ПЕРСОНАЛЬНЫЙ БЫТОВОЙ КОМПЬЮТЕР «ЭРУДИТ—002»

ПАСПОРТ

Часть 2

РАБОТА С ПЕРСОНАЛЬНЫМ БЫТОВЫМ КОМПЬЮТЕРОМ «ЭРУДИТ-002»

1991

ПЕРСОНАЛЬНЫЙ БЫТОВОЙ КОМПЬЮТЕР «ЭРУДИТ—002»

ПАСПОРТ

Часть 2

РАБОТА С ПЕРСОНАЛЬНЫМ БЫТОВЫМ КОМПЬЮТЕРОМ «ЭРУДИТ-002»

1991

ОГЛАВЛЕНИЕ

ПРЕДИСЛОВИЕ

Персональный бытовой компьютер «ЭРУДИТ-002» предназначен в основном для учебных целей и проведения досуга, но может быть использован для вычислений, организации персональной информационной системы, а также в других сферах деятельности. Для работы с компьютером необходимы цветной монитор (цветной телевизор) (при использовании монохроматического монитора значительно уменьшается информативность компьютера) и кассетный магнитофон: в процессе работы с компьютером на экран монитора выводится информация, а магнитофон используется в качестве внешней памяти. К «ЭРУДИТУ» также может быть подключено печатающее устройство. Отметим, что «ЭРУДИТ» имеет команды управления файлами микродрайвов: LINE, OPEN, CLOSE, MOVE, ERASE и CATALOGUE.

Бытовой компьютер «ЭРУДИТ» программно совместим с таким бытовым компьютером, как ZX Spectrum. Отметим, что в ряде социалистических стран созданы аналогичные компьютеры, общим признаком которых является наличие микропроцессора Z80A (США) или его аналога.

В паспорте не рассматриваются те возможности «ЭРУДИТА», которые широко не используются и представляют интерес только для узкого круга специалистов. Например, подробно не рассматривается возможность функции IN при опросе клавиатуры и др.

В первой главе книги Вы ознакомитесь с принципом действия компьютера. Во второй главе даются элементарные основы программирования на языке БЕЙСИК, а третья глава — как бы справочник языка БЕЙСИК, в котором конкретно указывается, что каждый оператор или функция может выполнить.

Глава I.

Знакомство с «Эрудитом»

1.1. КАК ВЫГЛЯДИТ «ЭРУДИТ».

«ЭРУДИТ» применяется в комплекте с цветным телевизором и кассетным магнитофоном.

Блок питания выполнен в виде отдельного блока и включается в сеть 220 В.

На рис. 1 показано расположение клавиатуры и гнезд коммутации «ЭРУДИТА». На верхней панели компьютера расположена клавиатура.

На рисунке:

1 — клавиатура;

2 — кнопка сброса (RESET);

3 — подключение магнитофона;
4 — подключение джойстика;
5 — подключение напряжения питания (5V);
6 — выход RGB сигнала;
7 — «Видео».

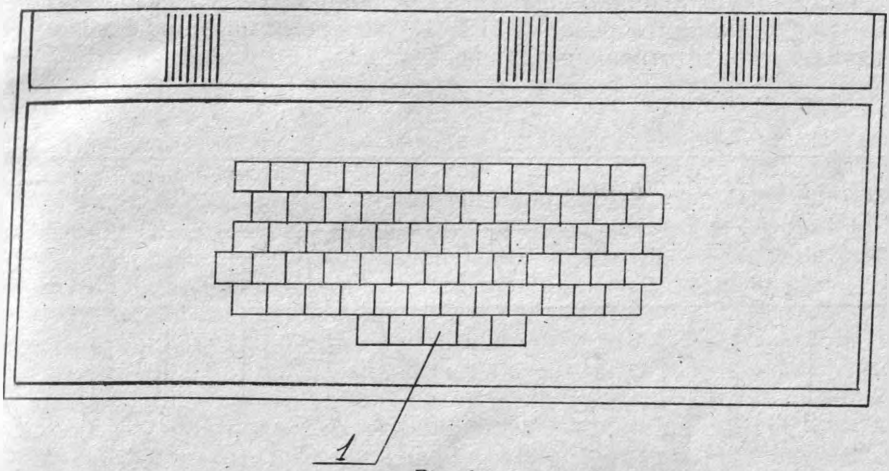


Рис. 1

8 — отверстия под регулировку цветов;
9 — переключение джойстика прямое/инверсное;
10 — изменение изображения на ТВ пр/инв.

Назначение основных клавиш управления пояснено во второй главе книги. На задней стенке «ЭРУДИТА» находятся все коммутационные гнезда и кнопка сброса RESET. Гнезда предназначены для подключения напряжения питания телевизора, магнитофона и других устройств.

1.2. НАСТРОЙКА ТЕЛЕВИЗОРА.

Компьютер «ЭРУДИТ» обычно подключается к цветному телевизору. Чтобы на экране была видна формируемая компьютером картинка, в телевизоре надо установить соответствующий канал, если подключение происходит через антенное гнездо.

Схема устройства сопряжения с цветным телевизором по высокой частоте потребителю не поставляется. В случае неисправности устройства обращайтесь на завод-изготовитель по адресу, указанному в I части паспорта.

В случае подключения «ЭРУДИТА» к цветному телевизору че-

рез RGB-вход необходимо соединить кабелем (согласно части I паспорта) RGB-выход компьютера с RGB-входом телевизора. Телевизор без RGB-входа необходимо доработать (согласно части I паспорта).

Включив питание компьютера и соединив его с телевизором, надо нажать кнопку сброса RESET. В это время на экране должна появиться приветственная надпись (см. рис. 2).

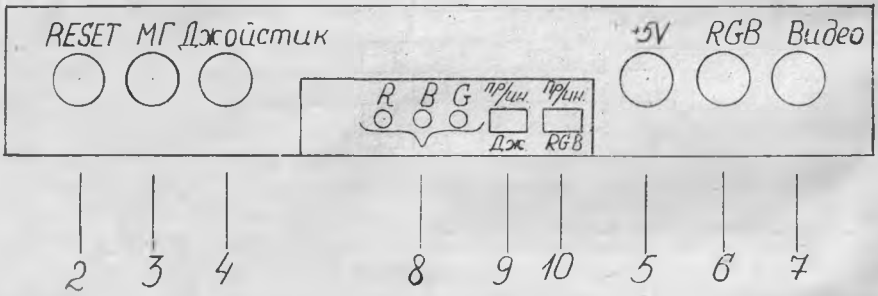


Рис. 2

Получив такое изображение, Вы можете проверить формируемые «ЭРУДИТОМ» цвета и тут же начать работу. Цвета проверяются следующим образом: нажмите клавишу с командой BORDER, а затем какое-нибудь число от 1 до 6. Приветственная надпись исчезает, и появляется слово BORDER, а после него число. Теперь нажмите клавишу, на которой написано ENTER. Край экрана—контур окрасится в цвет, который указан на цифровой клавише. Если цвета плохие, надо попробовать улучшить их с помощью ручек настройки цветов, находящихся на задней стенке компьютера, а затем ручек регулировки цветов телевизора.

При подключении к RGB-входу телевизора может получиться так, что все цвета окажутся противоположными тем, которые должны быть. Для устранения этого дефекта служит кнопка «инверсия RGB» на задней панели компьютера.

1.3. ЧТО МОЖЕТ ВЫПОЛНЯТЬ «ЭРУДИТ».

Теперь, когда компьютер включен и телевизор отрегулирован,

попробуйте нажать несколько любых клавиш. На экране появятся слова, буквы и цифры.

Теперь нажмите кнопку сброса RESET и можете начинать работу. Ниже приведено несколько примеров, которые на экране телевизора демонстрируют возможности «ЭРУДИТА».

1.3.1. *Как ввести и запустить программу.*

Каждая последовательность команд представляется в виде списка, который называется листингом. Вы заметите, что листинги программ складываются из частей, каждая из которых начинается номером 10, 20 и т. д. Каждая такая часть называется строкой программы (даже если она на экране занимает две или больше строк), в которой содержится одна или больше команд компьютеру.

В каждой строке программы Вы увидите полные слова или сокращения из двух или более букв, например, PRINT, LET, RND, PI, PAPER, GO TO. Они называются ключевыми словами, и Вы можете ввести их, нажимая букву за буквой. Надо просто найти клавишу, на которой написано нужное слово (например, PRINT написано на клавише P), и дальше все делать по правилам, приведенным в подразделе «Как вводить».

При вводе строки ее изображение появляется в нижней части экрана. Закончив ввод всей строки, нажмите клавишу ENTER. Теперь строка появится в верхней части экрана и одновременно запишется в память компьютера. Таким образом введите все строки. Если случайно нажали не ту клавишу, прочитайте подраздел «Как исправлять ошибки».

После ввода всех строк, нажмите клавишу R: на экране появится слово RUN. Теперь нажмите клавишу ENTER и «ЭРУДИТ» начнет выполнять введенную программу.

1.3.2. *Как вводить.*

Чтобы ввести какое-нибудь ключевое слово (или символ), прежде всего найдите клавишу, на которой написано это слово (или символ). После этого нажмите клавиши в той последовательности, как показано на рис. 3.

Полное описание, как пользоваться клавиатурой, приведено во второй главе.

Надпись сверху.
Нажмите клавишу.

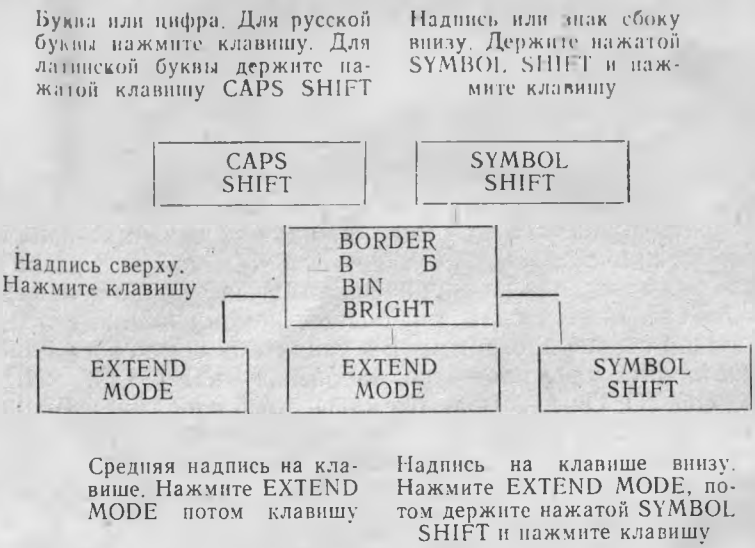


Рис. 3

1.3.3. *Надписи.*

```
10 BORDER 5
20 INK RND×7: PAPER RND×7
30 PRINT «ИЗМАИЛ»:
40 GO TO 20
```

Надпись «ИЗМАИЛ», написанная различными цветами, заполнит весь экран. В это время компьютер остановится, и в нижней части экрана появится надпись «ДАЛЬШЕ?». Чтобы изображение передвинулось вверх и можно было бы дальше печатать надписи, нажмите любую клавишу, за исключением N, ПРОБЕЛ, BREAK и STOP. Если Вы остановили программу, нажав указанные клавиши, то Вы можете ее запустить, нажимая клавишу RUN, а затем — ENTER.

Попробуйте в строке под номером 30 надпись «ИЗМАИЛ» заменить какой-нибудь другой надписью, например:

30 PRINT «ВИКТОРИЯ»:

Но не забудьте в конце поставить точку с запятой. Вместо надписи «ИЗМАИЛ» теперь по всему экрану будет воспроизведено указанное Вами имя.

1.3.4. *Как поменять программу.*

Подождите пока программа закончится или остановите ее, нажав клавишу BREAK. После этого нажмите клавиши в такой последовательности: CLS, ENTER, LIST и ENTER.

На экране появится листинг программы (список строк).

Посмотрите, которую строку Вы хотите поменять, а затем введите всю новую строку, не забывая и номер строки. В конце строки нажмите клавишу ENTER. Новая строка заменит старую.

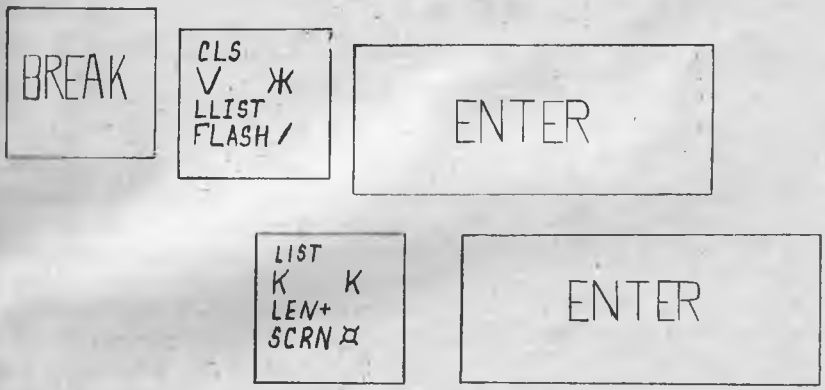


Рис. 4

Последовательно нажмите клавиши RUN и ENTER: компьютер начнет выполнять новую программу. Позже увидим, что менять программу можно и иначе — путем редактирования.

1.3.5. *Как запустить программу сначала.*

Некоторые из представленных здесь программ, например, ФЛАГ заканчиваются сами, выдавая сообщение О ВЫПОЛНЕНО с ука-

занным номером последней строки программы. Это значит, что вся программа выполнена. Чтобы запустить программу с начала, необходимо нажать RUN и ENTER.

Другие программы или выполняются все время, или сами автоматически запускаются с начала.

Такие программы останавливаются нажатием клавиши BREAK. Держите эту клавишу нажатой до тех пор, пока программа не остановится и будет выдано сообщение BREAK. Чтобы снова запустить программу, нажмите клавиши RUN и ENTER.

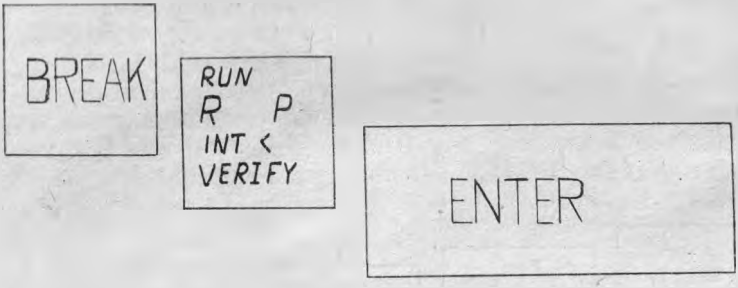


Рис. 5

1.3.6. *Как ввести новую программу.*

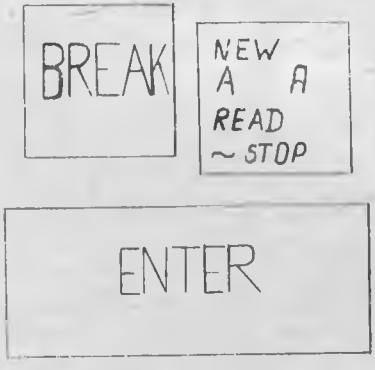


Рис. 6

Если Вы полностью проверили одну программу и хотите ввести совершенно новую, подождите, когда первая закончится или остановите ее, нажав клавишу BREAK.

Стереть старую программу из памяти компьютера можно двумя способами: первый способ — нажать последовательно клавиши NEW и ENTER, второй способ — просто нажать клавишу сброса RESET.

В обоих случаях экран станет черным, а затем появится приветствие.

1.3.7. *Узоры.*

```
10 LET N=INT (RND×11+3):DIM S    (N, I)
20 FOR I=1 TO N
```

```
30 LET S$(I)=CHR$ (RND×15+128)
40 NEXT I
50 LET P=INT (RND×4)
60 INK P+4:PAPER P
70 LET I=1:PRINT AT 0,0;
80 FOR J=1 TO 704
90 PRINT (1);
100 LET I=I+1
110 IF I>N THEN LET I=1; BEEP .02, J/16-15
120 NEXT J: GO TO 10
```

После запуска программы с помощью клавиш RUN экран заполнится цветным узором. Во время индикации узора будет слышен изменяющийся звуковой сигнал. Затем компьютер с верхней части экрана начнет формировать новый узор, потом еще один и т. д. Программа сама не заканчивается, остановить ее можно, нажав клавишу BREAK.

Попытайтесь в последней строке 120 указанное число 10 заменить на 50. Теперь на экране все время будет тот же рисунок, но будет изменяться его цвет.

1.3.8. *Двухцветная мозаика.*

```
10 BORDER 0: PAPER 5: CLS
20 INPUT «KX»; KX, «KY:»; KY, «D:»; D
30 OP E=0 TO 21
40 ET X=—KX+E*/22
50 OP =0 TO 31
60 ET У=—KУ+ */22
70 ET C=1 T (X*X+У*У)/2
80 IF C=INT (C) THEN PRINT AT E,S; PAPER 2; " "
90 NEXT S: NEXT E
```

После запуска программы в нижней части экрана индицируется надпись «KX:». Это компьютер просит Вас ввести какое-нибудь число. Введите, например, число 20, после чего не забудьте нажать клавишу ENTER. В данном случае сбоку появится другая надпись — «KY:». Опять таким же образом введите какое-нибудь число, например, 30. Компьютер попросит ввести еще одно число, индицируя надпись «D:». Введите, например, 5. Теперь компьютер начнет рисовать на экране мозаику из красных и белых квадратиков. Запустите программу снова и введите другое число. На экране появится другой рисунок.

Замените цвета мозаики, изменяя в 10 и 80 строках числа в пределах от 0 до 7.

