат». Выполнение упражнений позволяет учащимся освоить работу с графикой в режиме диалога. Освоив простейшие, но основные элементы, они могут в дальнейшем изучить и другие графические возможности персонального компьютера — вычерчивание кривых линий, построение графиков функций, создание более сложных графических изображений.

Л. ЗВАВИЧ
Учитель-методист
Л. САМОВОЛЬНОВА
Ведущий методист Управления информатики и электронно-вычислительной техники Минпроса СССР

Телепередачи для десятиклассников

Телепередача «Как устроена ЭВМ?». На наш взгляд, ее следует показать учащимся не только X, но и IX класса. Они увидят большое количество разнообразных компьютеров, что очень полезно в условиях безмашинного преподавания.

Передача проходит в виде лекции-беседы, с которой обращается к нам ее автор и постоянный ведущий — академик А. П. Ершов. Ее можно смотреть без специальной подготовки, однако было бы целесообразно сообщить учащимся план, например такой:

примеры ЭВМ;

основные части компьютера, его внешние устройства;

электричество и ЭВМ;

способы передачи информации в ЭВМ;

двоичная система счисления — ключ к кодированию информации;

заглянем внутрь машины: магистраль, байты и биты, адресные провода;

процессор изнутри (ДВК-2М): платы, печатный монтаж, микросхемы, внутренняя память;

интегральные схемы;

дисплей;

клавиатура; специальные клавиши, курсор;

магнитная лента, гибкий магнитный диск (дискета);

печатающее устройство (принтер).

Подробный рассказ обо всем этом будет иллюстрироваться показом всевозможных компьютеров, которые применяются в школе.

Трудной для понимания может оказаться двоичная система записи. В передаче она весьма удачно интерпретируется как взвешивание при помощи гирек в 1, 2, 4, 8 и т. д. граммов, каждая из которых используется не более одного раза. Так, например, чтобы «взвесить» число 40, нужно взять одну гирю в 32 г и одну в 8 г, а гири в 16, 4, 2 и 1 г не брать ни разу. Выражаясь более точно, число 40 можно записать как $40=1\cdot32+0\cdot16+1\cdot8+0\cdot4+0\cdot2+0\cdot1$. Таким образом, в двоичной системе число 40 будет представлено в виде 101000. И наоборот, если двоичная запись числа 1101, то его можно записать как $1\times8+1\times4+0\times2+1\times1=13$.

Чтобы дать возможность учащимся в какой-то мере представить запись целого числа (меньшего по модулю 32) в машине, можно, усадив в ряд шесть человек, попросить первого поднимать руку, если число положительное, и не поднимать, если число отрицательное; остальных — представить себе, что у них есть гирьки соответственно по 16, 8, 4, 2 и 1 г, и поднимать руку, если их гирька понадобится. Так, для представления числа +25 руки поднимут 1-й, 2-й, 3-й и 6-й ученики:
$+25=+(16+8+1)=+11001_2$;
числа —19 — 2-й, 5-й и 6-й:
$-19=-(16+2+1)=-10011_2$.

На самом деле в машине отрицательные числа записываются не совсем в таком виде. Число +19 будет представлено как
0000000000010011,
а число —19 в виде
1111111111101101.
Наиболее заинтересовавшимся этим вопросом учащимся можно порекомендовать книгу С. В. Фомина «Системы счисления» (популярные лекции по математике, вып. 40).

Телепередача «Новые конструкции алгоритмического языка» посвящена команде выбора и команде повторения с параметром, которую, как и в учебном пособии, называют циклом «для».

А. П. Ершов начинает с рассмотрения уже изученных команд ветвления и повторения (т. е. цикла «пока»). Он обсуждает вопрос о том, зачем нужна

команда выбора. Поскольку здесь есть ссылка на учебное пособие, то целесообразно предварительно ознакомить учащихся с его 6-м и 7-м параграфами.

Для лучшего понимания целесообразности применения команды выбора можно рассмотреть с учащимися (либо до, либо после передачи) два примера на составление программы. Они похожи на приведенные в передаче.

1. Двое играют в кости по следующим правилам: первый выигрывает, если выпадает нечетное число, второй — если четное, т. е. при выпадении 1, 3, 5 выиграл первый, иначе — выиграл второй. Всё.

Несколько изменим правила: пусть первый выигрывает, если выпадает простое число, а второй — если составное; если же имеет место третий случай, то фиксируется ничья. В такой ситуации и применяется команда выбора, позволяющая записать алгоритм точно и наглядно.

выбор
  при выпадении 2, 3, 5: выиграл первый
  при выпадении 4, 6: выиграл второй
  при выпадении 1: ничья
все

После аналогичного примера в передаче ставится вопрос о том, в каком порядке нужно записывать условия при записи команды выбора. Ведущий рассказывает о составлении программы, в которой перестановка таких условий приводит к ошибкам. Мы тоже рассмотрим подобный пример.

2. Определить по номеру месяца, к какому кварталу он относится.

выбор
  при номере месяца, меньшем 4: первый квартал
  при номере месяца, меньшем 7: второй квартал
  при номере месяца, меньшем 10: третий квартал
  при номере месяца, меньшем 13: четвертый квартал
все

Если здесь последнее условие поставить в самом начале, то машина отнесет любой месяц к четвертому кварталу.

В передаче весьма подробно рассмотрена разница между первым и вторым примерами, сделан вывод о том, когда возможны перестановки условий, а когда нет.

Чтобы учащиеся лучше уяснили команду «для», советуем перед просмотром напомнить им, что такое индексы. Можно, дав ряд чисел, попросить учеников найти сумму всех чисел, с первого по восьмое или с третьего по десятое, используя для этого цикл «пока».

Отметим в заключение, что во время просмотра передачи в основном надо смотреть и слушать. Материал излагается просто и доступно; восприятие его не должно вызвать затруднений.

Телепередача «Алгоритм вычисления функций». Ее цель — расширение и углубление знаний, полученных учащимися при изучении соответствующей темы по учебному пособию (§ 8). Это как бы продолжение передачи о решении задач по математике и физике (Информатика и образование. 1987. № 1). Она делится на две части.

В первой А. П. Ершов напоминает школьникам общее понятие функции и функциональной зависимости, сначала одной, затем нескольких переменных. Поскольку даваемые пояснения четкие, но весьма беглые, советуем поговорить с учащимися по этому поводу заранее. Может быть, в частности, использован материал учебника А. Н. Колмогорова «Алгебра и начала математического анализа» для IX—X классов (глава 1, § 2, 3).

В школьном курсе математики отсутствуют примеры функций нескольких переменных, поэтому их следует рассмотреть особо. Это можно сделать на уроках как информатики, так и математики и физики.

Для данного n $f(x)=x^n$ является степенной функцией аргумента x; при значении аргумента $x=3$ функция принимает значение $f(3)=3^n$.

Функция $f(n)=x^n$ является для всякого положительного x показательной функцией аргумента n; при значении $n=2$ функция принимает значение $f(2)=x^2$.

Функция $f(x; n)=x^n$ является функцией двух переменных x и n; при значении этих аргументов $x=3$, $n=2$ $f(3; 2)=$

$=3^2=9$, а при $x=2$, $n=3$ $f(2; 3)=2^3=8$.

Формула $S(t)=v_0t+at^2/2$ задает путь равнопеременного движения как функцию от аргумента t (времени) и в то же время может быть рассмотрена как функция трех переменных $S(v_0; a; t)=v_0t+at^2/2$. Аналогично $I=U/R$ может быть рассмотрена как $I(U)=U/R$, $I(R)=U/R$ и, наконец, как $I(U, R)=U/R$.

Учащиеся должны понять, что именно функции нескольких переменных главным образом и описывают соотношения в реальном мире.

На уроках стоит вспомнить и материал, связанный со сложными функциями, он также рассмотрен в указанном учебнике (гл. II, § 5). Так, если $\varphi(x)=1+x^2$, а $f(x)=\sqrt{x}$, то композиция первой и второй функций $f(\varphi(x))=\sqrt{1+x^2}$. Функцию $\gamma(x)=\sin(3x+1)$ можно рассматривать как композицию $\varphi(x)=3x+1$ и $f(x)=\sin x$, тогда $\gamma(x)=f(\varphi(x))$. Этот подход и используется в передаче.

Зрители вспомнят также и способы задания функций: аналитический, графический, табличный и др.

Таким образом, данная передача может послужить стимулом для повторения на новом уровне вопросов, связанных с функциональной зависимостью.

Во второй, главной, части рассматривается вычисление радиуса окружности, описанной около прямоугольника со сторонами y и x:

$$r(x; y)=1/2\sqrt{x^2+y^2}.$$

На этом примере будет объяснено, почему для алгоритмов вычисления функций нужно ввести некоторые специальные обозначения, т. е. расширить алгоритмический язык. На это нужно обратить особое внимание учащихся.

Возможно, будет полезно рассмотреть процесс вычисления значения функции $r(x; y)=1/2\sqrt{x^2+y^2}$ для каких-то конкретных x и y, например $r(5; 12)$:

I шаг $x^2=5^2=25$
II шаг $y^2=12^2=144$
III шаг $x^2+y^2=25+144=169$
IV шаг $\sqrt{x^2+y^2}=\sqrt{169}=13$
V шаг $r(x; y)=\frac{1}{2}\cdot 13=6{,}5$

В передаче будет рассмотрен соответствующий фрагмент алгоритма.

Вероятно, некоторую трудность для понимания может вызвать предложенный ведущим способ вычисления приближенного значения квадратного корня. Стоит заранее показать один из возможных вариантов этого вычисления, например по формуле

$$x_{n+1}=\frac{x_n+\frac{a}{x_n}}{2}\quad (a>0).$$

Не вдаваясь в теоретическое обоснование, вычислим приближенно квадратный корень из 3 (положим, что разность между двумя последовательными приближениями не должна превышать 0,1).

Пусть $x_1=1$, тогда $x_2=(1+3)/2=2$,

$|x_1-x_2|>0{,}1$; $x_3=(2+3/2)/2=1{,}75$;
$|x_2-x_3|>0{,}1$; $x_4=(1{,}75+3/1{,}75)/2=$
$=1{,}73...$; $|x_3-x_4|<0{,}1$, т. е. $\sqrt{3}\approx 1{,}7$.

После такого упражнения (полезно выполнить его на микрокалькуляторе) будут понятны соответствующие рассуждения из передачи.

В заключение укажем на пункты плана, которые стоит записать заранее в тетради учащихся для более организованного прослушивания:

примеры и символическая запись функции одной или нескольких переменных;

способы задания функций, примеры;

действие и результат действия. В чем разница между этими понятиями? Команда и результат ее выполнения;

алгебраическое выражение, его запись, двойной смысл этой записи;

для чего нужны специальные конструкции алгоритмического языка;

старая и новая записи алгоритма вычисления квадратного корня;

вызов алгоритма вычисления функции;

какие удобства дает введение алгоритмов вычисления функции;

список рекомендуемой литературы.

Эту передачу, как и большинство остальных, можно смотреть не только в системе всех передач, но и в отрыве от них.

С. КУЗНЕЦОВ
Кировоград
В. РАСПОПОВ
Канд. физ.-мат. наук
Киев

Моделирование работы программы

Математическое моделирование с применением ЭВМ является элементом алгоритмической культуры учащихся. Внеклассные занятия, будучи логическим продолжением обязательного курса информатики, дают возможность раскрыть суть метода математического моделирования на простых и выразительных примерах. Приведем один из них.

— Пусть каждый задумает число, — обращается учитель к ребятам. — Удвойте его и к результату прибавьте номер нашей школы. Теперь сумму умножьте на пять и прибавьте число учащихся, присутствующих на занятиях кружка. Все справились с заданием? А теперь число, которое вы задумали, отгадает компьютер.

И учитель передает кружковцам ПМК, в память которого он заранее ввел определенные величины и программу*:

Адрес	Команда	Код
00	ИП B	6L
01	—	11
02	ИП A	6—
03	÷	13
04	С/П	50
05	БП	51
06	00	00

⟨A⟩:=10; ⟨B⟩:=5·N+K, где N — номер школы, K — количество кружковцев.

Учащиеся по очереди вводят с клавиатуры результаты своих вычислений, пускают программу, неизменно получая, к своему удивлению, задуманные ими числа. Однако вскоре удивление пропадает. «Нужно с результатом проделать действия в обратном порядке», — почти одновременно догадываются некоторые из них.

Алгоритм отгадывания, по существу, готов:

вычесть из результата количество кружковцев;

разность разделить на пять;

из частного вычесть номер школы;

разность разделить на 2;

выдать результат вычислений в качестве ответа.

Кружковцы, владеющие техникой программирования на ПМК, без труда составляют программу в соответствии с найденным алгоритмом.

Адрес	Команда	Код
00	ИП 2	62
01	—	11
02	5	05
03	÷	13
04	ИП 1	61
05	—	11
06	2	02
07	÷	13
08	С/П	50
09	БП	51
10	00	00

⟨1⟩:=N, ⟨2⟩:=K.

Эта программа отлична от составленной учителем, хотя она выполняет ту же функцию. Вторая программа является математической моделью первой; программа учителя выступает в качестве объекта моделирования.

— Обратите внимание: моя программа короче, работает быстрее, а значит, она эффективнее. Составить ее мне помог предварительный анализ рассматриваемой ситуации, — подчеркивает учитель. — Обозначим задуманное вами число через X, а известные величины «количество кружковцев» и «номер школы» — через K и N. Получим уравнение

$$(2 \cdot X + N) \cdot 5 + K = a,$$

где a — результат вычислений.

Это математическая модель ситуации в виде формулы. Ваша математическая модель сразу представлена в виде алго-

---

* Эта и последующие программы написаны на языке программирования ПМК «Электроника Б3-34».

ритма. В данном случае модель в виде формулы предпочтительнее. Из уравнения находим

$$X=\frac{a-(5\cdot N+K)}{10}$$

и составляет более эффективный алгоритм вычислений:

вычесть из результата константу $5\cdot N+K$;

разность разделить на 10;

выдать результат вычислений в качестве ответа.

Очевидно, что показанная вам программа реализует этот алгоритм.

На таком простом примере учитель наглядно демонстрирует основные этапы математического моделирования с применением ЭВМ:

формализация задачи, когда в рассмотрение вводятся обозначения величин;

установление между величинами функциональной зависимости в виде формулы или алгоритма;

представление алгоритма в виде программы для ЭВМ.

Важно обратить внимание кружковцев на следующий факт.

В курсах физики, химии и других естественнонаучных дисциплин они изучают реальные объекты. В курсе информатики появляется новый объект исследования — ЭВМ, работающая по программе. Учащийся экспериментирует с предложенной учителем программой, т. е. устанавливает зависимость между аргументами программы и ее результатами, восстанавливает алгоритм ее работы. Этот алгоритм и является решением задачи.

Кружковцы, владеющие техникой программирования, могут пойти дальше. Они завершают упражнение составлением новой программы, функционально изоморфной исходной.

Обучение построению алгоритма не единственная цель математического моделирования работы исходной программы. Тут открываются широкие возможности познакомить учащихся с методом вычислительного эксперимента, имеющего большое значение в науке и технике. Приведем пример.

— Нидерландский физик Х. Гюйгенс, наблюдая движение маятника, обнаружил, что период колебания не зависит ни от массы, ни от угла отклонения от вертикали (при достаточно малых углах), — объясняет условие задачи учитель. — Период определяется только длиной маятника. Мною запрограммирована математическая модель (формула) зависимости периода T от длины l. Восстановите открытый Гюйгенсом физический закон, экспериментируя с программой.

Напомним, что формула, определяющая период колебаний математического маятника, имеет вид

$$T=2\pi\sqrt{\frac{l}{g}},\ \text{где}\ g=9{,}81\ \text{м/с}^2.$$

Программа для ее вычисления следующая:

Адрес	Команда	Код
00	ИП 9	69
01	÷	13
02	√	21
03	2	02
04	x	12
05	π	20
06	x	12
07	С/П	50
08	БП	51
09	00	00

Значение g заносится в адресуемую память: $\langle 9\rangle:=9{,}81$. Значение l (в метрах) набирается с клавиатуры, значение T (в секундах) выводится на индикатор ПМК.

Конечно, предлагать в качестве задачи вычислительный эксперимент с программой имеет смысл для тех кружковцев, которые не знакомы с точной формулой для расчета T. На занятиях создается ситуация, когда ребята как бы сами участвуют в открытии физического закона. Но объектом исследования служит математическая модель явления, реализованная в виде программы для ПМК.

Учитель советует кружковцам представить результаты эксперимента в виде таблицы.

l	0,25	0,5	1	2	
T	1,003	1,419	2,006	2,837	
l	4	5	7	9	16
T	4,012	4,486	5,308	6,018	8,024

Результаты вычислительного эксперимента анализируются.

— При увеличении длины l в четыре раза период T возрастает вдвое, при девятикратном увеличении l период становится в три раза больше, и при шестнадцатикратном увеличении длины значение T увеличивается вчетверо, — обращает внимание кружковцев на эти данные учитель.

В результате размышлений у учащегося возникает предположение о следующем законе: $T=k\sqrt{l}$. Значение коэффициента $k\approx 2$ находится путем подстановки конкретной пары значений T и l из таблицы. Таким образом, эмпирическим путем кружковцами «открыт» закон $T=2\sqrt{l}$. Данная формула с большой точностью описывает зависимость T от l в условиях Земли.

Рассмотренные выше задачи на исследование программ ЭВМ могут стать характерным типом заданий в новом школьном предмете. Здесь объектом исследования является работающая программа для ЭВМ, представляющая собой компьютерную модель некоторой реальной ситуации. Задача формулируется так: «играя» с программой, установить функциональную зависимость между аргументами и результатами компьютерной модели (объекта исследования). Метод решения задачи — математическое моделирование. Ответ — алгоритм, идентичный «засекреченному» в программе.

Возможности внеклассных занятий позволяют рассматривать в качестве компьютерных моделей комбинаторные игры с ЭВМ. В интересах формирования навыков математического моделирования важны такие особенности игр:

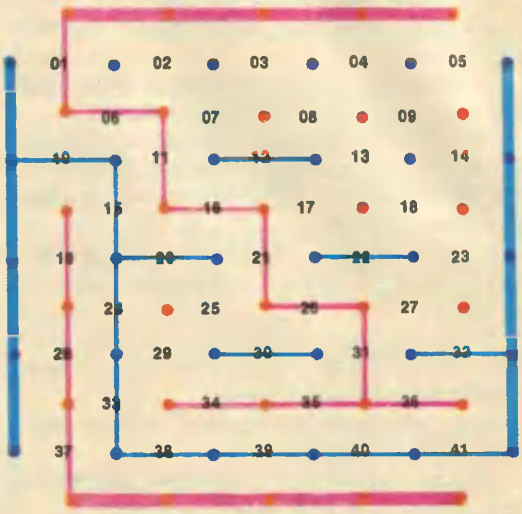
желание разгадать сущность «разумного» поведения компьютера побуждает ребят к созданию математической модели исследуемой программы;

эксперимент с игровой программой для установления зависимости между аргументами («мой ход») и значениями результата («ход компьютера») протекает естественным образом в процессе диалога с ЭВМ;

факт существования играющей программы придает уверенность в том, что построить математическую модель анализируемой игры возможно.

Играя с машиной, учащийся в конечном счете осознает, что в «разумном» поведении компьютера нет ничего таинственного: фактически состязание происходит с овеществленным в виде программы интеллектом ее автора.

Приведем пример построения на занятиях кружка математической модели классической комбинаторной игры.



В этой партии победил участник, начавший игру вторым (синие линии). Вот ее протокол: 01 10 06 12 11 20 16 22 21 30 26 32 31 40 36 31 35 39 34 38 37 33 28 24 19 15.

На рисунке изображено игровое поле. Участники по очереди проводят вертикальные и горизонтальные отрезки, соединяя ими точки своего цвета. Соединять точки по диагонали нельзя. Линии противников нигде не должны пересекаться. Выигрывает тот, кто первым построит ломаную, связывающую противоположные стороны доски своего цвета.

Задача моделирования игры решается кружковцами путем исследования готовой компьютерной модели:

Адрес	Команда	Код
00	ИП 8	68
01	+	10
02	П 9	49
03	К ИП 9	F9
04	—	11
05	С/П	50
06	БП	51
07	00	00

⟨0⟩:=8; ⟨2⟩:=16; ⟨8⟩:=12; ⟨B⟩:=16; ⟨Д⟩:=13; ⟨1⟩:=8; ⟨3⟩:=16; ⟨A⟩:=16; ⟨C⟩:=11 (для ПМК «Электроника МК-61» также ⟨E⟩:=8).

Программа имитирует поведение гроссмейстера в сеансе одновременной игры на многих досках. ЭВМ вступает в игру первой, соединяя отрезками красные точки, и весьма успешно использует свое, казалось бы, минимальное преимущество.

Партия начинается с того, что кружковец, играющий на доске под номером L, набирает на клавиатуре сообщение L 00 и нажимает клавишу С/П. В ответном ходе ПМК последние две цифры — номер отрезка, который машина «просит» нанести за нее на доску. Человек отвечает компьютеру в том же формате — «номер доски», «номер отрезка», а затем нажимает клавишу С/П.

Программа составлена таким образом, что сеанс одновременной игры завершается победой ПМК над всеми участниками. У кружковцев возникает законный вопрос: неужели ПМК, обладающий ограниченным объемом памяти, способен хранить всю информацию об игровых позициях на многих досках? Ответ может быть только один: между аргументами алгоритма игры и результатами существует простая функциональная зависимость. Тщательно тестируя программу, ребята вместе с учителем строят модель выигрышной стратегии в виде следующего алгоритма.

```
алг ИГРА (нат m, n)
    арг m
    рез n
нач цел K
    K:=m—[m/10]·10
    если K=0
        то n:=m+1
        иначе если K=1
                то n:=m—1
                иначе если K<6
                        то n:=m+4
                        иначе n:=m—4
                      все
            все
    все
кон
```

Здесь переменная, содержащая ход человека, обозначена через m, а ход ЭВМ — через n.

Кружковцы, владеющие техникой программирования на ПМК, могут написать не очень простую программу, реализующую этот алгоритм разветвляющегося типа; некоторые из них, формализуя эту задачу, вводят табличные величины, в результате получая более простой алгоритм и более короткую линейную программу игры. Но и в этом случае их программа будет отличаться от исходной, составленной с применением не описанных в инструкции особенностей ПМК. Учитель имеет возможность еще раз проиллюстрировать на примере этих трех программ, реализующих одну и ту же игру, понятие функциональной изоморфности компьютерных моделей.

Т. КОЖАНТАЕВ
г. Гурьев

Таблицы значений исполнения алгоритмов

Основным инструментом проверки алгоритмов, записанных на алгоритмическом языке, является исполнение их учащимися с помощью таблицы значений. Для них не существует обязательных правил оформления — учитель и учащийся могут изменять ее по своему усмотрению, в зависимости от целей конкретного урока. Предлагаемые здесь модификации таблицы значений и методические подходы к их использованию опробованы автором на занятиях в одной из школ г. Гурьева.

В соответствии с пробным учебным пособием таблица значений записывается отдельно от текста алгоритма; результаты каждого нового шага исполнения алгоритма заносятся в новую строку таблицы, что (при наличии в алгоритме команды повторения) может привести к

ее чрезмерному увеличению и затрудняет запись на классной доске. Удобнее разместить таблицу рядом с текстом алгоритма, записывая новые значения величин в клетки, расположенные на одном уровне с командами алгоритма. Пропуск выполнения строки обозначается стрелкой, конец ветвления, повторения, алгоритма — крестиком.

Приведем несколько примеров.

На рис. 1 (на обложке) заполнение показано «в динамике»: (1) — начальный вид, (2) — исполнение алгоритма, (3) — окончательный вид таблицы после исполнения алгоритма и определения результата.

При неоднократном исполнении алгоритма (для разных значений аргумента) таблицу можно продолжать вправо (рис. 2).

При исполнении алгоритма, содержащего команду повторения, для записи результатов каждого нового выполнения серии команд, входящей в команду повторения, используется новый столбец (рис. 3).

Здесь таблица С — и аргумент, и результат. Строку аргументов заполним сразу; в строку результата будем записывать значения элементов С[*i*], в итоге получим 5 величин — новое значение таблицы С.

В алгоритме решения биквадратного уравнения используются вложенные команды ветвления. Для упорядочения этой конструкции можно в таблице исполнения поставить открывающие (если) и закрывающие (все) скобки (рис. 4).

ЧТО МОЖЕТ ЭВМ?

Дышите глубже — вы взволнованны

Еще бы — идти к врачу, а попасть на прием к компьютеру! Но теперь это может случиться с каждым, и не только в столице. Сотрудничество разработчиков двух московских институтов — энергетического и областного научно-исследовательского клинического — увенчалось созданием специального анализатора. Его подключают к телефонной линии, пациент дышит в шланг, анализатор определяет необходимые параметры, кодирует их и передает ЭВМ,— может быть, за сотни километров. Проанализировав данные, машина выдает диагноз состояния органов дыхания или рекомендации по дальнейшему обследованию.

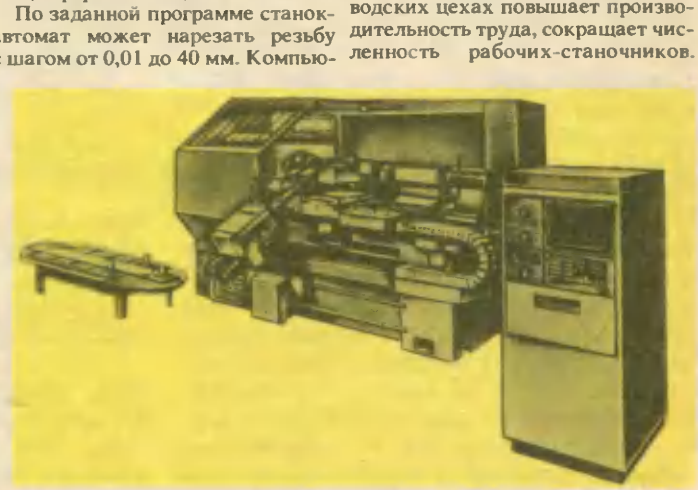
Дышите глубже!

Роботизированный комплекс

Роботизированные станочные комплексы, оснащенные микропроцессорами, значительно увеличивают эффективность современного производства. Один из них, созданный советскими инженерами, предназначен для автоматической обработки наружных и внутренних поверхностей сложных, сферических деталей.

По заданной программе станок-автомат может нарезать резьбу с шагом от 0,01 до 40 мм. Компьютер контролирует точность и режим обработки заготовок, включая и агрегаты комплекса. В его состав входит станок с ЧПУ 16К20ФЗ и робот-манипулятор, который подает заготовки на тактовый стол и передает на конвейер готовые детали. Наибольший их диаметр — 500 мм, длина — 900 мм.

Применение нового комплекса уменьшает время обработки деталей, улучшает качество готовой продукции. Его применение в заводских цехах повышает производительность труда, сокращает численность рабочих-станочников.



КАБИНЕТ ВТ

В. АШКЕНАЗЫ
Калининский государственный университет

Литерные величины на БК-0010

В языке программирования Фокал отсутствуют средства обработки литерных величин. Этот недостаток позволяет устранить доступный любому преподавателю информатики пакет подпрограмм обработки литерных величин, реализованный на КУВТ-86 в КВТ СПТУ № 16 г. Калинина.