1.3.9. *Мелодия.*

```
 10 REM П. И. Чайковский «ВО ПОЛЕ БЕРЕЗА СТОЯЛА»
 15 INPUT «ЗАДАЙТЕ ТЕМП»; T:LET T=60/4/T
 20 FOR I=1 TO 4: BEEP T,7:NEXT I
 30 BEEP 2*T,5:BEEP T,3:BEEP T,3
 40 BEEP 2*T,2:BEEP 2*T,0
 50 BEEP T,7:BEEP T,7:BEEP T,10:BEEP T,7
 60 BEEP T,5:BEEP T,5:BEEP T,3:BEEP T,3
 70 BEEP 2*T,2:BEEP 2*T,0
 80 FORI=1 TO 19:READA:READ B:BEEP A,B:NEXT I
 90 DATA 0,257,0.25,7,0.25,7,0.25,7,0.5,5,0.25,3,0,0.25,3
100 DATA 0.5,2,0.5,0,0.25,7,0.25,7,0.25,10,0.25,7,0.25,5,
         0.25,5,0.25,3,0.25,3,0.25,3,0.5,2,0.5,0
```

Компьютер воспроизводит мелодию широко известной песни «Во поле береза стояла». В начале программы «ЭРУДИТ» запрашивает темп исполнителя. Чем меньше будет число, тем быстрее он проиграет мелодию.

1.3.10. *Многоугольники.*

```
 10 BORDER 00: PAPER 6: CLS
 20 INPUT «ЧИСЛО УГЛОВ:»:N
 30 FOR R=0 TO 5
 40 LET X1=128: LET Y1=172
 50 PLOT X1, Y1
 60 FOR I=1 TO N
 70 LET X2=128+84*SIN (I*2 PI/N)
 80 LET Y2=88+84*COS (I*2*P1/N)
 90 DRAW PAPER R: INK 9: X2—X1, Y2—Y1
100 LET X1=X2: LET Y1=Y2
110 BEEP .05,R*Y1/18
120 NEXT I
130 BEEP 1,R*15—30
140 NEXT R
```

После запуска программы внизу экрана индицируется надпись «ЧИСЛО УГЛОВ.». Компьютер ждет, пока Вы укажете число углов многоугольника. Нажмите, например, клавиши 6 и ENTER. На экране вырисовывается шестиугольник, а затем несколько раз меняется цвет его сторон. Когда программа закончится, запустите ее вновь и введите какое-нибудь число.

Попробуйте в 90 строке между словами DRAW и PAPER вклю-

нить FLASH1. В этом случае вычерченные многоугольники будут мигать.

1.3.11. *Как исправить ошибки.*

Если Вы нажали не на ту клавишу или перед этим не нажали (в случае необходимости) клавишу SHIFT или EXTEND MODE, не отчаивайтесь. Нажмите клавишу DELETE, и последнее ключевое слово (знак, буква или цифра) исчезнет. Если хотите убрать и другие символы, держите клавишу DELETE в нажатом положении.

Если Вы сделали в строке ошибку и, не исправив ее, нажали клавишу ENTER, перед ошибкой может появиться мигающий знак вопроса. Тут же, нажав клавишу DELETE, ликвидируйте часть строки до ошибки и саму ошибку и, правильно набрав строку до конца, нажмите клавишу ENTER.

Если Вам все-таки удалось с помощью ENTER ввести строку с ошибкой, то программа может остановиться, а в нижней части экрана появится сообщение, в котором указан номер неверной строки. Тогда наберите всю эту строку заново и без ошибок. Теперь нажмите клавиши ENTER, RUN и ENTER: программа будет работать. Как редактировать, т. е. исправлять ошибки, не набирая заново всей строки, пояснено ниже.

1.3.12. *Флаг СССР.*

```
10 INK 6: PAPER 0: CLS
20 FOR I=1 TO 22
30 FOR=1 TO 32
60 PRINT PAPER 2; « »;
70 NEXT J
80 NEXT I
90 PLOT 45,164: DRAW—7,—21
100 DRAW 18 —12: DRAW—7,21
120 PLOT 40,139: DRAW—9,—9
130 DRAW 3,—3; DRAW4,4
140 DRAW 22, —22: DRAW 3,3
150 DRAW —22,22: DRAW 5,5
```

```
160 DRAW —6,0
170 PLOT 31,121: DRAW —7,—7
180 DRAW 3,—3: DRAW 6,6
190 PLOT 31,121: DRAW 23,23,164*PI/180
200 PLOT 33,171:DRAW 21,27,175*PI/180
```

1.3.13. *Переплетающиеся полосы.*

```
 10 BORDER 0: PAPER 0: CLS
 20 LET Z=1
 30 LET EP=0: LET EG=21
 40 LET SP=1: LET SG=30
 50 FOR I=1 TO 2
 60 LET RP=INT (RND*8)
 70 FOR J=1 TO 11
 80 FOR E=EP TO EG STEP Z
 90 PRINT AT E,SP—Z; PAPER R; « »
100 NEXT E
110 FORS=SP TO SG STEP Z
120 PRINT AT EG,S; PAPER R; « »
130 NEXT S
140 LET X=EP: LET EP=EG—Z: LET EG=X+Z
150 LET X=SP: LET SP=SG—Z: LET SG=X+Z
160 LET Z=—Z: NEXT J
170 LET Z=—1
180 LET EP=21: LET EG=0
190 LET SP=30: LET SG=1
200 NEXT I: G0 TO 20
```

Запустив программу, на экране наблюдаем вращающиеся в разные стороны разноцветные полосы. Программу можно остановить посредством нажатия BREAK клавиши.

Вставьте следующие строки:

```
 95 BEEP .01,E+J*1
125 BEEP .01,S+J*1
```

Теперь во время вращения полос будут слышны меняющиеся звуки.

1.3.14. *Баскетбол.*

```
 10 INK 6: PAPER 4: BORDER 4: CLS
 20 FOR K=0 TO P1 STEP P1/400
 30 LET X=42*SIN (K): LET Y=42*COS (K)
```

```
40 PLOT 128—X,133—Y: DRAW 2*X, 2*Y
50 NEXT K
60 FOR 1=1 TO 6
70 READ X1, Y1, X2, Y2
80 PLOT X1, Y1: DRAW OVER 1; X2, Y2, K*P1/180
90 NEXT 1
100 DATA 163,109,—74,40,87,102,100,41,72,68,86,130,35,44,
         129,170,143,22,47,103,106,0,—28,83,42,177,83,—28,
         —83,42
110 FOR 1=1 TO 16
120 READ X1, Y1, X2, Y2
130 PLOT X1, Y1: DRAW X2, Y2
140 NEXT 1
150 DATA 78,84,99,0,78,83,99,0,86,73,12,10,93,61,25,22,
         99,47,39,36,104,32,54,51,106,14,72,69,113,0,42,41,
         133,0,16,16,158,83,11,—10,138,83,24,—23,118,83,38,
         —37,98,83,54,—53,78,83,71,—69,102,39,40,—39,106,16,
         16,—16
160 FOR 1=0 TO 21
170 FOR J=1 TO 23 STEP 22
180 PRINT AT 1,J; PAPER RND*7; INK 9; «ДИНАМО»
190 NEXT J: NEXT I
```

После ввода программы на экране появятся баскетбольный мяч и корзина, а затем по сторонам экрана — надпись «ДИНАМО».

Замените в 180 строке слово «ДИНАМО» словом «СПАРТАК», или каким-нибудь другим словом, состоящим не более чем из 8 букв.

1.3.15. *Что дальше?*

Вы испробовали несколько программ. Если Вы хотите использовать их в будущем, можете записать их на магнитную ленту. Для этого посмотрите раздел «Как сохранить свои программы».

Если Вы хотите дальше экспериментировать с «ЭРУДИТОМ», то прочитайте о программировании во второй главе «Введение в программирование». В ней пояснены некоторые особенности программирования.

1.4. КАК ИСПОЛЬЗОВАТЬ ПОДГОТОВЛЕННОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

При вводе программы в «ЭРУДИТ» посредством нажатия клавиш Вы создаете последовательность электронно-закодированных

сигналов. Коды записываются в компьютерную память, где они и хранятся. Коды остаются в памяти до тех пор, пока Вы их или сотрете (например, вводя команду E или нажимая клавишу сброса RESET) или выключите «ЭРУДИТ».

Однако не всегда, когда Вы хотите работать с «ЭРУДИТОМ», обязательно вводить программу с клавиатуры. Вы можете использовать уже готовые программы, которые могут быть подготовлены непосредственно или автоматически введены в компьютер. Использование подготовленного программного обеспечения не только освобождает Вас от длительной работы по вводу программ с клавиатуры, но также позволяет Вам иметь библиотеку готовых программ, для подготовки которых Вы потратили бы несколько дней, а то и недель. Разработчики программного обеспечения готовят программы различного назначения, которые Вы потом можете использовать.

1.4.1. *Вопросы и ответы по программному обеспечению.*

Что называется программным и аппаратным обеспечением?

Программное обеспечение — это совокупность средств, облегчающих работу пользователя на компьютере. Сюда относится библиотека программ, написанная специально для «ЭРУДИТА», «Синклера» и т. д. Аппаратное обеспечение — это техническое оборудование, то есть все устройства, участвующие в работе «ЭРУДИТА».

Что называется загрузкой программы?

Закодированный сигнал записывается на магнитную ленту. Когда Вы прослушиваете записанное на ленту программное обеспечение, магнитофон воспроизводит последовательность сигналов, которая и составляет программу. Вам требуется только соединить кассетный магнитофон с «ЭРУДИТОМ», и кодовая система будет готова к непосредственной записи в память «ЭРУДИТА». Такой процесс называется загрузкой программы.

Почему программное обеспечение записывается на кассеты?

Кассеты удобны в эксплуатации, они не требуют специальных устройств. Для ввода программ, которые на них записаны, Вам необходим только недорогой кассетный магнитофон.

Какой кассетный магнитофон больше всего подходит?

Для работы с «ЭРУДИТОМ» больше всего подходит портативный кассетный магнитофон (моно или стерео) с питанием от электросети, имеющий линейный вход, телефонный выход и не имеющий автоматического шумоподавителя.

Нужен ли уход за программами, записанными на ленту?

Как и любой другой вид магнитной памяти кассеты с записанными программами могут быть повреждены воздействием сильного магнитного поля. Следовательно, не держите их вблизи приборов, потребляющих большие токи. Кассеты с программами также следует оберегать от пыли и влаги.

Годится ли любой вид программного обеспечения?

Нет. Годится только программное обеспечение, написанное специально для «ЭРУДИТА», «СИНКЛЕРА» и т. д.

1.4.2. *Как подключить кассетный магнитофон.*

Перед включением или выключением магнитофона следует вынуть кассету из магнитофона, чтобы она случайно не испортилась.

Перед загрузкой программы с ленты гнезда магнитофона «Линейный выход» и вход с микрофона должны быть соединены кабелем «№ 2» с гнездом компьютера «МГ».

1.4.3. *Как загрузить программу.*

Теперь, когда кассетный магнитофон подключен к «ЭРУДИТУ», все готово к загрузке и записи программы. Вы можете использовать ленту с готовой программой или со своими программами. В обоих случаях следует выполнить те же действия.

Включите кассетный магнитофон. Убедитесь, что «ЭРУДИТ» тоже включен, и вставьте кассету в магнитофон. Если компьютер выполняет ранее введенную программу, то подождите или остановите ее нажатием клавиши BREAK. Теперь для того, чтобы удалить эту программу из памяти «ЭРУДИТА», можете ввести команду NEW или нажать клавишу сброса RESET, но это необязательно, так как перед загрузкой новой программы память очищается сама. Поэтому необходимо помнить, что, если Вы загружаете новую программу, старая стирается.

Далее выполните операции, приведенные ниже. Если что-то не

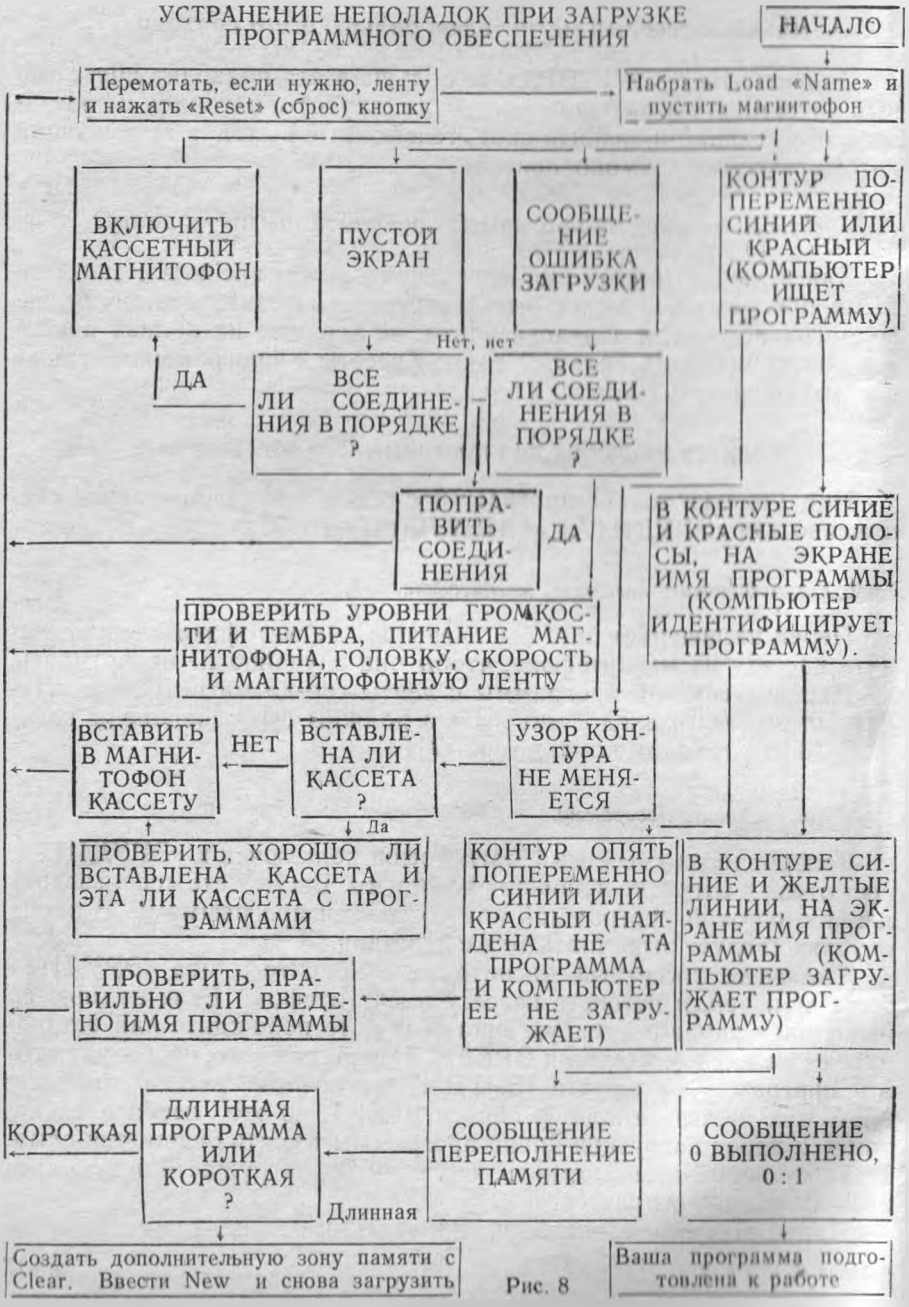

УСТРАНЕНИЕ НЕПОЛАДОК ПРИ ЗАГРУЗКЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ
НАЧАЛО
Перемотать, если нужно, ленту и нажать «Reset» (сброс) кнопку
Набрать Load «Name» и пустить магнитофон
ВКЛЮЧИТЬ КАССЕТНЫЙ МАГНИТОФОН
ПУСТОЙ ЭКРАН
СООБЩЕНИЕ ОШИБКА ЗАГРУЗКИ
КОНТУР ПОПЕРЕМЕННО СИНИЙ ИЛИ КРАСНЫЙ (КОМПЬЮТЕР ИЩЕТ ПРОГРАММУ)
Нет, нет
ДА
ВСЕ ЛИ СОЕДИНЕНИЯ В ПОРЯДКЕ ?
ВСЕ ЛИ СОЕДИНЕНИЯ В ПОРЯДКЕ ?
ПОПРАВИТЬ СОЕДИНЕНИЯ
ДА
В КОНТУРЕ СИНИЕ И КРАСНЫЕ ПОЛОСЫ, НА ЭКРАНЕ ИМЯ ПРОГРАММЫ (КОМПЬЮТЕР ИДЕНТИФИЦИРУЕТ ПРОГРАММУ).
ПРОВЕРИТЬ УРОВНИ ГРОМКОСТИ И ТЕМБРА, ПИТАНИЕ МАГНИТОФОНА, ГОЛОВКУ, СКОРОСТЬ И МАГНИТОФОННУЮ ЛЕНТУ
ВСТАВИТЬ В МАГНИТОФОН КАССЕТУ
НЕТ
ВСТАВЛЕНА ЛИ КАССЕТА ?
УЗОР КОНТУРА НЕ МЕНЯЕТСЯ
Да
ПРОВЕРИТЬ, ХОРОШО ЛИ ВСТАВЛЕНА КАССЕТА И ЭТА ЛИ КАССЕТА С ПРОГРАММАМИ
КОНТУР ОПЯТЬ ПОПЕРЕМЕННО СИНИЙ ИЛИ КРАСНЫЙ (НАЙДЕНА НЕ ТА ПРОГРАММА И КОМПЬЮТЕР ЕЕ НЕ ЗАГРУЖАЕТ)
В КОНТУРЕ СИНИЕ И ЖЕЛТЫЕ ЛИНИИ, НА ЭКРАНЕ ИМЯ ПРОГРАММЫ (КОМПЬЮТЕР ЗАГРУЖАЕТ ПРОГРАММУ)
ПРОВЕРИТЬ, ПРАВИЛЬНО ЛИ ВВЕДЕНО ИМЯ ПРОГРАММЫ
КОРОТКАЯ
ДЛИННАЯ ПРОГРАММА ИЛИ КОРОТКАЯ ?
СООБЩЕНИЕ ПЕРЕПОЛНЕНИЕ ПАМЯТИ
СООБЩЕНИЕ 0 ВЫПОЛНЕНО, 0 : 1
Длинная
Создать дополнительную зону памяти с Clear. Ввести New и снова загрузить
Ваша программа подготовлена к работе

Рис. 8

удалось, смотрите структурную схему «Устранение неполадок при загрузке программного обеспечения» (см. рис. 8).