Подпрограммы используют встроенную функцию FCHR. Для хранения литерных величин в оперативной памяти используется двухмерный массив, строки которого заполняются кодами символов запоминаемого текста. Каждая литерная величина (обозначим ее L_i) идентифицируется по индексу i, определяющему одну из строк массива. Первый элемент строки (элемент —128) содержит количество символов литерной величины. Первая строка массива (строка —128) зарезервирована для хранения промежуточных литерных величин. Имена всех переменных, используемых в подпрограммах, начинаются с буквы L, что следует учитывать при составлении вызывающих программ.

Рассмотрим некоторые из подпрограмм.

Ввод литерной величины (подпрограмма 100). Литерная величина L_{LI} с индексом LI=1,...,255 вводится с клавиатуры и запоминается в строке LI—128 двухмерного массива LT. Одновременно осуществляется контрольный вывод ее на экран. В соответствующей строке массива LT запоминаются также исправления литерной величины, производимые при вводе с помощью желтых клавиш редактирования. Начало ввода — при появлении двоеточия на экране; он завершается нажатием клавиши *ввод* или вводом 255-го символа, после чего в LT (LI—128,—128) заносится длина литерной величины — от 0 (пустой текст) до 255; на экране появляется «:» (конец ввода).

Текст подпрограммы:

```
100.1 T «:»; S LL=—127
100.2 S LT (LI—128, LL)=FCHR
      (FCHR(—1));
      I (LT(LI—128, LL)—13) 100.3, 100.4
100.3 S LL=LL+1; I (LL—128)100.2
100.4 S LT(LI—128,—128)=LL+127:
      X FCHR(124); R
```

Печать литерной величины (подпрограмма 101). На экран выводится литерная величина L_{LJ} с заданным индексом LJ из строки LJ—128 массива LT.

Текст подпрограммы:

```
101.1 S LL=—127
101.2 I (LT(LJ—128,—128)—LL—128)101.4
101.3 X FCHR(LT(LJ—128, LL));
      S LL=LL+1; G 101.2
101.4 R
```

Проверка равенства литерных величин (подпрограмма 102). Проверяется условие равенства двух литерных величин с индексами LX и LY. Если оно соблюдается, то переменная L0 принимает значение 1, в противном случае — 0.

Присваивание значения литерной величины (подпрограмма 103). Литерной величине с индексом LB присваивается значение литерной величины с индексом LA. Соответственно в LT (LB—128, —128) переносится и значение длины L_{LA}.

Соединение литерных величин (подпрограмма 104). Литерной величине с индексом LW присваивается значение соединения литерных величин с индексами LU и LV, причем допускается попарное равенство индексов. В LT (LW—128, 128) записывается значение суммарной длины, не превышающее

255. Избыточные символы (из текста с индексом LV) пропадают.

Вырезка из литерной величины (подпрограмма 105). Литерной величине с индексом LD присваивается значение фрагмента (от позиции L1 до позиции L2≥L1) литерной величины с индексом LC. Если L2<L1 или L1 больше длины L_{LC}, то в LD заносится пустой текст.

Текст подпрограммы:

```
105.1 I ((L2—1)*(LT(LC—128,—128)—L1)) 105.9
105.2 I (—L1) 105.4
105.3 S L1=1
105.4 I (L2—LT(LC—128,—128)—1)105.6
105.5 S L2=LT(LC—128,—128)
105.6 F LL=L1—128, L2—128; SLT(—128, LL—L1+1)=LT(LC—128,LL)
105.7 FLL= —127, L2—L1—127; S LT(LD—128, LL)=LT(—128, LL)
105.8 S LT(LD—128,—128)=L2—L1+1;R
105.9 S LT(LD—128,—128)=0;R
```

Частичное изменение значения литерной величины (подпрограмма 106). Символы в литерной величине с индексом LF, начиная с позиции L3 и кончая позицией L4≥L3, заменяются первыми L4—L3+1 символами литерной величины с индексом LE. Допускается LF=LE.

Перечисленные подпрограммы охватывают все операции обработки литерных величин, введенные в школьном курсе информатики. Нетрудно составить подпрограммы, обеспечивающие, например, определение позиции первого вхождения одной литерной величины в другую, и т. д.

В качестве примера использования описанного пакета рассмотрим задачу анализа формулы химического соединения. Программа выделяет из нее обозначения отдельных химических элементов и указывает число соответствующих атомов (напомним, что строчные латинские буквы имеют десятичные коды КОИ-7 от 97 до 122, а цифры — от 48 до 57).

```
1.01 X FCHR(12)
1.02 T «Введите химическую формулу.» ,!; S LI=1; D 100
1.03 S D=LT(—127,—128); I (D) 1.04,1.13
1.04 S LC=1; S LD=2; S L1=1; S L2=1; I (D—2)1.07
1.05 S Z=LT(—127,—126); I ((Z—97)*(122—Z)) 1.07
1.06 S L2=2
1.07 D 105; T ! «Элемент»; S LJ=2; D 101; T «.»
1.08 S K=1; S L1=L2+1; I (D—L1) 1.11
1.09 S Z=LT(—127, L1—128); I ((Z—48)*(57—Z))1.11
1.10 S K=Z—48; S L1=L1+1
1.11 T «Число атомов=»,% 1.0,K, «.»
1.12 S LD=1; S L2=D; D 105; G 1.03
1.13 Q
```

Пример диалога:

```
Введите химическую формулу...
:Na2CO3
Элемент Na. Число атомов=2.
Элемент C. Число атомов=1.
Элемент O. Число атомов=3.
```

Некоторое усложнение этой программы с предварительной записью в массив LT необходимых справочных текстов позволяет организовать, например, проверку правильности введенной химической формулы и диалоговый контроль знания учащимися обозначений и названий химических элементов.

А. КОВНЕР

Принципы реализации алгоритмического языка

Курс информатики в настоящее время рассчитан на «безмашинный вариант», но многие школы имеют возможность если не все занятия вести с использованием ЭВМ, то хотя бы изредка проводить практические работы с той или иной вычислительной техникой. При этом перед учителем встает проблема выбора языка, на котором учащиеся должны работать с машиной. Есть несколько путей ее решения: доставать специальное программное обеспечение, рассчитанное на школьный курс информатики; изучать с детьми язык программирования, реализованный на той ЭВМ, с которой им предстоит работать; реализовывать самостоятельно алгоритмический язык, описанный в учебных пособиях для IX и X классов. Каждый из них имеет свои недостатки.

Сложность первого в том, что так называемый Е-практикум реализован далеко не во всех операционных системах, к тому же это просто тренажер по написанию и отладке алгоритмов, на базе которого ничему, кроме принципов написания и работы алгоритма, научить нельзя, а хотелось бы иметь язык универсальный, на котором можно написать работающую, интересную программу.

При выборе второго изложенный в учебном пособии алгоритмический язык приходится изучать по программе, а учить параллельно два языка (алгоритмический и какой-то из языков программирования) за один час в неделю — задача сложная и для взрослого человека, не говоря уж о среднем десятикласснике.

Пойти третьим не позволяет профессиональная неподготовленность большинства учителей информатики.

Из этой ситуации каждый выкручивается как может, я рискнул пойти по третьему пути. Возможно, мой опыт окажется полезен и другим.

Выбор операционной системы. Обилие имеющихся в настоящее время ОС заставляет выбирать из них самую перспективную — уж если реализовывать язык, то в надежде, что в будущем результатами твоей работы смогут воспользоваться и другие. К тому же, если мы начинаем работать со школьниками на машине, то им приходится изучать ОС хотя бы на самом простом уровне, а изучать систему, имеющуюся лишь на одной-двух марках ЭВМ — значит даром терять учебное время.

На мой взгляд, оптимальной в этом плане является система типа UNIX — МНОС, ДЕМОС и т. д. Они все шире распространяются в Советском Союзе благодаря своей универсальности, мобильности, простоте работы и большому выбору предоставляемых возможностей.

Здесь описывается реализация алгоритмического языка в операционной системе МНОС РЛ1.2, разработанной ИПК Минавтопрома.

Формализация языка. Школьный алгоритмический язык на реализацию не рассчитан. Поэтому прежде всего приходится вносить в него ряд ограничений. Желательно, чтобы они были минимальными, так как дети изучают язык по учебному пособию и большие отличия от него приносят множество трудностей как для учителя, так и для ученика. Эти ограничения или изменения должны приближать алгоритмический язык к универсальным языкам программирования, чтобы облегчить школьникам переход на любой из них.

На алгоритмический язык были наложены следующие ограничения.

1. Описания переменных должны располагаться только в строке после слова «алг» в скобках.

2. Описания вспомогательных переменных должны располагаться только в строке после слова «нач».

3. Описания переменных различного типа отделяются друг от друга не запятой, а точкой с запятой, например: вещ А,Б; нат Ц.

4. Все переменные и служебные слова

должны выделяться в тексте пробелами с двух сторон.

5. Названия вспомогательных алгоритмов должны состоять только из одного слова.

6. Нет действия «возведение в степень»; вместо него можно пользоваться стандартной функцией POW(X,Y).

7. Все выражения должны записываться в одну строчку, т. е. отсутствует дробная черта.

8. Знаки арифметических действий записываются следующим образом: + — сложение; — — вычитание; / — деление; * — умножение.

9. Строки «арг» и «рез» должны присутствовать в тексте программы, даже если в них нечего писать (например, если программа аргументов не имеет). Это ограничение не распространяется на вспомогательные алгоритмы вычисления значения функции. Также необходимы скобки после имени алгоритма.

10. Если переменная является одновременно и результатом, и аргументом, ее надо записывать только в строку «рез».

11. При обращении из основной программы к вспомогательному алгоритму его результаты надо помечать знаком «&» (например: БИД (A, B,&C); здесь C — результат).

Ограничения 1—4 нужны для упрощения перевода и лучшего оформления программы. Если описания не помещаются в указанных строках, надо пользоваться знаком переноса строки (см.: Расширения языка).

Ограничение 5 стандартно для большинства языков и приучает к определенной культуре записи программ.

Ограничение 6 зависит от реализации, но не является вредным, так как многие современные языки (Паскаль, Си) не содержат возведения в степень, а школьники должны понимать, что это действие является для машины значительно более сложным, чем другие, и для его реализации необходима специальная программа.

Ограничения 7—8 являются стандартными почти для всех языков программирования. Необходимо учить школьников записи алгебраических выражений в строчку, если мы хотим научить их хотя бы азам программирования.

Ограничения 9—10 зависят от реализации.

Ограничение 11 введено сознательно. Опыт работы показывает, что больше всего ошибок у школьников при работе со вспомогательными алгоритмами возникает из-за того, что при обращении они забывают значения параметров. Наложенное ограничение заставляет школьника при обращении к вспомогательному алгоритму еще раз просмотреть его текст и внимательнее разобраться в его работе.

Расширения языка, их необходимость. Как уже отмечалось, цель реализации — получить язык, с одной стороны, как можно более совпадающий с языком учебных пособий, с другой — работающий, как язык программирования. Реализуемый язык должен давать возможность писать, отлаживать и выполнять интересные для ученика, полезные для его общего развития программы. Хотелось бы иметь возможность писать на нем программы, полезные для школьных курсов, и простейшие обучающие программы.

Для этого введены следующие дополнения.

1. В языке можно использовать математические функции из стандартного математического пакета языка Си.

2. Добавлены функции ввода-вывода:

ВЫВОДТ( ) — вывод произвольного текста, записанного в скобках;

ВЫВОДТНС( ) — то же, но после вывода текста не переводится строка;

ВЫВОДВ(A) — вывод вещественного значения переменной A;

ВЫВОДВНС(A) — то же, но после вывода значения не переводится строка;

ВЫВОДЦ(A) — вывод целого или натурального значения переменной A;

ВЫВОДЦНС (A) — то же, но после вывода значения не переводится строка;

ВЫВОДЛ(A) — вывод литерного значения переменной A;

ВЫВОДЛНС(A) — то же, но после вывода значения не переводится строка;

ВВОДВ(A) — ввод вещественного числа и присваивание его значения переменной A;

ВВОДЦ(A) — ввод целого или нату-

рального числа и присваивание его значения переменной А;

ВВОДЛ(А) — ввод строки и помещение ее в литерную переменную А.

3. Для написания программ с большим числом переменных и записи в строчку длинных формул вводится знак переноса строки со следующими правилами: конец обрыва строки помечается «#», и в начале продолжения ставится знак «#»; при трансляции разрыв строки заменяется на один пробел.

4. Для улучшения оформления программ и упрощения работы с уже написанными вспомогательными алгоритмами вводится знак комментария. Комментарием считается весь текст, расположенный в строке после слова «КОМ:». Текст на строке перед словом «КОМ:» комментарием не считается и воспринимается транслятором. Комментарием также считаются строка после слова «КОН» и пустые строки.

Запись программ. Основная программа и вспомогательные алгоритмы должны быть записаны каждый в отдельный файл, причем название файла с основной программой должно иметь окончание .E, вспомогательного алгоритма — .EV, алгоритма вычисления значения функции — .EF.

Чтобы получить выполнимую машиной программу, необходимо вызвать на трансляцию одновременно с основной программой все используемые ею вспомогательные алгоритмы. Это делается с помощью команды ER ⟨имя файла с основной программой и имена файлов со вспомогательными алгоритмами⟩ ⟨имя выполняемой программы⟩, например: ER PR.E PR1.EV PR2.EV PR3.EF PR

В результате выполнения этой команды в памяти машины образуется файл PR с текстом программы на языке машинных кодов. Чтобы выполнить программу, ученик должен дать команду PR.

Таким образом, чтобы воспользоваться программой, нужно каждый раз проделать довольно большую работу: создать (а при наличии ошибок и отредактировать) один или несколько файлов, собрать в одной команде имена файлов с текстами основной программы и всех вспомогательных алгоритмов; для этого иногда нужно найти необходимый файл, просмотреть его содержимое. Все это создает определенные трудности для преподавания, но зато дает возможность, изучая алгоритмический язык, одновременно знакомиться с операционной системой машины, привыкать работать с файлами. Можно создавать вспомогательные алгоритмы для многократного использования в разных программах, библиотеки вспомогательных алгоритмов — в общем, знакомиться почти со всеми видами применения языков программирования высокого уровня.

Язык может быть использован для кружковой работы, для написания простейших игровых и обучающих программ.

Способ реализации. Основной частью транслятора являются программы для простейшего макропроцессора операционных систем МНОС и ДЕМОС (ММ), с помощью которого алгоритмы построчно переводятся на соответствующие конструкции языка Си.

Например, при реализации команды «ВЫБОР»:

строка ВЫБОР переводится как IF(1<0) ↓;

строка ПРИ ⟨условие⟩:⟨команда⟩ как → ELSE IF (⟨условие⟩)↓ ⟨команда⟩

строка ИНАЧЕ ⟨команда⟩ как →ELSE ↓ ⟨команда⟩

строка ВСЕ как →

В результате работы макропроцессора получается файл с программой на языке Си.

Для основных алгоритмов и для вспомогательных программы различны.

Оболочка транслятора, написанная на управляющем языке указанных операционных систем — SHELL, по окончанию имени файла различает основной и вспомогательный алгоритмы и вызывает для них макропроцессор с соответствующей программой. Тексты на языке Си помещаются в один файл, к нему подключается математическая библиотека и вызывается Си-компилятор.

Все этапы работы транслятора отображаются на экране терминала, что дает возможность на уроках поговорить о его устройстве и трансляторах вообще.

Недоработки. Работа над транслято-

ром еще не закончена, не реализованы операции над литерными переменными в том виде, в котором они изложены в учебном пособии для X класса. Основным недостатком работающей в настоящее время версии транслятора является отсутствие диагностики ошибок, что вносит значительные трудности в работу, но и это временное явление.

Переносимость. Без внесения каких-либо изменений транслятор может быть установлен на ЭВМ СМ-4, СМ1420 и аналогичных в операционных системах МНОС или ДЕМОС. Для установки на ЭВМ серии ЕС в операционной системе ДЕМОС необходимо добавить вспомогательную программу, заменяющую русские идентификаторы и названия функций на латинские, так как Си-компилятор в ОС ДЕМОС ЕС не воспринимает русских идентификаторов.

Действующая в настоящее время версия транслятора со всеми вспомогательными файлами занимает всего 2K байта дисковой памяти (40 блоков).

Замечания по программированию. Несомненно, можно было написать эффективный транслятор с алгоритмического языка с описанными выше ограничениями и дополнениями. Его можно было бы написать на Си, с тем чтобы на выходе получать программу на ассемблере, но эта задача под силу только профессионалам высокой квалификации. Конвертор же на Си или на какой-либо другой язык может написать и непрофессионал, причем, если нет хорошего макропроцессора, подобный конвертор можно написать на любом языке высокого уровня, конечно, если в нем есть развитые средства для работы с файлами и текстовыми строками.

```
КОМ:                                    ПРИМЕР 1
КОМ: ПРОГРАММА ВЫЧИСЛЯЕТ ПЛОЩАДЬ КРИВОЛИНЕЙНОЙ ТРАПЕЦИИ,
КОМ: ОГРАНИЧЕННОЙ ГРАФИКОМ ФУНКЦИИ Y=SIN X, ПРЯМЫМИ X=A,
КОМ: X=B И ОСЬЮ АБСЦИСС, ЗНАЧЕНИЯ A И Б, ЧИСЛО РАЗБИЕНИЙ
КОМ: ЗАПРАШИВАЮТСЯ И ВВОДЯТСЯ С ТЕРМИНАЛА.
КОМ: СТРУКТУРЫ ЯЗЫКА — ВВОД, ВЫВОД, МАТЕМАТИЧЕСКИЕ
КОМ: ФУНКЦИИ, ЦИКЛ «ПОКА».

АЛГ ПЛОЩАДЬ (ВЕЩ A,B,ПЛ; НАТ H)
          АРГ A,B,H
          РЕЗ ПЛ
НАЧ ВЕЩ Y, X; НАТ Й
      ВЫВОДТ(ВВЕДИТЕ ЛЕВУЮ ГРАНИЦУ КРИВОЛИНЕЙНОЙ ТРАПЕЦИИ)
      ВВОДВ(A)
      ВЫВОДТ(ВВЕДИТЕ ПРАВУЮ ГРАНИЦУ КРИВОЛИНЕЙНОЙ ТРАПЕЦИИ)
      ВВОДВ(B)
      ВЫВОДТ(ВВЕДИТЕ ЧИСЛО РАЗБИЕНИЙ)
      ВВОДЦ(H)
      Y:=(B—A)/H; ПЛ:=0; X:=A; Й:=1
      ПОКА Й<=H
            НЦ
              ПЛ:=ПЛ+SIN(X)*Y
              X:=X+Y
              Й:=Й+1
            КЦ
      ВЫВОДТ(ПЛОЩАДЬ КРИВОЛИНЕЙНОЙ ТРАПЕЦИИ РАВНА)
      ВЫВОДВ(ПЛ)
КОН ВЫЧИСЛЕНИЕ ПЛОЩАДИ
MAIN( )←
DOUBLE A,B,ПЛ;
UNSIGNED H;

DOUBLE У,X;
UNSIGNED Й;
PRINTE(«ВВЕДИТЕ ЛЕВУЮ ГРАНИЦУ КРИВОЛИНЕЙНОЙ ТРАПЕЦИИ\N»);
SCANF(«%LF»,&A);
PRINTF(«ВВЕДИТЕ ПРАВУЮ ГРАНИЦУ КРИВОЛИНЕЙНОЙ ТРАПЕЦИИ\N»);
SCANF(«%LF»,&B);
PRINTF(«ВВЕДИТЕ ЧИСЛО РАЗБИЕНИЙ\N»);
SCANF(«%D»,&H);
```

```
Y=(B—A)/H;
ПЛ=0;
X=A;
Й=1;
WHILE(Й<=H)
{

ПЛ=ПЛ+SIN(X)*У;
X=X+Y;
Й=Й+1;
}

PRINTF («ПЛОЩАДЬ КРИВОЛИНЕЙНОЙ ТРАПЕЦИИ РАВНА \N»);
PRINTP («%G/N»,ПЛ);
}


КОМ:                                          ПРИМЕР 2
КОМ: ВСПОМОГАТЕЛЬНЫЙ АЛГОРИТМ, ВЫЧИСЛЯЮЩИЙ
КОМ: ЗНАЧЕНИЕ ФУНКЦИИ — НАИБОЛЬШИЙ ЭЛЕМЕНТ
КОМ: ЛЮБОЙ ПРЯМОУГОЛЬНОЙ ТАБЛИЦЫ.
КОМ: ОБРАЩЕНИЕ В ОСНОВНОЙ ПРОГРАММЕ:
КОМ: МАКСЭПРТ (А, Б, Ц), ГДЕ А И Б —
КОМ: ЦЕЛЫЕ ЧИСЛА — КОЛИЧЕСТВО СТРОК И СТОЛБЦОВ
КОМ: СТОЛБЦОВ ТАБЛИЦЫ, Ц — ИМЯ ТАБЛИЦЫ.
КОМ: СТРУКТУРЫ ЯЗЫКА — ПРЯМОУГОЛЬНАЯ
КОМ: ТАБЛИЦА, ВЛОЖЕННЫЕ ЦИКЛЫ ТИПА «ДЛЯ».

АЛГ ВЕЩ МАКСЭПРТ(НАТ А, Б; ВЕЩ ТАБ Ц[1:А, 1: Б ])
НАЧ НАТ Й , Ю
       ЗНАЧ := Ц[1,1]
       ДЛЯ Й ОТ 1 ДО А
         НЦ
           ДЛЯ Ю ОТ 1 ДО Б
             НЦ ЕСЛИ ЗНАЧ<Ц[Й , Ю]
                  ТО ЗНАЧ := Ц[Й , Ю]
                ВСЕ
             КЦ
         КЦ
КОН

EXTERN DOUBLE МАКСЭПРТ();
DOUBLE МАКСЭПРТ(А , Б , Ц)
UNSIGNED А , Б ;
DOUBLE Ц[25][81];

{
DOUBLE ЗНАЧ;
UNSIGNED Й , Ю;
FOR(Й=1 ; Й<=А ; ++(Й))
{
FOR(Ю=1 ; Ю<=Б ; ++(Ю))
{
IF(ЗНАЧ<Ц[ Й ][ Ю ])
{
ЗНАЧ=Ц[ Й ][ Ю ];
}

}

}
RETURN(ЗНАЧ);
}
```

В. БРЯБРИН, И. ГОРЯЧАЯ

«Эрудит»

Подача учебного материала в увлекательной игровой форме значительно повышает эффективность его восприятия. Это учтено в инструментальной системе «Эрудит», созданной в помощь учителю иностранных языков.

Учебный материал для упражнений представляется в системе параллельными текстами на двух языках — Я1 и Я2, для которых допустимы следующие типы подстановок:

Я1	Я2
ИЯ	РЯ
РЯ	ИЯ
ИЯ	ИЯ

Здесь символами РЯ и ИЯ обозначены «родной» и «иностранный» языки. В их качестве могут фигурировать различные национальные языки (русский, английский, испанский, немецкий и др.), а также сочетания типа «язык математических выражений (Я1) и значения математических выражений (Я2)», «даты (Я1) и исторические события (Я2)». Иначе говоря, систему «Эрудит» можно использовать для обучения почти любому предмету.

Элементы (слова, фразы, формулы), представленные на языке Я1, составляют опорную композицию, элементы на языке Я2 — игровую композицию. Для каждого элемента опорной композиции ученику предстоит подобрать определенный элемент из игровой композиции. Иными словами, задача ученика — установить правильные соответствия между элементами языков Я1 и Я2. Тип соответствий и способ предъявления упражнения ученику, а также комментарий к упражнению задаются учителем с помощью специальных средств.

Элементы игровой композиции могут быть представлены в системе как явно (т. е. изображаться на экране дисплея), так и неявно. В первом случае ученик должен с помощью специальных маркеров, перемещаемых по экрану клавишами управления, выбрать соответствующие элементы из Я1 и Я2. Во втором на экране видны лишь элементы опорной композиции, а соответствующие им из игровой композиции нужно воспроизвести (например, ввести перевод иностранного текста с клавиатуры).

Сеанс обучения проходит в режиме игры, на протяжении которой ученику за правильность и скорость выполнения упражнений начисляются очки с учетом степени его подготовленности и сложности упражнения. Перед началом сеанса система запрашивает имя ученика и в дальнейшем общается с ним по имени. Итог сеанса-игры сопоставляется с достижениями десятки лучших и при попадании в эту десятку заносится туда вместе с именем обучаемого. Игровой режим позволяет поддерживать азарт и интерес обучаемого к повторению сеансов. Задача учителя — подобрать упражнения так, чтобы обеспечить прогресс в изучении предмета.

У системы есть два режима работы: «учитель» и «ученик»; нужный выбирается в начале работы с помощью специального графического меню и особого маркера в виде человечка — Ведущего. Этот веселый человечек выполняет роль посредника между пользователем и ЭВМ и появляется всякий раз, когда запрашивается помощь системы.

В режиме «учитель» преподаватель указывает название интересующей его учебной темы. На экран дисплея при этом выводится список всех имеющихся к данному моменту тем. Если вводится новое название, то оно заносится в каталог и заводится соответствующая этому названию новая тема. Далее учителю предоставляется возможность работы с двумя типами объектов: с упражнениями или уроками.

При выборе одного из них на экране дисплея появляется список упражнений (или уроков) с комментариями к ним. В качестве комментария к упражнению выступает формулировка задания в данном упражнении. Комментарием к уроку может быть, например, описание цели урока. Учитель имеет возможность выбрать из предъявленного ему списка

конкретный объект (упражнение или урок) для дальнейшей работы с ним (модификации, уничтожения и др.) либо создать новый объект. Эти действия осуществляются с помощью Ведущего.

При создании нового упражнения учитель должен задать в нем тип соответствия — один из двух предлагаемых ниже.

Простое соответствие (1:N): опорная композиция содержит лишь один элемент языка Я1, игровая композиция содержит N элементов языка Я2. В этом случае элементу опорной композиции соответствует единственный элемент из игровой композиции. Учителю следует помнить, что при введении игровой композиции этот единственный соответствующий правильному ответу элемент должен быть первым.

Множественное соответствие (N:N): в опорной и игровой композициях содержится одинаковое число элементов. Соответствующие друг другу элементы языков Я1 и Я2 должны располагаться друг против друга, т. е. идентифицироваться одним и тем же порядковым номером.

В описываемой реализации N может принимать значения от 1 до 8.

Вот несколько примеров упражнений первого типа (1:N):

подобрать для слова, словосочетания или предложения на языке Я1 правильный перевод из предложенных вариантов ответа на языке Я2;

найти синоним (или антоним) к слову на языке Я1 из списка предложенных слов на языке Я2;

отгадать загадку (для описанного на языке Я1 явления, предмета или процесса выбрать правильное название из нескольких вариантов на языке Я2).

Пример упражнения второго типа (N:N): имеется набор фраз — правильный, логически связный текст на языке Я1 (диалог, стихотворение и т. п.) и соответствующее ему множество неупорядоченных фраз на языке Я2; требуется расставить фразы языка Я2 в правильном порядке.