1. Вставьте кассету в магнитофон и перемотайте ее до начала.

2. Нажмите клавишу LOAD. На экране появится слово LOAD. Тогда наберите имя программы в кавычках, например: LOAD «PROG 1».

Теперь экран телевизора будет выглядеть, как показано на рис. 9.

Компьютер будет искать на ленте программу с указанным названием. Если названия программы между кавычками не будет, то компьютер начнет загружать первую попавшуюся ему программу. В этом случае следует набрать такую команду: LOAD""

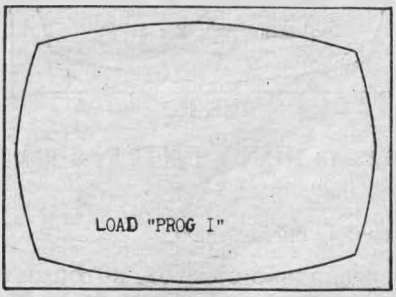


Рис. 9

3. Нажмите клавишу ENTER. Экран очистится.

4. Включите магнитофон. Контур экрана начнет окрашиваться то в красный, то в голубой цвет. Это указывает на то, что «ЭРУДИТ» выбирает программу.

5. Когда по контуру экрана покажутся красные и голубые полосы, скользящие вверх или вниз, это будет свидетельствовать о том, что «ЭРУДИТ» начинает принимать сигнал.

6. На экране появляется слово ПРОГРАММА:, после которого следует имя программы; или БАЙТЫ:, после которого следует название или буква. Это значит, что компьютер удачно выбрал программу.

7. Появление вновь красных и голубых полос говорит о том, что компьютер ждет начала загрузки программы.

8. По контуру появляется рисунок из желтых и синих линий. Это показывает, что «ЭРУДИТ» считывает программу из кассеты. Если программа очень длинная, загрузка может продолжаться несколько минут.

9. Если программа разбита на секции, то 7, 8 и 9 операции могут повторяться по нескольку раз.

10. После считывания программа может начать выполняться автоматически. Не забудьте при этом выключить магнитофон.

11. Если после считывания программы она автоматически не выполняется, то появляется сообщение 0 ВЫПОЛНЕНО, 0:1. Остановите магнитофон.

Рис. 10

12. Нажмите клавиши RUN и ENTER, и программа начнет выполняться.

1.4.4. *Советы по программному обеспечению.*

Ниже приведено несколько советов, которые помогут Вам сэкономить время.

1. На футляре каждой кассеты аккуратно напишите названия всех входящих в нее программ в той последовательности, в какой они записаны на листе. Названия программ запишите полностью, так как они должны предлагаться компьютеру.

2. Если Ваш магнитофон имеет счетчик ленты, то применяйте его для быстрого нахождения программы на ленте.

Установите счетчик в нулевое положение в начале ленты, после чего введите LOAD с названием программы, которой нет на этой ленте. Включите магнитофон. «ЭРУДИТ» синдицирует на экране названия всех программ, не загружая их в память. Запишите показания счетчика ленты рядом с названием соответствующей программы. Это даст Вам возможность в любой момент быстро найти нужную программу.

3. Если лента перемотана до нужной программы или Вы не знаете названия этой программы, введите команду LOAD" ". Между кавычками не должно быть пробела. «ЭРУДИТ» загрузит первую найденную программу. Если на экране появится название не той программы, которая Вам необходима, нажмите клавишу BREAK, перемотайте ленту до следующей программы и попробуйте все с начала.

2.1. КЛАВИАТУРА — ПАНЕЛЬ УПРАВЛЕНИЯ ВАШИМ КОМПЬЮТЕРОМ

Чтобы «ЭРУДИТ» слушался Ваших команд, в него надо ввести программу, написанную на каком-нибудь из языков программирования. Обычно транслятор программного языка, т. е. переводчик на понятный машине язык, хранится на магнитной ленте, а перед использованием загружается в память машины. Но транслятор одного языка — БЕЙСИКА — записан в постоянную память, поэтому программу, написанную на этом языке, можно вводить с клавиатуры или ленты сразу после включения компьютера. Это основной программный язык «ЭРУДИТА». Версия языка БЕЙСИКА, которая используется в «ЭРУДИТЕ», простая, но мощная. Она (насколько это позволило) приближена к разговорному английскому языку. Кстати, в «ЭРУДИТЕ» реализована особенность, которая значительно облегчает введение программы: каждое ключевое слово вводится нажатием всего лишь одной клавиши. Ключевые слова — это специальные слова БЕЙСИКА, которые указывают компьютеру, что он должен выполнять. Например, PRINT — индикация данных на экране, INPUT — ввести данные при помощи клавиатуры. Во многих компьютерах ключевые слова необходимо вводить букву за буквой, как в печатной машинке, и это необходимо делать без единой ошибки. В «ЭРУДИТЕ» необходимо всего лишь нажать клавишу, и на экране появится все ключевое слово.

БЕЙСИК «ЭРУДИТА» имеет свыше 80 ключевых слов, которые располагаются на 36 клавишах (26 буквенных и 10 цифровых). Ввиду того, что «ЭРУДИТ» использует такое множество команд БЕЙСИКА, многие клавиши имеют не одно, а несколько ключевых слов. Многие клавиши фактически имеют не только ключевые слова, но и буквы, цифры, знаки и даже графические символы, которые могут быть использованы в программах. Необходимый символ клавиши — SYMBOL SHIFT, CAPS SHIFT, EXTEND MODE, CAPS LOCK, GRAPH.

В дальнейшем будет точно указано, как выбрать любой символ или ключевое слово, находящиеся на клавиатуре (см. рис. 11).

2.2. РАБОТА С КЛАВИАТУРОЙ

Каждая клавиша «ЭРУДИТА» может выдать до шести различных ключевых слов, букв, цифр или знаков. В то же время набор необходимого ключевого слова или символа не сложен, если Вы ознакомитесь с одной особенностью «ЭРУДИТА». Когда Вы нажи-

маете клавишу, то воспроизводимое на экране зависит от режима работы компьютера в данный момент времени. Различные режимы работы позволяют Вам вводить разную информацию, например, ключевые слова, буквы или графические символы. Преимущество заключается в том, что при работе с клавиатурой «ЭРУДИТА» фактически помогает Вам выбирать режим работы клавиатуры, и Вы вводите команды и информацию в требуемой последовательности. Ниже приведено точное описание каждого режима.

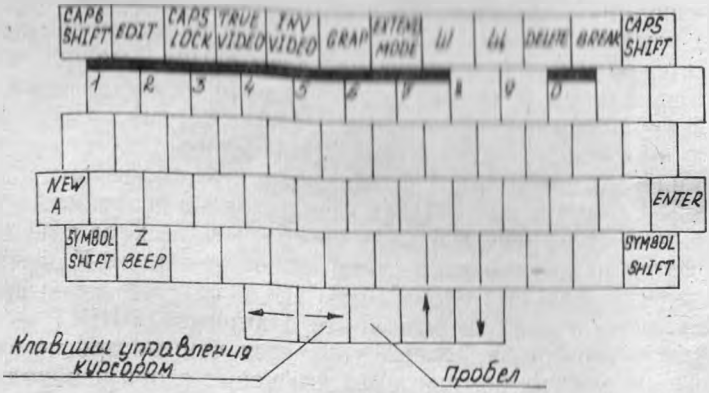


Рис. 11

Примечание:

CAPS LOCK — эта клавиша используется для перехода с латинского алфавита на русский и наоборот.

EDIT — эта клавиша используется для редактирования строки программы (см. 39 стр.).

GRAPH — эта клавиша используется для выбора графических символов на цифровых клавишах от «1» до «8» и из потребителем устанавливаемой графики.

NEW — эта клавиша очищает зону бейсика оперативной памяти компьютера.

SYMBOL SHIFT — держа эту клавишу нажатой и, нажимая другие клавиши, Вы выберете ключевые слова или знаки, написанные синим цветом. Предварительно нажав клавишу «EXTEND MODE» (режим «E»), Вы получите ключевые слова или знаки, записанные внизу клавиши (синяя надпись в желтой рамке).

КЛАВИШИ УПРАВЛЕНИЯ КУРСОРОМ

— эти клавиши перемещают курсор по направлениям, указанным на клавишах (во время редактирования программ для управления движущихся графических символов).

BEEP — эта клавиша выдает ключевое слово, которое управляет встроенным громкоговорителем компьютера.

ПРОБЕЛ — эта клавиша дает пробел (как и в пишущих машинках).

TRUE VIDEO и INV VIDEO — эти клавиши вставляют в строку программы управляющие коды для получения прямого или инвертированного изображения.

ЦИФРОВЫЕ КЛАВИШИ

— эти клавиши не только выдают цифры, но могут вставить в программу коды, которые управляют цветом (см. 75 стр.). Ключевые слова, написанные внизу клавиш от «4» до «0» (за исключением «8»), используются только во время работы с микродрайвами.

BREAK — эта клавиша останавливает выполнение программы. Она не стирает программу из памяти компьютера.

CAPS SHIFT — эта клавиша вместе с буквенной клавишей дает буквы латинского алфавита.

ENTER — эта клавиша сообщает компьютеру, что введенная информация уже полная и ее можно принять.

DELETE — эта клавиша используется для удаления неправильного ключевого слова, буквы, цифры, символа (см. 19 стр.).

EXTEND MODE — эта клавиша используется для перехода в режим «E» (для выбора ключевых слов, написанных желтым цветом, а с клавишей «SYMBOL SHIFT» для выбора нижних ключевых слов и символов (синяя надпись в желтой рамке).

2.2.1. *Режим ключевых слов*

Включите «ЭРУДИТ», нажмите кнопку RESET. На экране появится приветствие. Далее нажмите клавишу ENTER. В левом нижнем углу экрана появится мерцающая буква K.

Мерцающий квадратик называется курсором. Он показывает, в каком месте экрана появится вводимая информация, а буква К указывает, что компьютер работает в режиме ключевых слов (от слова «keyword» — ключевое слово). Нажмите какую-нибудь буквенную клавишу. На экране появится верхнее ключевое слово клавиши, записанное над буквой.

Нажмите, например, Q и на экране появится ключевое слово PLOT. Сотрите его, нажав клавишу DELETE, проверьте другие клавиши в этом режиме. Цифровые клавиши покажут цифры, а буквенные клавиши — верхние ключевые слова.

Вновь нажмите клавишу DELETE, чтобы появился К курсор. Затем нажмите клавишу SYMBOL SHIFT и, удерживая ее в нажатом состоянии, нажмите какую-нибудь клавишу с буквой. В этом случае на экране появится ключевое слово или знак, обозначенный синим цветом сбоку от буквы. Если вместо клавиши с буквой нажмете клавишу с цифрой, на экране появится знак, обозначенный синим цветом между цифрой и нужным ключевым словом.

2.2.2. *Режимы русских и латинских букв.*

Воспроизведя ключевое слово или символ в режиме ключевых слов, компьютер автоматически переходит в другой режим: на это указывает мерцающая в квадратике курсора буква L вместо К. Это режим прописных букв (от слова «Letter» — буква — первая буква). Поскольку прописные латинские буквы заменены русскими, то нажав какую-нибудь клавишу с буквой, на экране увидите букву русского алфавита. Нажмите цифру, на экране появится цифра. Если Вы хотите получить латинскую букву, нажмите и держите в нажатом состоянии клавишу CAPS SHIFT, а затем нажмите клавишу с буквой.

Если Вы захотите всю информацию ввести латинскими буквами, то вначале нажмите клавишу CAPS LOCK. В этом случае буква индикации курсора поменяется на С. «ЭРУДИТ» перешел в режим заглавных букв (от слова «capitala» — заглавные буквы — первая буква), и Вы всегда будете получать большие латинские буквы, нажав клавишу с буквой. Однако клавиши с цифрами в данном режиме будут выбирать те же самые цифры. Для возвращения в режим (L) русских букв необходимо еще раз нажать клавишу CAPS LOCK.

2.2.3. *Расширенный режим.*

Следующий режим, называемый расширенным режимом, воспроизводится нажатием клавиши EXTEND MODE. В этом случае

буква индикации курсора поменяется на E (от слова «extended» — расширенный — первая буква). Нажмите какую-нибудь клавишу с буквой. На экране появится среднее ключевое слово клавиши. Например, при нажатии клавиши B получите BIN. Для того чтобы получить нижнее ключевое слово клавиши или знак, необходимо нажать желаемую клавишу SYMBOL SHIFT и, удерживая ее в нажатом положении, нажать желаемую клавишу с буквой. В этом случае, нажав, например, клавишу B, получим BRIGHT. Следовательно, в расширенном режиме Вы получаете среднее или нижнее ключевое слово. В расширенном режиме работы компьютер после нажатия клавиши (в том числе и EXTEND MODE) автоматически возвращается в режим русских или латинских букв.

2.2.4. *Графический режим.*

Пятый режим, называемый графическим режимом, включается нажатием клавиши GRAPH. В этом случае буква индикации курсора поменяется на G (от слова «graphios» — графический — первая буква). Нажмите какую-нибудь клавишу с цифрой от 1 до 8. На экране появится графический символ, изображенный на клавише. Далее нажмите клавишу CAPS SHIFT и какую-нибудь клавишу с цифрой.

В этом случае на экране появится графический символ, но на этот раз черный и белый цвета поменяются местами. При желании перейти в прежний режим, необходимо еще раз нажать клавишу, так как компьютер автоматически не может переключиться из графического режима в другой.

2.2.5. *Как выбрать ключевое слово или символ.*

Здесь Вам предоставляется возможность увидеть, каким образом можно выбрать любые ключевые слова, знаки или символы, расположенные на клавишах с буквами или цифрами. При выборе функции клавиши обратите внимание на то, в каком месте она на клавише записана и после этого, используя три нижеприведенных примера (см. табл. 2.1), решите вопрос о необходимости замены существующего режима.

Решая вопрос о режиме работы компьютера, всегда обращайте внимание на курсор.

2.3. РЕДАКТИРОВАНИЕ ПРОГРАММ

Во время ввода программы в «ЭРУДИТ» иногда возникает необходимость исправления ошибок в командах или строках програм-

мы или же замены этих строк или команд. Это легче всего выполнить путем редактирования программы.

2.3.1. *Как исправить ошибку.*

Если Вы в память компьютера попытаетесь ввести строку или команду, в которых имеется ошибка БЕЗ СМЫСЛА, то компьютер перед ошибкой будет индицировать мигающий вопросительный знак. Для исправления ошибки необходимо клавишами сдвига курсора влево или вправо (←, →) установить его справа от ошибки. Затем либо сотрите ошибочную информацию, нажав клавишу DELETE, либо введите недостающие ключевые слова или символы. После того нажмите клавишу ENTER.

Например, допустим, что Вы хотели перемножить 7 и 8 и, желая ввести знак *, забыли нажать клавишу SYMBOL SHIFT. Фактически Вы ввели PRINT 7B8.

«ЭРУДИТ» не может выполнить данную команду, потому, после нажатия ENTER, она индицирует мигающий вопросительный знак перед B, а это значит, что ошибка в этом месте программы. Вам необходимо передвинуть курсор так, чтобы он располагался справа от ошибки, после этого стереть B, нажав клавишу DELETE. Далее, чтобы компьютер выполнил правиль-

1	2	3	4

Режим букв (C)

Только клавиша B

[CAPS LOCK]

потом клавиша Б

[SYMBOL SHIFT]

и клавиша *

Графический режим (G)

[GRAPH] потом

только клавиши

от A до И — графика, установленная потребителем

Режим букв (C)

[CAPS LOCK] и клавиша 3

[CAPS LOCK] потом [SYMBOL SHIFT]

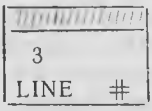
и клавиша #

Графический режим (G)

[GRAPH] потом клавиша

[GRAPH] потом [CAPS SHIFT]

и клавиша

Клавиша буквы		Клавиша цифры	
			
Действие	Результат	Действие	Результат
1	2	3	4
Режим ключевых слов (K)		Режим ключевых слов (K)	
Только клавиша BORDER		Только клавиша 3	
SYMBOL SHIFT и клавиша *		SYMBOL SHIFT и клавиша #	
Расширенный режим (E)		Расширенный режим (E)	
EXTEND MODE потом клавиша BIN		EXTEND MODE пурпурная бумага потом клавиша	
EXTEND MODE потом SYMBOL SHIFT и клавиша BRIGHT		EXTEND MODE потом SYMBOL SHIFT и клавиша пурпурные чернила	
		EXTEND MODE потом CAPS SHIFT и клавиша LINE	
Режим букв (L)		Режим букв (L)	
Только клавиша В		Только клавиша 3	
CAPS SHIFT и клавиша Б		SYMBOL SHIFT и клавиша #	
SYMBOL SHIFT и клавиша *			

по введенную команду, нажмите SYMBOL SHIFT и B (выбор *), а затем ENTER.

2.3.2. *Как отредактирвать строку программы.*

Когда Вы пишете программу, Вы создаете последовательность пронумерованных строк команд, называемую листингом. После вво-

да программы Вы можете просмотреть его, нажимая LIST и ENTER. Вы можете заметить знак >, расположенный перед какой-либо строкой программы. Если его нет, то нажмите клавиши (↑, ↓) сдвига курсора вверх или вниз и удерживайте их до момента появления знака, который называется программным курсором. Если Вы теперь нажмете клавишу EDIT, строка, которую обозначает программный курсор, будет записана в конце программы, где ее можно исправить, как показано выше, используя сдвиг курсора и клавишу DELETE. Для ввода исправленной строки в программу достаточно нажать клавишу ENTER. Аналогично редактируются и другие строки программы.

Если необходимо программный курсор сдвинуть очень далеко, введите команду LIST с номером необходимой строки и только после этого нажмите клавишу EDIT. В обоих случаях необходимая строка окажется в нижней части экрана, и ее можно будет исправить.

Для стирания всей строки проще всего набрать номер этой строки и после этого нажать клавишу ENTER. Если Вы с помощью RUN запустили программу, в которой имеется ошибка, то в нижней части экрана Вам будет выдано сообщение об ошибке.