Ввод текстов опорной и игровой композиций учитель осуществляет с помощью обычного текстового редактора в два окна, симметрично расположенные на экране дисплея. Окна, в зависимости от указанного учителем типа соответствия в создаваемом упражнении, предварительно могут быть разделены на N равных частей.

Закончив работу с окнами, учитель заполняет комментарий, который будет предъявляться ученику перед выполнением упражнения (пример: «отгадать загадку, набрав на клавиатуре правильный ответ»).

Наконец учитель задает сложность упражнения (от 1 до 9) и выбирает способ показа (метод предъявления упражнения ученику на экране дисплея).

Окончив формирование нового упражнения, педагог сообщает (с помощью специальной клавиши) об этом системе. Она присваивает созданному упражнению номер и возвращает учителя на предыдущий уровень, где с помощью Ведущего он может либо выбрать новый объект для работы, либо подняться на уровень выше для выбора типа объекта (упражнение или урок).

Если объектом работы является созданное ранее упражнение, то учитель может:

изменить содержание этого упражнения, его сложность и способ показа;

удалить упражнение из данной темы;

просмотреть содержимое других упражнений данной темы;

«пометить» упражнение, с тем чтобы не забыть включить его в состав нового урока.

При формировании нового урока педагог просматривает список существующих в данной теме упражнений с комментариями к ним и выбирает те, которые считает нужными. Здесь облегчат работу помеченные упражнения, поскольку они выделяются специальным образом (после создания нового урока они перестают быть помеченными). В описываемой реализации урок может состоять не более чем из десяти упражнений. Преподавателю предоставляются также средства для введения комментария к уроку, изменения содержимого урока, исключения из его состава некоторых упражнений и включения других.

При создании нового урока система автоматически присваивает ему номер.

Упражнения, включенные в урок, будут предъявляться ученику в заданной учителем последовательности.

В режиме «ученик» система запрашивает название темы. Получив ответ, она выводит на экран дисплея список уроков, и обучающийся выбирает один из них, после чего можно сразу приступить к выполнению урока или предварительно просмотреть десятку лучших.

Перед началом сеанса обучения система запрашивает имя обучающегося и степень его подготовленности (от 1 до 9). Во время урока ученик может:

регулировать (запрещать или разрешать) звуковое сопровождение упражнений;

прервать выполнение упражнения, с тем чтобы перейти к следующему (система при этом подсчитает количество очков, набранных учеником);

прервать урок и выбрать другой (в этом случае прерванный урок не засчитывается для ученика).

Время на выполнение каждого упражнения ограничивается и фиксируется с помощью высвечивающихся на экране дисплея часов. Для каждого метода предъявления упражнений есть учитывающий его специфику алгоритм вычисления времени, отводимого на выполнение упражнения. Общим для всех этих алгоритмов является учет типа соответствия в упражнениях, значения N (числа элементов в игровой композиции) и сложности упражнения. Очки за выполненное упражнение начисляются по формуле

$$S = S_{пр} - S_{шт},$$

где

$$S_{пр} = PVLX(C1 \cdot T1/T2),$$

$$S_{шт} = \frac{KL \cdot T2 \cdot C2}{X \cdot T1}.$$

В свою очередь, P=A—B, где A — число правильно найденных учеником соответствий, B — общее число соответствий между элементами языков Я1 и Я2; K — признак полностью неверного ответа, равен 1 при P=0, в остальных случаях K=1; V=1/(M+1), где M — число неправильных попыток ответа при выполнении упражнения; L — степень подготовленности ученика (от 1 до 9), указанная им самим перед началом сеанса; X — сложность упражнения (от 1 до 9); T1 — время, выделенное системой на выполнение упражнения; T2 — время, затраченное учеником на выполнение упражнения; C1 и C2 — коэффициенты корректировки, подбираемые экспериментально.

В конце урока система предлагает учащемуся выполнить те упражнения, которые он пропустил, после чего вычисляет оценку урока, суммируя оценки за отдельные упражнения. Ученик может попытаться повысить ее, проходя урок еще и еще, чтобы, например, попасть в десятку лучших.

Методы предъявления упражнений дают еще одну возможность варьировать сложность заданий. Системой определен набор способов показа, позволяющих одно и то же упражнение предъявлять ученику в различных формах, что способствует более прочному усвоению учебной информации.

Содержимое опорной и игровой композиций может изображаться на экране дисплея, например, в одном из следующих видов:

как стоящие рядом столбцы элементов;

как передвигающиеся (возможно, с различными скоростями) на экране объекты.

В первом из них требуется выбрать ответ из заданного набора возможных значений. Это схема облегченного типа, в ней содержится элемент подсказки. Здесь используются изображения двух человечков, выражающих свое мнение по поводу ответов ученика различными жестами. Каждый из человечков отвечает за свою композицию: левый — за опорную, элементы которой выводятся на экран в виде столбца в порядке, заданном учителем; правый — за игровую, элементы которой располагаются в случайном порядке (с помощью датчика случайных величин) в столбце напротив. Управляющими клавишами ученик должен установить человечков напротив соответствующих элементов и зафиксировать свой ответ нажатием клавиши «Исполнение». Если ответ верен, то соответ-

ствующие элементы из игровой и опорной композиций помещаются друг против друга и выделяются цветом.

Другой способ предъявления упражнений дает ученику возможность ответить на заданный вопрос без подсказки. Игровая форма, в которую облечен этот способ, использует плавающие по экрану объекты (рыбки), за которыми закреплены элементы опорной композиции. В течение обусловленного промежутка времени ученик должен ввести ответ с клавиатуры, при этом он может успеть опробовать несколько вариантов. По окончании упражнения полезно показать ученику правильные ответы, которые он не смог найти.

В обоих описанных способах предусмотрена система поощрений за правильное выполнение упражнения, активно использующая элементы мультипликации, звуковое сопровождение и другие приемы поддержания интереса.

Система реализована на языке Турбо-Паскаль 3.0 в операционной системе MS—DOS 3.x (типа ДОС-16) для персональных компьютеров IBM PC/XT/AT и ЕС 1841 с оперативной памятью не менее 128К байт и цветным графическим дисплеем.

Для работы с нестандартными алфавитами рекомендуется использовать стандартный цветной графический адаптер со встроенным в ПЗУ специальным знакогенератором или усовершенствованный графический адаптер с загружаемыми знакогенераторами.

А. МАСЛОВ, О. ТАИРОВ, В. ТРУШ

Физиолого-гигиенические аспекты использования персональных ЭВМ в учебном процессе

Длительная работа с различными компьютерными устройствами повлекла за собой изменение состояния здоровья операторов ЭВМ. Наиболее существенное отрицательное влияние оказывают несовершенные видеотерминальные устройства и эргономическая конструкция рабочего места, которые вызывают нарушения зрительной функции и жалобы на мышечный дискомфорт. Необходимо отметить, что их влияние на детский и подростковый организм изучено крайне недостаточно, причем наиболее опасное воздействие связано, по всей очевидности, с видеотерминалами на электроннолучевых трубках. Полно не исследовано также влияние постоянного шума, микроклиматических условий (повышенная температура, сухость и ионизация воздуха) и воздействие потока информации, поступающего в вынужденном режиме.

В настоящее время актуальным вопросом использования ЭВМ в средней школе является соблюдение санитарно-гигиенических нормативов, определяющих прежде всего качество телевизионных дисплеев. Важнейшими параметрами дисплеев являются: уровни электромагнитного излучения в инфракрасном, СВЧ, рентгеновском и ультрафиолетовом диапазонах; яркость и контраст изображения; размер знаков; частота и уровень пульсации яркости. Многочисленными исследованиями установлено, что нарушение этих нормативов приводит к неблагоприятным последствиям для здоровья почти у 70 % взрослых операторов. Снижение качества изображения на экранах дисплеев приводит к ухудшению зрения и к ряду неврологических симптомов (головным болям, повышенному утомлению и др.).

У детей и подростков, центральная нервная система (ЦНС) и зрительные органы которых находятся в стадии созревания, степень и вероятность развития неблагоприятных последствий при работе с видеотерминалами может быть гораздо выше, чем у взрослых. Следует особо отметить, что незрелая ЦНС у детей находится в состоянии повышенной судорожной готовности по сравнению со взрослыми, поэтому для них высока опасность развития фотоэпилепсии, связанной с мерцанием экрана и частотой смены кадров (обычно 50 Гц).

Работа с дисплеями низкого оптического качества может приводить к быстрому ухудшению остроты зрения, причем у детей с генетической предрасположенностью к

миопии оно может проявиться довольно рано. Совершенно очевидно, что снижение остроты зрения влечет за собой многочисленные последствия — снижение успеваемости, уровня социальной адаптации, профпригодности и т. д.

Вопросы физиолого-гигиенической оценки влияния видеотерминалов и снижения неблагоприятных последствий работы с ЭВМ в настоящее время находятся в стадии интенсивных исследований. Всесторонне изучив влияние характеристик видеотерминалов на состояние различных физиологических систем, рабочая группа экспертов Всемирной организации здравоохранения (ВОЗ) в 1985 г. рекомендовала новые, более строгие гигиенические нормативы, которые во многих странах были приняты.

По инициативе Управления информатики и электронно-вычислительной техники Минпроса СССР осенью 1986 г. создана межведомственная экспертная группа по выработке нормативов на видеотерминалы школьных ЭВМ. С учетом рекомендаций ВОЗ, действующих в СССР санитарно-гигиенических норм и требований, ГОСТов, а также результатов исследований, проводимых в институтах АПН СССР и АМН СССР, экспертной рабочей группой были разработаны и в декабре 1986 г. утверждены Минздравом СССР и Минпросом СССР «Временные санитарно-гигиенические нормы и правила к устройству видеотерминалов школьных ЭВМ. Часть I. Требования к дисплеям».

Наличие этого документа позволило уже в конце 1986 — начале 1987 г. государственным комиссиям, принимавшим ОКР по темам «КУВТ «Корвет», «КУВТ УК НЦ», «Индикатор школьной ПЭВМ на базе стандартного телевизионного приемника «Юность», обосновать отказ от рекомендации к использованию представленные по результатам этих ОКР мониторы в составе школьных ПЭВМ.

Ранее вопросы о закупке конкретных типов зарубежных ЭВМ, а также о выпуске отечественных ПЭВМ рассматривались без участия гигиенистов и в отсутствие соответствующих нормативных документов. Лишь с осени 1986 г. экспертная рабочая группа организовала систематическое гигиеническое обследование видеотерминалов школьных ЭВМ. В НИИ дефектологии АПН СССР и НИИ общей и коммунальной гигиены АМН СССР было изучено более 50 типов зарубежных и 7 типов отечественных ПЭВМ. Обнаружено, что подавляющее большинство из них не соответствуют современным гигиеническим нормам. Большинство отечественных видеотерминалов были разработаны на основе использования существующих бытовых телевизоров («Электроника», «Юность», «Шилялис», «Сапфир»), которые по своим техническим характеристикам, особенно по разрешающей способности и минимальному размеру точки, в принципе не могут достичь гигиенических нормативов на видеотерминалы. Как подчеркивается в специальном докладе рабочей группы ВОЗ, из самого хорошего бытового телевизора в лучшем случае можно сделать плохой видеотерминал для ЭВМ.

Сейчас в школах практически нет ЭВМ с видеотерминалами, удовлетворяющими утвержденным гигиеническим нормам. Более того, подавляющее большинство находящихся в школах ЭВМ не соответствуют нормам даже по важнейшим (кстати, технически легко достижимым) оптическим характеристикам: контрасту, яркости, резкости. К таким ЭВМ, в частности, относятся КУВТ-86, КУВТ «Ямаха», ЭВМ «Агат». Использование дисплеев низкого качества может оказывать весьма серьезное влияние на остроту зрения школьников, приводить к быстрому утомлению. Этот вывод, содержащийся в большинстве печатных источников и в материалах ВОЗ, убедительно подтвержден исследованиями, проведенными в институтах АМН СССР и в НИИ физиологии детей и подростков АПН СССР. В этих исследованиях показано, что использование ЭВМ «Агат» в обучении школьников IX—X классов курсу «Основы информатики и вычислительной техники» оказывает специфическое влияние: значительно снижает умственную работоспособность, приводит к неблагоприятным изменениям функционального состояния центральной нервной системы, сердечно-сосудистой системы и зрительного анализатора. Причем жалобы на усталость глаз отмечались у 85 % школьников. Существенно, что объективно регистрируемое ухудшение состояния центральной нервной системы и зрения отмечалось уже через 20—25 мин работы с видеотерминалом.

Эти неблагоприятные последствия можно уменьшить, лишь введя существенные ограничения на время работы учащихся с видеотерминалами, не соответствующими гигиеническим нормативам. Этому и служат «Временные рекомендации по режимам занятий учащихся за видеотерминалами ЭВМ».

Утвержденные Минздравом СССР и Минпросом СССР от 28.01.87 г. режимы ограничивают занятия с использованием видеотерминалов третьего класса (типа КУВТ-86, КУВТ «Ямаха», ЭВМ «Агат») суммарной

(Окончание на с. 91.)

Журнал в журнале
для школьников, студентов,
учащихся СПТУ
и техникумов

Издается при участии
ЦК ВЛКСМ

МОЛОДЕЖНАЯ ИНИЦИАТИВА

2.

В НОМЕРЕ

Что такое «Редактор шрифтов»

Вновь предоставляем слово школьнику. Приглашаем участвовать в работе «Молодежной инициативы» всех юных программистов.

Одной из особенностей ПЭВМ фирмы «Ямаха» является возможность произвольно изменить внешний вид символа на экране дисплея. Соответствующие коды записываются в ячейки видеопамяти. Так можно «замещать» используемые символы любыми другими. В частности, речь может идти о национальных алфавитах или о любых других, в частности таких, с помощью которых можно создавать всевозможные изображения.

С целью предоставления пользователю удобного средства для создания новых символов разработана программа «Редактор шрифтов». Она позволяет изменить внешний вид любого символа или группы символов и сохранить полученный шрифт. Работа ведется в диалоговом режиме. После запуска редактора на экран выводятся все символы шрифта и квадратная матрица 8×8 точек, содержащая изображение символа в увеличенном виде. Перемещая по экрану курсор, можно выбрать любой символ и затем отредактировать его, после чего выбрать для редактирования другой символ, и т. д.

Созданный в результате такой работы шрифт можно сохранить, записав на магнитный диск. Шрифт может быть записан на диск в двух форматах: в виде файла, загружаемого командой bload, и в виде Бейсик-программы, выполнение которой приведет к формированию в ПЭВМ данного шрифта. В первом случае загрузка шрифта может быть осуществлена из любой программы, в которую нужно вставить в соответствующем месте оператор bload, причем с помощью нескольких таких операторов можно менять шрифт внутри программы. Во втором случае выведенная на диск Бейсик-программа может быть пристыкована к разрабатываемой программе, что позволит передавать ее по сети обычным образом.

Символам на экране можно придавать любую форму. Благодаря этому «Редактор шрифтов» может использоваться для воспроизведения таких символов, с помощью которых удобно создавать красивые изображения. Наиболее ощутимый эффект это дает при создании рисунков, содержащих повторяющиеся элементы («кирпичная кладка», «звездное небо», «трава» и т. д.). «Редактор» предоставляет возможность менять цвет изображения.

Предусмотрена вторая страница «экрана», названная в программе «подэкраном». На «подэкране» создается изображение путем расстановки в нужных местах соответствующим образом отредактированных символов. В любой момент можно вернуться обратно к «экрану», выбрать другой символ и, вновь вернувшись на «подэкран», создавать таким образом достаточно сложные изображения. Рисунки также можно сохранить на диске для использования в других программах. Это позволяет создавать различные заставки в разработанных программах. Вывод таких рисунков на экран происходит практически мгновенно. Можно выводить рисунки на печатающее устройство. Программа «Редактор шрифтов» написана на Бейсике с включением ряда фрагментов на языке ассемблера.

Мы можем передать ее всем заинтересованным лицам и организациям. Со всеми вопросами обращаться по адресу: 644099, Омск, наб. Тухачевского, 14. Омский государственный педагогический институт им. А. М. Горького, кафедра информатики и вычислительной техники.

А. Опачанов,
ученик X класса
школы № 37 г. Омска

С компьютером — на «ты»

«Сначала поздоровайся. А то она огорчится и не будет играть», — тихо шепчет за спиной Толи Квашина Виталий Смирнов. И поясняет мне, человеку, на его взгляд, явно несведущему в компьютерах: «Знаете, какая она обидчивая! Мы однажды играли в крестики-нолики, а ей что-то не понравилось. Так на экране появилась надпись: «С *хулиганами не играю*».

Это был необязательный урок, и проходил он в один из последних дней прошлого учебного года. Но сюда, в студенческий вычислительный центр Челябинского политехнического института, V Б школы № 31 пришел охотно. Ребята быстро расселись у дисплеев — и началось... Студент-пятикурсник приборостроительного факультета Андрей Юлатов едва успевал бегать от одного дисплея к другому. Хорошо еще, что эти игровые обучающие программы составлял он сам — легче ориентироваться.

Это был эксперимент — обучение работе на ЭВМ пятиклассников.

«Я считаю, одиннадцать лет — оптимальный возраст для начала общения с компьютером, — говорит начальник студенческого вычислительного центра А. П. Погодин. И продолжает: — Совсем не обязательно учить ребят языку программирования, хотя восприимчивость школьников гораздо выше, чем у студен-

тов, не говоря уж о взрослых людях. На мой взгляд, пятикласснику вполне достаточно дать основные представления: об алгоритме, исполнителе, переменной и т. д. То есть сначала ребенок учится свободно манипулировать этими понятиями, а впоследствии, в процессе работы, легко сможет овладеть любым машинным языком».

Уже три года существует факультет «Юный программист», организованный при ЧПИ по инициативе учебно-методического управления и кафедры прикладной математики. Удостоверения об окончании факультета получили более 150 учащихся школ и СПТУ. Кроме того, 67 ребят из районов области прошли обучение на открытом в 1986 году заочном отделении.

Успешно работает летний лагерь для старшеклассников «Компьютерная эра». За право поехать сюда среди школьников разворачивается настоящее соревнование. К открытию лагеря в этом году политехники готовятся особенно тщательно. Студенческий вычислительный центр и кафедра электронно-вычислительных машин предоставили ребятам двенадцать рабочих мест с графическими цветными дисплеями, библиотеку программ — обучающих и игровых. Преподаватели института подготовили цикл лекций для школьников по новейшим направлениям развития электронно-вычислительной техники. Практику у школьников будут вести студенты-старшекурсники. Несмотря на солидный багаж знаний, им пришлось немало времени потратить на изучение дополнительного материала, который пока отсутствует в учебных планах. Думается, что подобные специализированные лагеря могли бы с успехом организовать на базе своей вычислительной техники другие вузы, а также крупные предприятия.

Большой популярностью пользуется вычислительный центр, созданный политехниками в школе № 31. Пять персональных компьютеров ДВК-1 не только установили в школе, но и дополнительно разработали устройство сопряжения ЭВМ с магнитофоном, чтобы можно было сохранить наработанные программы до следующего урока (у ДВК-1 нет внешней памяти), поставили устройство для распечатки программ, каждый компьютер снабдили цветным графическим дисплеем. А сотрудники кафедры прикладной математики университета решились на такой эксперимент: подключили один комплекс к кафедральной ЭВМ СМ-4. Теперь школьники на расстоянии четырех километров могут использовать возможности более мощной вычислительной машины.

С созданием первого школьного вычислительного центра в ЧПИ не успокоились. Ребята из школы № 31 могут учиться работать с компьютером, а как же другие? Предполагается создать такие центры и в других школах и СПТУ, но необходима активная помощь базовых предприятий.

А пока политехники предложили такой выход: в январе, феврале и июне институтские вычислительные залы пустуют — студенты на сессии и каникулах. Есть возможность предоставить технику в распоряжение учащихся. Проведение занятий преподаватели берут на себя.

Но если городские школьники и учащиеся СПТУ имеют возможность увидеть настоящую ЭВМ хоть краешком глаза, то сельские — лишены и этой малости. Сборы были недолги. Два персональных компьютера бережно укутали в пеноплен, загрузили в маленький тряский автобус и — в путь. За неделю сотрудники студенческого вычислительного центра побывали у девятиклассников семи школ Красноармейского района. Были прочитаны лекции, продемонстрированы возможности ЭВМ. И главное — ребята смогли посидеть за пультом компьютера, нарисовать картинки на экране дисплея, поиграть в различные игры. Учителя высказали пожелание почаще практиковать такие полезные встречи.

Идея передвижного класса очень заманчива. Институты и предприятия города могли бы поделиться ЭВМ ради такого прекрасного начинания. Есть чем вооружить компьютерные десанты. Да и проводятся они не ради благотворительности и не по принуждению.

Реальность компьютерных десантов доказана. Инициатива челябинцев одобрена. Подобный опыт, как нам известно, есть и в других регионах, в Омске например. Но одним отрядом всю огромную область не обеспечишь. Да и некоторые села неблизко расположены от города. А нехватка автобусов ощущается острейшая. Очевид-

но, настала пора выпуска микроавтобусов с 3—5 ЭВМ в салоне. Кстати, такой опыт уже есть у наших зарубежных коллег. Передвижные компьютерные классы могут смягчить дефицит и нехватку вычислительной техники.

Позади уже почти два года, в течение которых девятиклассники всей страны изучают новый предмет — «Основы информатики и вычислительной техники». К нему готовились и в нашей области: учителя и преподаватели СПТУ проходили обучение на базе Челябинского государственного университета, политехнического, педагогического институтов и областного института усовершенствования учителей. В помощь преподавателям нового предмета были разработаны учебное пособие по обучению языку Бейсик, сборник задач по основам информатики, лабораторный практикум, методические рекомендации «Психолого-педагогические аспекты проблемы компьютеризации» и др. Они стали хорошим подспорьем в новом деле. Свидетельство тому — целая папка заявок на эти пособия из других областей страны.

На одной проблеме хотелось бы остановиться. В двенадцатой пятилетке предусматривается организовать массовый выпуск персональных компьютеров. В нашей стране начинается новый этап компьютеризации — внедрение ЭВМ в повседневный быт. Технология использования персональных компьютеров непрерывно упрощается, однако обращение с ними под силу лишь подготовленному человеку. Начиная их массовый выпуск, необходимо обеспечить и соответствующую подготовку массового заинтересованного покупателя-пользователя. На наш взгляд, роль высшей школы здесь ведущая. В первую очередь студенты и преподаватели должны уметь пользоваться персональным компьютером. А для этого нужно насытить ими вузы. Возможность формирования сетей из них позволит во многом снять и остроту проблемы материальной базы компьютеризации обучения.

А. НАЙН

Это вы можете

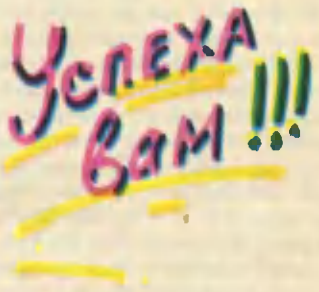
Кто не слыхал о чуде технике — ЭВМ, понимающей речь! Однако нужно ли смотреть на нее почтительно, «снизу вверх»? Как сказать...

В Ленинграде создан электронный анализатор речи «Барс» — микро-ЭВМ, способная распознавать 200 слов с надежностью 97 %. На анализ одного слова требуется 0,1 с. Анализатор можно подключить параллельно клавиатуре другого компьютера — появится возможность вводить данные и команды не только пальцами, но и голосом, производительность труда возрастет.

Отличный результат отличной работы! Больше всего восхищает то, что все программы речевого анализа хранятся в ПЗУ емкостью всего 4К байт, а образцы голоса оператора, на общение с которым настроен «Барс»,— в ОЗУ такой же емкости. Ведь даже у БК-0010 память больше!

А раз так, то не попробовать ли и вам создать такую программу? Ну, не на 200, а на 20 слов, и пусть тратит на анализ секунду или две... Как ввести в БК-0010 запись речи, рассказано в журнале «Наука и жизнь» (1987, № 4).

Для начала можно попробовать вывести речь на экран в виде графика и проанализировать вид основных звуков... А впрочем, решайте сами, что делать!



Что же может ЭВМ?

Собственно, ЭВМ способна только хранить, пересылать, сравнивать, складывать числа и менять их знак (ну, и еще кое-что — но это несущественно). Все чудеса, ей приписываемые,— заслуга не ее, а программистов. Так что правильнее было бы спросить: «Что может программист?»

Программист - профессионал может очень многое, но сейчас речь пойдет о школьниках. Итак, это вы можете!

Программа рисует карикатуру. Очень забавную, весьма узнаваемую и немного ехидную. Обычно рисующие программы весьма сложны, а столь «мастеровитые» — тем более, но эту вы сможете написать сами. Эффективность ее работы обусловлена алгоритмом — очень простым, как все гениальное. Его автор — С. Бреннан.

В памяти ЭВМ хранится описание «среднестатистического» лица — набор координат 186 характерных точек, соединив которые в определенной последовательности прямыми линиями мы получим рисунок этого лица. Пользователь должен ввести 186 элементов второго массива — координаты аналогичных точек (5 — для радужной оболочки глаза, 11 — для челюсти и т. д.) чьего-то лица, которые нетрудно рассчитать, начертив координатную сетку на фотографии этого человека. После этого ЭВМ, пользуясь своими небольшими умениями, должна вычислить разницу между координатами соответствующих точек (первой из первого массива и первой из второго и т. д.) и увеличить ее на X процентов. В зависимости от величины X получается слегка шаржированное, карикатурное или «сюрреалистическое» изображение оригинала,— естественно, после того, как вновь полученные точки будут соединены отрезками прямых.

Более подробное описание алгоритма приведено в журнале «В мире науки» (1986, № 12).

У нас есть компьютер

В конце 1986 г. в средней школе № 14 г. Черновцы в кабинете информатики появились БК-0010. Сначала десятиклассники осваивали клавиатуру, простейшие алгоритмы, потом перешли к изучению языка Фокал и составлению программ.

В школе работает кружок информатики, которым руководит преподаватель физики Григорий Александрович Гринберг. Это человек, увлеченный своим делом, добрый наставник. Он сумел привить ребятам любовь к работе на ЭВМ.

Заниматься в кружке может каждый. Шестиклассник А. Рабинович составил игровые программы «Пятнашки», «Морской бой», «Воздушный бой», «Футбол», а также цветовые заставки к играм.

— Покажи мне одну из программ,— прошу я его.

Мальчик загрузил в ЭВМ «Пятнашки». На дисплее появилась таблица с разбросанными в беспорядке цифрами. Их необходимо составить в нужном порядке: от 1 до 15. При помощи клавиш со стрелками «вверх», «вниз», «вправо», «влево» цифры «перегоняют» в нужную клетку. Если в процессе игры будет допущена ошибка, ЭВМ перестанет реагировать на команды и на дисплее появится: «Будь внимателен». Исправив ошибку, можно играть дальше.

Сергей Скалицкий составляет учебные программы решения квадратных и линейных уравнений. В будущем он мечтает стать инженером-программистом.

Все свободное время проводит в кружке студент-физик Черновицкого университета Константин Лищинский. Интересна его учебно-игровая программа «Логика» для десятиклассников. Она идет 40—45 мин, в течение которых машина обучает, задает вопросы, оценивает ответы. После загрузки компьютера программой на дисплее высвечиваются логические схемы и таблицы. Здесь используется двоичная система счисления. Игра делится на три уровня-этапа. Если на втором допущено определенное количество ошибок, машина возвращает игру на первый уровень. Цель программы — научить учащихся разбираться в логических элементах И, ИЛИ, Не, объяснить устройство ЭВМ.