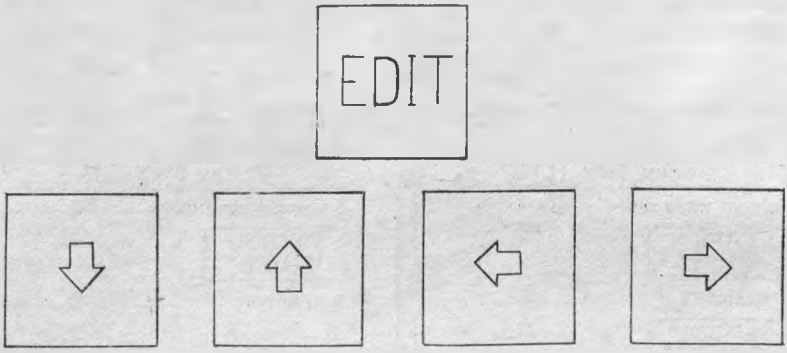


Рис. 12

2.4. ВЫЧИСЛЕНИЯ

Вычисления «ЭРУДИТОМ» выполняются очень быстро и точно. Вначале введите команду.

PRINT 5+4

При нажатии клавиши ENTER команда исчезает и на экране появляется ответ — число 9.

Для вычислений «ЭРУДИТ» использует пять знаков, называемых знаками арифметических операций. Вы можете их проверить при помощи команды PRINT. Введение таких команд, как PRINT превращает Ваш «ЭРУДИТ» в калькулятор. Однако она может выполнить гораздо больше операций, чем обыкновенный калькулятор. Например, может одновременно индицировать как само выражение, так и результат. Введите следующую команду:

PRINT «5+4=»; 5+4

Компьютер будет индицировать следующий ответ:

5+4=9

Это получается потому, что команда PRINT индицирует на экране без изменений все то, что заключено в кавычках, таким образом, на экране появится 5+4=. Символы, заключенные в кавычки, составляют строку. Точка с запятой указывает «ЭРУДИТУ», что результат необходимо индицировать непосредственно за знаком равенства (подробнее об этом смотрите в подразделе 2.4.6).

2.4.1. *Знаки, используемые в вычислениях.*

В табл. 2.2 приведены так называемые «знаки арифметических операций», которые используются для проведения математических операций. Обратите внимание на то, что в компьютере не применяются такие знаки, как × или :.

Таблица 2.2

Символ	Клавиша	Функция	Пример
+	K	Сложить два числа	6+3=9
—	J	Вычитать одно число из другого или указать отрицательное число	6—3=3 —6—3=—9
*	B	Умножить два числа	6×3=18
/	V	Разделить одно число на другое	6/3=2
↑	H	Возвести первое число в степень, которую указывает второе число	6^3=216

2.4.2. *Ваша первая программа.*

После того, как введенная команда выполнена, компьютер попросту ее «забывает». Если хотите, чтобы компьютер повторил расчеты, Вы можете это записать в виде программы. Введите следующую пронумерованную строку программы и нажмите клавишу ENTER.

10 PRINT 5+4

В данном случае ключевое слово PRINT с атрибутами называется не командой, а оператором. После нажатия ENTER оператор не будет сразу выполняться, как бы это произошло в случае с командой. Вместо этого компьютер запишет строку этой команды в верхней части экрана.

Номер у строки вынуждает «ЭРУДИТ» записать в память введенную строку и не выполнять ее до тех пор, пока не будет об этом указано отдельно.

Все строки, составляющие программу, выполняются лишь в том случае, когда Вы запустите программу специальной командой RUN. Теперь нажмите последовательно RUN и ENTER. На экране появится результат — число 9.

Операторы команды всегда выполняются в той последовательности, которая указывается номерами строк, а нумерация строк обычно производится через десять, чтобы в дальнейшем, при необходимости, можно было включать новые строки.

Теперь попробуйте реально поработать с «ЭРУДИТОМ». Введите сяедующую программу:

Таблица чисел

```
10 LET N=1
20 PRINT N; " ";
30 LET N=N+1
40 GO TO 20
```

Рис. 13

Не забудьте в конце каждой строки нажать клавишу ENTER, а после того, когда вся программа введена, — RUN и ENTER.

На экране появляются все числа от 1 до 203. Теперь нажмите какую-нибудь клавишу, кроме N, ПРОБЕЛ, STOP и BREAK. В данном случае на экране появится новый ряд чисел.

Эта программа использует переменную. В нашем случае переменной является N. В качестве переменной может быть буква или слово (за исключением русских букв Ё, Ч, Ш, Щ, Ъ, Э, Ю).

Значение переменной изменяется в процессе выполнения команды. Оператор LET, находящийся в 10 строке, используется для присвоения переменной значения, равного 1. Оператор PRINT из 20 строки индицирует на экране значение переменной и затем оставляет пробел. В 30 строке вновь используется оператор LET, но теперь для увеличения значения переменной на единицу. Оператор GO TO (одно ключевое слово) из 40 строки возвращает программу в 20 строку, которая теперь индицирует 2. Так будет повторяться до тех пор, пока весь экран не заполнится числами.

2.4.3. *Как заставить программу запрашивать числа.*

Остановите предыдущую программу, нажав клавишу BREAK. Теперь введите новую строку:

10 INPUT N

Эта строка заменяет в программе 10 строку. Теперь, после запуска программы, экран очистится, а курсор окажется в нижней, левой части экрана — компьютер ждет от Вас ввода числа. Введите какое-либо число и нажмите клавишу ENTER. На экране появится числовой ряд, начинающийся с введенного Вами числа. Так происходит потому, что оператор INPUT N присваивает переменной значение введенного Вами числа. Оператор INPUT заставляет компьютер во время выполнения программы запрашивать информацию.

2.4.4. *Программирование таблицы умножения.*

Сотрите старую программу, нажав кнопку сброса RESET, и введите нижеприведенную.

Таблица умножения

```
10 LET X=1
20 INPUT N
30 PRINT N; «*»; X; «=»; N*X
```

```
40 LET X=X+1
50 GO TO 30
```

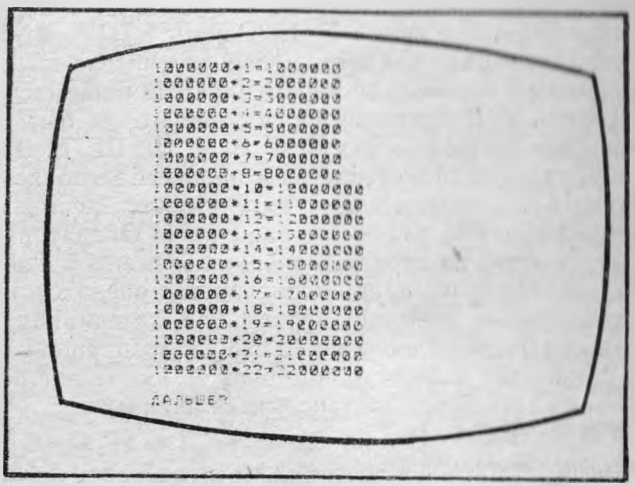


Рис. 14

Данная программа заставляет «ЭРУДИТ» умножать.

Введите какое-нибудь число, и на экране появится таблица умножения для данного числа. Нажмите любую клавишу, за исключением BREAK и ПРОБЕЛ, и таблица умножения будет продолжаться.

Нажмите клавишу BREAK, а после этого опять запустите программу при помощи RUN и создайте новую таблицу.

На рис. 14 приведено изображение на экране при введенном числе 1 000 000.

2.4.5. *Для чего применяются скобки.*

Иногда в выражениях для вычислений Вы будете применять скобки. Введите следующие две команды и сравните результаты:

```
PRINT 6+2/4
PRINT (6+2)/4
```

Первый результат будет 6,5, а второй — 2. Разные результаты получаются потому, что компьютер при вычислениях применяет свою приоритетную систему. Первым выполняется возведение в степень (↑), а затем умножение (*) или деление (/) и, наконец, сложение (+) или вычитание (—), но всегда первыми выполняют-

ся вычисления, заключенные в скобки. Поэтому, выполняя первую нашу команду, компьютер сначала делит 2 на 4, а затем результат 0,5 прибавляет к 6. Выполняя вторую команду, компьютер сначала прибавляет 2 к 6, а затем делит на 4.

2.4.6. *Как использовать разделительные знаки.*

«ЭРУДИТ» использует ряд разделительных знаков. Они очень важны, так как многие из них являются как бы инструкцией для компьютера, в зависимости от которой он так или иначе понимает строку программы или индицирует данные на экране.

; Точка с запятой. При употреблении ее в операторе PRINT компьютеру поступает указание, что элементы данных, находящиеся по обе стороны от точки с запятой, необходимо располагать на экране один за другим.

: Двоеточие. Сигнализирует о том, что в строке программы заканчивается один оператор и начинается новый.

" Кавычки. Любые символы, заключенные в кавычки, трактуются не как числа или переменные, а как текст. Кавычками начинается и заканчивается строка.

, Запятая. При употреблении ее в оператор PRINT компьютеру поступает указание, что элемент данных, находящийся после запятой, необходимо индицировать или с центра экрана или с начала следующей строки. Не применяйте запятой в дробных числах.

. Точка. Применяется в качестве запятой в дробных числах.

' Апостроф. При применении его в операторе PRINT компьютеру поступает указание, что элемент данных, находящийся за апост-

рофом, необходимо индицировать с начала следующей строки.

2.5. ЦВЕТА И КАК ИХ ИСПОЛЬЗОВАТЬ

«ЭРУДИТ» может создать восемь разных цветов. Каждый цвет имеет свой кодовый номер (см. табл. 2.3.).

Таблица 2.3

Коды цветов «ЭРУДИТА»

Номер кода	Цвет
0	Черный
1	Синий
2	Красный
3	Пурпурный
4	Зеленый
5	Голубой
6	Желтый
7	Белый

В табл. 2.3. показаны цвета и даны их коды. Вам не нужно запоминать эти коды — цифровые клавиши, которые им соответствуют, отмечены соответствующим цветом. Фактически оттенки цвета, которые Вы получаете на экране своего телевизора, зависят от Вашего телевизора и положения ручек регулировки цвета, контраста и яркости.

2.5.1. *Три способа применения цвета.*

Вы можете использовать цвета тремя способами: указывая цвета контура экрана, бумаги и чернил.

Телевизионный экран делится на две части: в центре находится рабочая часть экрана, на которой могут индицировать любые данные, а остальная часть — по краям — называется контуром, где не могут быть индицированы какие-либо данные, но можно менять только цвет.

Вся рабочая часть экрана разбита на квадратики — позиции символов. В одном квадратике индицируется один символ. Каждый квадратик состоит из 64 точек (8×8). Цвет, в который окрашиваются индицируемые символы, точки и линии, называется цветом чернил, а цвет фона — цветом бумаги. Цвета бумаги и чернил могут быть установлены для всей рабочей области экрана или только для отдельной позиции символа. Одна позиция символа может иметь только один цвет чернил и бумаги.

Когда Вы включаете «ЭРУДИТ», автоматически устанавливаются начальные цвета: цвет чернил — черный, а контура и бумаги — белый. Вы можете тут же поменять цвета, введя непосредственные команды установки цветов:

BORDER 7 — устанавливает цвет контура (в этом случае белый);

INK 2 — устанавливает цвет чернил (красный);
PAPER 6 — устанавливает цвет бумаги (желтый).

Теперь нажмите кнопку сброса RESET, введите и запустите простую программу:

```
10 PRINT «*»;
20 GO TO 10
```

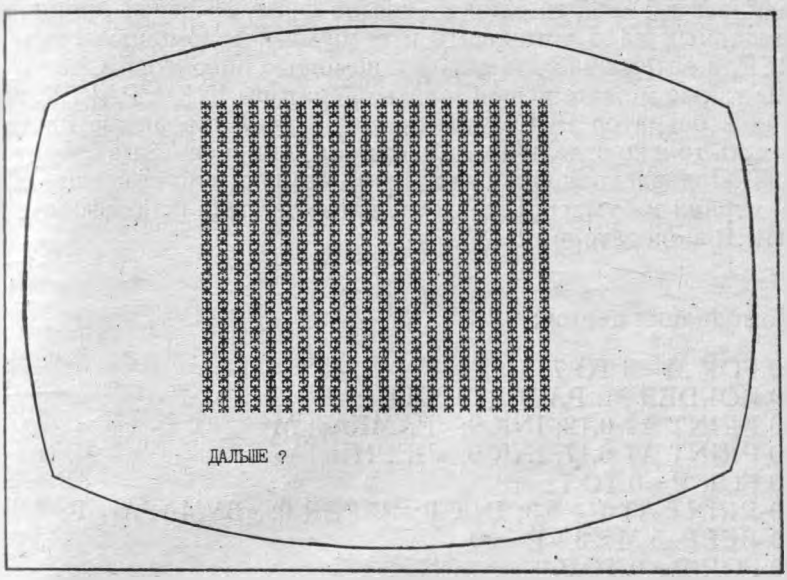


Рис. 15

На экране появляется рисунок, состоящий из черных звездочек на белом фоне. Теперь нажмите клавишу BREAK и введите несколько команд установки цветов — BORDER, INK и PAPER с различными комбинациями кодов цветов, после каждого ввода запуская программу. Рисунок на экране будет тот же, но различных цветов.

2.5.2. *Программирование цветных изображений*

Если Вы хотите видеть текст, надписи, таблицы и картинки в различных цветах, то можете использовать в программах ключевые слова BORDER, INK и PAPER. Цвет контура меняется сразу же, как только компьютер встречает в строке программы оператор

BORDER. Оператор INK, присутствующий в строке программы, устанавливает новый цвет чернил только для индицированных далее символов. Цвет символов, индицированных до оператора INK, не меняется. Оператор PAPER, встретившийся в строке, меняет цвет фона, но не по всему экрану, а только в тех квадратиках (позициях символов), в которых индицируются символы уже после встречи PAPER. Если Вы хотите, чтобы вся рабочая часть экрана окрасилась в какой-то один из цветов фона, то перед индицированием данных надо установить цвет бумаги с помощью оператора PAPER, а затем очистить экран с помощью оператора CLS.

Вы также можете использовать операторы INK и PAPER, включив их в оператор PRINT. В данном случае указанные цвета будут иметь только отдельные символы, сындицированные оператором PRINT. Приведенная ниже программа демонстрирует все цвета контура, чернил и бумаги, а также показывает, как использовать INK и PAPER в операторе PRINT.

Комбинация цветов:

```
10 FOR M=0 TO 7
20 BORDER M: PAPER M: CLS
30 PRINT AT 0,12; INK 9; «РАМКА»; M
40 PRINT AT 6,17; INK 9; «ЧЕРНИЛА»
50 FOR P=0 TO 7
60 PRINT AT P+8,3; INK P; PAPER 9; «БУМАГА»; P; " ";
70 BEEP .5,M×8+P—30
80 FOR R=0 TO 7
90 PRINT INK R; PAPER R; " "; R;
100 BEEP .05,P×4+10
110 NEXT R
120 NEXT P
130 NEXT M
```

Коды цветов задают три переменные: M — контура, P — бумаги, R — чернил. Операторы BEEP создают звук, а строки программы, начинающиеся ключевыми словами FOR и NEXT, обозначают начало и конец программных циклов, изменяющих коды цветов от 0 до 7. Обратите внимание на то, что и оператор INK и оператор PAPER могут иметь код цвета, равный 9. Оператор INK 9 делает цвет чернил контрастным, т. е. черным или белым по отношению к цвету, соответствующему цвету бумаги, такому, чтобы символ был хорошо виден. Аналогично оператор PAPER 9 делает цвет бумаги контрастным по отношению к цвету чернил той же самой позиции.

9.5.3. *Программирование цветных диаграмм.*

Нижеприведенная программа формирует цветные столбчатые диаграммы (см. рис. 16). Они показывают количество жителей двенадцати европейских стран (Югославии, Польши, Великобритании, Португалии, Румынии, Франции, Испании, ФРГ, ГДР, Венгрии, Италии и Чехословакии) в виде желтых столбцов с числом наверху. В строке с номером 50 между кавычками введите два пробела.

В операторе DATA записаны данные, которые считываются при помощи оператора READ, находящегося в 30 строке. Во время действия каждого цикла, начинающегося с 20 строки и кончающегося 80 строкой, из списка DATA считывается одно число, значение которого присваивается переменной G. А теперь добавьте к уже введенным строкам еще и те, которые приведены ниже, а 90 строку замените новой. Внизу будут нарисованы еще и зеленые диаграммы, указывающие на площадь этих стран (в сотнях тысяч кв. км).

Диаграммы

```
10 BORDER 0: PAPER 1: CLS
20 FOR S=4 TO 26 STEP 2
30 READ G
40 FOR E=21 TO 22—G STEP—1
50 PRINT AT E,S; PAPER 6; " "
60 NEXT E
70 PRINT AT 21—G,S; INK 9; G×3
80 NEXT S
90 DATA 7,12,18,3,8,18,13,20,6,4,19,5
```

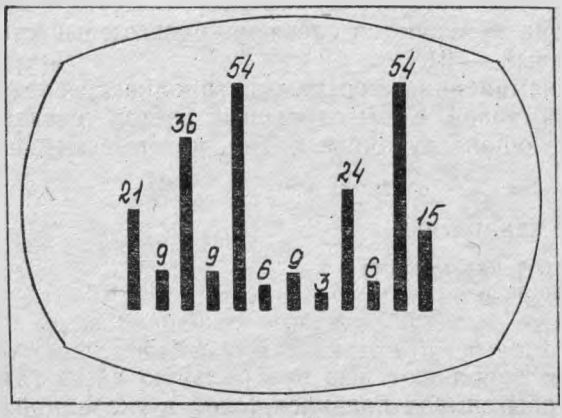


Рис. 16

```
71 READ
72 FOR E=21 TO 22— STEP—1
73 PRINT AT E,; PAPER 4; " "
74 NEXT E
75 PRINT AT 21—, ; INK 9; PAPER 6;
90 DATA 7,3,12,3,18,2,3,1,8,2,18,5,13,5,20,2,6,1,4,1,
        19, 3, 5, 1
```

2.6. ГРАФИКА

Графику, формируемую «ЭРУДИТОМ», условно можно разделить на графику низкой и высокой разрешающей способности. Оба типа графики могут использоваться на экране одновременно.