Лищинским разработана программа для черновицкого предприятия ковровых изделий «Восход». На составление рисунка ковра она затрачивает не более 2 с, причем возможны 10 тыс. разных вариантов.

Вера Байкова, учащаяся X Б класса СШ № 14, разработала тесты по профориентации. При вводе программы в компьютер на дисплее появляются два вопроса: один — слева, другой — справа — об избрании той или иной профессии. Если мы выбрали вопрос слева, нажимаем кнопку со стрелкой «влево». На дисплее высвечивается информация о качествах, необходимых для данной профессии. С помощью такой программы можно оценить правильность выбора.

Программы кружка используются в СПТУ-13, в индустриальном и политехническом техникумах города.

— С введением в школьный курс нового предмета ОИВТ мы столкнулись с определенными трудностями,— говорит Г. А. Гринберг.— У нас есть совершенно новая техника, однако отсутствуют общие подходы к организации кружковой работы. Учителя не могут дать школьникам достаточных знаний в области программирования. Необходимо централизованное тиражирование программ, нужна специальная литература.

Новые компьютеры БК-0010 рассчитаны только на игровые и учебные программы, а ограничиваться этим просто невозможно. Думается, что работа стала бы интересней, если бы кружок смог обмениваться опытом, программами с другими кружками нашей страны, решать те или иные вопросы, совместно изобретать что-то.

Хотелось бы, чтобы в журнале «Информатика и образование» появилась новая рубрика «Запишите мой адрес», через которую был бы возможен обмен мнениями, программами. Нужно чаще проводить различные региональные, всесоюзные конкурсы, школы, летние лагеря юных программистов, где они смогут общаться, узнать много нового и интересного — ведь с ними будут заниматься профессионалы.

— Григорий Александрович, каковы задачи и перспективы кружка?

— Главная задача — организация творческой работы на любом уровне: от игры до составления сложных программ. Что касается перспектив, думаю разделить кружок на две группы. Первая — общеобразовательная, дающая ученикам начальные знания в области информатики и вычислительной техники; вторая — для тех, кто выбрал программирование в качестве профессии.

Курбатова Елена,
учащаяся X А класса
СШ №14
г. Черновцы, юнкор газеты
«Молодой буковинец»

Автоматизация программирования графики

Вы захотели написать программу, которая выведет на экран дисплея рисунок, чертеж, схему.

Подождите! Не торопитесь составлять алгоритм, подсчитывать число символов и позиций; чтобы все располагалось как надо, не торопитесь писать программу и отлаживать ее.

Если вы воспользуетесь графическим редактором «Автограф», разработанным автором настоящей статьи, то, затратив ровно столько времени, сколько требуется, чтобы с помощью нескольких клавиш и несложной инструкции нарисовать на экране требуемое изображение, вы автоматически получите желаемую программу.

Одной из самых существенных характеристик ПЭВМ является, пожалуй, наличие широких графических возможностей. Обычно они зависят от набора операторов графики языка программирования, наличия светового пера и специальных графических режимов. Однако использование графических операторов требует знания базового языка программирования, а световое перо зачастую отсутствует в комплектах ПЭВМ.

«Автограф» должен был решить следующие задачи:

научить ПЭВМ «Ямаха», не имеющую светового пера, выполнять любые графические построения на экране дисплея, не требуя от пользователя знания какого-либо языка программирования (а именно таким нужно предполагать пользователя);

обеспечить возможность запоминания рисунка и передачу его по сети.

Отказ от традиционных способов решения второй задачи привел к идее запоминания и передачи рисунка по сети в виде программы на Бейсике-MSX. Тут же выяснилось, что эта идея предоставила вначале непредусмотренную, но, пожалуй, самую важную возможность — автоматизацию программирования графики.

Таким образом, «Автограф» имеет, по меньшей мере, три особенности:

не требует от пользователя знания языков программирования;

предоставляет более широкий спектр графических возможностей, чем световое перо;

для всякого рисунка, выполненного на дисплее, машина автоматически составляет программу на Бейсике, которая способна воспроизвести (разумеется, на ПЭВМ, функционирующей в системе MSX) этот рисунок.

Избранный подход позволяет автоматизировать процесс программирования графическо-текстовых кусков обучающих и других программ, может быть применен и в других ситуациях для автоматизации процесса программирования; его можно использовать для расширения Бейсика с целью включения в Бейсик-программу параметризованных участков (включая метки), способных под воздействием управляющей команды автоматически превращаться в реальные строки Бейсик-программы, а также для автоматических модификаций Бейсик-программы.

«Автограф» реализован в двух вариантах. Первый рассчитан на нецветной дисплей и позволяет дорисовывать картинку через некоторое время. Он может использоваться на ученических (не имеющих дисковода) машинах КУВТ «Ямаха»; незначительным недостатком его является периодическое прерывание процесса рисования на несколько секунд, чего можно избежать во втором варианте благодаря использованию дисковода. Конечный результат работы обоих вариантов одинаков.

На базе «Автографа» можно построить любую про-

грамму графики Бейсика-MSX, игровые и графические программы для детей как школьного, так и дошкольного возраста, обучающую программу по графике Бейсика, позволяющую идти от графических конструкций к соответствующим конструкциям языка.

«Автограф» можно адаптировать к любой машине, в которой предусмотрены арифметические действия, анализ строковых величин и графические операторы (проведение отрезков, построение окружностей и прямоугольников, управление цветом).

Кроме уже упомянутых «Автограф» имеет следующие возможности, отсутствующие у светового пера:

восстановление перпендикуляра к данной прямой;

проведение прямой, параллельной данной прямой;

9 скоростей ведения курсора;

9 толщин линий;

рисование окружности.

Именно «Автографом» была создана программа, выдающая пользователю, забывшему инструкцию, рисунок-«подсказку» (см. фото). А расшифровывается он так.

Чтобы задать скорость курсора, нужно нажать F1 и цифру, соответствующую номеру скорости (количеству точек, через которые курсор пройдет в единицу времени). По умолчанию скорость равна 1, самая большая скорость — 9.

Для рисования прямоугольника следует зафиксировать концы одной из его диагоналей, нажимая F2. Чтобы поставить точку, нужно нажать «RETURN». Точка появляется над левым верхним краем курсора.

Для проведения отрезка, параллельного данному, нужно отметить концы данного отрезка, а потом начало будущего параллельного отрезка клавишей F3. Длина нового отрезка будет равна длине исходного.

Для проведения перпендикуляра из данной точки отрезка отметить концы отрезка и потом точку, от которой будет восстанавливаться перпендикуляр, клавишей F4. Длина перпендикуляра будет равна длине отрезка.

Для распечатки рисунка на принтере нужно нажать F5. Помните, что при этом рисунок должен быть белым, а фон, на котором рисуем,— голубым!

Для очистки экрана и перехода к программной фик-

сации изображения нужно нажать F6 (SHIFT+F1).

Для изменения цвета изображения нужно нажать F7 (SNIFT+F2) и набрать номер желаемого цвета. По умолчанию задается номер 15 (белый).

Для изменения толщины линий нужно нажать F8 (SHIFT+F3) и задать толщину. По умолчанию она равна 1, самая большая — 9.

Для рисования окружности следует нажать F9 (SHIFT+ +F4) и задать радиус (для рисования отрезка нужно нажать ESC в его концах).

Для закрашивания определенной области установленным цветом нужно курсор поместить внутрь этой области и нажать F10 (SHIFT+ +F5). Цвет границ должен совпадать с текущим цветом закраски.

Курсор двигается с помощью стрелок вверх, вниз, вправо, влево. Все символы, кроме служебных (приведенных выше), печатаются на экране при наборе с клавиатуры. Для стирания последнего введенного оператора необходимо нажать BS.

Желающие получить материалы по «Автографу» могут обращаться по адресу: Одесса, ул. Свердлова, 83. Одесское облоно. Стеценко И. А. Тел. 28-37-73.

М. АЛЬТ
Одесса

Советы физиолога

1. Если вы носите очки для работы на близком расстоянии, необходимо работать на ЭВМ в очках.

2. Если подряд идут два занятия по информатике, между ними необходимо проводить комплекс физических упражнений для снятия общего утомления. Этот комплекс надо менять не реже одного раза в месяц.

(Продолжение со с. 80.)

еженедельной длительностью для учащихся IX—IX классов — 20 мин, VII—VIIIклассов — 15 мин, V—VI классов — 10 мин.

Некоторые отечественные ПЭВМ, используемые в школах, снабжены профессиональными дисплеями (например, ДВК-2М). По большинству параметров (хотя и не всем) они соответствуют существующим нормативам. Устанавливаемая нормативами еженедельная суммарная длительность работы за такими дисплеями существенно выше — от 180 мин для учащихся X класса до 60 мин для учащихся V—VI классов, при ограничении непрерывной длительности работы с ЭВМ 15—25 мин.

Практически вся электронно-вычислительная техника в школах в настоящее время представлена видеотерминалами третьего класса, время работы с которыми резко ограничено. Естественно, это сужает возможность широкого использования вычислительной техники в учебном процессе.

Следует надеяться, что утверждение научно обоснованных санитарно-гигиенических норм и требований к видеотерминалам ЭВМ приведет к выпуску гигиенически безвредных школьных ЭВМ.

Постановлением Государственного комитета СССР по вычислительной технике и информатике «Об обеспечении серийного выпуска мониторов для комплектации школьных ПЭВМ с учетом «Временных санитарно-гигиенических норм и правил к устройству видеотерминалов школьных ЭВМ» № 8 от 19 марта 1987 г. предусмотрены необходимые меры по обеспечению школьных ЭВМ видеотерминалами, наиболее полно отвечающими санитарно-гигиеническим требованиям.

От редакции.

В следующем номере мы предполагаем опубликовать комплекс физических упражнений для снятия усталости у школьников.

ЧТО МОЖЕТ ЭВМ?

Лазер плюс процессор

Лазер и микропроцессор — основные части оригинального измерителя скорости горячего проката. С его помощью быстро и точно, а главное, не касаясь раскаленных заготовок, можно определить их длину, скорость передвижения по прокатному стану. Такой измеритель создан учеными Сибирского отделения АН СССР. Устройство было апробировано на редукционном стане в трубоэлектросварочном цехе Новосибирского металлургического завода имени А. Н. Кузьмина. Как показали испытания, прибор способен с высокой точностью измерять скорости в диапазоне от 0,5 до 10 м в секунду. При этом оптический датчик устанавливается на расстоянии до полутора метров от технологической линии.

От оптического датчика, выполненного по компенсационной схеме лазерного доплеровского анемометра, данные о перемещении изделий поступают в микропроцессор, который, обрабатывая их, передает необходимую информацию в систему управления.

Лазерный измеритель можно применять для определения скорости не только горячего проката, но и изделий с поверхностью, рассеивающей свет, на металлургических, машиностроительных, стекольных заводах.

Помощник геологов

Сейчас вычислительная техника широко применяется для обработки данных геофизических исследований, связанных с поиском месторождений полезных ископаемых. Обычно информацию, полученную в ходе полевых работ, направляют в вычислительный центр, где специалисты, анализируя результаты вычислений, могут оценить перспективы геофизических исследований в районе. А нельзя ли анализировать полученную информацию на месте, в полевых условиях? Оказывается, можно. В этом геологам поможет полевой вычислительный комплекс ПВК-Э, созданный специалистами научно-производственного объединения «Нефтегеофизика» совместно с учеными Сибирского отделения Академии наук СССР.

В состав комплекса входит ЭВМ «Электроника 60М, с памятью ОЗУ емкостью 512К байт, алфавитно-цифровым и графическим дисплеями, графопостроителем, накопителем на магнитной ленте типа СМ. В качестве графического дисплея можно использовать цветной телевизор. Датчики регистрирующих станций «Цикл-2» или «Импульс» посылают в ЭВМ геофизическую информацию. Компьютер сообщает геологам о подземных зонах повышенной электропроводимости, параметрах изучаемого разреза почвы. Теперь специалисты могут делать экспресс-анализ результатов гравитационной и магнитной разведки непосредственно в полевых условиях.

Полевой вычислительный комплекс монтируется в автофургоне и легко перевозится с места на место.

В Госпрофобре СССР

Временное положение

о порядке разработки, экспертизы, тиражирования, поставки и использования педагогических программных средств вычислительной техники в системе профтехобразования

Общие положения

Настоящее Временное положение подготовлено на основе «Положения о порядке разработки, производства, поставки и использования программных средств вычислительной техники, а также автоматизированных систем и систем обработки информации», утвержденного постановлением ГКНТ, Госснаба СССР, Госстандарта, Госкомцен СССР, Госкомтруда СССР, Госстроя СССР, ЦСУ СССР, Минфина СССР, Минрадиопрома, Минприбора, Минэлектронпрома и Минпромсвязи в соответствии со стандартами Единой системы программной документации (ЕСПД) и другими действующими стандратами (приложение 1).

Положение устанавливает общий порядок разработки, экспертизы, тиражирования, поставки и использования педагогических программных средств вычислительной техники (ППС) и регулирует отношения в этой области между организациями и учебными заведениями системы профтехобразования.

Педагогические программные средства вычислительной техники предназначены для использования в учебно-воспитательном процессе учебных заведений профтехобразования.

ППС представляет собой программу или программы на носителе данных с программной и инструктивно-методической документацией, разработанные по действующим стандартам и другим нормативным документам, прошедшие опытную проверку и экспертизу, принятые в производство, изготовленные по утвержденной в установленном порядке технологии и зарегистрированные в отраслевом фонде алгоритмов и программ.

Разработка, производство и сопровождение ППС относятся к деятельности по созданию особо сложной продукции.

Разработка ППС для учебных заведений профтехобразования производится на основе «Координационного плана создания, экспертизы и тиражирования педагогических программных средств вычислительной техники системы профтехобразования на 1986—1990 гг.», утвержденного Госпрофобром СССР.

Научное руководство вопросами разработки ППС в системе профтехобразования осуществляет Всесоюзный научно-исследовательский институт профессионально-технического образования (ВНИИ профтехобразования). Он же является головным учреждением по организации работы по созданию ППС и координации деятельности по внедрению электронно-вычислительной техники в учебно-воспитательный процесс.

Разрабатываемые педагогические программные средства подразделяются на четыре основные группы:

а) программные средства по различным учебным предметам и дисциплинам, изучаемым в системе повышения квалификации и в учебных заведениях профтехобразования; по производственному обучению;

б) программные средства, предназначенные для проведения внеурочной работы и технического творчества учащихся и студентов;

в) программные средства, предназначенные для решения информационных и управленческих задач в системе профтехобразования;

г) другие программные средства, в том числе специализированные пакеты прикладных программ (ППП) для автоматизированных обучающих систем (АОС).

Заказчиками ППС могут быть:

госпрофборы союзных республик

ВНИИ профтехобразования

Свердловский инженерно-педагогический институт (СИПИ)

Всесоюзный научно-методический центр профессионально-технического обучения молодежи (ВНМЦ), Всесоюзный институт повышения квалификации руководящих работников и специалистов профтехобразования (ВИПК) (на основании трудовых соглашений), базовые предприятия средних профтехучилищ, средние профтехучилища и индустриально-педагогические техникумы.

Разработчиками ППС могут быть:

ВНИИ профтехобразования, ВНМЦ, СИПИ, ВИПК, научно-исследовательские, опытно-конст-

---

Утверждено Государственным комитетом СССР по профессионально-техническому образованию

рукторские организации и вычислительные центры других ведомств, высшие учебные заведения, опорные средние профтехучилища и индустриально-педагогические техникумы по внедрению электронно-вычислительной техники в учебно-воспитательный процесс, средние профтехучилища и техникумы, авторы-разработчики.

При разработке ППС функции заказчика и разработчика могут быть совмещены.

За разработчиками ППС (коллективы или отдельные лица) сохраняется авторство.

Оплата за разработку ППС, если она не является плановым заданием, осуществляется заказчиком в соответствии с действующими положениями и нормативами.

Порядок разработки и экспертизы педагогических программных средств вычислительной техники

Разработка и экспертиза ППС как продукции производственно-технического назначения осуществляется в соответствии с ГОСТ 15.001—73 «Разработка и постановка продукции на производство. Основные положения», а также стандартами ЕСПД.

Заказчик составляет и предъявляет разработчику заявку-исходные требования на разработку ППС. Форма заявки на разработку ППС приведена в приложении 2.

При создании ППС выделяются следующие стадии:

разработка технического задания;
разработка технического проекта;
подготовка рабочего проекта;
внедрение.

Техническое задание (ТЗ) разрабатывается на основе исходных требований заказчика и является основным документом для разработки ППС и документации на него. ТЗ подготавливается разработчиком и согласовывается с заказчиком.

Заказчик согласовывает техническое задание в срок не более 15 дней со дня получения.

ТЗ разрабатывается в соотвествии с ГОСТ 19. 201—78. Дополнительные требования на ППС приведены в приложении 3.

Технический проект (ТП) состоит из педагогического сценария, содержащего отобранный и укомплектованный материал для данного вида программной продукции, схемы алгоритма и описания программы (сведения о логической структуре и функционировании программы).

Рабочий проект (РП) состоит из программного изделия (программы на носителях данных) и программной документации, в обязательном порядке включающей текст программы и инструкцию пользователю, отражающую специфику и методические вопросы применения данного ППС.

Разработанная и отлаженная программа проходит приемочные испытания. Они проводятся для оценки соответствия ППС сопроводительной документации, ТЗ и дидактическим требованиям, а также с целью оценки уровня качества.

В проведении приемочных испытаний ППС участвуют разработчики, педагоги, методисты и представитель заказчика.

По результатам приемочных испытаний составляется протокол по форме, приведенной в приложении 4.

В случае необходимости проводится доработка ППС, вносятся коррективы в педагогический сценарий, программу и сопроводительные документы.

На стадии «Внедрение» осуществляется окончательное оформление всех материалов и документов ППС для представления их на заключительную экспертизу в экспертный совет по оценке педагогических программных средств вычислительной техники, предназначенных для использования в учебных заведениях профтехобразования.

В экспертный совет на заключительную экспертизу представляются следующие материалы и документы ППС:

техническое задание;

педагогический сценарий, схемы алгоритма и описание программы;

текст программы;

программное изделие — программа на носителе данных (магнитный диск или лента);

протокол приемочных испытаний;

инструкция пользователю.

Заключительная экспертиза ППС проводится в соответствии с «Положением об экспертном совете по оценке педагогических программных средств вычислительной техники, предназначенных для использования в учебных заведениях профтехобразования».

При проведении экспертизы в общем случае проверяют:

соответствие ППС дидактическим целям обучения, современному уровню технических и методических требований;

комплектность документации ППС.

По результатам проведения заключительной экспертизы составляется экспертное заключение. Форма экспертного заключения приведена в приложении 5.

Тиражирование, поставка и использование педагогических программных средств вычислительной техники

ППС с заключением экспертного совета направляется в Ученый совет Госпрофобра СССР для принятия решения об их тиражировании и внедрении в систему профтехобразования. Объем тиражирования различных программных продуктов определяется в каждом конкретном случае отделом информатики и электронно-вычислительной техники и Учебно-методическим управлением Госпрофобра СССР.

Тиражирование ППС в системе профтехобразования осуществляет Всесоюзное объединение производственных предприятий Госпрофобра СССР (ВООП).

При ВНИИ профтехобразования создается отраслевой фонд алгоритмов и программ, в который включается:

педагогический сценарий, схема алгоритма и описание программы;

текст программы;

программное изделие — программы на носителе данных (магнитный диск или лента);

инструкция пользователю;

специальные пакеты прикладных программ (ППП).

Поставка ППС в учебные заведения профтехобразования производится в соответствии с действующим в Госпрофобре СССР порядком.

В комплект ППС при поставке входят:

программное изделие — программа на носителе данных (магнитный диск или лента);

инструкция пользователю.

По отдельному требованию могут также поставляться более подробные методические указания пользователю и текст программы.

В учебных заведениях профтехобразования учет ППС осуществляется в соответствии с дествующим порядком в составе основных средств или малоценных и быстроизнашивающихся предметов.

Условия эксплуатации ППС должны соответствовать предъявляемым техническим требованиям.

Приложение 1

Перечень стандартов, на основе которых разработано Временное положение

1. ГОСТ 19.001—77	Единая система программной документации. Общие положения.
2. ГОСТ 19.002—80	ЕСПД. Схемы алгоритмов и программ. Правила выполнения.
3. ГОСТ 19.003—80	ЕСПД. Схемы алгоритмов и программ. Обозначения условные графические.
4. ГОСТ 19.004—80	ЕСПД. Термины и определения.
5. ГОСТ 19.101—77 (СТ СЭВ 1626—79)	ЕСПД. Виды программ и программных документов.
6. ГОСТ 19.102—77	ЕСПД. Стадии разработки.
7. ГОСТ 19.103—77	ЕСПД. Обозначение программ и программных документов.
8. ГОСТ 19.104—78 (СТ СЭВ 2088—80)	ЕСПД. Основные надписи.
9. ГОСТ 19.105—78 (СТ СЭВ 2088—80)	ЕСПД. Общие требования к программным документам.
10. ГОСТ 19.106—78 (СТ СЭВ 2088—80)	ЕСПД. Требования к программным документам, выполненным печатным способом.
11. ГОСТ 19.201—78 (СТ СЭВ 1627—79)	ЕСПД. Техническое задание. Требования к содержанию и оформлению.
12. ГОСТ 19.202—78 (СТ СЭВ 2090—80)	ЕСПД. Спецификация. Требования к содержанию и оформлению.
13. ГОСТ 19.301—79 (СТ СЭВ 3747—82)	ЕСПД. Программа и методика испытаний. Требования к содержанию и оформлению.
14. ГОСТ 19.401—78 (СТ СЭВ 3746—82)	ЕСПД. Текст программы. Требования к содержанию и оформлению.
15. ГОСТ 19.402—78 (СТ СЭВ 2092—80)	ЕСПД. Описание программы.
16. ГОСТ 19.403—79	ЕСПД. Ведомость держателей подлинников.
17. ГОСТ 19.404—79	ЕСПД. Пояснительная записка. Требования к содержанию и оформлению.
18. ГОСТ 19.501—78	ЕСПД. Формуляр. Требования к содержанию и оформлению.
19. ГОСТ 19.502—78 (СТ СЭВ 2093—80)	ЕСПД. Описание применения. Требования к содержанию и оформлению.
20. ГОСТ 19.503—79 (СТ СЭВ 2094—80)	ЕСПД. Руководство системного программиста. Требования к содержанию и оформлению.
21. ГОСТ 19.504—79 (СТ СЭВ 2095—80)	ЕСПД. Руководство программиста. Требования к содержанию и оформлению.
22. ГОСТ 19.505—79 (СТ СЭВ 2096—80)	ЕСПД. Руководство оператора. Требования к содержанию и оформлению.
23. ГОСТ 19.506—79 (СТ СЭВ 2097—80)	ЕСПД. Описание языка. Требования к содержанию и оформлению.
24. ГОСТ 19.507—79 (СТ СЭВ 2091—80)	ЕСПД. Ведомость эксплуатационных документов.
25. ГОСТ 19.508—79	ЕСПД. Руководство по техническому обслуживанию. Требования к содержанию и оформлению.
26. ГОСТ 19.601—78	ЕСПД. Общие правила дублирования, учета и хранения.
27. ГОСТ 19.602—78	ЕСПД. Правила дублирования, учета и хранения программных документов, выполненных печатным способом.
28. ГОСТ 19.603—78 (СТ СЭВ 2089—80)	ЕСПД. Общие правила внесения изменений.
29. ГОСТ 19.604—78 (СТ СЭВ 2089—80)	ЕСПД. Правила внесения изменений в программные документы, выполненные печатным способом.
30. ГОСТ 15.001—73	Разработка и постановка продукции на производство. Основные положения.

Приложение 2

_____

(наименование организации-заказчика, адрес)

ЗАЯВКА

на разработку педагогического программного средства

_____

(наименование)

1. Цель, назначение и область применения ППС _____

_____

(тип учебного заведения, название предмета, темы, профессия и др.)

_____

2. Предполагаемый разработчик (соисполнитель)

_____

3. Срок выполнения заявки (год, квартал)

_____

4. Источники финансирования _____

_____

Дата

Должность Личная подпись Расшифровка подписи

_____ _____ _____

Заключение организации-разработчика с указанием ориентировочной стоимости всех работ по разработке продукции.

Должность Личная подпись Расшифровка подписи

_____ _____ _____

Дата

Примечание. Исходные требования заказчика являются неотъемлемой частью заявки и содержат:

назначение ППС;

вид вычислительной техники и язык программирования;

условия эксплуатации;

другие требования.

Исходные требования выполняются в виде приложения к заявке.

Приложение 3

Дополнительные требования к техническому заданию на разработку педагогических программных средств вычислительной техники

Техническое задание (ТЗ) на разработку педагогических программных средств вычислительной техники в общем случае должно состоять из следующих разделов:

наименование и область применения;
основания для разработки;
назначение разработки;
технико-педагогические требования;
стадии и этапы разработки;
порядок контроля и приемки;
приложения.

В техническом задании допускается уточнять содержание разделов, вводить новые разделы или объединять отдельные из них.

В разделе «Наименование и область применения» указывают наименование учебного предмета (раздела, темы и т. д.) и профессии, на которые разрабатывается данное ППС, или наименование курса, краткую характеристику области его применения, общую характеристику комплекса (вида) учебно-вычислительной техники, для которого создается ППС.

В разделе «Основания для разработки» в общем случае указывают полное наименование документа (документов), на основании которого разрабатывается ППС.

Основанием для разработки являются: приказы Госпрофобра СССР, госпрофобров союзных республик, Координационный план создания экспертизы и тиражирования педагогических программных средств вычислительной техники системы профтехобразования, хозяйственный договор, договор о творческом сотрудничестве, трудовое соглашение.

В разделе «Назначение разработки» указывают цели создания и применения ППС в учебно-воспитательном процессе, функциональные особенности и условия эксплуатации данного ППС. Дается ссылка на аналогичные существующие ППС или указывается, что оно разрабатывается впервые.

В разделе «Технико-педагогические требования» указывают:

технические требования, определяющие показатели качества ППС и эксплуатационные характеристики ЭВМ;

дидактические требования, предъявляемые к ППС в соответствии с целями и задачами обучения;

методические требования, исходя из рациональной методики преподавания, организации деятельности преподавателя и учащихся.

В разделе «Стадии и этапы разработки» устанавливают необходимые стадии, этапы и содержание работ (перечень программных документов, которые должны быть разработаны, согласованы, утверждены), а также сроки разработки и определяют исполнителей.

В разделе «Порядок контроля и приемки» указывают документы, подлежащие согласованию и утверждению на различных этапах разработки, виды испытаний и общие требования к приемке ППС.

В приложениях к техническому заданию при необходимости приводят:

перечень научно-исследовательских и других работ, обосновывающих разработку;

схемы алгоритмов, таблицы, описания, обоснования, расчеты и другие документы, которые могут быть использованы при разработке;

другие источники разработки.