2.6.1. *Графика низкой разрешающей способности.*

Вся рабочая часть экрана имеет 768 позиций символов — 24 строки по 32 позиции. Однако графика низкой разрешающей способности при помощи оператора PRINT может быть создана только на 22 верхних строках в основной части экрана, которая имеет всего 704 позиции символов. Две нижние строки предназначены для индикации различных сообщений компьютера и вновь вводимых данных. В этих строках желаемые символы можно индицировать, введя оператор INPUT.

Для установления места каждой позиции, находящейся в основной части экрана, используются два числа: первое указывает на номер строки, второе — на номер столбца. Строки экрана нумеруются сверху вниз: верхняя строка имеет номер 0, а нижняя — 21. Столбцы экрана нумеруются слева направо: левый столбец имеет номер 0, а правый — 31.

Представляемая ниже программа заполняет разными цветами все позиции символов. Коды случайных цветов создает функция RND, генерирующая случайные числа, которые меньше 1 и больше 0.

Случайные квадратики

```
10 BORDEP 1 : INK RND×7
20 PRINT "■";
30 G0 TO 10
```

Квадратики заполняют всю центральную часть экрана. Если Вы хотите, чтобы символ появился точно в указанной позиции, то вместе с PRINT надо использовать ключевое слово AT. Оно пишет-

ся после PRINT, затем указывается номер строки, ставится запятая и указывается номер столбца. Теперь ставится точка с запятой, после которой записывается информация, которая должна появиться, начиная с указанной позиции. Например,

команда

PRINT AT 11,16; «*»
печатает звездочку в 11 строке 16 столбца.

2.6.2. *Создание цветных узоров*

Для получения цветных узоров в программах очень удобно использовать циклы FOR NEXT. Эти циклы представляют собой такие части программы, которые повторяются очередное количество раз. В первой строке цикла можно указать компьютеру, сколько раз цикл надо повторять: после ключевого слова FOR пишется переменная с присвоением начального значения, а после TO — конечное значение. Последняя строка цикла обозначается ключевым словом NEXT.

Часто бывает полезно вложить один цикл в другой. Нижеприведенная программа демонстрирует, как могут использоваться два цикла FOR NEXT, вложенные один в другой, если менять цвета INK и позиции AT.

Цветные узоры

```
10 BORDER 0 : PAPER 5 : CLS
20 FOR X=1 TO 7
30 FOR Y=0 TO 15
40 PRINT AT Y+X—1,15—Y;
   INK X; "■ "
50 PRINT AT Y+X—1,16+Y;
   INK X; "■ "
60 NEXT Y
70 NEXT X
```

2.6.3. *Как выбрать графические символы.*

Клавиатура «ЭРУДИТА» имеет набор графических символов, который без труда разрешает программировать графику низкой разрешающей способности. Вы найдете эти символы на цифровых клавишах от 1 до 8.

Чтобы сформировать графические символы на экране, надо нажать GRAPH и тут же какую-либо из указанных клавиш. Графи-

ческие символы появятся внизу экрана. Изображенная черным цветом часть каждого символа обозначает цвет бумаги, а остальная — чернил. Нажимая какую-нибудь из этих клавиш одновременно с CAPS SHIFT, получаем графические символы противоположных цветов.

Закончив ввод графических символов в строках программы, компьютер следует перевести обратно в начальный режим посредством нажатия клавиши GRAPH.

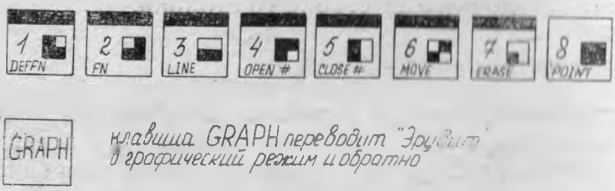


Рис. 17

2.6.4. *Программирование картинок.*

Используя графику низкой разрешающей способности, Вы можете «рисовать» картинки, предварительно установив позиции и цвета необходимых для этого графических символов. Пользуясь крупной сеткой, приведенной в конце книги на рис. 3.2, Вы можете создать свою картинку. Затем, пользуясь изображениями графических символов на цифровых клавишах 1—8, введите одну за другой составляющие картину строки программы. Для создания на экране крупных частей картинки, которые складываются из одинаковых символов того же цвета, используйте циклы FOR NEXT.

Нижеприведенная программа является примером такого программирования. Попробуйте запустить программу, вводя первые четыре строки, затем восемь строк, тринадцать, а далее — каждый раз прибавляя по одной строке. Вы увидите, как из частей складывается вся картинка — башня Гедиминаса.

Строки 20—40 создают зеленую основу. Следующие четыре строки рисуют саму башню, а строки 90—130 — амбразуры. Следующие затем строки создают вершину башни (140 строка), флаг (150 и 160 строки) и надпись.

Башня замка

```
10 BORDER 0: PAPER 5: CLS
20 FOR E=19 TO 21 : FOR S=0 TO 31
30 PRINT AT E, S; INK 4; « ■ »
40 NEXT S: NEXT E
50 FOR X=1 TO 3 : FOR У=1 TO 4
```

```
60 FOR S=13—X TO 18+X
70 PRINT AT 2+X×4+Y, Z; INK 2; « ■ »
80 NEXT S : NEXT Y : NEXT X
90 FOR X=4 TO 12 STEP 4
100 PRINT AT 4+X,15; INK 0; PAPER 2; «  »
110 PRINT AT 5+X,15; INK 0; PAPER 2; «  »
120 PRINT AT 6+X,15; INK 0; « »
130 NEXT X
140 PRINT AT 6,12; INK 2; «  »
150 PRINT AT 1,16; INK 6; «  »
160 PRINT AT 2,16; INK 4, PAPER 2; «  »
170 PRINT AT 20, 13; INK 9; PAPER 4; «LIETUVA»
```

2.6.5. *Как пользоваться циклами* FOR NEXT.

Циклы FOR NEXT всегда начинаются со строки, в которой есть ключевые слова FOR и TO и переменная с указанными начальным и конечным значениями. Например:

```
80 FOR C=1 TO 6
```

Здесь переменной цикла является C. Операторы, имеющиеся в строках, следующих после этой строки, компьютер будет выполнять указанное количество раз: в нашем примере — шесть раз. В этих операторах может использоваться и переменная цикла C.

Цикл FOR NEXT всегда кончается ключевым словом NEXT, после которого указана переменная цикла, например:

```
100 NEXT C
```

Если цикл выполняется впервые, то переменной цикла присваивается начальное значение, находящееся перед TO, и во время каждого последующего выполнения значение переменной увеличивается на единицу, пока не достигнет конечного значения. В нашем случае вначале C присваивалась единица, затем значение C становилось равным 2, 3, 4, 5 и 6.

Если хотите, чтобы во время каждого цикла переменная менялась не на 1, а на какую-нибудь другую величину, называемую шагом, то ее следует указать в первой строке цикла после ключевого слова STEP, например:

```
30 FOR E=1 TO 6 STEP 0,5
```

В этом случае переменная цикла во время каждого цикла будет меняться через каждые 0,5, т. е. приобретет значения 1, 1,5, 2, 2,6, 3, . . ., 5,5 и 6.

Циклы можно вложить один в другой. Например, в программе «Комбинация цветов», приведенной в подразделе 2.5.2, один в другой вложены три цикла. Это значит, что «средний» цикл выполняется указанное количество раз во время выполнения «внешнего» цикла, а «внутренний» цикл выполняется еще большее число раз — он проходит все свои значения во время выполнения каждого «среднего» цикла. Надо только запомнить, что последний начатый цикл должен закончиться первым, т. е. соответствующие ключевые слова NEXT должны идти в порядке, обратном FOR.

2.6.6. *Графика высокой разрешающей способности.*

Применяя графику высокой разрешающей способности, Вы можете создать детализированные картинки из прямых и кривых отрезков.

В этом случае минимальным элементом графики является точка. Каждая позиция, используемая в графике низкой разрешающей способности, составлена из 64 таких точек (8×8), т. е. в основной части экрана содержится 45056 точек: 256 по горизонтали и 176 по вертикали. Графические точки называются пикселями (от англ. «picture cells» — элементы картины, клеточки). Как и для символа графики низкой разрешающей способности, так и для указания места пикселя на экране, используются два числа. Однако координаты пикселя указываются в обратном порядке: сначала указывается координата горизонтальной оси — X-координата (для графики низкой разрешающей способности это соответствует столбцу), а затем координата по вертикальной оси — Y-координата (соответствует строке). По горизонтальной оси пиксели нумеруются как и позиции графических символов слева направо: от 0 до 255. По вертикальной же оси пиксели нумеруются наоборот (по сравнению с позициями графических символов) — снизу вверх: от 0 до 175. Таким образом, экран графики высокой разрешающей способности соответствует обычной системе координат X, Y — пиксель с координатами 0; 0 находится в нижнем левом углу, а пиксель с координатами 255, 175 — в верхнем правом (см. рис. 26).

2.6.7. *Обозначение и вычерчивание точек.*

Графика высокой разрешающей способности использует только три ключевых слова: PLOT, DRAW и CIRCLE.

Ключевое слово PLOT, за которым следует горизонтальная и вертикальная координаты, разделенные запятой, индицирует на экране пиксель цвета чернил в указанной позиции. Например:

PLOT 128,87

DRAW чертит линию между двумя точками. Первая точка — это ранее указанный пиксель, а вторая — это пиксель, место которого на экране указывают два разделенных запятой числа, идущие сразу после DRAW. Но это не абсолютные координаты второго пикселя, а их изменения относительно координат ранее указанного пикселя. Ранее указанный пиксель — это пиксель, указанный в последнем предыдущем операторе PLOT или DRAW. Если оператор DRAW применяется в программе впервые и до этого не было ни PLOT, ни другого DRAW, то в качестве более раннего (предыдущего) принимается пиксель с нулевыми координатами.

Например, строки программы:

```
30 PLOT 100,80
40 DRAW — 20,30
```

чертят линию от пикселя с координатами 100,80 до пикселя с координатами 80,100. Из этого примера видно, что используемые DRAW изменения координат могут быть отрицательными. Попробуйте эту программу:

Песочные часы

```
10 INK 4
20 PLOT 80,152
30 DRAW 96,—126
40 DRAW —96,0
50 DRAW 96,126
60 DRAW —96,0
```

Рис. 18

Оператор PLOT устанавливает начальную позицию в левом верхнем углу экрана. Затем четыре DRAW оператора чертят четы-

ре зеленые линии. Теперь добавьте к программе 70-ю строку, а 10-ю замените новой.

```
10 BORDER 2 : PAPER 6 : INK 1 : CLS
70 CIRCLE 128,88,80
```

Запустите программу еще раз, и вокруг песочных часов будет начерчена окружность. Как видите, после ключевого слова CIRCLE указаны три числа. Первые два числа — это координаты центра вычерчиваемой окружности, а третье — радиус окружности, выраженный количеством пикселей.

Третий параметр можно указать и в операторе DRAW, но в этом случае DRAW будет чертить не линии, а дуги окружностей. Начальная и конечная точки дуг указываются так же, как и для прямых линий, а третье число указывает угол этой дуги в радианах. Изгиб дуги в ту или иную сторону зависит от знака третьего числа. Попробуйте с помощью DRAW начертить прямую, а затем добавить третье число в пределах от 2 до —2, и посмотрите что получится.

2.6.8. *Как заштриховать фигуры.*

Вы можете легко сформировать сплошные фигуры при помощи графики высокой разрешающей способности, располагая множество линий одну возле другой. Это можно осуществить при помощи цикла FOR NEXT, изменяя позиции, указанные оператором DRAW.

Попробуйте такую программу:

Заштрихованные треугольники

```
10 BORDER 4: PAPER 6: INK 2: CLS
20 FOR X=24 TO 232
30 PLOT X, 160
40 DRAW 256—2*X,—144
50 NEXT X
```

При выполнении этой программы линии располагаются сверху вниз. При помощи оператора PLOT указываются верхние точки линии, X-координаты которых меняются от 24 до 232. Нижние точки указываются оператором DRAW: X-координата изменяется в обратном порядке — от 232 до 24. Если в 20-й строке примените ключевое слово STEP, то вычерченные линии не сольются.

Попробуйте, например, следующую замену:

```
20 FOR X=24 TO 232 STEP 4
```

Вы получите фигуру, похожую на двойной веер. Так происходит потому, что в данном случае координата конца строк изменяется не через 1, а через 4.

2.6.9. *Вычерчивание эскизов.*

Вообще для создания узоров или рисунка нет необходимости каждый раз писать новую программу. Вместо этого можно написать программу, которая дала бы возможность формировать рисунок непосредственно на экране. Ниже приведена простая программа, позволяющая это выполнить.

Программа чертит короткие, Вами указанные линии, начиная с пикселя с координатами 10,10. Программа начинается с оператора, требующего указать код чернил, бумаги и длины вычерченных линий. Далее находящийся в 40 строке оператор INPUT ждет, пока Вы укажете направление черчения. Введя символ «В», линию направите вверх, «Н» — вниз, «Л» — влево, «П» — вправо. Знак «¤», расположенный в названии переменной K¤, указывает на то, что переменная является не числовой, а символьной, т. е. переменной присваиваются не числовые значения, а строки. Оператор I указывает направление вычерчивания линий.

Эскизы

```
10 INPUT «ЧЕРНИЛА»; R, «БУМАГА:»; P, «ДЛИНА:»; 1
20 BORDER R: PAPER P: INK R: CLS
30 PLOT 10,10
40 INPUT K¤
50 IF K¤ =«В» THEN DRAW 0,1
60 IF K¤ =«Н» THEN DRAW 0,—1
70 IF K¤ =«П» THEN DRAW 1,0
80 IF K¤ =«Л» THEN DRAW —1,0
90 GO TO 40
```

2.6.10. *Выбор рабочих вариантов программы с помощью операторов* IF *и* THEN

В 50—80 строках программы «Эскизы» используются операторы IF и THEN. Они позволяют компьютеру решить, что делать дальше. В этом случае он проверяет — является ли введенный символ одним из символов «В», «Н», «П» или «Л». Если это так, то компьютер выполняет соответствующий оператор DRAW — чертит линию в указанном направлении.

После ключевого слова IF всегда указывается условие, выполнение которого проверяет компьютер. Если условие выполняется, то затем выполняются операторы, указанные в той же строке после

ключевого слова THEN. Если нет, то программа сразу выполняет следующую строку. Если после THEN надо указать более одного оператора, то они разделяются двоеточиями, например:

```
80 IF A=10 THEN PRINT «ХОРОШО!»: LET B=B—1 : GO
   TO 30.
```

2.6.11. *Создание узоров.*

Используя графику низкой или высокой разрешающей способности или обе сразу, «ЭРУДИТ» позволяет создать любой узор или картинку. Вначале при создании картинок удобно пользоваться экранной сеткой, приведенной на рис. 26. Затем следует составить программу вычерчивания линии или фигуры в нужных позициях. Для создания узоров и картинок удобно применять циклы FOR NEXT. Во время каждого цикла можно менять цвет и позицию символа или линии, как правило, по постоянному закону. В нижеприведенной программе использован этот метод.

Квадраты

```
10 BORDEREO : PAPER 0 : CLS
20 FOR R=6 TO 1 STEP —1
30 FOR E=12—P * 2 TO 9+R * 2
40 FOR S=17—P * 2 TO 14+R * 2
50 PRINT AT E,S; INK R; « ■ »
60 NEXT : NEXT E : NEXT R
```

В этой программе использованы три FOR NEXT цикла. «Внешний» цикл с переменной R — «чернила» — меняет цвет чернил и величину квадрата, который формируется в этот момент. В то же время циклы с переменными E — «строка» и S — «столбец» — меняют позиции строки и столбца графического символа «■ ». Попробуйте печатный символ квадрата «■ », указанный в 50 строке, заменить каким-нибудь другим символом.

2.6.12. *Случайные эффекты и подпрограммы.*

В программе «Квадраты» цвета квадратов точно установлены. Но иногда возникает необходимость в том, чтобы после каждого пуска программы цвета получались различными. Цвета, позиции и другие параметры после каждого цикла программы можно получить разными, используя функцию генерирования случайных чисел P. Если она генерирует случайные числа, например, между нулем и семью, то в программе следует написать так: RND * 7.

Нижеприведенная программа чертит на экране симметричные узоры, складывающиеся из графических символов. Используемая в ней функция P меняет код графических символов, а также цвета чернил и бумаги. Оператор 1M, находящийся в 10 строке, создает двухмерный массив K, имеющий 64 элемента — цифровые переменные, которые указываются при записи названия массива и в скобках — индексы: K(1,1), K(1,2), . . ., K(8,7), K(8,8).

Циклы с переменными I и J создают 16 случайных кодов, которые указывают на какой-нибудь графический символ. Цикл с переменной M устанавливает для каждого из этих графических символов по три симметричных ему графических символа. Таким образом создаются коды 64 графических символов, которые присваиваются переменным массива K.

Оператор DIM B 1000, находящийся в 90-й строке, передает управление подпрограмме. Подпрограмма — это часть программы, которая заканчивается оператором RETURN. Чаще всего это последовательность операторов, необходимых в нескольких местах программы. Оператор GO SUB, как и оператор GO TO, передает управление указанной строке — в нашем случае с номером 1000.