Форма титульного листа технического задания

_____

(наименование организации-разработчика)

УТВЕРЖДАЮ

_____

(Подпись должностного лица

_____

организации-разработчика, дата)

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

на разработку _____

_____

(наименование педагогического программного средства)

СОГЛАСОВАНО

_____

(Подпись должностного лица

_____

организации-заказчика, дата)

Приложение 4

Протокол приемочных испытаний ППС

Приемочная комиссия в составе:
председателя _____

(фамилия, инициалы, должность

_____

организация)

и членов комиссии 1. _____

(фамилия, инициалы

_____

должность, организация)

2. _____

назначенная приказом (распоряжением) по _____

_____

(наименование организации)

№ _____ от _____ провела испытания _____

(наименование ППС)

В результате приемочных испытаний комиссия установила:

_____

(даются выводы о соответствии программного

_____

изделия, сопроводительной документации ТЗ

и дидактическим требованиям и оценивается уровень

качества продукции)

Председатель комиссии (подпись)
Члены комиссии (подписи)

Приложение 5

УТВЕРЖДАЮ

(подпись председателя экспертного органа, Ф., И., О., дата)

ЭКСПЕРТНОЕ ЗАКЛЮЧЕНИЕ

на

(наименование ППС или его части)

На рассмотрение представлены следующие материалы

(наименование представленных материалов)

разработанные

(наименование организации-разработчика или Ф., И., О. автора)

в соответствии с

(наименование документа, являющегося основанием для разработки ППС)

В результате экспертизы установлено

(излагается краткий текст заключения, общая оценка и выводы)

Предлагается при доработке внести в программное изделие и документацию следующие изменения и дополнения:

Заключение принято
на заседании экспертного совета
протокол № _____ от (дата, год)
Секретарь экспертного совета (подпись)

Положение

об Экспертном совете по оценке педагогических программных средств вычислительной техники, предназначенных для использования в учебных заведениях профтехобразования

Общие положения

Экспертный совет по оценке педагогических программных средств вычислительной техники (в дальнейшем — совет) осуществляет экспертизу педагогических программных средств (ППС), предназначенных для использования в учебных заведениях профтехобразования.

Совет создается при головной научной организации в системе Госпрофобра СССР — Всесоюзном научно-исследовательском институте профессионально-технического образования (ВНИИ профтехобразования г. Ленинград).

В своей деятельности совет руководствуется Единой системой программной документации, нормативами и методическими документами ГКНТ, ГКВТИ, приказами и решениями Госпрофобра СССР, отраслевыми нормативно-методическими документами, Временным положением о порядке разработки, экспертизы, тиражирования, поставки и использования педагогических программных средств вычислительной техники в системе профтехобразования и настоящим Положением.

Совет в своей деятельности подотчетен Госпрофобру СССР.

Задачи и функции совета

Рассмотрение и оценка поступающих ППС на их соответствие дидактическим целям обучения, современным техническим и методическим требованиям.

Подготовка экспертного заключения (приложение 1) на ППС и представление их Ученому совету Госпрофобра СССР.

Рассмотрение предложений и рекомендаций по использованию ППС, разработанных в других организациях и ведомствах, а также имеющихся зарубежных ППС.

Утверждено Государственным комитетом СССР по профессионально-техническому образованию

Состав совета

Состав совета формируется из числа ведущих ученых, специалистов и инженерно-педагогических работников. В него входят представители Госпрофобра СССР, ВНИИ профтехобразования, ВНМЦ, СИПИ, ВИПКа, техникумов и профтехучилищ, специалисты предприятий и учреждений других министерств и ведомств.

Состав совета утверждается Госпрофобром СССР. Изменения в составе совета согласуются с руководством Госпрофобра СССР.

В заседаниях совета могут принимать участие разработчики ППС, приглашенные практические работники профтехобразования и другие специалисты, не являющиеся членами совета (без права решающего голоса).

Порядок работы совета

Заседания совета проводятся под руководством председателя, а в его отсутствие — заместителя председателя. Протокол каждого заседания направляется в Госпрофобр СССР.

Работа совета осуществляется на основе годовых планов.

Материалы для рассмотрения на заседаниях совета подготавливаются секретарем совета из числа поступивших на экспертизу и заключение.

В перечень материалов входит:

техническое задание;

педагогический сценарий, схема алгоритма и описание программы;

текст программы;

программное изделие — программа на носителе данных (магнитный диск или мини-кассета);

протокол приемочных испытаний;

инструкция пользователю.

Экспертиза представленных ППС осуществляется путем исполнения программы на ЭВМ, ее анализа и проверки основных приемов и режимов подачи учебной информации.

Совет правомочен принимать решения, если на заседании присутствует не менее половины его состава.

Решение совета считается принятым, если за него проголосовало более 50 % членов совета, участвовавших в заседании. Решения принимаются открытым голосованием, если не было договоренности о проведении тайного голосования.

ППС, прошедшие заключительную экспертизу в совете, направляются в Ученый совет Госпрофобра СССР для принятия решения об их тиражировании и внедрении в систему профтехобразования. Объем тиража различных программных продуктов определяется в каждом конкретном случае Отделом информатики и электронновычислительной техники и Учебно-методическим управлением Госпрофобра СССР.

Все спорные вопросы по принятым решениям и рекомендациям разрешаются Госпрофобром СССР.

Обязанности председателя, заместителя председателя и секретаря совета

Председатель осуществляет общее руководство работой совета по экспертизе ППС:

вносит предложения в Госпрофобр СССР по составу совета;

руководит обсуждением на заседаниях совета вопросов, связанных с оценкой представленных ППС;

утверждает план работы совета.

Заместитель председателя выполняет следующие функции:

осуществляет координацию деятельности с различными органами и учреждениями профтехобразования по совершенствованию ППС, методической документации, направленной на дальнейшее повышение их эффективности и качества;

осуществляет непосредственное руководство экспертизой ППС в рамках совета и подготовкой заключений на них;

своевременно рассматривает предложения членов совета по совершенствованию его работы и всей системы экспертизы;

выполняет обязанности председателя совета в его отсутствие.

Секретарь совета ведет организационную работу и делопроизводство:

составляет проекты планов работы совета и осуществляет контроль за их выполнением;

организует заседания совета (получение и сортировка необходимых материалов, передачи их членам совета для ознакомления и изучения, рассылка извещений и др.);

подготавливает проекты заключений совета;

оформляет протоколы заседаний совета;

контролирует выполнение решений и рекомендаций совета.

Члены совета обязаны присутствовать на его заседаниях, тщательно изучать материалы к заседанию совета, активно участвовать в их обсуждении и рассмотрении, в срок выполнять поручения председателя совета или его заместителя по экспертизе поступающих ППС.

Члены совета участвуют в его работе на общественных началах.

Т. Драгныш

Главное — учеба

В первый свой приход сюда я столкнулась с интересным фактом.

Мне показывал электронно-вычислительные машины своего училища мастер производственного обучения Г. В. Годжелло. Во время нашего разговора подошел мальчик, поодаль стояла женщина.

— Георгий Владимирович, можно, я покажу маме, чем мы здесь занимаемся?

Сказал спокойно: было ясно, что уверен — не откажут.

Они сели за новенькую «Ямаху» и, тихо переговариваясь, провели около часа в кабинете вычислительной техники.

Вряд ли нуждается в комментариях этот эпизод. Когда человеку хочется поделиться с близкими чем-либо,— как правило, это важно для него. Прекрасно, когда этим «самым важным» становится учеба!

Заинтересовало в училище многое. Например, оригинально использован ленинградский опыт переоборудования лингафонного кабинета в кабинет вычислительной техники. Немало изобретательности проявлено преподавателями, чтобы творчески перенести находки коллег «на московскую почву».

Второй раз приехав в училище, я попала на экзамен. Торжественность момента подчеркивали и цветы на столе, и взволнованный вид ребят, толпившихся у дверей.

Вот приглашается для ответа очередной выпускник. Это Дима Макарочкин. Он окончил школу № 170, ему 17 лет. СПТУ заканчивает с красным дипломом. Он получил специальность оператора ЭВМ. Отвечает смело, уверенно; получает «отлично». Когда подхожу к нему после ответа, чтобы поговорить, он шу-

тит, что выучил 22 билета, а ему попался билет № 23.

— Я собираюсь поступать в МИРЭА, на факультет кибернетики и автоматики,— говорит Дима.

А за столами готовятся к ответу учащиеся. Девочки охотнее садятся отвечать к ласковой Ларисе Алексеевне. Те, кто побойчее, идут к Георгию Владимировичу. В ответах ребят чувствуются глубокие знания. Экзамен напоминает, скорее, аналогичное событие в вузе, настолько все серьезно и значительно.

После идем к Ольге Юрьевне Селезневой, преподавателю информатики. Застаем ее в окружении ребят. Идет обычная для любой школы, но почему-то удивительная здесь беседа об... аккуратности заполнения дневников.

Знакомимся. Некоторые ребята, например Андрей Ручкин и Олег Белоусов, из группы электромехаников по ремонту и обслуживанию ЭВМ с большим интересом отнеслись к созданию в нашем журнале молодежной рубрики, собираются подписаться на журнал «Информатика и образование».

...Вернувшись домой, рассказываю дочери об увиденном в СПТУ-199...

Педагогический опыт

Э. ЛИБА
Директор Ныоской средней школы. Эстонская ССР

От арифметики до информатики

История Ныоской школы начинается с 1686 года. Ученики известного эстонского просветителя Форселиуса организовали тогда в Эстонии первые деревенские школы. В них кроме азбуки, пения, чтения и закона божьего по просьбе родителей детей обучали и основам арифметики. Последовательное обучение математике начинается с 60-х годов прошлого века, после реформы крестьянских народных школ.

При Советской власти новое содержание получило образование и в нашей школе. В 1953 году семилетняя школа была преобразована в среднюю, первый выпуск (14 десятиклассников) был в 1957 году. В 1963 году построено новое здание. В следующем году при участии вычислительного центра Тартуского государственного университета и Тыраверской обсерватории Института астрономии и астрофизики АН ЭССР начали работать классы с углубленным изучением физики и математики, где обучались учащиеся из разных районов республики, преимущественно из сельских местностей. В 1973 году при школе открыли интернат на 200 учеников. Ныне здесь установлены 25 комплектов вычислительной техники, на которых обучаются почти 700 учеников, около 100 учителей, воспитателей, инженеров и других работников школ. Среди базовых предприятий школы — Институт астрономии и астрофизики АН ЭССР, кафедра общей физики и вычислительный центр Тартуского государственного университета.

У нас проходят практику студенты Тартуского государственного университета, педшколы Таллинского педагогического института.

В 1986 году на базе ЭВМ «Урал», которую университет передал нашей школе, был основан вычислительный центр. Здесь используются ЭВМ «Наири-3-1», «ЕС-1020», «Агат».

— ЭВМ — неотделимая часть нашей школы, — рассказывает учитель математики Харри Кеэрутая. — Начиная с 1980 года на базе ЭВМ «Наири-3-1» разрабатываются задания по физике и химии, которыми пользуются в других школах республики. Классные руководители используют ЭВМ для подведения итогов успеваемости учеников и посещаемости ими школы.

Для лучшего использования компьютеров нами составлены первые программы для персональных ЭВМ. Среди учеников есть энтузиасты, способные реализовать на ЭВМ проекты сложных учебных программ. К сожалению, у нас пока нет надежных персональных ЭВМ, а это мешает внедрению в школьную практику перспективных методов обучения. Несмотря на то что ученые республики разработали для школ 3 персональных компьютера («Тарту», «Юку», «Энтель»), их производство пока еще не налажено. Для школы, располагающей кадрами и имеющей ценный опыт обучения учащихся работе на ЭВМ, нужно создавать необходимые условия для дальнейшего развития. Это важно, потому что наш вычислительный центр является учебной базой и для других школ Эстонской ССР.

— Живу в Ныо, — говорит Тоомас Саарсен, студент I курса университета, окончивший Ныоскую школу с серебряной медалью. — Интерес к математике появился еще в V классе. На ЭВМ начал работать с VIII класса, по примеру старшего брата. Был членом ученического научного общества. Составил программы модификации имитатора для ЭВМ «Тарту». Уже во время учебы в школе работал в школьном вычислительном

центре, получая там зарплату. В 1984 и 1985 годах принимал участие в летней школе программистов в Новосибирске, где получил диплом I степени. Интересуюсь фотоделом, музыкой, электроникой.

Рассказывает старшеклассник Тарви Мартенс:

— Я сам из Пярну. Учился в Пярнуской средней школе. Интерес к точным наукам появился уже в начальной школе. Учитель Тийт Эрм стал моим руководителем по математике. Когда я учился в V классе, я стал победителем районной олимпиады среди учащихся VI класса. Во время республиканской олимпиады был на экскурсии в Ныоской средней школе. Увидел вычислительный центр, школу и сделал выбор — после окончания VIII класса буду здесь учиться.

С ЭВМ основательно познакомился только в Ныо. Под руководством учителя Уно Пууса работал сначала на ЭВМ «Найири-3-1», позднее на персональных компьютерах. Сейчас я член секции математики-кибернетики ученического научного общества и научный секретарь южного района республики. На II Прибалтийской конференции ученических научных обществ в 1986 году стал лауреатом. Сейчас занимаюсь разработкой трансляторов языков для «Агата» и участвую в передачах телешколы эстонского телевидения, посвященных компьютерам. В школе учусь на «4» и «5». Ежегодно принимаю участие в олимпиадах точных наук. Увлекаюсь музыкой. Окончил Ныоскую музыкальную школу по классу фортепиано, пою в хоре и играю в ансамбле.

Собираюсь поступать в ТПИ по специальности «Электронно-вычислительные машины» или «автоматические системы управления».

— Живу в Тыравере. На ЭВМ начал работать с 8 лет под руководством отца, — говорит ученик IX класса Вейко Саар. — Сейчас на работу с ЭВМ у меня уходят все свободные вечера, примерно 30 часов в неделю. После окончания школы хочу поступать в университет.

В 1986 году Ныоскую среднюю школу окончили 80 учащихся, из них 22 учились только на «5» и «4». С золотой медалью окончили школу двое: Марти Райдал, который сейчас студент университета, и Яак Таммемяги — студент Таллинского политехнического института. С серебряной медалью окончили школу семеро. Все они учатся в вузах.

Широкое применение математики и ее методов в объективном познании окружающей нас природы весьма характерно для развития науки. Если в конце прошлого и начале этого столетия математика подчинила себе физику, химию и экономическую науку, то сейчас она вторгается даже в сферу гуманитарных знаний. Поэтому дальнейшее развитие той или иной отрасли науки в наше время во многом зависит от подготовки специалистов с математическим складом мышления и умением применять математические методы.

В. СОБОЛЕВ
Канд. техн. наук. Пензенский приборостроительный техникум

Учить по-новому

Внедрение станков с числовым программным управлением, гибких производственных модулей, робототехнических комплексов способствует появлению профессий более высокого интеллектуального уровня.

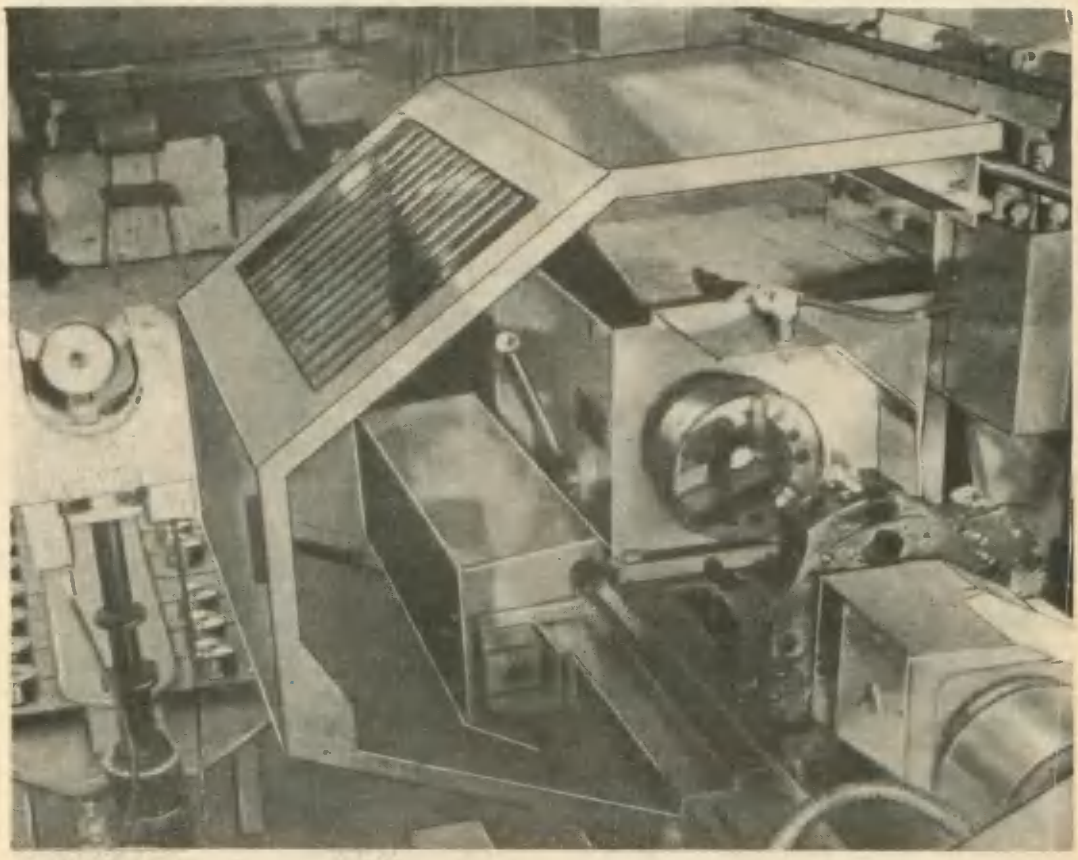
На заводах появляются наладчики электронно-вычислительной техники, операторы автоматизированных устройств, включая и станки с числовым программным управлением, другие специалисты. Их подготовка требует нового качества подхода к преподаванию предметов. Массовое использование принципиально новой техники стимулирует приобретение рабочими новых специальностей.

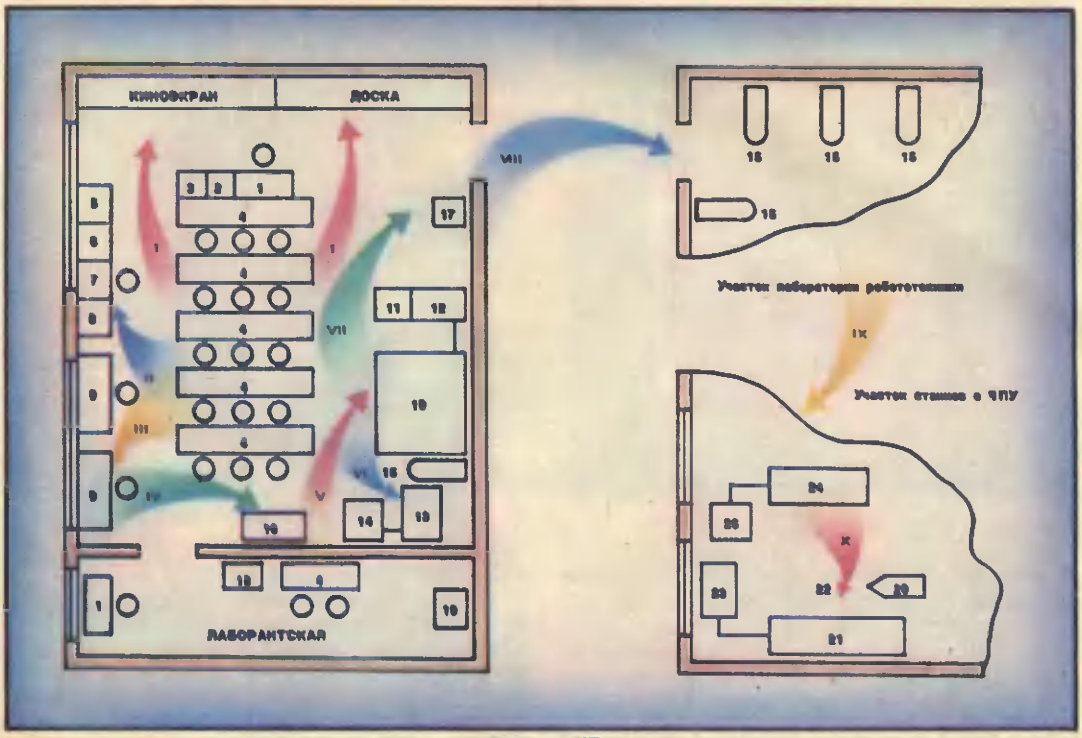
Необходимо готовить специалистов с широким техническим кругозором, обладающих компьютерной грамотностью. В современном производстве нужен специалист, который в одном лице мог бы сочетать качества технолога, программиста, оператора, наладчика станка с ЧПУ. Это позволит сосредоточить в одних руках технологическую и программную подготовку, обеспечивающую быструю и качественную работу по пуску или переналадке станка с ЧПУ. Кроме того, повысится ответственность специалиста за конечный результат своей работы. Технико-экономическая эффективность от внедрения новой техники во многом обеспечивается современной подготовкой специалиста.

Гибкое автоматизированное производство не будет столь гибким, если в разработке нормативно-технической документации по освоению новой техники будут задействованы многие специалисты, каждый из которых не в состоянии выполнить от начала до конца весь объем работы.

В. Т. Соболев проводит экскурсию по своей лаборатории

Станок с ЧПУ и автоматическим манипулятором, установленный в лаборатории Пензенского приборостроительного техникума





Сейчас все шире используются системы автоматизированного проектирования (САПР) конструкторско-технологической документации, автоматизированные рабочие места конструктора, технолога. Персональные ЭВМ позволяют на экране дисплея воспроизводить текст технической документации, давать распечатки на ее форматах единого образца и одновременно получать управляющие программы для гибких производственных систем ГПС.

Поэтому будущий специалист по робототехническим и автоматизированным комплексам должен получить в учебных заведениях профессиональную подготовку не по отдельным, изолированным друг от друга предметам, а сразу по всему циклу производственной цепочки: разработка изделия — выпуск.

Сейчас в техникуме созданы целевые лаборатории — программно-автоматизированного оборудования систем ГПС и САПР, роботезированных технологических комплексов. Оснащены эти лаборатории современной техникой. Здесь есть станки с ЧПУ, гибкий модуль, промышленные роботы разных типов, устройства связи станков с ЧПУ и автоматизированных комплексов с ЕС ЭВМ, графопостроитель, автоматизированные рабочие места конструктора, технолога и программиста.

При работе с учащимися изложение учебного материала идет в той последовательности, которая позволяет им быстро изучить теорию и приобрести необходимые практические навыки по работе с программно управляемым автоматизированным оборудованием.

Вместе с имеющимися в лаборатории устройствами она оснащена рабочими местами для преподавателя и учащихся. Это связано с тем, что лабораторные и практические работы весьма сложны и преподаватель, прежде чем допустить учащихся к работе на станках с ЧПУ, должен провести курс теоретических занятий. Целесообразно проводить их в стенах такой лаборатории, ибо суть лекционного материала будет лучше усвоена учащимися и вызовет у них определенный интерес к изучению предмета. На рисунке показан вариант плана целевой лаборатории в Пензенском приборостроительном техникуме и схематично дана последовательность изложения и закрепления теоретического материала практической работы по курсам «Техническое и программное обеспечение работы станков с ЧПУ и РТК» и «Оборудование и основы построения ГПС».

Л. МАТВЕЕВА

Информатика и... туризм

Можно ли совместить занятия по школьной программе и учебно-тематические экскурсии?

Такая форма работы появилась в г. Ровно в 1980 г. Ее развили в г. Иванове, поддержали в ряде городов Российской Федерации и других союзных республик. Опыт БММТ «Спутник» Ивановского ОК ВЛКСМ не перенимается механически, к нему подходят творчески. Но суть его остается — экскурсии организуются по программам Минпроса СССР в учебное время за счет часов, отведенных на организацию тематических экскурсий и изучение родного края.

Прочно вошли в школьное расписание учащихся девятых классов Иркутской области уроки истории, химии, физики, географии, биологии, литературы и информатики, организуемые БММТ «Спутник» Иркутского ОК ВЛКСМ.

Например, трехдневный урок биологии на озере Байкал. Ребята знакомятся с флорой и фауной озера, осматривают музей природы или охотоведения при Иркутском сельскохозяйственном институте, участвуют в беседах об охране окружающей среды, о защите от загрязнения, о сохранении памятников природы.

Такие специализированные заезды организуются в Иркутской области с ноября 1986 г. В них уже участвовали около 500 учащихся 17 районов.

Двухдневные учебные экскурсии для учащихся десятых классов школ Курской области вошли в практику совместной работы БММТ «Спутник» и областного отдела народного образования недавно, но уже есть положительные отзывы об уроках истории на знаменитой Курской дуге, уроках биологии в Курском сельскохозяйственном институте, на станции юных натуралистов и предприятиях агропромышленного комплекса.

Интересна учебно-тематическая программа БММТ «Спутник» Тувинского обкома комсомола. В ней наряду с организацией учебных экскурсий предусмотрены обмен опытом пионерской и комсомольской работы по краеведению и туризму в школе, встречи с работниками горкома, обкома комсомола, вечера дружбы на базе городских школ г. Кызыла. Один день полностью посвящен профориентационной работе. Учащиеся посещают базовые предприятия профтехучилищ, знакомятся с системой обучения в них, встречаются с преподавателями и руководством.

Совсем недавно в расписаниях общеобразовательной школы появились уроки информатики и вычислительной техники. Сразу же встала задача: как помочь учащимся овладеть новым предметом, как сделать его до-

ступнее. И снова на помощь пришли учебные экскурсии. В Красноярском крае большой популярностью пользуются четырехдневные тематические заезды учащихся IX—X классов под девизом «Наука, компьютеризация и туризм», которые проводятся совместно Красноярским университетом и Академгородком. В программе — посещение вычислительного центра университета, лекции и практические занятия на персональных компьютерах. В ноябре — декабре 1986 г. участниками заездов стали 180 старшеклассников.

Интересно проводит учебные экскурсии по информатике и вычислительной технике БММТ «Спутник» Тюменского обкома ВЛКСМ. На уроках информатики в Тюмени уже побывали около 1500 десятиклассников сельских школ области.

Заезды организуются за счет учебных часов, отведенных школьной программой на изучение предмета «Основы информатики и вычислительной техники», и проводятся потоками по 120 человек (четыре группы). Программа рассчитана на три дня. Контроль за комплектованием групп по районам области, за соблюдением графика заезда, оплата путевок осуществляются органами народного образования.

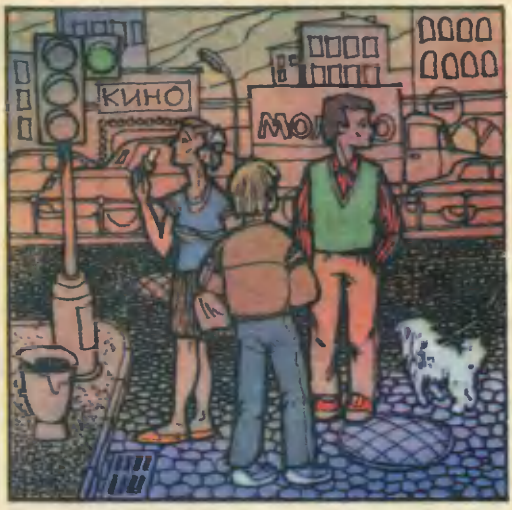
Для размещения туристов БММТ заключен договор на аренду мест на турбазах «Нефтяник» и «Верхний бор».