Симметричные рисунки

```
10 BORDER 0 : PAPER 0 : CLS :
   DIM K(8,8)
20 LET P=INT (RND*4)
30 INK P+4 : PAPER P
40 FOR I=1 TO 4 : FOR J—1 TO 4
50 LET K(1,J)=INT (RND+16) :
   RESTORE
60 FOR M=1 TO K(I,J)+1
70 READ K (I,9—J), K(9—I,J)
   K(9—I,9—J)
80 NEXT M : NEXT J : NEXT I
90 GO SUB 1000
100 PAUSE 100 : GO TO 20
1000 LET X=1 : PRINT AT 0,0;
1010 FOR Q=0 TO 21
1020 FOR W=1 TO 4 : FOR Y=1 TO 8
1030 PRINT CHR¤ (K(X,Y)+128) :
1040 BEEP .01, (K(X,Y)+3
1050 NEXT Y : NEXT W
1060 LET X=X+1
1070 IF X=9 THEN LET X=1
1080 NEXT Q
```

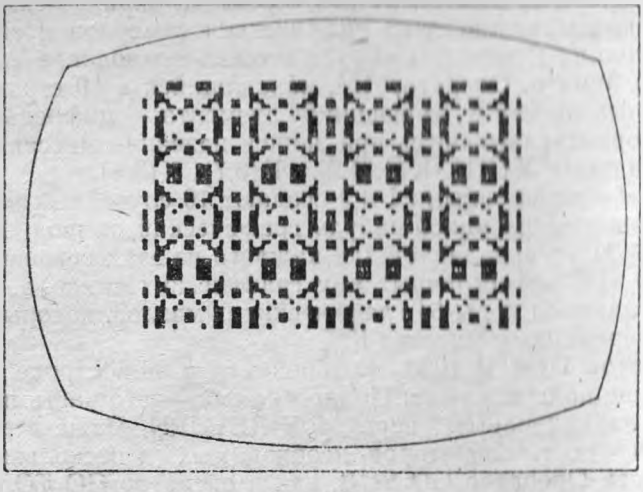


Рис. 19

```
1090 RETURN
2000 ДАТА 0,0,0,2,4,8,1,8,4,3,12,12,8,1,2,10,5,10,9,9,6,
     11,13,14,4,2,1,6,6,9,5,10,5,7,14,13,12,3,3,14,7,11,
     13,11,7,15,15,15
```

Однако компьютер запоминает в программе место оператора, вызывавшего подпрограмму.

Когда выполнение подпрограммы закончится, оператор RETURN вернет управление обратно на то место, с которого была вызвана подпрограмма. В нашем случае — управление будет возвращено 100 строке. Таким образом, подпрограмму можно запускать с любого места программы желаемое число раз.

Подпрограмма, начинающаяся с 1000 строки, заполняет экран фрагментами узора, хранящимися в массиве K. Функция CHR ¤, находящаяся в 1030 строке, создаст из цифрового кода символ, который индицируется на экране. Так как коды графических символов от 128 до 143 (см. таблицу 3.3), то к кодам, считываемым из массива K, добавляется число 128. Функция INT, находящаяся в 50 строке, из чисел с дробной частью создает целые числа. Оператор PAUSE 100 останавливает на 2 секунды дальнейшее выполнение программы.

Программа заполняет узором весь экран (см. рис. 19). Во время индицирования каждого символа слышен короткий звук, тон которого зависит от кода символа. Затем компьютер, подождав не-

сколько секунд, начинает индицировать следующий узор. Программа останавливается нажатием клавиши BREAK.

2.6.13. *Использование обоих видов графики одновременно.*

Нижеприведенная программа использует графику и высокой и низкой разрешающей способности. Сначала введите и запустите первую часть программы. На экране программа, использующая операторы PLOT и DRAW, нарисует море, корпус корабля и паруса. Затем находящийся в 120 строке оператор PRINT изобразит флаг.

Корабль

```
10 BORDEP 0 : PAPER 1 : INK 6 : CLS
20 FOR Y=1 TO 23 STEP 2
30 PLOT 0,Y : DRAW INK 5; 255,0
40 NEXT Y
50 FOR I=0 TO 7
60 PLOT 15—1,24+1
70 DRAW INK 4; 58+3* 1,0
80 NEXT I
90 FOR X=—32 TO 39
100 PLOT 40,103 : DRAW X,—67
110 NEXT X
120 PRINT AT 8,3; INK 2; « ■■ »
130 PLOT 40,111 : DRAW OVEP 1; 0, —79
```

Когда введете и вторую часть программы и опять запустите программу, на экране постоянно будет сверкать лазерный луч, создающий на синем фоне желтые звездочки. Этот луч чертится от носа корабля до пикселя со случайными координатами X, Y. Эти координаты затем переводятся в координаты графики низкой разрешающей способности, указывающие позиции печатания звездочек.

```
140 LET X=RND* 167+88
150 LET Y=RND* 151+24
160 LET E=INT ((175—Y)/8)
170 LET S=INT (X/8)
180 PLOT 88,32 : DRAW OVER 1; X—88, Y—32
190 BEEP.01,X/5
200 PLOT 88,32 : DRAW OVER 1; X—88, Y—32
210 PRINT AT E,S; « * »
220 GO TO 140
```

Оператор OVER 1, используемый в 180 и 200 строках, позволяет

первому DRAW оператору начертить лазерный луч, а второму — погасить его, не меняя при этом остальной части картинки, а находящийся в 130 строке оператор OVER 1 при вычерчивании мачты корабля заставляет чертить линию цвета бумаги по поверхности, затушеванной цветом чернил.

2.6.14. FLAS, BRIGHT и INVERSE

Эти ключевые слова также помогают управлять цветом. После каждого из этих ключевых слов можно указать 0 или 1. Эти слова можно использовать в операторе PRINT, ставя при этом после них точку с запятой. FLASH 1 вынуждает квадратик символа (позицию) мигать — менять цвета чернил и бумаги. Если в этих операторах вместо 1 поставим 0, то будет восстановлен нормальный режим индикации.

Попробуйте эти операторы внедрить в уже опробованные программы. Например, в программе «Квадраты» в 50 строке квадратик замените звездочкой и одновременно присоедините строку:

```
15 INVERSE 1
```

Теперь звездочки на экране будут черными (цвета бумаги), напечатанными на цветных полосах (цвета чернил). Попробуйте и с INVERSE 0.

Чтобы посмотреть, как действуют BRIGHT и FLASH, внедрите в программу «Симметричные узоры» две такие строки:

```
15 BRIGHT 1
16 FLASH 1
```

Мигание экрана можно остановить, введя команду FLASH 0:CLS.

В этих примерах применяется воздействие на всю центральную часть экрана. Если надо ограничить действие FLASH, BRIGHT и INVERSE только для некоторых позиций экрана, то эти ключевые слова надо применять в операторе PRINT — тогда они будут влиять только на те символы, которые будут печататься с PRINT оператором.

2.6.15. *Как создать свои символы.*

Набор символов «ЭРУДИТА» не ограничивается только графическими и другими символами, расположенными на клавиатуре. В специальной зоне памяти могут быть записаны данные о символах, созданных Вами. Последние называются символами «графики,

установленной потребителем». В каждой программе можно создать 21 такой символ.

Каждый символ на экране создается в квадратике из 64 пикселей: восемь строк по восемь пикселей, т. е. один символ занимает одну позицию сетки графики низкой разрешающей способности.

2.6.16. *Создание графических символов.*

Прежде всего нарисуйте сетку из 64 клеток (8×8), как показано на рис. 20. Теперь, заштриховав некоторые квадратики, создайте свой символ. Каждый заштрихованный квадратик соответствует пикселю цвета чернил, а незаштрихованный квадратик соответствует пикселю цвета бумаги. Далее следует записать двоичные коды символа — каждый заштрихованный квадратик соответствует 1, а незаштрихованный — 0. Ниже приведен чертеж символа, изображающего человека.

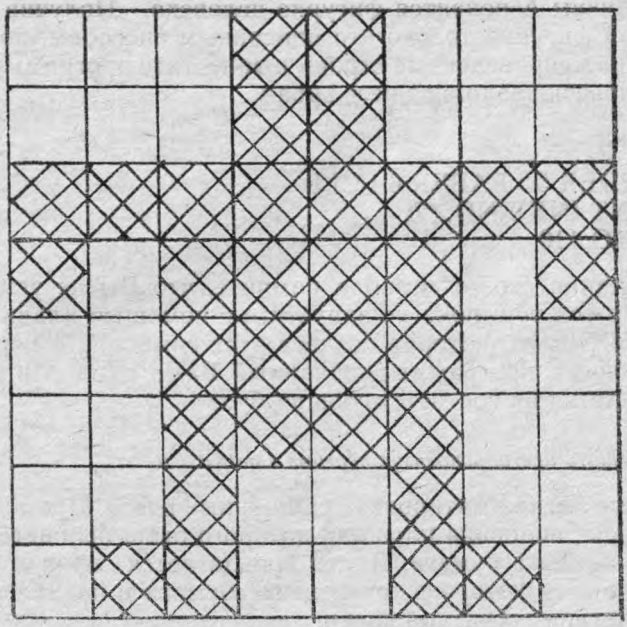


Рис. 20

Каждый символ графики, устанавливаемый потребителем, закрепляется за какой-либо буквенной клавишей от A до U. Коды

символов вводятся в память при помощи восьми операторов POKE USR, в которых указывается, за какой буквенной клавишей закрепляется вновь созданный символ и его двоичный код, начинающийся ключевым словом BIN. Ниже приведена часть программы, вводящая в память коды символа, изображающего человека, и закрепляющая этот символ за клавишей A.

```
10 POKE USR «A»,BIN 00011000
20 POKE USR«A»+1,BIN 00011000
30 POKE USR«A»+2,BIN 11111111
40 POKE USR «A»+3,BIN 10111101
50 POKE USR«A»+4,BIN 00111100
60 POKE USR «A»+5,BIN 00111100
70 POKE USR «A»+6,BIN 00100100
80 POKE USR «A»+7,BIN 01100110
```

Теперь запустите эту программу и нажмите клавиши GRAPH и A. Вместо буквы A появится фигурка человека. Получив символ человеческой фигурки только что описанным способом, добавьте к программе нижеприведенные строки и запустите программу. Разноцветные человечки заполнят весь экран.

Человечки

```
100 BORDER 5 : PAPER 6 : CLS
110 PRINT INK RND×7;
120 GO TO 110
```

При создании своего символа помните, что Вы не увидите его на экране до тех пор, пока не запустите с помощью клавиши RUN ту часть программы, которая создает этот символ. До первого запуска этой части программы, созданный Вами символ в листинге будет выглядеть как соответствующая буква.

2.6.17. *Как смешать цвета, используя пестрые квадратики.*

Вы можете легко имитировать смешанные цвета. Для этого надо создать символ, половина площади которого была бы цвета чернил, а половина — цвета бумаги. Далее предлагается программа, создающая символ, у которого точки цвета бумаги и чернил расположены как клетки шахматной доски.

```
10 FOR X=0 TO 6 STEP 2
20 POKE USR «A»+X, BIN 10101010
30 POKE USR «A»+X+1, BI 01010101
40 NEXT X
```

В программе указываются две строки символов, которые компьютер записывает в память четыре раза подряд.

Когда Вы запустите эту программу, то на экране увидите (нажав, конечно, GRAPH и A) пестрый квадратик. Если Вы примените такой метод в программе, которая использует ключевые слова установки цветов, то получите цвет, соответствующий смеси цветов бумаги и чернил.

2.6.18. *Упрощение установленной графики с помощью READ и DATA*

Создать свои графические символы легче, используя десятичные числа вместо двоичных и ключевые слова READ и DATA. Прежде всего надо перевести восемь двоичных чисел, состоящих из 0 и 1, в десятичные числа. Для этого можно использовать ключевые слова PRINT BIN, за которыми следует двоичное число, например:

```
PRINT BIN 00111100
```

После выполнения такой команды «ЭРУДИТ» индицирует на экране 60 — десятичный эквивалент двоичного числа 00111100. В случае символа, изображающего человека, восемью десятичными числами будут: 24, 24, 255, 189, 60, 60, 36, 102.

Теперь Вы можете применять операторы READ DATA, позволяющие легко оперировать большими массивами чисел, используемых в программе.

В операторе (или в нескольких операторах) DATA записываются числа или строки, разделенные запятыми. Операторы DATA могут быть в любом месте программы — это не влияет на ее работу. В операторе READ указывается название переменной (или несколько названий): цифровой переменной, если в операторе DATA записаны числа, либо название символьной переменной, если записаны строки.

Когда «ЭРУДИТ» встречает оператор READ, он находит первый имеющийся в программе оператор DATA, берет первое значение из его списка и пересылает его переменной, которая указана в операторе READ. В следующий раз оператор READ присваивает переменной следующее значение из списка DATA и т. д. Когда список одного оператора DATA кончается, значения берутся из следующего.

Ниже приведена программа, создающая тот же символ, изображающий человека.

```
10 FOR I=0 TO 7
20 READ S
```

```
30 POKE USR «A»+I,S
40 NEXT I
50 DATA 24, 24, 255, 189, 60, 60, 36, 102
```

Программа записывает в память восемь десятичных чисел, создающих символ. Если Вы хотите присвоить символ другой клавише, то замените в 30 строке A на желаемую букву, а если хотите создать другой символ, то в находящемся в 50 строке операторе DATA укажите другие восемь чисел, разделенных запятыми. Когда Вы захотите получить на экране созданный символ, запустите программу, а затем нажмите клавишу GRAPH и соответствующую буквенную клавишу.

2.6.19. *Вычерчивание шахматной доски.*

Здесь представлена программа, которая на экране вычерчивает шахматную доску, а затем располагает в исходных позициях фигуры. Вы можете ввести цветные символы, используя коды управления цветом (см. рис. 21). Сначала введите и запустите программу «Замена букв».

Замена букв

```
1 LET X=256* PEEK 23676+ PEEK 23675
2 FOR I=0 TO 167
3 POKE X+1, PEEK (15880+1)
4 NEXT I
```

Введите эту программу и запустите с помощью RUN, а затем уничтожьте (только не командой NEW), введя вместо нее четыре пустых строки.

Дальше введите следующую программу:

Шахматная доска

```
10 FOR I=1 TO 6
20 READ SO
30 GO SUB 1000
40 NEXT I
50 BORDER 0 : PAPER 5 : CLS
60 FOR E=7 TO 14 STEP 2 :
   FOR S=12 TO 19 STEP 2
70 PRINT AT E,S; „ ■
   AT E+I,S; „ ■ ”
80 NEXT S : NEXT E
```

```
90 PRINT AT 7, 12; «B A R Q K R A B»
100 PRINT AT 8, 12; «P P P P P P P P»
110 PRINT AT 13, 12; «P P P P P P P P»
120 PRINT AT 14, 12; «B A R Q K R A B»
130 STOP

1000 FOR J=0 TO 7
1010 READ W
1020 POKE USR S✱ +J,W
1030 NEXT J
1040 RETURN

2000 DATA «P», 0, 0, 16, 56, 56, 16, 124, 0
2010 DATA «B», 0, 84, 124, 56, 56, 124, 124, 0
2020 DATA «A», 0, 16, 56, 120, 24, 56, 124, 0
2030 DATA «R», 0, 16, 40, 68, 108, 56, 124, 0
2040 DATA «K», 0, 16, 56, 16, 56, 68, 56, 0
2050 DATA «Q», 0, 84, 40, 16, 108, 124, 124, 0
```

Коды графических символов, изображающих шахматные фигуры, даны в строках от 2000 до 2050: клавише P присваивается образ пешки, B — ладьи, A — коня, R — слона, K — короля, Q — королевы. Программа, записывающая эти коды в память, в зону графики, устанавливаемой потребителем, находится в строках от 1000 до 1040.

2.6.20. *Коды управления цветом.*

Для установления цвета можно вместо ключевых слов I K и PAPER использовать коды управления цветом, которые применяются в PPI T операторах после первых кавычек. В листинге после этих кодов появятся символы установленного цвета. Такой же цвет они будут иметь и во время выполнения программы.

Желая использовать коды управления цветом, прежде всего следует нажать клавишу EXTEND MODE, а затем цифровую клавишу одновременно с CAPS SHIFT или без нее. Ниже показано, как выбрать желаемый цвет.

2.7. МУЛЬТИПЛИКАЦИЯ

Мультипликация создается при передвижении по экрану символов или линий. Это сделать не сложно: надо просто менять позиции индикации символов или вычерчивания линий. Для этого удобнее всего использовать FOR NEXT циклы.

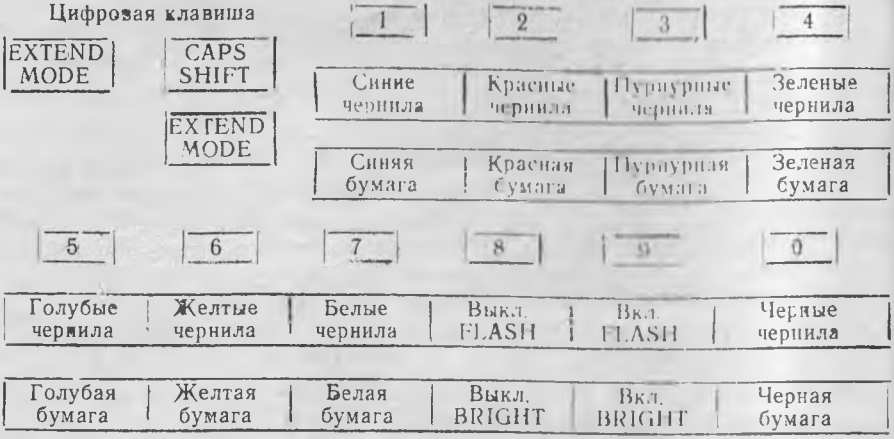


Рис. 21

2.7.1. *Вертикальные и горизонтальные движения.*

Введите и запустите приведенную ниже программу. Она создает два графических символа: клавише «A» присваивается образ водолаза, а «B» — выделяемых им пузырьков воздуха.

Погружающий водолаз

```
10 BORDER 3 : PAPER 5 : CLS
20 FOR X=0 TO 7
30 READ Y
40 POKE USR «A»+X, Y
50 NEXT X
60 DATA 24, 24, 255, 189, 60, 60, 36, 102
70 FOR X=0 TO 7
80 READ Y
90 POKE USR «B»+X, Y
100 NEXT X
110 DATA 96, 144, 144, 102, 9, 9, 6, 0
120 INPUT «СКОРОСТЬ : »; V
130 FOR E=0 TO 20
140 PRINT AT E, 4; INK 0:„ "
150 PRINT AT E+1, 4; INK 2; „ "
```

```
160 PAUSE V
170 NEXT E
```

После запуска программы в нижней части экрана сразу появится надпись из 120 строки — компьютер просит указать скорость движения. Введите какое-нибудь число от 1 до 50. Чем число меньше, тем скорость движения будет больше. После введения скорости на экране появляется водолаз, который опускается сверху вниз, оставляя после себя полосу пузырьков.

Графические символы создают операторы с 20 по 110 строку. Сама мультипликация индицируется при выполнении цикла, содержащегося в 130—170 строках. Во время выполнения цикла в одной строке индицируются пузырьки, а в следующей, ниже — водолаз. Затем команда, содержащаяся в 160 строке, заставляет компьютер подождать указанный Вами интервал времени (1 соответствует 1/50 сек.), и цикл повторяется с начала. Только теперь графические символы индицируются на одну строку ниже. Это выполняется до тех пор, пока водолаз не достигнет 21 строки.

А теперь присоедините к программе эти строки:

Идущий водолаз

```
180 FOR S=4 TO 30
190 PRINT AT 21, S; « »
200 PRINT AT 21, S+1; INK 2; «  »
210 PAUSE V
220 NEXT S
```

Теперь водолаз, опустившись вниз, начинает двигаться направо, пока не достигнет края экрана, т. е. 31 столбца.

Горизонтальным перемещением водолаза управляет цикл OP EXT, записанный в дополнительных строках. Во время выполнения каждого цикла на месте символа водолаза индицируется пробел. Затем водолаз индицируется в следующей позиции той же строки. Запомните, что всегда лучше сначала уничтожить символ в старой позиции перед тем, как индицировать его в новой позиции. Это позволит избежать мигания во время мультипликации.

2.7.2. *Столкновение фигур.*

Во многих компьютерных играх при столкновении двух фигур или при попадании в цель воспроизводят какие-нибудь дополнительные эффекты.

Если два символа индицируются соответственно в позициях E, (E — строка, -столбец), и V, H (V — вертикальная ось, H — горизонтальная ось), тогда, если E=V, а S=H, то они находятся в той же позиции. Это можно описать оператором

160 IF E=V AND S=H THEN PRINT «BUM!!!»

Другой способ установления столкновения — это проверка цвета символа. Если в позицию, в которой находится один символ, сындицируете другой символ иного цвета, то достаточно проверить не изменился ли в этой позиции цвет чернил. Теперь введите и запустите приведенную ниже программу, в которой как раз и используется этот метод.

В программе используются два символа: клавише «A» присваивается изображение человека, которое в предыдущей программе представляло водолаза, (теперь оно будет изображать парашютиста), а клавише «B» — взрыва. В различных местах экрана появились человечки — парашютисты, опускающиеся вниз, а в «небе» постоянно мерцает лазерный луч. Если он попадает в опускающегося парашютиста, то последний «взрывается».

Парашютисты

```
Строки 10 — 100 из программы «Опускающийся водолаз»
110 DATA 145, 82, 44, 121, 158, 52, 74, 137
120 INK 7
130 LET H=RND * 31
140 FOR V=0 TO 20
150 PRINT AT V, H,; «   »; AT V+1, H;
16 INK 0; «A»
160 LET X=RND * 255 : LET Y=RND * 167
170 PLOT 0, 8 : DRAW OVER 1; X, Y
180 BEEP 0.01,11
190 PLOT 0, 8 : DRAW OVER 1; X, Y
200 IF ATTR (V+I,H) =47 THEN GO TO 500
210 NEXT V
220 PRINT AT 21, H; INK 2; «A»
230 GO TO 130
500 PRINT AT V+1, H; FLISH I; PAPER 2; «B»
510 BEEP 0.5, —3
520 GO TO 130
```

Позицию парашютиста в цикле FOR NEXT указывают переменные V и H. Переменной H присваивается случайное значение (в 130 строке), поэтому парашютисты могут начинать свое падение с любой точки верхней строки. Парашютисты окрашены в черный

цвет, а лазерный луч — в желтый. Когда оператор DRAW, находящийся в 170 строке, прочерчивает лазерный луч через позицию, в которой находится парашютист, то последний становится желтым, как и луч. Далее находящаяся в 200 строке функция ATTR проверет цвета позиции, в которой находится парашютист. Если цвет бумаги голубой (код 5), а цвет чернил — желтый (код 6), то получаемое число 46 (код чернил плюс код бумаги, умноженный на 8), и управление передается 500 строке, с помощью которой имитируется взрыв — символ парашютиста заменяется символом взрыва.

2.7.3. *Отскакивание фигур.*

Во многих графических программах требуется, чтобы столкнувшиеся фигуры отскакивали от препятствия, например, от краев экрана. Нижеприведенная программа показывает, как это сделать. По краям экрана рисуются ворота: из букв «Л» — левые, из букв «П» — правые. По экрану — «футбольному полю», отскакивая от краев, перемещается мяч (буква «О»). При попадании мяча в ворота генерируется звуковой сигнал. Над воротами генерируется счет голов.

Футбол

```
10 BORDER 1 : PAPER 6 : CLS
20 FOR E=7 TO 14
30 PRINT AT E, 0; «Л»; AT E,31; «П»
40 NEXT E
50 LET RK=0 : LET RD=0
60 LET V=INT (RND * 20)+1 :
   LET H=INT (RND* 28)+2
70 LET X=1 : LET Y=1
80 PRINT AT V,H; OVER 1; «О»
90 PRINT AT 0,2; RK; AT 0, 27;
   RD : BEEP .2, 15
100 PRINT AT V,H; OVER 1; «О»
110 LET V=V+Y : LET H=H+X
120 IF V=0 OR V=21 THEN LET Y=—Y
130 IF H=0 OR H=31 THEN LET X=—X
140 IF SCREEN * (V,H)=«Л» THEN LET PK=PK+1 : GO
    TO 80
150 IF SCREEN * (V,H)=«П» THEN LET RD=RD+1 :
    GO TO 80
160 PRINT AT V, H; OVER 1; «О»
170 PAUSE 4 : GO TO 100
```

Цикл, описанный в 20—40 строках, рисует ворота. Переменная РК — результат попадания в левые ворота, RD — в правые, В 60 строке записаны начальные (случайные) координаты мяча. Координаты следующей позиции мяча записаны в 110 строке как результат прибавления к предыдущим координатам 1 или —1, т. е. значения переменных X и Y, которые, достигнув края экрана, изменяются на противоположные в строках 120 и 130. В следующих двух строках — 140 и 150 функция SCREEN $ проверяет — находится ли символ «Л» или «П» в позиции V, H, т. е. попал ли мяч в ворота. Если да, то увеличивается результат попадания в ворота и генерируется звуковой сигнал.

2.8. КАК СОЗДАТЬ МУЗЫКУ И ЗВУКОВЫЕ ЭФФЕКТЫ

«ЭРУДИТ» имеет звуковой синтезатор, который может значительно оживить Ваши программы, создавая музыкальные звуки широкого диапазона и специальные звуковые эффекты. Их легко применять, даже если Вы слабо разбираетесь в музыке. Звуковой сигнал, создаваемый компьютером, воспроизводится внутренним динамиком компьютера.

2.8.1. *Программирование звуков.*

Для синтезирования звука в «ЭРУДИТЕ» отводится один оператор — BEEP. В нем указываются два числа или две цифровые переменные, разделенные запятой. Первая указывает длительность синтезируемого сигнала в секундах, а вторая — высоту тона. Тон выражается в полутонах. Ноте «до» первой октавы соответствует 0, «до» соответствует 1, а «си» малой октавы соответствует —1 и т. д.

Запустите нижеприведенную программу и Вы услышите весь диапазон создаваемых «ЭРУДИТОМ» звуков.

```
10 FOR T=—60 TO 69
20 PRINT AT 11, 15; T; «   »
30 BEEP .3, T
40 NEXT T
```

«ЭРУДИТ» воспроизводит весь диапазон звуков от самого низкого тона (—60) до самого высокого (69). Вы, конечно, заметите, что самые низкие тона слышатся как щелчки, а самые высокие не слышны.

На рис. 22 приведены коды нотных тонов нескольких основных октав. Для получения ноты на полтона выше прибавьте к значению кода 1, а на полтона ниже — отнимите 1.

«ЭРУДИТ» позволяет получать различные звуковые эффекты. Чаще всего они получаются при размещении оператора BEEP внутри цикла и быстром изменении кода тона. Запустите нижеприведенные программы и Вы получите звуковые эффекты. Обратите внимание, что длительность звука очень малая — сотые доли секунды.

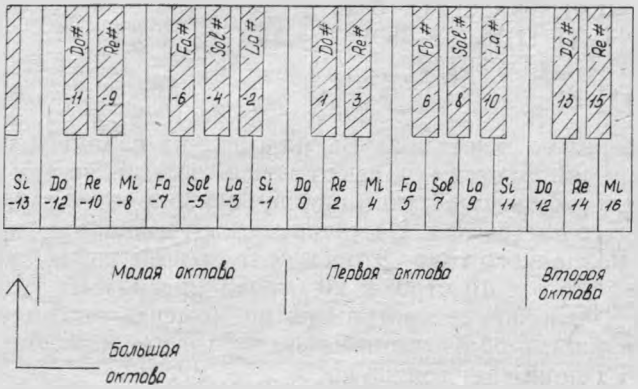


Рис. 22

Эти программы сами не заканчиваются, их надо остановить нажатием клавиши BREAK.

Бульканье

```
10 LET T=INT (RND*50)—25
20 BORDER (T+25)/7
30 BEEP .08,T : BEEP .08, T+10 : BEEP 08,T+5
40 GO TO 10
```

Эта программа воспроизводит постоянно три ноты со случайным кодом. Каждый раз, перед исполнением этих нот, меняется цвет контура, зависящий от высоты нот.

Двигатель

```
10 LET X=10 : LET I=1
20 BEEP .005,X
30 BEEP .005,24—X
40 LET X=X+1
50 IF X>38 OR X<10 THEN LET I=—1
60 GO TO 20
```

Эта программа поочередно создает два звука, из которых сначала один повышается, другой — понижается, а затем наоборот и т. д.

Пианино

```
10 LET T=CODE INKEY
20 IFT=0 THEN GO TO 10
30 PRINT AT 11,15; (T—30)/2; «  »
40 BEEP .04, (T—30)/2
50 GO TO 10
```

Эта программа ждет, пока Вы нажмете на клавишу, после чего на экране индицируется код высоты тона ноты, и до тех пор, пока Вы будете держать нажатую клавишу, генерируется звуковой сигнал соответствующего тона. Нажатие каждой из клавиш приводит к генерации различного тона. Это зависит от кода символа клавиши, устанавливаемого в 10 строке. 20 строка программы задерживает дальнейшее выполнение программы до момента нажатия какой-нибудь клавиши. Обратите внимание на то, что нажатие клавиши CAPS SHIFT понижает тон звука.

2.9. КАК СОХРАНИТЬ СВОИ ПРОГРАММЫ

Сохранить свои программы можно, записав их на магнитную ленту. Для этого необходимо подключить к «ЭРУДИТУ» кассетный магнитофон, как это было описано ранее. Для записи на магнитную ленту содержащейся в памяти компьютера программы применяется ключевое слово SAVE. Записанная на ленту программа может быть считана при помощи ключевого слова LOAD, о чем было описано ранее. Ниже показано, как записать программу на ленту и как проверить правильность записи.

2.9.1. *Запись своей программы на ленту*

1. Прежде всего подключите «ЭРУДИТ» к кассетному магнитофону, как это было показано в главе I.
2. Если кассетный магнитофон имеет регулятор уровня записи, установите его примерно на $^3/_4$ максимального уровня. Если такого регулятора нет, уровень записи магнитофона будет регулироваться автоматически.

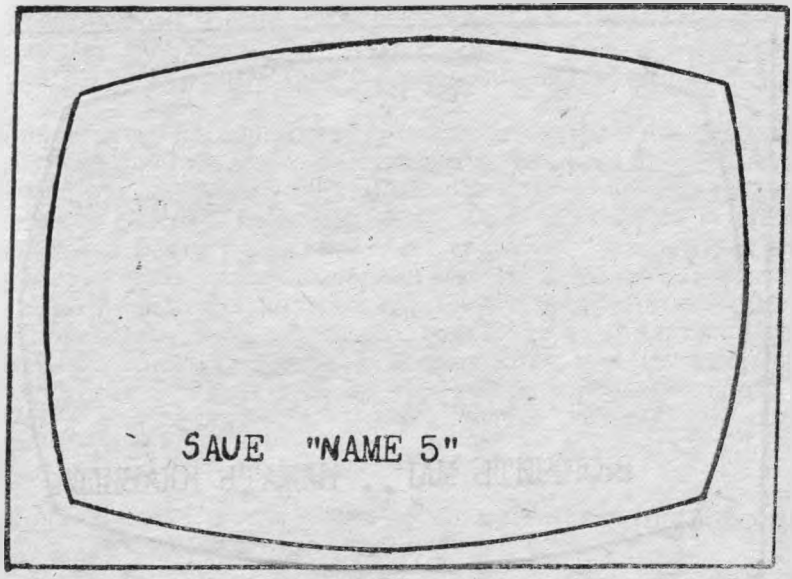


Рис. 23

3. Нажмите клавишу SAVE, затем в кавычках укажите имя выбранной Вами программы, например:

SAVE «NAME» 5»

Имя может быть представлено комбинацией букв и цифр, содержащей до десяти символов. Нажмите клавишу ENTER. Строка с оператором SAVE исчезнет, и появится команда запуска ленты.

4. Включите режим записи магнитофона, затем нажмите любую клавишу «ЭРУДИТА».

5. Теперь подождите, пока «ЭРУДИТ» запишет программу на ленту. Вначале Вы увидите синие и красные полосы, медленно перемещающиеся по экрану. Затем неожиданно на короткое время появятся синие и желтые линии, это «ЭРУДИТ» записывает на ленту имя программы.

6. После короткого перерыва вновь появятся синие и красные полосы, а затем — синие и желтые линии. Теперь «ЭРУДИТ» записывает непосредственно программу. На этот раз запись будет более длительной — в зависимости от объема программы.

7. Когда на ленту будет записана вся программа, на экране появится сообщение 0 ВЫПОЛНЕНО, 0:1. Остановите ленту. Теперь при желании можно проверить правильность проведенной записи.

Рис. 24

2.9.2. *Как проверить программу.*

Несмотря на то что компьютер записал программу на ленту, запись может оказаться неудачной. В «ЭРУДИТЕ» предусмотрена возможность ее проверки.

Процедура проверки называется верификацией. Прежде всего перемотайте ленту на начало записи программы. Соедините компьютер и магнитофон кабелем считывания. Теперь нажмите клавишу VERIFY и затем наберите в кавычках имя Вашей программы. Нажмите клавишу ENTER и клавишу воспроизведения. Вы снова увидите на экране ту же последовательность красных, синих и желтых линий. На экране появится название программы, которое сохраняется в течение времени верификации.

После того, как закончится вторичное мерцание синих и желтых линий, на экране появится сообщение:

0 ВЫПОЛНЕНО, 0:1

Это означает, что «ЭРУДИТ» сравнил программу, записанную на ленту, с программой, хранящейся в памяти, и установил их пол-

ную идентичность. Теперь Вам действительно ясно, что запись на ленте является хорошей.

Если Вы не получите такого сообщения, значит что-то не в порядке.

Прежде всего проверьте все по таблице «Устранение неполадок при загрузке программного обеспечения» (см. рис. 8), т. к. может оказаться, что программа записана хорошо, но во время верификации неверно считана. Если Вы обнаружили ошибку, то исправьте ее, перемотав ленту назад, и снова сравните программы. Если компьютер опять неудачно выполнил верификацию, то проверьте по таблице «Устранение неполадок при записи программного обеспечения», представленной на рис. 25, только не нажимайте кнопку сброса или, а также не выключайте компьютер, т. к. Вы уничтожите хранящуюся в памяти программу, не имея хорошей ее копии на ленте.

2.9.3. *Автоматический запуск программ.*

В команде SAVE после имени программы можно указать LINE с номером строки, например:

SAVE «ВОДОЛАЗ» LINE 1

В этом случае процедура записи остается та же, но во время верификации в команде VERIFY после имени программы — в нашем случае «ВОДОЛАЗ» — слово LINE с номером строки указывать нельзя. Программы, записанные при участии LINE 1, начнут выполняться автоматически после их считывания. В таком случае нет необходимости применять RUN (но не забудьте остановить ленту после начала выполнения программы).

Номер строки, указанный после LINE, указывает компьютеру, с какой строки необходимо запустить программу. Надпись LINE 1 годится для всех программ, запускаемых с первой строки. Если же номер первой строки программы больше единицы, то компьютер автоматически начнет выполнять программу со строки, номер которой больше единицы.

2.9.4. *Использование ключевых слов* CODE, SCREEN $ *и* DATA *во время записи*

Ключевое слово SAVE также может использоваться совместно с ключевыми словами CODE или SCREEN $. Это нужно тогда, когда необходимо записать на ленту часть содержания памяти «ЭРУДИТ», в том числе и информацию из экранной памяти, в которой хранится изображение, создаваемое на экране. Когда на ленту необходимо записать массив, ключевое слово SAVE также может

быть использовано вместе с ключевым словом DATA. Подробнее об этом смотрите в третьей главе книги.