Программа пребывания в Тюмени включает ежедневные практические занятия по «Основам информатики и вычислительной техники» в течение каждого из трех дней, экскурсию по городу, посещение картинной галереи и музея «Штаб-квартира В. К. Блюхера».

Основная часть пребывания включает в себя учебные занятия на ЭВМ на базе Тюменского университета, индустриального института, студотрядов, на вычислительных центрах Гипротюменьнефтегаза, облагропрома.

Программа практических занятий по основам информатики и вычислительной техники составлена и утверждена органами народного образования.

Занятия со школьниками ведут студенты



вузов и инженеры-программисты вычислительных центров на добровольных началах.

20 декабря 1986 г. отделом науки ОК ВЛКСМ на базе кафедры вычислительной математики университета был проведен семинар с инженерами и студентами по программе учебных занятий, отдельным приемам методики преподавания.

Занятия с юными туристами ведутся интересно. Молодые «преподаватели» в дополнение к программе придумывают различные развлекательные, развивающие игры.

Позиция областного отдела народного образования понятна: в области 550 средних школ, и лишь в 23 школах имеются микро-ЭВМ. Учебно-тематические экскурсии — полезный вид работы.

Вот как ответила на вопрос: «Нужны ли подобные экскурсии?»— ученица X класса Московской средней школы Тюменского района: «Уроки информатики, начиная с IX класса, давались нелегко. Когда первый раз увидела ЭВМ, испугалась: как буду на ней работать?! Прошло три дня, и я успешно решала квадратные уравнения, строила графики функций, играла с машиной в игры. Во время практики получала только «5». Думаю, что поездка в Тюмень поможет мне и в дальнейшей учебе по информатике».

Е. ПОЛАТ
Канд. пед. наук

Проблемы использования компьютеров в системе образования развитых капиталистических стран

Компьютер вошел в систему школьного образования многих стран с конца 70-х как предмет и как средство обучения. Во многих странах внедрение компьютера в школы с самого начала контролировалось государством. Например, в Великобритании, первой стране капиталистического мира, в которой компьютер пришел в школу, в 1981 г. разработаны государственные программы внедрения компьютерного обучения в школы Англии и Уэльса («Микроэлектроник эдьюкэйшенэл програм»), Шотландии («Микроэлектроник дивелопмент програм»). В 1982 г. принята программа внедрения компьютера в начальную школу («Микроэлектроник энд праймэри эдьюкэйшен»). Во Франции в 1982 г. одобрена программа, в соответствии с которой каждая школа к 1988 г. будет оборудована восемью компьютерами. В январе 1985 г. утвержден план «Информатика для всех», который предусматривает массовое оснащение школ, коллежей, лицеев микропроцессорной техникой.

Что касается США, ФРГ, Австралии, то там внедрение компьютеров в обучение отдано на рассмотрение местных органов образования (штатов в США, Австралии, земель в ФРГ). Причем, если в Австралии и ФРГ этот процесс полностью контролируется местными органами образования, в США главенствующую роль играет частный бизнес. Фирмы, руководствуясь коммерческим интересом, стремятся привязать школы к собственной продукции.

В странах, где школьная политика находится под непосредственным контролем правительства (Франция, Великобритания), школам предоставляются государственные субсидии на приобретение техники, прикладных программ. Например, во Франции[1] в соответствии с планом «Информатика для всех» предусматривается в ближайшее время установить в школах 120 тыс. микро-ЭВМ модели «Томсон Т07 70» и «Эксельвизион-100», техника поставляется в комплекте с пакетами прикладных программ [1]. В Великобритании 27 тыс. школ пользуются 50 %-ной субсидией, которую предоставляет им департамент профессионального образования. В то же время программы школы приобретают за свой счет. Парк машин в школах развитых капиталистических стран уже довольно значителен и постоянно растет. Так, в настоящее время в американской начальной и средней школе[2] — более 1 млн. компьютеров. Это вдвое больше, чем два года назад. Более 15 млн. учащихся и 500 тыс. учителей использовали компьютеры в 1984/85 учебном году. Свыше 7 тыс. начальных и 16 тыс. средних школ (это 1/4 всех школ США) имеют по 15 компьютеров и более. В США начальное звено обучения имеет в среднем 3,5 машины на каждую школу, среднее звено — 7 компьютеров, старшее — 11 [3]. В Великобритании каждая школа имеет в среднем 5 компьютеров. Но разумеется, техника распределяется по школам неравномерно. Небольшой опрос, который провел Дерин Уотсон (Великобритания), показал, что в одних школах парк компьютеров составлял 16 машин, в других — одну [4]. В лучшем положении оказываются школы, получающие дополнительные субси-

[1] Материалы по Франции подготовлены А. Н. Литвиновой.
[2] Материалы по США подготовлены Т. Г. Жарковской.

дии от частных организаций. В Великобритании это классические школы и так называемые паблик скулз.

Используются самые разнообразные модели машин. В некоторых странах разработаны специальные рекомендации для школ в отношении приобретения техники. В Великобритании это «РМЛ Линк 480 Зет» (фирма «Рисёч Машин»), «Синклер Спектрум» и «Би Би Си Акорн Модель Б» [2]. Во Франции — указанные выше модели. В Австралии — отечественный компьютер «Майкроби». Широко используются и зарубежные образцы, главным образом американские «Эппл», «Коммодор», «Ай Би Эм», «Рэдио шэк». Самый большой объем продажи персональных компьютеров не только в США, но и во всем мире приходится на долю компании «Эппл». Только в штате Алабама в 1985/86 учебном году 1400 школ закупили 6231 персональный компьютер «Эппл-2».

Все это говорит о том, что процесс внедрения компьютерной техники в школы развитых капиталистических стран с каждым годом набирает силу. Чем объясняется феномен повышенного внимания к компьютеру со стороны не только органов образования, но и правительств? Только ли потребностями современного производства, технологии, подготовки кадров? Сейчас уже стало совершенно очевидно, что нет. В основе такого небывалого интереса к новому техническому средству лежат две причины, имеющие поистине стратегическое значение для всего социального развития общества. Первая заключается в характере современного и особенно будущего развития экономики любой страны мира. Говорят, что в экономическом соревновании выиграет та страна, корпорация, фирма, которая будет иметь лучший доступ к информации. То есть первая причина — в актуальности, стратегической значимости информационного обеспечения, следовательно, информационной технологии. Вторая, как указывает известный американский педагог Ант. Маллан, заключается в том, что компьютер нужен самой системе образования как средство развивающего обучения [6].

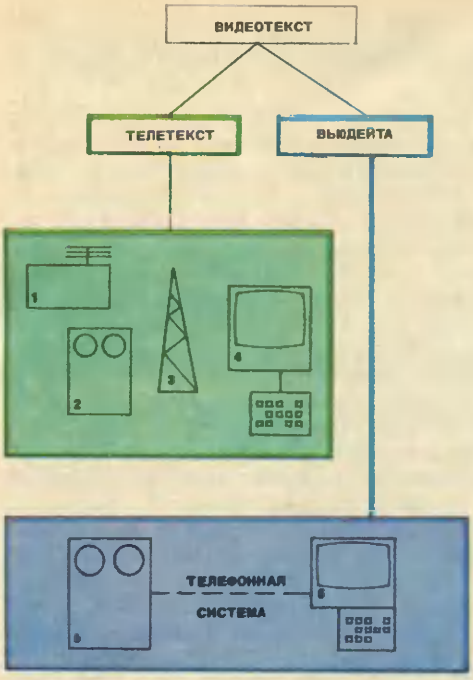
Отсюда в политике компьютерного обучения на Западе четко прослеживаются две тенденции: 1) рассматривать компьютер в качестве важного звена информационной технологии; 2) использовать компьютер в массовой школе как средство обучения различным предметам школьной программы.

Остановимся на указанных тенденциях подробнее.

Компьютер как элемент информационной технологии

В настоящее время подавляющее большинство стран вынуждены констатировать все увеличивающийся разрыв между уровнем теоретической и практической подготовки учащихся школ и потребностями современной науки и техники. Проблема настолько актуальна, что она стала предметом обсуждения Международного конгресса ЮНЕСКО в 1981 г. по естественнонаучному и трудовому обучению и вопросам национального развития. В 1984 г. утверждена долгосрочная программа международного сотрудничества по этой проблеме (1984—1989). В Великобритании издан специальный правительственный указ, который предписывает обязательное включение в программы школ теоретических разделов, отвечающих современному уровню развития науки и техники, формирование у учащихся соответствующих практических умений и навыков. Прогрессивные педагоги во многих странах начали движение «Научные знания всем». Вместе с тем, как отмечают ведущие дидакты мира, выполнение этой насущной задачи чрезвычайно сложно, поскольку объем информации в любой отрасли ежегодно удваивается, а в таких областях, как медицина, биология, экономика, утраивается. Только в области медицины, например, в 1970 г. было опубликовано 390 тыс. статей, более тысячи статей в день [7]. Вот почему, как считают английские педагоги Дэвид Лэйтон [8], Джерри Веллингтон [4] и другие, ни современный учитель самой высокой квалификации, ни современный учебник не в состоянии уследить за потоком новейших сведений, которые появляются в той или иной области знаний. Нужны новые надежные источники информации, доступные как учителю, так и учащимся. Перед органами образования различных стран мира стоит задача освоения всего арсенала информационной технологии. При этом механические способы накопления, хранения и выдачи информации постепенно заменяются электронными средствами — компьютерами и их сетями.

Во Франции в 9040 наиболее крупных школах, а также в 2733 коллежах и 500 лицеях смонтированы локальные информационные системы «Нанорезо». Эта система разработана фирмой «Леанор» специально для группового обучения и представляет собой сочетание профессионального 16-разрядного центрального компьютера фирмы «Бюль» (рабочее место учителя) и микрокомпьютеров фирмы «Томсон» (рабочие ме-



1. ТВ студия

2. Центральная ЭВМ хранящая в памяти сотни фреймов

3. Передатчик

4. Модернизированная ТВ установка со специальным интерфейсом, позволяющим соединиться с центральной ЭВМ

5. Центральная ЭВМ хранящая в памяти тысячи фреймов

6. Модернизированная ТВ установка со специальным интерфейсом

ста учащихся). Система «Нанорезо» рассчитана на одновременную работу с 31 рабочим местом учеников.

Компьютерное обучение в школе дополняется использованием других информационных средств — видеодисков, видеотекстов и др.

Видеодиски уже сейчас используются для передачи, получения и хранения информации. Они стали существенной частью информационной технологии. Изображение и звук телевизионной передачи могут быть записаны на лазерный видеодиск. Лазер считывает с видеодиска информацию, которая передается на экран телевизора. Лазерные диски имеют существенные преимущества: отсутствие контакта между проигрывающей головкой и диском, что означает их долговечность; возможность записи огромного количество единиц информации — 30 томов всего на одном диске; легкость и четкость управления. Последнее, возможно, наиболее важно. Диски, которые могут быть управляемы пользователем, называются интерактивными дисками. Видео становится интерактивным, когда предлагает вопросы и варианты ответов для выбора или предлагает пользователю задать собственные вопросы или дать собственные ответы. Ответы пользователя определяют отбор информации. Этот диалог между зрителем и видео обычно контролируется компьютерной системой, программа которой предлагает различные варианты и пути решения проблемы. Пока интерактивные видео находятся в начале своего развития, но развитие это перспективно для целей образования.

Системами видеотекса являются телетекст и вьюдейта. Вьюдейта действует на основе телефонной связи. Информация, запрошенная абонентом по телефону с любого расстояния, поступает на экран специального телевизора или на дисплей персонального компьютера. Это интерактивная система. Вся информация хранится в огромном банке данных центрального компьютера. Пользователь имеет возможность запросить нужную ему информацию либо заложить в память компьютера собственную. Примером системы вьюдейта служит в Великобритании «Престель», в Японии — «Кептен».

Система телетекста передает информацию по телевизионным каналам также из банка данных центрального компьютера. Примерами такой системы в Великобритании являются «Сифэкс 1», «Сифэкс 2» (Би Би Си), «Орэкл» (Ай Ти Ви). Это неинтерактивные системы. Пользователь может получить любую запрашиваемую информацию, но сам передать на хранение свою информацию не может.

Для пользования указанными системами телевизоры снабжаются специальным устройством. Нажимая клавишу на панели этого устройства, можно соединиться с центральным компьютером. Нажатие другой клавиши дает индексы информации, которой располагает центральный компьютер.

Описанные выше информационные системы уже сейчас используются в школах Великобритании, Франции, Японии, хотя пока нельзя говорить о массовом их применении, в первую очередь из-за чрезвычайной дороговизны абонентной платы за использование каналов связи. Доступны они пока немногим элитным школам, а также достаточно состоятельным индивидуальным абонентам.

В то же время эти примеры показывают перспективы развития и влияния информационной технологии на систему образования будущего. Системы, позволяющие обеспечить доступ практически к любой информации в считанные минуты, секунды, мо-

гут принципиально изменить подход к проблеме школьного образования и самообразования, к выбору методов и организационных форм обучения. Особое значение, как отмечается в материалах ЮНЕСКО, приобретают различные информационные системы для обучения на расстоянии, что актуально для развивающихся стран, где особо остро стоит проблема подготовки педагогических кадров.

Рассмотрение компьютерного обучения в школе как части широкого использования информационной технологии приводит как следствие к другой отчетливо просматриваемой в педагогической литературе Запада тенденции, а именно: применение компьютера в массовой школе как средства развивающего обучения.

Компьютер как средство обучения

В ряде стран (Франция, Австралия) с самого начала было принято направление на использование компьютера в качестве средства обучения. В других странах (США, Канада, Великобритания, Дания, ФРГ) вначале компьютер в школе рассматривался и как предмет изучения, и как средство обучения. В настоящее время практически во всех указанных странах компьютер используется как средство обучения. В плане изучения даются лишь основы компьютерной грамотности, так называемые клавиатурные навыки, элементы программирования, в начальной школе — на языке Лого, в средней — на языке Бейсик или другом в зависимости от используемой техники. Углубленно информатику изучают в спецшколах, профессионально ориентированных на эту область знания.

Такая направленность компьютерного обучения поставила проблемы, которые в настоящее время дискутируются в печати, а именно: целесообразность организации специальных кабинетов ВТ и определение общедидактических и частнометодических функций компьютера.

Единой точки зрения на организацию компьютерного обучения нет. Высказываются точки зрения и за создание кабинетов ВТ и против. Приведем аргументацию английского педагога Джерри Веллингтона [4].

Аргументы за кабинет ВТ:	*Аргументы против кабинета ВТ:*
обеспечивается лучшая сохранность оборудования; легче устанавливать технику; более подходящие условия эксплуатации ВТ (соответствующая мебель, гигиенические условия, постоянный подвод питания и т. д.); легче проводить профилактику и уход (поддерживать температурные условия, удалять пыль)	трудности составления расписания уроков с использованием ВТ в курсах различных дисциплин; резко сокращаются возможности интеграции компьютера с другими средствами обучения, его использования в ходе лабораторных работ, практикумов и пр.

Способы использования компьютера в качестве средства обучения различны. Это и работа всем классом и группами, и индивидуальная работа с компьютером. Перечисленные способы обусловлены не только наличием (или нехваткой) достаточного количества аппаратных средств, но и дидактическими целями. Так, если в классе имеется только компьютер учителя или если учитель ставит перед собой задачу организации коллективной работы по поиску решения задач, постановки проблемы и т. д., он организует работу класса на основе учительского компьютера. Надо сказать, что такой подход в ряде случаев оказывается даже более продуктивным, чем индивидуальная работа учащихся с персональным компьютером или дисплеем.

Так, английские педагоги Джон Никол, Жаклин Дин [9] из Университета Экзетер организовали сравнительный эксперимент в ходе преподавания истории. В одном классе учитель организовал коллективную работу учащихся на базе учительского компьютера, в другом — на основе индивидуальной работы с персональными компьютерами. Учащиеся должны были найти логическую цепочку вопросов компьютеру, ответы на которые привели бы к решению проблемы. в первом случае учитель организовал коллективное обсуждение каждого последующего вопроса (инициатива предоставлялась учащимся). Только одобренный большинством учащихся и наиболее убедительный вопрос вводился учителем в компьютер, и все ученики могли видеть на экране дисплея, насколько он был успешен, в каком направлении следует продумывать следующий вопрос. В другом классе диалог с машиной учащиеся вели самостоятельно. Эксперимент показал, что решение проблемы в первом случае занимало значительно меньше времени и проходило очень оживленно. Во втором случае разброс во времени среди учащихся был большой, были огорчения и радости. Работа принесла удовлетворение от коллективного общения, обмена мыслями. Видимо, в педагогическом процессе выбор спо-

соба использования компьютера следует ставить в прямую зависимость от дидактической задачи (в данном случае задача заключалась в развитии логического мышления учащихся, построения определенного алгоритма), возраста учащихся (это были дети 10—11 лет), сложности программы. Необходимо сочетание различных форм работы с компьютером: коллективной, групповой, индивидуальной.

Использование компьютера в качестве средства обучения выдвигает задачу разработки научно обоснованной типологии программ как на уровне дидактики, так и на уровне частных методик с учетом специфики каждого предмета.

В педагогической литературе можно встретить различные подходы к построению типологии обучающих программ. Мне приходилось уже об этом писать[1]. Приведу еще один возможный подход, на который в литературе 1984—1985 гг. ссылаются многие педагоги Великобритании. Это подход, разработанный в 1977 г. Кемисом, Аткином и Райтом [4].

Они выделяют четыре вида программ:

обучающие программы;

поисковые программы;

проверочные программы;

свободные программы.

Программирование для компьютера в рамках компьютерного обучения первоначально развивалось из принципов программированного обучения, что предполагало расщепление задачи на шаги, построение алгоритма ее решения. Эти принципы и сейчас находят широкое применение в составлении обучающих программ. Однако беда состоит в том, что многие учителя и составители программ не могут выйти за рамки этого в определенных пределах и для определенных дидактических целей весьма продуктивного подхода, тем самым резко ограничивая потенциальные возможности компьютера как средства развивающего обучения. Ведь на современном этапе важной целью использования компьютеров должно быть развитие умения ставить проблемы, искать их рациональные решения, ориентироваться в постоянно растущем потоке информации. Поисковые программы вовлекают ученика в поиск правильного решения. Понятия и выводы теории раскрываются ученику постепенно по мере использования программы. В отличие от обучающих программ, где компьютер дает задание и контролирует действия ученика по ходу решения задачи, усвоения определен-

[1] См.: Полат Е. С. Компьютер и школа // Физика в школе. 1984. № 2.

ного задания, выполнения упражнения и пр., в поисковых программах компьютер выступает в роли посредника между учеником и поставленной в определенной ситуации проблемой. Эта ситуация может быть реальной (например, некий производственный процесс), исторической (например, Англия времен викингов), теоретической (например, теория частиц) или даже воображаемой (например, город будущего). В настоящее время поисковые программы реализуются как моделирующие программы различных типов. Они могут моделировать лабораторный эксперимент, природное явление или производственный процесс.

Преимущества такого подхода:

в экономии материальных средств, лабораторного оборудования, реактивов; в экономии времени на подготовку эксперимента;

в возможности изучения процесса, который иным способом в условиях класса продемонстрировать невозможно;

в простоте контроля переменных при моделировании.

Проверочные программы позволяют ученику вмешиваться в программу, проверять собственные идеи, гипотезы. В отличие от предыдущего подхода, где модель задается программистом в готовом виде, здесь ученик сам создает модель, проверяет ее, модифицирует. Такие программы, к сожалению, еще достаточно редки в компьютерном обучении.

Наконец, так называемые свободные программы предполагают применение компьютера в качестве средства, экономящего труд, время, умственные затраты при выполнении расчетов, графических построений. В этом случае компьютер используется как вспомогательный инструмент при выполнении тех или иных заданий.

Опыт работы с компьютером в последние годы, совершенствование техники обучающих программ позволили расширить приведенную выше типологию за счет следующих типов программ:

«Редактор текстов»;

программ создания собственного или использования готового банка данных (которые могут понадобиться в поисковых, проверочных и свободных программах);

программ использования компьютера для управления, контроля за другими системами, приборами.

Общедидактические подходы к типологии программ, учитывающие дидактические функции компьютера, позволяют методистам выделить его методические функции с учетом специфики предмета. Так, в преподавании математики Дэвид Джессон [4] выделяет пять функций:

1) выполнение упражнений. Здесь компьютер представляет пользователю вопросы (ранжированные по трудности), упражнения...;

2) электронная доска. Компьютер используется как цветное, динамическое визуальное средство обучения;

3) моделирование;

4) исследование. Из числа предлагаемых вариантов, данных ученик выбирает и аргументирует собственное решение;

5) математические расчеты в курсах других дисциплин.

В преподавании иностранных языков Грэхэм Дэвис [4] выделяет следующие типы программ:

1) программы упражнений;

2) программы тестирования, предусматривающие диагностику ошибок, систематизацию результатов выполнения заданий группой и отдельным учеником;

3) обучающие программы. Их использование начинается с введения информации, затем следуют упражнения. Предусматривается более детальная диагностика ошибок, разветвления, коррекция;

4) программы запроса. Они используют банк данных. Реализуется также возможность запроса упражнений на подкрепление полученной учеником информации, сбор и хранение сведений, указывающих, какие знания (упражнения) ученик не усвоил (не выполнил);

5) программы моделирования. Обычно такие программы включают сценарии, в которые ученик может вносить свои изменения;

6) лингвистические игры: кроссворды, ребусы, предложения или тексты, игры на догадку и пр.

При использовании компьютера в обучении родному языку Малькольм Солт [4] считает целесообразным использовать следующие типы программ:

1) выполнение упражнений;

2) программы обучения, где даются комментарии к правилам, практические рекомендации, помогающие ученику выполнять задания;

3) программы моделирования. Эти программы могут быть полезны в классной работе, обсуждениях, анализе, развитии умения логического построения высказываний, планировании сочинений, изложений и пр.;

4) программы решения проблем (особенно на языке Лого). Как отмечал еще в 1980 г. С. Пейперт, такие программы развивают речь в процессе аргументации при постановке и реализации проблем;

5) «Редактор текста».

В преподавании общественных дисциплин (истории, географии) Эшли Кент [4] вы-

деляет следующие функции компьютера:

1) статистический анализ;

2) моделирование: а) модели и системы; б) упражнения на принятие решений, игры;

3) закрепление знаний и навыков.

Кроме того, компьютер в преподавании указанных предметов может использоваться для ознакомления учителя (или учащихся) с новой информацией;

как электронная доска;

для контроля и оценки знаний, систематизации этих сведений для учителя;

управления видео (создание комплекса).

При использовании компьютера в преподавании естественнонаучных дисциплин Клив Опи [4] указывает на способность компьютера выполнять быстро сложные работы: представлять на экране данные эксперимента в различных графических формах; обеспечивать интерактивное моделирование экспериментов, проведение которых либо невозможно, либо затруднено в обычном классе; контролировать многочисленные параметры: напряжение, силу тока, температуру, силу света, временные интервалы, давление и т. д.; служить для сбора и хранения всевозможных данных эксперимента; контролировать внешние устройства.

Использование компьютера в качестве средства обучения ставит на повестку вопрос об интеграции его с другими средствами обучения, а также разработку и внедрение в учебный процесс различных микропроцессорных устройств. В настоящее время в начальном обучении Великобритании, США широко используются такие микропроцессорные устройства, как «черепаха» (самостоятельно либо совместно с персональным компьютером). С его помощью и с помощью языка Лого можно осуществить разнообразные графические построения, создавать рисунки на бумаге и на экране телевизора. Используются микропроцессорные устройства в лабораторных работах по физике, химии, биологии, природоведению, помогая накапливать, обрабатывать, систематизировать данные эксперимента, производить необходимые расчеты. Эти устройства могут быть подключены к компьютеру, и тогда все данные, результаты эксперимента, графики могут высвечиваться на дисплее. Примером таких устройств являются электронные лабораторные устройства «Вела», «Джипси» и т. д.

Проведенный анализ педагогической литературы развитых капиталистических стран за 1984—1986 гг. позволяет сделать следующие выводы относительно перспектив развития компьютерного обучения в школах этих стран:

1. Применительно к близкой перспективе можно отметить четко прослеживаемые тенденции использовать компьютер в качестве средства развивающего обучения различным предметам школьной программы.

2. В перспективе заметна тенденция к постепенной замене традиционных средств обучения средствами информационной технологии (видеодиски, видеотексты и др.).

3. Ставится вопрос о необходимости разработки научно обоснованных типологий программ как на общедидактических, так и на частнометодических уровнях.

Литература

1. *Поли А.* Стратегия введения информатики и новых технологий в систему образования Франции // Материалы Международного симпозиума в г. Пловдиве. 1984. Окт.

2. *Russel T.* Computers in the Primary School. Plymouth, 1985.

3. Educational Technology. 1986. June.

4. *Wellington.* Children, Computers and the Curriculum. L., 1985. 267 p.

5. *Dean Ch., Whitlock Q.* A Handbook of Computer Based Training. L., 1983. 259 p.

6. *Mullan A. P.* Children and Computers in the Classroom. L., 1984. 157 p.

7. Microcomputers and the Curriculum / Ed. by A. V. Kelly. L., 1984. 187 p.

8. Innovations in Science and Technology Education / Ed. by Dr. D. Layton. Vol. 1.

9. *Nichol C., Dean C.* Pupils, Computers and History Teaching // New Horizons in the Educational Computing. L., 1983. P. 190—203.

В системе образования США, Великобритании в последние годы применяются такие виды обучающих систем, как телеконференции (аудиоконференции и видеоконференции). Аудиоконференции — это диалоги на расстоянии посредством телефонных линий с помощью специальных аппаратов, напоминающих телефоны. Видеоконференции позволяют вести диалог на значительных расстояниях с помощью телевизионных передач. Партнеры видят друг друга на телеэкране (как в знакомых нам телемостах). Диалоги могут при необходимости сопровождаться демонстрацией на телеэкране разнообразной информации — текстовой, иллюстративной и пр. Такие системы позволяют вести обучение и получать консультации на расстоянии, что имеет особое значение для отдаленных регионов, в частности для развивающихся стран, где остро ощущается нехватка квалифицированных преподавательских кадров.

Например, на Аляске (США) климатические условия настолько суровы, что «дистанционное обучение» — единственный выход.

В Великобритании теле- и видеоконференции используются в дополнение к традиционному обучению. Они имеют особое значение для централизованного инструктирования школьной администрации, повышения квалификации учителей, связи министерства с местными органами образования, позволяют оперативно решать многочисленные вопросы, не отвлекая педагогов от прямых служебных обязанностей. Преподаватели Плимутского политехнического университета используют аудиоконференции, чтобы облегчить студентам усвоение учебного материала, дать дополнительные сведения по различным курсам. На медицинском факультете Йельского университета телеконференции используются для одновременных консультаций или занятий со студентами, которые находятся на практике в 18 медицинских центрах, разбросанных по всему Уэльсу. В Северной Ирландии с помощью телеконференций осуществляется связь между Ольстерским университетом и его тремя филиалами, расположенными на значительных расстояниях друг от друга.

В последние годы в английских школах начинают использовать спутниковую связь, прежде всего в преподавании иностранных языков. Трансляция через прямые радиовещательные спутники открывает новые возможности для передачи фрагментов уроков, например, из страны изучаемого языка в любой регион земного шара, где имеются соответствующие наземные приемные станции. Радиовещание (открытое, учебное, сельский радиофорум, радиошколы) приобретает особое звучание как одна из форм неформального образования населения отдаленных районов и в этом качестве используется в настоящее время в Нигерии, Кении, Танзании.

Все большее распространение получают разнообразные видеосистемы. Специальные программы для систем видео и компьютера создаются корпорацией Би Би Си (Великобритания), например известная программа «Думздей прожект». Два диска содержат свыше 50 000 иллюстраций, 150 000 страниц текста, 25 000 карт. Программа имеет страноведческую направленность и создана по аналогии с известной в Британии книгой — «Думздей бук», которая содержит перепись всех земель Англии 1086 г., сделанную по приказу короля Вильгельма I. Современная видеокомпьютерная версия, разработанная к 900-летию упомянутой книги, дает любую информацию о любых областях Великобритании наших дней. Ученик с помощью компьютера может запросить на экран сначала фоновую карту страны, а затем в зависимости от полученного задания запрашивает данные об определенной местности, ее основных промышленных, сельскохозяйственных центрах, полезных ископаемых, народонаселении и т. д. Каждую новую видеоинформацию можно «накладывать» на предыдущую, постепенно формируя общую картину. Подсчитано, что, если пользователь захочет прочесть на телеэкране только текстовой материал этих дисков, ему потребуется три полных года, а чтобы просмотреть весь банк данных, ему потребуется семь лет. Такая видеокомпьютерная система Би Би Си, куда входят упомянутые выше два диска, видеопроигрыватель и микрокомпьютер, стоит 3000 фунтов стерлингов. Это, конечно, дорого и не по карману обычным школам. Поэтому компания приняла решение о выпуске минисистемы, которая будет включать ограниченное количество текста при сохранении полного набора иллюстраций и карт и может сочетаться с микрокомпьютером, имеющимся в большинстве школ страны. Помимо этого, специально для школ Би Би Си выпускаются видеодиски «Телевижн Шекспир», куда включены шесть наиболее известных произведений Шекспира; «Спаси свою жизнь» (видеоинформация по оказанию медицинской помощи) и др.

В США примером такой системы является «Электронная энциклопедия». Один видеодиск включает 20 томов Академической национальной американской энциклопедии (9 млн. слов, свыше 30 000 статей, 1ІОМ байт информации.) Поиск информации осуществляется мгновенно благодаря специальному лазерному устройству. Эта информация может быть тут же распечатана или перенесена в память машины.

Нам пишут

Один за всех...

...и все за одного. Этот старый, добрый, надежный принцип работает (как ни удивительно) даже в приложении к микрокалькулятору.

4 апреля 1987 г. в школе № 8 Калининграда Московской области шел урок практической части курса ОИВТ с использованием ПМК. Десятиклассники отрабатывали бригадно-эстафетный метод работы. Все используемые на уроке ПМК, которых было меньше, чем учеников, содержали одну и ту же программу. Группа, работающая с конкретным ПМК, получала результат работы предыдущей группы, «подмешивала» его к своим исходным, полученным в начале занятий данным, вводила в калькулятор, вычисляла результат и передавала его «эстафетой» следующей группе. В качестве объектной была использована программа нахождения НОД академика А. П. Ершова (Информатика и образование. 1986. № 3. С. 70) после устранения в ней опечаток и некоторой доработки. Ее текст приводится ниже. Коллективная работа велась «на отметку» в ограниченный отрезок времени. «Передача эстафетной палочки» осуществлялась демонстрацией результата, высвечиваемого на индикаторе. «Госприемка», т. е. преподаватель, контролировала правильность результата и в случае надобности требовала доработки.

Два раза произошло непредвиденное: в самый ответственный момент отказывали ПМК. Преподаватель вынужден был, спасая положение, «ввести в бой» ПМК тех групп, которые «эстафету» уже передали...

Только потом пришло озарение. Да ведь эстафетный метод — это «класс одного ПМК»! Есть же музеи одной картины, приносящие немалые пользу и радость! А выражение «класс одного ПМК» не стало заголовком только потому, что на самом деле калькуляторов было около 10 и «один за всех» не отработан в чистом виде на достаточном числе уроков.

А сколько этот вариант может дать подвижникам информатики и их ученикам! В цепочке «безмашинный вариант ОИВТ» — «класс одного ПМК» — «КВТ» одно лучше другого, если идти слева направо, но в экономическо-временно́м пространстве мы ближе ко второму. Каждый настоящий преподаватель информатики, особенно в сельской местности — а таких большинство,— это энтузиаст в лучшем смысле слова, и он приобрел или готов приобрести ПМК. Если еще и не пожалеть калькулятор для общего дела!...

Один ПМК может научить каждого в классе навыкам работы с готовой программой, умению записывать программу в память, программировать несложные формулы, не бояться ни малых, ни больших ЭВМ.

За исключением тех, кто непосредственно занят передачей «эстафеты», ученики могут и должны заниматься копированием текстов готовых программ в

тетради, развивать навыки их комментирования, перевода в школьный алгоритмический язык и т. п. Если калькуляторов два или три, то эстафета легко превращается в увлекательное и полезное соревнование между «командами».

Приводим текст программы вычисления НОД.

00	X→П1	41
01	→	14
02	X→П2	42
03	П→X1	61
04	П→X2	62
05	—	11
06	Fx≠0	57
07	17	17
08	Fx≥0	59
09	13	13
10	X→П1	41
11	БП	51
12	03	03
13	/—/	0L
14	X→П2	42
15	БП	51
16	03	03
17	П→X1	61
18	С/П	50
19	БП	51
20	00	00

Для запуска программы необходимо нажать В/О, набрать одно число, нажать В↑, набрать другое число и нажать С/П.

Б. Тарасенко

А. Бурденко

На чем учить компьютерной грамотности сегодня?

Введение в школьный курс обучения новой дисциплины «Основы информатики и вычислительной техники» поставило ряд вопросов, из которых самыми острыми являются: чему учить и на чем учить? Если ответ на первый вопрос в какой-то мере дается в пробном учебном пособии по основам информатики и вычислительной техники, то ответ на второй вопрос еще ждет своего решения.

На наш взгляд, обучение программированию без использования вычислительной техники столь же малоэффективно, как и учеба игре на музыкальном инструменте без него. И дело здесь, как показывает опыт, не столько в том, что учащийся не приобретает в этом случае навыков общения с ЭВМ, сколько в том, что без реализации программ на ЭВМ учащийся, не видящий результата своего труда, быстро теряет интерес к учебе. Наоборот, обучение программированию с «выходом на машину» является для учащегося настолько увлекательной работой (и игрой одновременно), что многие из них не прекращают этой работы даже во время перемены.

На какой же вычислительной технике обучать школьника программированию сегодня? «Большие» стационарные ЭВМ со сложными операционными системами, которыми в настоящее время оснащены вычислительные центры, мало подходят для этой цели. Во-первых, из-за высокой стоимости и малочисленности. Во-вторых, такие машины требуют многочисленного и высококвалифицированного обслуживающего персонала. В-третьих, работа на них требует знания сложного для школьника алгоритмического языка и сложной системы команд связи с машиной при вводе и выводе информации. Все это приводит к тому, что «выходы на машину» школьника возможны в лучшем случае 1—2 раза в неделю. Отладка каждой программы может растянуться на длительный период времени. За «деревьями» трудностей общения с машиной теряется «лес» самого программирования.

Опыт обучения обычной грамоте свидетельствует, что если дать учащемуся самые простые средства — бумагу и карандаш и организовать постоянную работу с этими средствами под

руководством учителя, то успех обеспечен. Переход в дальнейшем на другой алфавит или на более совершенные инструменты для письма сведется к приобретению чисто технических навыков.

Итак, по нашему мнению, для успешного обучения компьютерной грамотности школьник должен иметь на первом этапе доступ к надежной в работе и простой в эксплуатации ЭВМ, обладающей сравнительно несложной системой команд. Такие ЭВМ с соответствующим программным обеспечением уже создаются. Но процесс оснащения ими всех общеобразовательных школ, организация технического обслуживания и ремонта такой техники потребуют значительного времени и определенных материальных затрат.

А между тем имеются простые в употреблении, надежные в работе и дешевые вычислительные средства, которыми можно оборудовать учебные кабинеты в школе уже сейчас. Речь идет о программируемых микрокалькуляторах (ПМК) или, как еще их называют, микро-ЭВМ индивидуального пользования. Такие микро-ЭВМ с полным правом можно считать «грифельной доской» второй грамотности. Устройство моделей «Электроника БЗ-34, МК-54, МК-56, МК-61, МК-52» допускает возможность реализации всех основных вычислительных процессов: линейных, разветвляющихся и циклических, а также позволяет использовать подпрограммы, прямую и косвенную адресацию, организовать хранение информации (данных и программ) в памяти. Конечно, по быстродействию и возможностям, объему оперативной памяти ПМК сильно уступают стационарным ЭВМ, но для учебных целей и для решения относительно простых задач их возможности вполне достаточны. По этой причине даже в вузах, имеющих вычислительные центры, студенты I и II курсов выполняют часть вычислительных работ на программируемых микрокалькуляторах, приобретая начальный уровень компьютерной грамотности. И только на старших курсах, когда действительно появляется необходимость, студенты работают в вычислительном центре.

Программируемые микрокалькуляторы доступны для пользователя. Они дешевы (один экземпляр ПМК «Электроника МК-54» стоит 65 руб.), портативны, удобны для ввода и вывода информации, их входной язык прост. Длина программы, которая помещается в памяти вышеназванных моделей ПМК, достигает 98—105 шагов. Сегодня на рабочем месте инженера находится программируемый микрокалькулятор, который стал так же необходим многим специалистам, как в недалеком прошлом логарифмическая линейка. По некоторым данным, использование ПМК в инженерных и научных расчетах снимает в ряде случаев до 30 % нагрузки со стационарных ЭВМ вычислительных центров.

Исходя из вышеизложенного, руководство Одесского высшего инженерного морского училища им. Ленинского комсомола решило оборудовать учебный кабинет по изучению «Основ информатики и вычислительной техники в подшефной школе № 21 Одессы настольными программируемыми микрокалькуляторами «Электроника-56». Для этого школе было передано 16 специализированных столов с подведенным электрическим питанием и 34 ПМК. Преподаватели училища разработали плакаты по курсу «Основы информатики и вычислительной техники», которые были изготовлены на Одесском художественно-производственном комбинате и переданы школе. Кроме того, сотрудники училища подготовили и передали школе методическое учебное пособие по программированию на отечественных микрокалькуляторах. Первые занятия в школе по «Основам информатики и вычислительной техники» провели преподаватели кафедры высшей математики училища. В настоящее время кафедра поддерживает с педагогическим коллективом подшефной школы постоянную связь, оказывает методическую помощь.

Применение программируемых микрокалькуляторов органически вписывается в традиционные разделы школьного курса математики и физики. При многократных вычислениях значений функции по одной и той же формуле, которые приходится выполнять школь-

нику при построении графиков функций и т. п., использование ПМК предоставляет возможность ознакомить старшеклассника с идеологией современной прикладной математики, численными методами решения сложных задач на ЭВМ. Например, на ПМК легко реализуются программы численного решения трансцендентных уравнений, методы численного интегрирования. На уроках математики ПМК сделает наглядными многие отвлеченные теоретические построения, позволит продемонстрировать сходимость числовой последовательности к пределу, численно проверить возникающие у учащихся гипотезы о свойствах геометрических фигур.

На уроках физики ПМК является эффективным инструментом для обработки результатов лабораторных работ. Кроме того, используя ПМК, можно учитывать в математической модели изучаемого физического явления силы сопротивления, нелинейность уравнений движения и многое другое. Широкое и постоянное применение ПМК в школе позволит выработать у школьников алгоритмическое мышление, т. е. заложить прочный фундамент компьютерной грамотности.

С. ОСИПОВА

Красноярский институт цветных металлов — школе

Кафедра высшей математики Красноярского института цветных металлов помогает школе в решении задач, связанных с компьютеризацией. При кафедре работает кружок «Юный программист» для учащихся IX—X классов. Ниже приведена программа занятий кружка, ориентированная на изучение программирования для микрокалькулятора (МК) «Электроника Б3-34» на первом году обучения.

I. Лекционные занятия

1. Роль вычислительной техники в современном обществе.

Постановление Секретариата ЦК ВЛКСМ, коллегии ГКНТ СССР, Президиума АН СССР, коллегии Минвуза СССР «Об участии комсомольцев и молодежи в развитии, эффективном применении вычислительной техники и изучении основ ее использования». Простейшие примеры инженерных задач, решаемых с помощью ЭВМ (1 ч).

2. Понятие алгоритма, примеры. Способы представления алгоритмов: предписания, формулы, блок-схемы (3 ч).

3. Алгоритмическая запись, оператор (команда), программа. Линейные, разветвляющиеся и циклические алгоритмические процессы (4 ч).

4. Функциональные возможности микрокалькулятора Б3—34, его структурная схема. Представление чисел в МК. Режим ручных вычислений. Одноместные операции (4 ч.)

5. Работа микрокалькулятора в автоматическом режиме. Программирование линейных алгоритмических процессов (2 ч.)

6. Команды переходов, программирование разветвляющихся и циклических процессов (4 ч).

7. Подпрограммы (1 ч).

8. Косвенная адресация (2 ч).

II. Практические занятия

1. Ввод чисел. Нормализация. Программирование линейных алгоритмических процессов. Составление программы вычисления площади треугольника по

формуле Герона. Вычисление углов треугольников с использованием теоремы косинусов (2 ч).

2. Программирование разветвляющихся и циклических процессов. Составление программы нахождения корней трансцендентного уравнения методом половинного деления. Составление программы, включающей подпрограмму (8 ч).

Все занятия строились таким образом, чтобы каждая задача имела понятный для учащихся смысл; программировались вычисления, часто используемые на уроках математики и физики. Включение вычислительных упражнений прикладного характера позволило сделать работу учащихся более содержательной и целенаправленной. Большое внимание уделялось закреплению у школьников навыков поэтапного решения задачи от ее постановки и написания алгоритма в виде блок-схемы до отладки программы и анализа полученного результата.

Работа кружка часто проходила в виде семинаров, на которых школьники докладывали о результатах своей деятельности демонстрировали самостоятельно составленные программы. Лучшие работы были представлены на краевую конференцию НОУ, проводимую КГУ, и получили хорошую оценку.

По договору между институтом и межшкольным учебно-производственным комбинатом Кировского района учащиеся девятых классов были распределены в институт для прохождения летней производственной практики по специальности «оператор ЭВМ». На практике школьники ознакомились с работой вычислительного центра института, изучили программирование на Бейсике, составили следующие программы:

1. Нахождение корней квадратного уравнения.
2. Вычисление площади треугольника по формуле Герона.
3. Решение систем линейных алгебраических уравнений по формулам Крамера.
4. Простое и двойное разветвление.
5. Табулирование функции с помощью операторов цикла.
6. Табулирование функций с двумя переменными (вложенные циклы).
7. Вычисление факториала.
8. Игровые ситуации.

Все программы были отлажены и просчитаны на ЭВМ «Мера-60» в ВЦ института.

Сотрудники кафедры высшей математики постоянно оказывают методическую помощь учителям района, приглашают их на занятия по программированию на Бейсике и Фортране, разрабатывают для них методические пособия по некоторым разделам вычислительной математики.

Ю. КОЛОСОВ

Зачем вычислять?

Введение в школу и другие учебные заведения курса ОИВТ ставит множество вопросов, среди которых не последний — о роли вычислений.

Начнем с калькуляторов. В последнее время наблюдается фетишизация счета на них, некая мода. Часто студенты (будущие специалисты!), решая задачу, вместо обдуманной прикидки физического результата, что для инженера особенно ценно, начинают лихорадочно подставлять цифры, не данные, а именно цифры, вычисляют, получают в итоге опять же цифры, а не результаты расчета. Все полученное считают ответом, не думая о положении запятой, числе значащих цифр... Смысл инженерного расчета полностью теряется.

Была дана задача: определить скорость движения, если путь в 100 км пройден за 3 часа. Подсчитывают: 33 333 333. Отвечают — 33 миллиона 333 тысячи 333. Спрашиваю, чего: метров, секунд, рублей? Наконец осознают, это — скорость приблизительно

равная 30 км/час, т. е. не 33 миллиона и т. д., а 33 км/час. Это студенты, а каково же тогда школьникам, особенно при решении абстрактной задачи, в которой нельзя прикинуть результат?

Кстати, практическую пользу прикидочных расчетов наглядно подтверждает практика работников торговли, которым в ряде случаев удобнее (для производительности) рассчитать не точную цифру, а приблизительную. Поэтому они обычно не пользуются калькуляторами, а считают в уме. Вот бы у кого поучиться моим студентам!

Полагаю, что излишнее увлечение вычислениями на калькуляторах отучает от понимания физического смысла задачи, элементарной математики, прикидочных расчетов — основы инженерной интуиции, грамотности инженера.

Не подумайте, что я против вычислительной техники: нет, я ее сторонник, но сторонник разумного использования, практической отдачи.

А что такое информатика? До последнего времени этим термином традиционно обозначали технологию обмена научно-технической информацией, документалистику, библиотечное дело. Однако развитие электронно-вычислительной техники заставило по-новому взглянуть на использование информации в сфере управления экспериментами, оборудованием, разнообразными процессами в экономике, общественной жизни и других областях деятельности человека. В связи с этим информатика становится комплексной научной и технологической дисциплиной, в которой должны изучаться проблемы сбора, переработки информации и создания вычислительных управляющих систем. Наконец, в рамках этой дисциплины могут рассматриваться вопросы проектирования и внедрения вычислительно-управляющих и исполнительных систем в конкретные технологии, процессы, оборудование. Встраиваться могут микро- и мини-ЭВМ, программируемые управляющие калькуляторы и, наконец, специализированные микропроцессорные контроллеры, т. е. вся та техника, которая является базовой при изучении ОИВТ. Именно поэтому информатика так быстро входит в жизнь и, разумеется, в учебные процессы всех уровней — от школ до вузов. Таким образом, я за комплексное понимание термина «информатика» и соответствующей дисциплины.

При изучении информатики в первую очередь должны быть определены цели, а затем поставлены задачи, которые предполагается решать средствами вычислительной техники.

Напомню интересные результаты анкетирования школьников, приведенные в статье Д. Славина «Каждый урок — эксперимент» (Информатика и образование. 1986. № 3). Один из типичных ответов: «Мы делаем какие-то странные вещи, которые трудно как-то применять в практических целях. Мы осложняем свою жизнь и решение элементарных задач... Но я еще ни разу не видел применения этих навыков».

Итак, зачем, с какой целью нужно учиться вычислять?

Не для того, чтоб разучиться считать, а чтобы использовать результаты вычислений (для управления экспериментом, оборудованием, процессом, экономикой, мышлением).

Безусловно, на уровне школы в качестве цели может стоять задача научиться использовать средства вычислительной техники для расчетов как таковых (пригодится в дальнейшем), но эта задача должна решаться не в ущерб изучению других дисциплин — той же математики и тем более физики, химии, биологии (назовем их технологическими). На уровне же профессионального образования в вузах, техникумах, СПТУ главная цель — «вычислять, чтобы управлять»; в какой-то мере это целесообразно и на уровне школ, учитывая растущую тенденцию к профориентации.

Но может быть, к комплексному пониманию информатики преждевременно подходить на стадии школьного обучения, так как в школах нет подходящей техники? Адекватная поставленной задаче техника существует — это программируемые управляющие калькуляторы типа МК-46, МК-64. Они вполне вписываются в учебный процесс — как в методическом

аспекте (требуют изучения программирования, могут использоваться в автоматизированном эксперименте, в качестве управляемых ЭВМ), так и в финансовом (невысокая стоимость — порядка 300 руб.); приобрести их можно по безналичному расчету.

Сравнительно невысокое быстродействие этих калькуляторов при управлении реальными объектами рассматривается как недостаток, в учебном же процессе это превращается даже в определенное преимущество, обеспечивая наглядность процесса управления (как киносъемка в замедленном темпе), а значит, и лучшее усвоение предмета.

Наконец, если на калькуляторах МК-46, МК-64 закрыть табло и клавиши, то получается «модель» микропроцессорного контроллера (или микропроцессора, по широко распространенной терминологии); если же добавить перепрограммируемую память, которая будет хранить набранные программы и запоминать их при отключении питания (калькулятор типа МК-49), то полезность такого прибора в учебном процессе возрастет еще больше. Не случайно идея применения управляющих калькуляторов неоднократно высказывалась в различных публикациях.

Наш первый опыт показал хорошую применимость МК-64 на практике. Удалось поставить серию лабораторных работ по управлению процессами, наиболее характерными для нашей бытовой отрасли, в основном тепловыми. Но подобные процессы встречаются и в пищевой, легкой, лесоперерабатывающей промышленности, агрономической практике. Заводом «Физэлектроприбор» разработан лабораторный комплекс на базе МК-64, который еще лучше вписывается в школьную практику. Гамма входных периферийных устройств, также разработанных заводом, позволяет ставить опыты с набором практически всех физических параметров, которые могут встретиться в школьных дисциплинах.

Следует помнить о той огромной роли, которую могут (и должны!) сыграть в учебном процессе микропроцессоры — хотя бы потому, что за пятилетку их использование в народном хозяйстве должно возрасти в 10 раз. Уже сейчас можно было бы на основе микропроцессора 1807, дешевого (3 р. 55 к.) и надежного, создать школьный демонстрационный комплект, стоимость которого не превысит 150 рублей.

Безусловно, необходимо также предусмотреть возможность подключения аналоговой аппаратуры к создаваемым школьным комплектам учебной вычислительной техники («Корвет», «Электроника УК-НЦ»).

Еще раз подчеркну: использование вычислительной техники в качестве управляющей при изучении курса ОИВТ необходимо, если мы хотим получить от этого курса практическую отдачу.

Информация

Конференция в Новосибирске

В соответствии с планом мероприятий Минпроса СССР в г. Новосибирске с 13 по 17 апреля проходила Всесоюзная научно-практическая конференция «Методика разработки педагогических программных средств», в ходе которой решались следующие задачи:

обобщение опыта разработки педагогических программных средств (ППС);

определение методических установок по разработке и созданию ППС;

определение основных путей использования ЭВТ в учебно-воспитательном процессе;

проведение экспериментальной проверки программно-методического комплекса «Информатика-87» для КУВТ «Ямаха».

В работе конференции приняли участие 127 делегатов от всех республик Советского Союза, в том числе представители МП СССР, АПН СССР, АН СССР, НИИ педагогики и институтов усовершенствования учителей (ИУУ) минпросов союзных республик, педагогических институтов, преподаватели школ и ПТУ.

На конференции работали 3 секции:

технология разработки и использования ППС;

опыт разработки и использования программных средств по курсу ОИВТ;

опыт разработки и использования программных средств по общеобразовательным предметам.

Большой интерес вызвали доклады сотрудников Вычислительного центра Сибирского отделения АН СССР Н. Г. Глаголевой, П. А. Земцова, Е. В. Налимова, Л. С. Бараз о реализации системы программ для языка Рапира на ПЭВМ «Корвет» и «Ямаха». Целый комплекс инструментальных средств продемонстрировали сотрудники лаборатории вычислительных методов мехмата МГУ Д. В. Варсанофьев и А. Н. Дымченко. Об опыте разработки программного обеспечения для БК-0010 в Латвийской ССР рассказал заведующий лабораторией НИИФТТ Латвийского госуниверситета Ю. Я. Кузьмин. Он представил разработанный в республике пакет программ, включающий систему «Рига», Т-язык, программы-исполнители и др.

В процессе работы конференции методисты республиканских ИУУ ознакомились с программно-методическим комплексом «Информатика-87», предназначенным в соответствии с программой 36-часового спецкурса для подготовки руководящих и педагогических кадров народного образования по основам информатики и вычислительной техники.

Созданная на конференции рабочая группа обобщила замечания и предложения, поступившие от участников конференции, и внесла необходимые изменения и дополнения в проект комплекта нормативно-технической и методической документации (НТМД), обеспечивающего единый порядок создания, тиражирования и сопровождения ППС.

Конференция отметила, что в короткий срок усилиями Минпроса СССР и АПН СССР была проделана большая организационно-методическая работа, позволившая скоординировать усилия разработчиков по созданию ППС. В настоящее время уже созданы и апробированы первые отечественные ППС, некоторые из которых демонстрировались в ходе конференции и вызвали большой интерес.

Вместе с тем были выделены острые проблемы:

недостаточно разработаны научные основы компьютеризации образования;

отсутствует единая система разработки и тиражирования ППС и учебно-методических материалов по их сопровождению;

ощущается острая нехватка надежных средств вычислительной техники, позволяющих решать поставленные задачи;

отсутствует научно обоснованный перечень тем для создания ППС по курсу ОИВТ и другим общеобразовательным предметам;

отсутствует должная координация разработки ППС;

отсутствует единая система информационного обеспечения процессов разработки и внедрения ППС в практику народного образования;

ощущается острая нехватка современных инструментальных рабочих станций для опережающих разработок ППС;

ощущается острая нехватка методических, спра-

вочных и руководящих материалов по созданию и применению ППС.

Всесоюзная научно-практическая конференция приняла рекомендации.

Минпросу СССР:

ускорить решение вопроса о создании отраслевого фонда алгоритмов и программ;

утвердить рассмотренный на конференции временный комплект НТМД по созданию ППС;

обеспечить первоочередное оснащение разработчиков ППС необходимыми техническими и программными средствами.

Минпросу СССР и АПН СССР:

с целью выявления состояния знаний учащихся IX и X классов по курсу ОИВТ и использования этих сведений для дальнейшего совершенствования курса и методов его преподавания провести итоговую контрольную работу в отдельных регионах страны;

проанализировав преподавание курса ОИВТ в IX и X классах, подготовить методические рекомендации по изучению основных тем курса. Опубликовать эти материалы в журнале «Информатика и образование»;

собрать предложения по усовершенствованию учебного и методического пособий по курсу ОИВТ для совершенствования материалов учебно-методического комплекта.

АПН СССР:

ежегодно подготавливать план разработки ППС в стране на основе научно обоснованного перечня тем учебного плана, где применение ЭВМ наиболее эффективно;

обеспечить информационное обслуживание научных и педагогических работников, занятых разработкой ППС;

шире привлекать практических работников школы для разработки ППС;

ускорить разработку концепции создания ППС, включив в нее разработку описания структуры деятельности педагога и обучаемого с целью определения методически оправданных областей применения ЭВМ в учебном процессе;

подготовить в 1987 г. научно-методический перечень по всем учебным предметам для создания ППС. Ежегодно дополнять и уточнять этот перечень с учетом накопления опыта и результатов НИР;

определить приоритетность языков программирования, используемых для разработки ППС;

шире практиковать взаимную стажировку специалистов-разработчиков ППС;

подготовить предложения по созданию научно-производственного объединения для разработки, тиражирования и сопровождения ППС.

Всесоюзная школа в Нарве

По инициативе Минпроса СССР и ЦК ВЛКСМ в целях повышения творческой активности, совершенствования профессиональной подготовки молодых ученых и специалистов, обмена опытом преподавания информатики и разработки программного обеспечения в г. Нарве (Эстонская ССР) с 31 марта по 8 апреля проходила третья Всесоюзная школа «Электронно-вычислительная техника в учебном процессе». Ее организацию и проведение осуществлял один из ведущих вузов страны — Тартуский государственный университет.

Сюда съехались более ста молодых ученых, преподавателей, специалистов в области информатики и вычислительной техники со всех концов страны. Обсуждались общие проблемы компьютеризации образования, происходило знакомство с эстонским передовым опытом применения ЭВМ в учебном процессе.

Открывая школу, первый заместитель министра просвещения ЭССР Ф. В. Опер дал обзор состояния преподавания курса ОИВТ в школах республики, рассказал об итогах прошедшего республиканского съезда учителей. Во всех школах Эстонии скоро начнется преподавание по экспериментальному учебному плану в V и X классах (с постепенным вовлечением в каждый последующий год еще одного класса). В результате школьники получат возможность выбирать предметы, факультативы и углубленные курсы по своим интересам и способностям. Предполагается ввести три интегрированных курса: граждановедение, «Человек и природа», «Эстетическая культура».

Председатель республиканского Совета молодых ученых и специалистов, член бюро ЦК ЛКСМ ЭССР Ю. Кахн рассказал о задачах и опыте работы совета. О направлениях развития и внедрения информационной технологии в обществе сообщил директор Института научно-технической информации и технико-экономических исследований У. Агур. Он дал обзор уровня информационной технологии за рубежом, остановился на социальных аспектах и проблемах, порожденных проникновением технических средств во все области человеческой деятельности.

О методической и программной поддержке действующего и перспективного курсов ОИВТ, перспективе разработок компьютерных курсов по отдельным учебным предметам, о публикациях в журнале «Информатика и образование» в 1987/88 учебном году в помощь учителю, об оснащении учебных заведений вычислительной техникой рассказали сотрудники Управления информатики и ЭВТ Г. Баранова, Т. Крюкова.

Начальник отдела учебной информатики ИПИ АН СССР С. Христочевский в своем докладе отметил, в частности, необходимость разработки концепции применения ВТ в учебном процессе, проведения научно-исследовательских работ по выработке требований, предъявляемых к средствам ВТ, используемым для обучения, ускорения работ по созданию базового и инструментального программного обеспечения действующих и перспективных КУВТ, педагогических программных средств для школы и вуза.

О состоянии и перспективах подготовки педагогических кадров педвузами страны, в частности о подготовке учителей информатики и экспериментальной работе по освоению ВТ студентами других специальностей, доложил инспектор вузов

Управления учебных заведений Минпроса СССР В. Ефимов.

Сотрудник проблемной лаборатории Свердловского педагогического института М. Сапир продемонстрировал комплекс учебно-методического и программного обеспечения курса информатики для педвуза и школы, созданный и используемый в Свердловске.

Во время работы школы активно обсуждались педагогические и психологические вопросы компьютеризации. Особый интерес вызвали выступления ст. н. с. НИИ СиМО АПН СССР Т. Сергеевой «Реализация дидактических принципов обучения в условиях применения компьютеров в учебном процессе» и заведующего лабораторией НИИ ОПП АПН СССР В. Рубцова о психолого-педагогических проблемах использования компьютеров в образовании.

Участникам школы были представлены новые разработки средств вычислительной техники, среди них — КУВТ «Корвет», принятый межведомственной комиссией в январе 1987 г. (этими комплектами будут оснащаться средние учебные заведения и вузы страны); новая профессиональная ЭВМ ЕС-1840; персональный компьютер «Юку», созданный специалистами Института кибернетики АН Эстонской ССР; ПЭВМ «Тарту» — разработка Тартуского госуниверситета.

Важной частью работы школы явились демонстрация и обсуждение программного обеспечения на различных видах ВТ, использующихся для преподавания в школах и вузах.

Были рассмотрены вопросы, связанные с организацией работы комиссии при Минпросе СССР по ППС, обсуждался комплект нормативно-технической и методической документации (НТМД), определяющий единый порядок разработки, тиражирования и сопровождения ППС.

Всесоюзная школа молодых ученых и специалистов в области компьютеризации образования затронула актуальные вопросы использования ЭВТ в учебном процессе. В ходе работы участниками были обсуждены методологические, психологические, методические и технические аспекты проблемы создания ППС, рассмотрены вопросы разработки и эксплуатации современных средств вычислительной техники для системы образования.

В целях более эффективного использования электронно-вычислительной техники в учебном процессе, повышения качества ППС, совершенствования методики преподавания курса «Основы информатики и вычислительной техники» участниками Всесоюзной школы были сформулированы следующие рекомендации и предложения.

Считать целесообразным привлечь к работе комиссии по приемке ППС специалистов в области психологии, гигиены, экономики и юриспруденции. В связи с этим обратиться в Управление информатики и ЭВТ Минпроса СССР с предложением о включении в состав комиссии указанных специалистов.

Обсудив временный комплект НТМД, считать целесообразным рекомендовать его к утверждению с учетом высказанных замечаний. НИИ ИВТ АПН СССР откорректировать содержание НТМД по результатам приемки ППС комиссией с последующим опубликованием внесенных изменений в журнале «Информатика и образование».

Обсудив проект требований к перспективному комплекту учебной вычислительной техники (КУВТ), рекомендовать НИИ ШОТСО АПН СССР при его доработке учесть следующие положения:

соответствие параметров дисплеев КУВТ санитарно-гигиеническим требованиям;

обеспечение полной программной совместимости моделей учебных ЭВМ;

необходимость разработки и включения в состав КУВТ унифицированного комплекса технических средств (ЦАП, АЦП,...) для проведения лабораторных экспериментов в учебных заведениях по дисциплинам естественнонаучного цикла с использованием ЭВМ.

Рекомендовать Минпросу СССР рассмотреть вопрос о необходимости разработки сопроцессоров для различных типов ЭВМ, используемых в учебных заведениях, с целью обеспечения их программной совместимости.

Рекомендовать Минпросу СССР, Минвузу СССР, Госпрофобру СССР выйти с предложением в Госкомиздат СССР об издании бюллетеня по разрабатываемым ППС в качестве приложения к журналу «Информатика и образование».

С целью обеспечения непрерывной компьютерной подготовки студентов педагогических вузов и педагогических отделений университетов рекомендовать Минпросу СССР и Минвузу СССР усовершенствовать учебные планы, устранив существующий временной разрыв в преподавании соответствующих дисциплин.

Передвижные компьютерные классы

Передвижные компьютерные классы помогут организации изучения основ информатики в различных районах страны.

В Свердловской области в 1985/86 учебном году Уральский политехнический институт организовал студенческий десант, который познакомил школьников с современной вычислительной техникой. Студенты выезжали в школы и организовали там компьютерные классы, проводя в них занятия.

В Томской области советом по внедрению вычислительной техники при отделе науки и учебных заведений обкома КПСС проведен в июне — июле 1986 г. компьютерный рейд на теплоходе, где были установлены ЭВМ ДВК-2М и «Электроника-60». Учащиеся школ и профтехучилищ прибрежных районов прослушивали вводные лекции и участвовали в практических занятиях.

В зимние каникулы компьютерный десант в составе аспирантов и студентов Томского института автоматизированных систем управления и радиоэлектроники на микроавтобусе объехал школы двух районов области.

Студенты провели теоретические и практические занятия с учащимися на ЭВМ, которую и привезли с собой.

В Ереване по инициативе отдела вычислительной техники Госплана Армянской ССР в салоне автобуса «Икарус», аренду которого финансируют Минпрос и Госпрофобр республики, оборудован компьютерный класс на 14 ПЭВМ.

Несомненно, накоплен положительный опыт организации передвижных классов, хотя следует отметить, что наиболее острой помимо вопросов приобретения и переоборудования транспортных средств является проблема надежности и работоспособности вычислительной техники. Передвижные компьютерные классы могут обеспечить проведение практических занятий по информатике в пяти-шести ПТУ (350—400 учащихся). Их начальное знакомство с ЭВМ, экскурсии учащихся в передвижные компьютерные классы можно практиковать в период организации оснащения ПТУ вычислительной техникой.

Семинар в Москве

15 января 1987 г. на ВДНХ СССР по инициативе ГУНО, Главпрофобра и МГК ВЛКСМ состоялось совещание-семинар актива молодых преподавателей информатики Москвы. Цель совещания — выработать конкретные предложения на ближайшее время для решения задач компьютерного всеобуча с учетом возможностей города. Вел совещание один из его организаторов — В. Н. Тихоненков, председатель комиссии по ВТ и учебной компьютеризации при МГК ВЛКСМ. Собравшимся было предложено провести совещание в форме дискуссии с жестким регламентом выступлений: от 1 до 3 минут. Такая организация работы была одобрена и позволила, по мнению учительницы школы № 444 Первомайского района Ю. Н. Пахомовой, высказаться всем желающим, заставляя их формулировать свои мысли точно, кратко.

Взволнованно и горячо обсуждали участники совещания наболевшие вопросы. Какой быть информатике в школе? Как оптимально организовать учебный процесс? Какими критериями руководствоваться в своей педагогической деятельности? Какие методы и формы обучения использовать? Как правильно организовать кружковую работу? Как рационально и грамотно использовать уже имеющуюся технику, а если ее нет, то на какую ориентироваться? Правовые отношения между базовым предприятием и школой, присвоение соответствующей квалификации выпускникам школы, распределение дефицитных гибких дисков, отсутствие широкой информации о работе коллег — да разве можно перечислить все вопросы, которые задает себе каждый творчески работающий преподаватель информатики? Но одна проблема протянулась красной нитью на фоне остальных — разработка и тиражирование качественных педагогических программных продуктов.

Сегодня всем очевидна необходимость создания единого координационного центра, в котором бы высококвалифицированные педагоги, психологи, программисты, гигиенисты, дизайнеры, а возможно, и представители других специальностей занимались бы разработкой, апробацией и тиражированием педагогических программных продуктов на научно разработанной основе. Но как непросто преодолеть межведомственные барьеры! И получается, что «лебедь рвется в облака, рак пятится назад, а щука тянет в воду». Ситуация, при которой «воз и ныне там», не устраивает всех, кому по-настоящему дорого дело компьютерного всеобуча. Поэтому на встречу с молодыми педагогами пришли представители институтов АПН СССР, Института информатики АН СССР, НИИ школ МП РСФСР, ГИВЦ Минпроса СССР, Министерства культуры. Они выступили перед собравшимися с информацией по состоянию дел, с предложениями о сотрудничестве. В частности, директор УПК № 3 Бабушкинского района, являющийся одновременно директором ГИВЦ Минпроса СССР И. И. Логвинов проинформировал пользователей ПЭВМ типа «Ямаха» о возможности оказания им консультативной помощи на общественных началах сотрудниками ГИВЦ. Старший научный сотрудник НИИ ШОТСО С. С. Смирнов сообщил о создании экспертной комиссии по аттестации педагогических программных продуктов в рамках Минпроса СССР, при которой предполагается организовать проблемные группы. Он пригласил всех желающих подключиться к этой работе.

В результате обмена мнениями и обсуждения перечисленных проблем был сформулирован ряд предложений. Например, создать в Москве неформальную организацию типа центра или клуба, в которой могли бы участвовать все желающие.

III Всесоюзный...

С 16 по 19 марта 1987 г. в Пущине состоялся III Всесоюзный семинар «Разработка и применение программных средств ПЭВМ в учебном процессе». Он был организован ИПИ АН СССР и НИВЦ АН СССР.

В семинаре участвовали представители АН СССР, Минпроса СССР и минпросов союзных республик, АПН СССР, Госпрофобра и Минвуза СССР и минвузов союзных республик.

Открыл семинар директор НИВЦ АН СССР А. Н. Молчанов. Затем с сообщениями выступили ученый секретарь ИПИ АН СССР В. Н. Захаров, начальник отдела ИПИ АН СССР С. А. Христочевский. Они рассказали о выполнении решения II Всесоюзного семинара, о перспективных задачах ИПИ АН СССР по компьютеризации школы, о содержании комплексного научного проекта «Учебная информатика», выполняемого в рамках сотрудничества АН социалистических стран, и в связи с этим предложили внести изменения в направления работ Всесоюзного семинара.

Представитель Управления ИиВТ Минпроса СССР Т. В. Крюкова рассказала о приемке отечественных КУВТ («Корвет» и УК НЦ), о проекте руководящего документа (РМД1—РМД5) «Педагогические программные средства» (ППС), созданного в НИИ ИиВТ АПН СССР (г. Новосибирск), о планируемых разработках ППС.

Участники семинара получили комплекты НТМД для обсуждения за «круглым столом».

Представители ИПИ АН СССР сообщили о результатах паспортизации кабинетов информатики на базе КУВТ «Ямаха». Паспортизация показала, что в основном все пользователи КУВТ «Ямаха» вели работы по

разработке ПС учебного назначения (инструментальные и прикладные);

подготовке / переподготовке учителей;

созданию программной поддержки курса «Основы информатики и ВТ».

Для развития работ по созданию и применению программных средств учебного назначения, для обеспечения обмена информацией между пользователями КУВТ «Ямаха» и по решению II Всесоюзного семинара в Пущине (22—27 июня 1986 г.) ИПИ АН СССР подготовил к печати Перечень ПС для КУВТ. Перечень ПС получили все участники семинара.

На семинаре были представлены сообщения по следующим направлениям работ:

перспективы развития отечественных учебных ПЭВМ;

разработка инструментальных ПС для КУВТ, опыт разработки и применения ППС в учебном процессе;

проблемы преподавания курса «Основы информатики и ВТ».

С большим вниманием участники семинара заслушали доклады представителей организаций — разработчиков учебных ПЭВМ А. Б. Либерова и И. О. Лозового «Отечественные КУВТ («Корвет» и УК НЦ)». Докладчики продемонстрировали возможности ЭВМ, рассказали о перспективах разработок.

Большой интерес вызвали сообщения В. А. Каймина (МИЭМ), Н. И. Повякель («Горсистемотехника», Киев), Е. В. Утлинского (Ярославский пед. ин-т), И. С. Левина (НТО АН, Ленинград), Е. Э. Елинер.

Были показаны программные средства, разработанные в следующих организациях:

механико-математическом факультете МГУ («Е-практикум-87», «Микромир-86»);

ВЦ СО АН СССР, Новосибирск (текстовой редактор TOR и отладчик DBG);

ИПИ АН и др. (сервисная программа работы с магнитофоном);

Омском государственном пед. институте (редактор шрифтов);

Свердловском инженерно-педагогическом институте (...станок и др.);

ИК АН УССР, Киев (автоматизированный словарь);

НИВЦ АН СССР, Пущино (тренажер по английскому языку);

МЭИ (лаб. работы по физике);

ЛГУ, Рига (лаб. работы по химии);

ВЦ ЛГПИ, Ленинград («Математическая смекалка»).

На семинаре заслушаны сообщения по структуре, методике преподавания и программной поддержке курса «Основы информатики и ВТ» в

ВМК МГУ, Москва (программные средства поддержки курса);

МИФИ, Москва (инструментальные средства для конструирования обучающих программ);

НИИ педагогики Минпроса УССР, Киев (программная поддержка курса);

УГУ, Свердловск (обучающие ППС);

СПИ, Свердловск (передвижной комплекс компьютерного всеобуча);

школе № 278, Ленинград (курс ОИиВТ на основе языка Лого).

За «круглым столом» участники семинара обсудили вопросы о проекте руководящего методического документа ППС; о языке программирования для учебных ПЭВМ; о создании перспективных (на уровне мировых стандартов) моделей учебных ПЭВМ и прикладных программных средств учебного назначения, учитывающих современные требования педагогической науки и рекомендации психологов и эргономистов.

По итогам работы семинара были приняты рекомендации:

Минпросу СССР решить вопрос о дополнительной закупке дискет и кассет для КУВТ «Ямаха» и обеспечить ими пользователей ИКУВТ и БКУВТ;

просить ИПИ АН СССР (отдел учебной информатики) организовать экспертную группу по приемке и аттестации инструментальных программных средств для учебных целей;

ИПИ АН СССР подготовить к 1988 г. предложения по унификации и стандартизации в системах учебной информатики. Участникам семинара передать в ИПИ АН СССР свои предложения;

НИИ ИиВТ АПН СССР организовать оперативное информирование (1 раз в полгода) о разработанных учебных программных средствах в журнале «Информатика и образование»;

разработчикам КУВТ («Корвет» и УК НЦ)

включить в язык Бейсик процедуры, обеспечивающие структурирование программ, а также возможность переопределения символов для нужд национальных языков и некоторых учебных предметов;

считать целесообразным создание временных творческих коллективов, объединяющих работы специалистов по вычислительной технике, программному обеспечению, а также педагогов и эргономистов, для конкурсной разработки перспективных учебных ПЭВМ. Участникам семинара передать свои предложения по архитектуре и базовому программному обеспечению в ИПИ АН СССР;

просить Минпрос СССР обратиться в ГКВТИ СССР для решения вопроса о финансировании инициативных разработок инструментальных и прикладных программных средств учебного назначения и об авторском праве на программные средства.

На конференции «Школьная информатика»

Во Дворце культуры работников просвещения Ленинграда — учителя, ученые, преподаватели вузов, студенты и, конечно, школьники, учащиеся СПТУ. Проходит шестая научно-техническая конференция «Школьная информатика»*. Председатель оргкомитета конференции — академик А. П. Ершов, заместитель председателя — лауреат Государственной премии СССР профессор М. Б. Игнатьев. Работают секции: обучающие программы, системное программное обеспечение, прикладное программное обеспечение, информационные системы, игровые программы, целевые игры.

По представленным докладам просматривается география страны: Таллин, Москва, Йошкар-Ола, Новосибирск, Ульяновск, Свердловск, Владивосток, Тбилиси и др. Какие итоги можно подвести? Прежде всего, значительно вырос уровень работ, возросла их практическая направленность. Так, многие разработки ленинградских школьников выполнены по заказам предприятий Ленинграда и оплачены этими предприятиями, другие — уже нашли применение в школе.

Успехи эти не случайны. За каждым районом города закреплены курирующие вузы, ведущие подготовку учителей, предоставляющие свою вычислительную технику. Отчет вузов-кураторов о внедрении вычислительной техники в учебный процесс школ, СПТУ и техникумов Ленинграда был заслушан на конференции. За каждой школой закреплены также базовые предприятия, чьи заказы по-

* Конференция состоялась в марте этого года.

Участники конференции в кабинете вычислительной техники Центра школьной информатики Октябрьского района, где они продемонстрировали свои программы, поиграли в компьютерные игры

Школа юного программиста Ленинградского института авиационного приборостроения

зволяют сделать труд школьников-программистов нужным и полезным обществу. При Совете ректоров города созданы комиссия «Школьная информатика», координирующая работу по компьютеризации обучения, и центры школьной информатики — городской и районные.

На конференции поставлен ряд проблем. Прежде всего, необходимо совершенствовать систему отбора представленных на конкурс работ. Стоит подумать и о том, чтобы через оргкомитет выявлять наиболее перспективные, актуальные работы и рекомендовать их в фонд алгоритмов и программ для накопления программного продукта. Многие из представленных школьниками пакетов обучающих программ по курсам физики, математики, химии, русского и иностранного языков, музыки уже используются в учебном процессе. Но выяснилось, что школы не хотят «делиться» своими разработками, опасаясь за авторство. Нужно координировать работу школьных программистов, подумать о защите авторских прав, избегать дублирования работ, чтобы труд школьников стал более эффективным и полезным.

И конечно же двух дней на проведение конференции явно недостаточно.

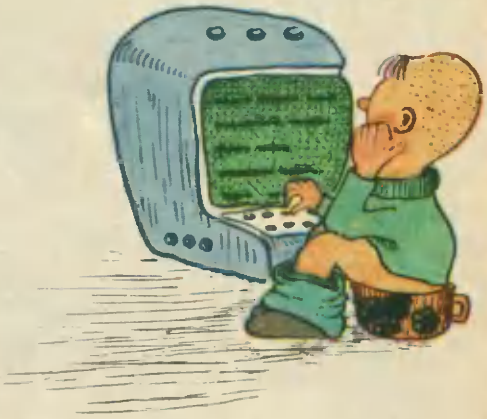
Идет пленарное заседание. На трибуну выходит академик А. П. Ершов. Зал встречает его восторженно. Андрей Петрович отвечает на многочисленные вопросы школьников. Ребят интересуют стоимость и надежность вычислительной техники, перспективы развития персональных школьных ЭВМ, методика обучения и многое другое... Наступает момент награждения. Вручает награды член-корреспондент АН СССР лауреат Ленинской премии С. С. Лавров. Дипломами оргкомитета конференции отмечены 68 работ. В зале немало и огорченных лиц. И понять это огорчение можно, но выбирали самые лучшие работы.

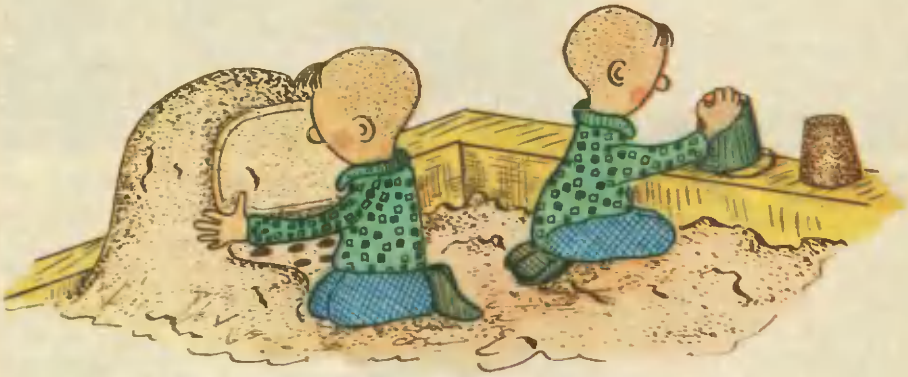
Ко времени проведения конференции закончила работу проводимая в виде игры «Ваш ход, компьютер» заочная школа юных читателей газеты «Ленинские искры». Победителям игры тоже вручены награды на конференции. Главный приз — МК-61 — получил ленинградский семиклассник Женя Данилушкин. В числе призеров игры есть и третьеклассник, и четвероклассники. Интересно, что среди докладчиков на конференции «Школьная информатика» вот уже второй год ученики III—VI классов.

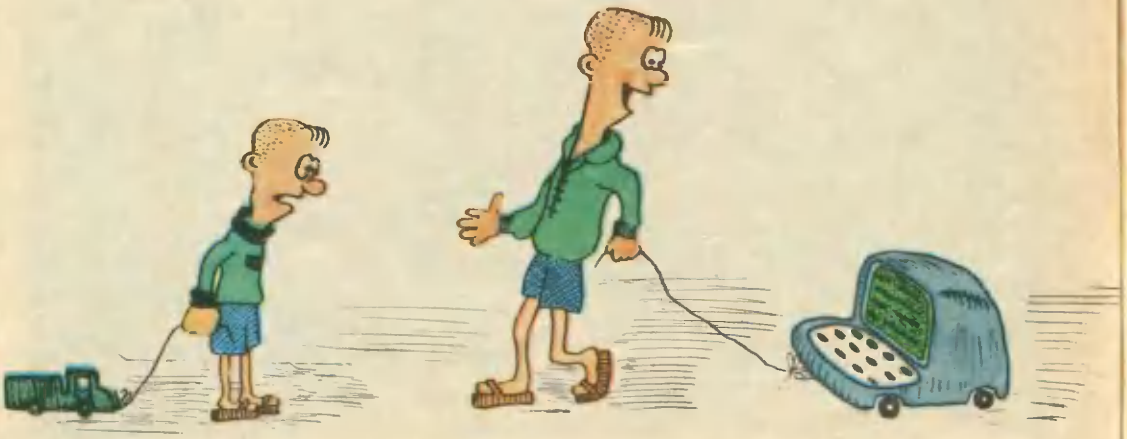
Заканчивает работу конференция. Пустеют залы и коридоры Дворца культуры, бывшего Юсуповского дворца. До встречи в следующем году! Творческих удач вам, ребята!

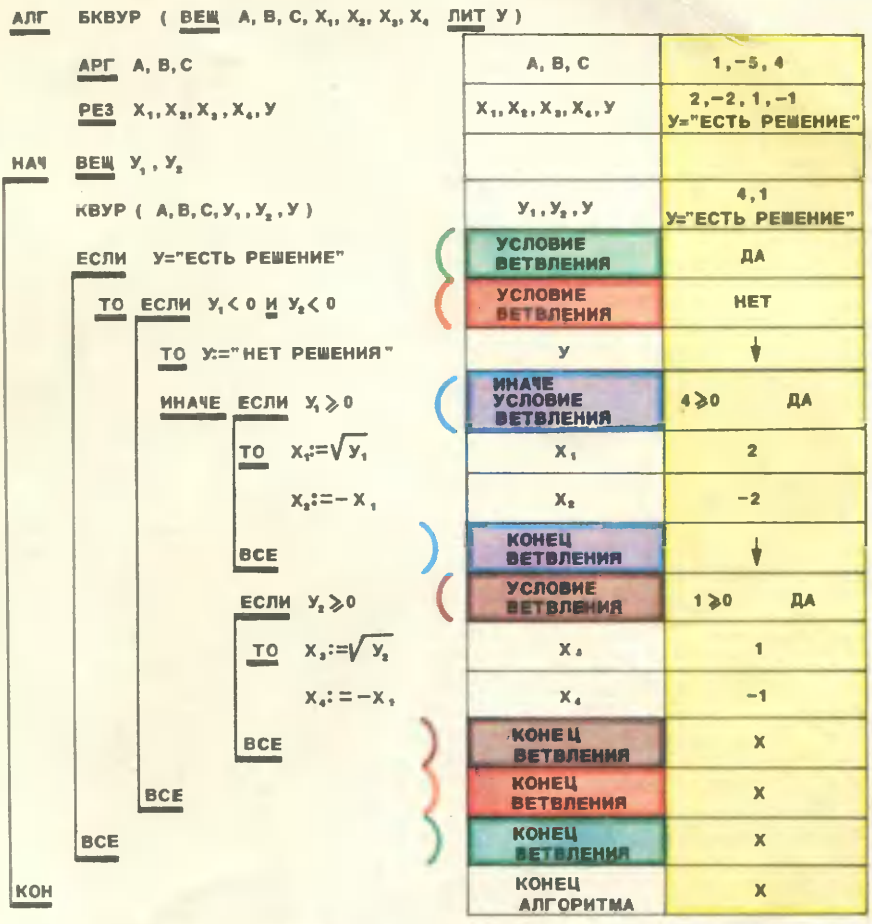
Г. Бровина

Весёлый урок









На второй и третьей страницах обложки — иллюстрации к статье Т. Кожантаева «Таблицы значений исполнения алгоритмов».

Цена 60 коп.

70423

ИНФО

4 '87

ИНФОРМАТИКА И ОБРАЗОВАНИЕ

Информатика и образование, 1987, № 4, 1—128