УСТРАНЕНИЕ НЕПОЛАДОК ПРИ ЗАПИСИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

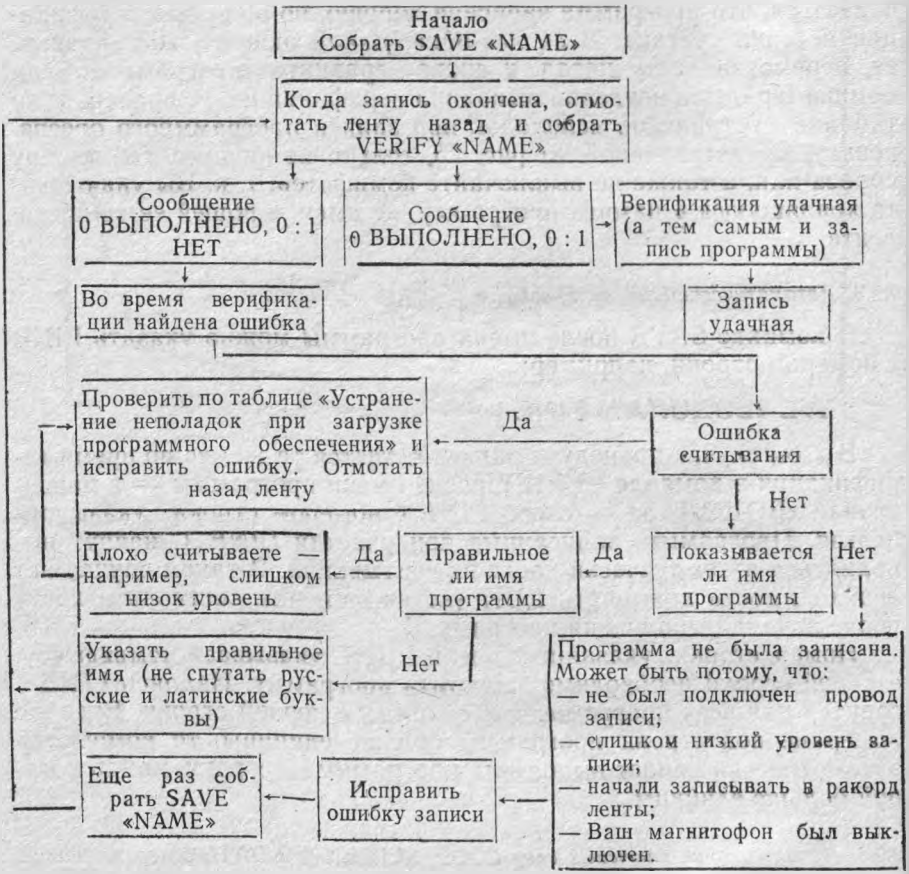


Рис. 25

Глава 3

БЕЙСИК «ЭРУДИТА»

3.1. УКАЗАНИЯ ПРОГРАММИСТУ ОТНОСИТЕЛЬНО КЛЮЧЕВЫХ СЛОВ БЕЙСИКА «ЭРУДИТА»

Ниже описаны все возможные ключевые слова «ЭРУДИТА», для каждого из которых указаны следующие данные:
набор на клавиатуре;
класс;
назначение;
использование;
формат.

3.1.1. *Классы ключевых слов*

Ключевое слово может принадлежать к одному из четырех возможных классов.

Команда

Это ключевое слово обычно вместе с атрибутами, которое указывает компьютеру на выполнение определенных действий и которое может быть использовано как непосредственная команда. Выполняется сразу после введения, т. е. после ENTER.

Например: RUN
PRINT A

Оператор

Это ключевое слово обычно вместе с атрибутами, которое указывает компьютеру на выполнение определенных действий и которое может быть использовано в строке программы. Выполняется только тогда, когда программа запускается оператором P.

Например: 10 PRINT A
20 CIRCLE X, Y, Z

Функция

Это ключевое слово, которое создает значение какого-то вида, функция представляет собой часть команды или оператора.

Например: ABS,
SIN

Логическая операция

Это ключевое слово, которое используется в командах и операторах для логического выражения. Оно используется при определении или изменении логических условий. «ЭРУДИТ» имеет три логические операции: AND, OR и NOT.

3.1.2. *Числа и переменные*

Числа

Точность чисел, хранящихся в памяти: 9 или 10 цифровых знаков.

Диапазон обрабатываемых чисел: от $4 * 10^{-39}$ до 10^{38}.

Используемые переменные

Цифровые. Имя любой длины, начинающееся буквой. Пробелы игнорируются.

Символьные. Имя из одной буквы, после которой идет « ».

Массивы. О переменных и индексах массивов сказано в описании оператора DIM.

3.1.3. *Формат ключевых слов*

Формат ключевых слов выражает синтаксис каждого ключевого слова, т. е. правильное сочетание ключевого слова и других факторов, таких как значение и переменные. В табл. 3.1 представлены сокращения, используемые при описании формата.

Таблица 3.1.

Сокращение	Пояснение	Пример
Цифр-конст	Цифровая константа (число)	30.4
Цифр-перем	Цифровая переменная (переменная, значение которой число)	PEREM
Цифр-выр	Цифровое выражение (любая возможная комбинация цифровых констант, переменных и ключевых слов, создаваемые значения которых — числа)	PEREM*3.1 RNO*8
Целочисл-конс Целочисл-перем Целочисл-выр	Цифра константа, переменная или выражение, значение которых округляется до ближайшего целого числа	

Продолжение

Сокращение	Пояснение	Пример
Симв-конст	Символьная константа или строка (любая комбинация символов между кавычками)	«УКРАИНА»
Симв-перем	Символьная переменная (переменная, значение которой — строка)	A¤
Симв-выр	Символьное выражение (любая возможная комбинация символьных констант, переменных и ключевых слов, создаваемые значения которых — строки)	A¤ + «SA» A¤ (5 TO9)
буква	Любая латинская или русская буква (кроме Ю, Ш, Щ, Э, Ч, Ъ и Е)	DИ
буква	Любая латинская или русская буква (кроме Ю, Ш, Щ, Э, Ч, Ъ и Е), после которой следует символ «¤»	P¤ Ж¤
условие	Условие	A=2 AND S<12
оператор	Любой оператор Бейсика, который можно применить вместе с другим оператором	IF 1=2 THEN STOP
[ ]	Конструкция ключевого слова, которую можно пропустить или которая может повторяться	

3.1.4. *Знаки Бейсика «ЭРУДИТА»*

Таблица 3.2

Знак	Действие/Применение	Знак	Действие/Применение
¤	Символьная переменная	=	Равно
'	Начинает печатание с новой строки	:	Отделяет операторы в одной строке программы
(	Открывающая скобка	/	Деление
)	Закрывающая скобка	*	Умножение
<=	Меньше или равно	.	Десятичная точка
<>	Неравно	;	Продолжает печатание со следующей позиции. Отделяет операторы, применяемые внутри другого оператора.
>=	Больше или равно		
<	Меньше		
>	Больше	"	Начинает и заканчивает строку
↑	Возведение в степень	,	Печатает с 0 или 16 позиции.
—	Вычитание (Отрицательное)		Отделяет значения, следующие после ключевого слова
+	Сложение (Положительное) Объединение		

Таблица 3.3

ТАБЛИЦА СИМВОЛОВ ГБК «ЭРУДИТ»

	0	1	2	3	4	5	6	7	8	9
0					TRUE VIDEO	INV VIDEO	PRINT запятая	EDIT	курс.←	курс.→
10	курс.↓	курс.↑	DELETE	ENTER	число	режим GRAPH	управл. INK	управл. PAPER	управл. FLAH	управл. BRIGHT
20	управл. INVERSE	управл. OVER	управл. AT	управ. TAB						
30			пробел	!	"	#	¤	%	&	'
40	(	)	*	+	,	−	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	Е	\	ъ	↑	_	Ю	А	Б	Ц
100	Д	Е	Ф	Г	Х	И	Й	К	Л	М
110	Н	О	П	Я	Р	С	Т	У	Ж	В
120	Ь	Ы	З	Ш	Э	Щ	Ч	©	□	■
130	▗	▄	▟	▙	▚	▜	▖	▞	▛	▜
140	▂	▆	▚	■	ГРАФИКА A	ГРАФИКА B	ГРАФИКА C	ГРАФИКА D	ГРАФИКА E	ГРАФИКА F
150	ГРАФИКА G	ГРАФИКА H	ГРАФИКА I	ГРАФИКА J	ГРАФИКА K	ГРАФИКА L	ГРАФИКА M	ГРАФИКА N	ГРАФИКА O	ГРАФИКА P
160	ГРАФИКА Q	ГРАФИКА R	ГРАФИКА S	ГРАФИКА T	ГРАФИКА U	RND	INKEY¤	PI	FN	POINT
170	SCREEN¤	ATTR	AT	TAB	VAL¤	CODE	VAL	LEN	SIN	COS
180	TAN	ASN	ACS	ATN	LN	EXP	INT	SQR	SGN	ABS
190	PEEK	IN	USR	STR¤	CHR¤	NOT	BIN	OR	AND	<=
200	>=	<>	LINE	THEN	TO	STEP	DEF FN	CAT	FORMAT	MOVE
210	ERASE	OPEN #	CLOSE #	MERGE	VERIFY	BEEP	CIRCLE	INK	PAPER	FLASH
220	BRIGHT	INVERSE	OVER	OUT	LPRINT	LLIST	STOP	READ	DATA	RESTORE
230	NEW	BORDER	CONTINUE	DIM	REM	FOR	GO TO	GO SUB	INPUT	LOAD
240	LIST	LET	PAUSE	NEXT	POKE	PRINT	PLOT	RUN	SAVE	RANDOMIZE
250	IF	CLS	DRAW	CLEAR	RETURN	COPY				

3.2. КЛЮЧЕВЫЕ СЛОВА И ИХ ИСПОЛЬЗОВАНИЕ

ABS — Абсолютное значение
Absolute value

Набор на клавиатуре
EXTEND MODE
G
Функция

ABS определяет абсолютное значение цифровой величины.

Как использовать ABS

После ABS указывается цифровая величина. Выражение должно быть заключено в скобки. Например:

```
50 LET A=ABS (B—C)
```

Пример

Команда

```
PRINT ABS—45.8
```

сындицирует на экране 45.8

Формат

```
ABS цифр-конст
ABS цифр-перем
ABS (цифр-выр)
```

ACS — Арккосинус
Arc Cosine

Набор на клавиатуре
EXTEND MODE
SYMBOL SHIFT W
Функция

ACS вычисляет значение угла по его косинусу.

Как использовать ACS

После ACS указывается цифровая величина. Выражение должно быть заключено в скобки. Например:

```
70 LET A=ACS(B/C)
```

Указываемая величина есть косинус нужного угла, она должна быть в пределах от —1 до 1. Результат функции ACS — значение угла, выраженное в радианах. Если хотите получить значение в градусах, его надо умножить на 180/PI.

Пример

Команда

```
PRINT 180/PI×ACS 0.5
```

Указываемая величина есть синус нужного угла и должна быть в пределах от —1 до 1. Но результат функции ASN — значение угла, выраженное в радианах. Если хотите получить значение в градусах, его надо умножить на 180/PI.

Пример

Команда
PRINT 180/PI* 0.5

сындицирует на экране 30 — угол, выраженный в градусах, синус которого 0.5.

Формат

ASN — цифр-конст
ASN — цифр-перем
ASN — (цифр-выр)

AT — Около
AT

Набор на клавиатуре

SYMBOL SHIFT I

См. INPUT, LPRINT, PRINT

ATN — Арктангенс
Arc TaNgent

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT E

Функция

ATN выдает значение угла по его арктангенсу.

Как использовать ATN

После ATN указывается цифровая величина. Выражение должно быть заключено в скобки, например:

70 LET A=ATN(B/C)

Указываемая величина и есть тангенс нужного угла. Результат функции ATN — значение угла, выраженное в радианах. Если хотите получить значение угла в градусах, его надо умножить на 180/PI.

Пример

Команда
PRINT 180/PI*ATN 1

сындицирует на экране 45 — угол, выраженный в градусах, тангенс которого 1.

Формат

ATN — цифр-конст
ATN — цифр-перем
ATN — (цифр-выр)

ATTR Атрибуты
ATTributes

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT I.

Функция

ATTR устанавливает атрибуты указанной на экране позиции символа. Атрибуты — это цвета чернил и бумаги, степень яркости и мерцания.

Как использовать ATTR

После ATTR указываются все цифровые величины, разделенные запятой и заключенные в скобки, например,

90 IF ATTR(V,H) = 124 THEN 0 TO 1000

Первая величина, идущая после ATTR — в нашем случае — это номер строки экрана, которая может принимать значения от 0 до 23. Вторая величина H — это номер столбца, значения которого могут быть от 0 до 31. Результат функции ATTR — число от 0 до 255. Это число — сумма кодов атрибутов, выполненная следующим образом:

Цвет чернил	Код цвета (0—7)
Цвет бумаги	Код цвета, умноженный на 8
Яркость	64
Мерцание	128

Пример

Если символ в позиции 19,20 имеет зеленый цвет чернил (код 4), белый цвет бумаги (код 7), увеличенную яркость, но не мерцает, то команда

PRINT ATTR (19,20)

сындицирует на экране 124 (4+8*7+64+0).

ATTR в двоичном формате

Результат ATTR — байт, в котором 7-й (старший) бит равен 1, если включено мерцание, и равен 0, если мерцание выключено. 6-й бит равен 1, если яркость увеличена. Если яркость нормальная, то 6-й бит равен 0. Биты от 5-го до 3-го указывают код цвета бумаги (в двоичном формате), а биты со 2-го до 0-го — код цвета чернил.

Формат

ATTR (цифр-выр, цифр-выр)

BEEP Свисток
BEEP

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT

Оператор/команда

_____ BEEP генерирует через громкоговоритель ноту с указанными длительностью и высотой тона.

Как использовать BEEP

BEEP используется в программе как оператор либо как непосредственная команда. После BEEP указываются две цифровые величины, разделенные запятой, например:

30 BEEP A, B

Первая величина (A) может быть в пределах от 0 до 10 и определяет длительность ноты в секундах. Вторая величина (B) может быть в пределах от —60 до 69 и устанавливает высоту тона ноты полутонами, 0 — это «до» первой октавы, нижние тона — со знаком минус, высокие — с плюсом.

Пример

Команда BEEP 1.5,2 сгенерирует ноту «ре» первой октавы длительностью в полторы секунды.

Формат

BEEP цифр-выр, цифр-выр

BIN — Двоичное число
BINary number

Набор на клавиатуре

EXTEND MODE
B

Функция

BIN переводит число из двоичной системы в десятичную.

Как использовать BIN

После BIN указывается двоичное число, состоящее из единиц и нулей (их может быть до 16), например,

60 POKE USR «B», BIN 11100101

Результат функции BIN — десятичное значение двоичного числа. Обычно функция используется вместе с POKE и USR, как показано выше, создавая устанавливаемые потребителем графические символы, когда 1 соответствует пикселю цвета чернил, а 0 — пикселю цвета бумаги.

Пример

Команда
PRINT BIN 10101110

сындицирует на экране число 174 — десятичное значение указанного двоичного числа.

Формат

BIN (1) (0)

BORDER Контур
BORDER

Набор на клавиатуре

B

Оператор/команда

BORDER определяет цвет контура вокруг рабочей части экрана.

Как использовать BORDER

BORDER может использоваться как непосредственная команда или как оператор в программе. После BORDER указывается цифровая величина, например:

40 BORDER 11/3

Величина, следующая после BORDER, является округленной до ближайшего целого числа и устанавливает следующие цвета контура:

0 черный,
1 синий,
2 красный,
3 пурпурный,
4 зеленый,
5 голубой,
6 желтый,
7 белый.

Помните, что BORDER также устанавливает цвет нижней части экрана. В отличие от INK и PAPER BORDER не может быть использован (размещен) в операторе PRINT.

Формат

BORDER целочисл-выр

BRIGHT Яркость
BRIGHT

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT B

Оператор/команда

BRIGHT индицирует указанные символы более ярко.

Как использовать BRIGHT

BRIGHT может быть использована в качестве непосредственной команды, но обычно используется как оператор в программе. После BRIGHT указывается цифровая величина, например:

```
90 BRIGHT 1
```

Указываемая величина округляется при необходимости до ближайшего целого числа и может иметь значения 0, 1 или 8. BRIGHT 1 дает более яркую окраску символам, выводимым после этого операторами PRINT или INPUT. BRIGHT 8 указывает, что яркость символов должна остаться такой, какой она была до использования оператора PRINT или INPUT. BRIGHT 0 устраивает действия BRIGHT 1 и BRIGHT 8, и далее символы индицируются с нормальной яркостью.

BRIGHT также может быть размещен внутри операторов индикации: PRINT, INPUT, PLOT, DRAW и CIRCLE. В этом случае BRIGHT указывается после ключевого слова, но перед данными или перед параметрами индикации. Сразу после BRIGHT указывается ранее установленная цифровая величина, после которой следует запятая, например:

```
30 PRINT BRIGHT 1, «ВНИМАНИЕ!»
```

В этом случае влияние BRIGHT имеет частотный характер и применяется только для индикации символов, точек и линий этого оператора. Помните, что BRIGHT 1 увеличивает яркость всей позиции символа (8×8) даже, если Вы в ней высвечиваете только одну точку.

Формат

BRIGHT целочисл-выр (;)

CHR $	Символьная строка ChaRacter atring

Набор на клавиатуре

EXTEND MODE
V

Функция

Все существующие на клавиатуре ключевые слова и символы, а также определенные потребителем графические символы составляют набор символов «ЭРУДИТА». Используя CHR ¤ и номер кода символа, можно получить строчное отображение каждого символа. В наборе символов есть также несколько символов, управляющих индикацией символов. Эти коды могут использоваться в программе. Перед CHR ¤ используют PRINT. Полный набор символов и номера кодов представлены в табл. 3.3.

Как использовать CHR

После CHR ¤ указывается цифровая величина, например:

30 PRINT CHR ¤ A

Если после CHR ¤ имеется выражение, то оно заключается в скобки. Величина, следующая после CHR ¤ (в нашем случае A), округляется до ближайшего целого числа. Если она находится в пределах от 32 до 255, то результат CHR ¤ — символ клавиатуры, графический символ, устанавливаемый потребителем, или ключевое слово символического вида. Коды употребляемых символов «ЭРУДИТА» от 32 до 95 и от 97 до 126 соответствуют ASCII кодам. Если в данном примере переменной A присваиваем значение 77, то оператор на экране выдаст индикацию «M».

CHR ¤ и управляющие коды

CHR ¤ со значением от 1 до 31 или возвращает обратно управляющие коды, или вообще их не использует. CHR ¤ 6 (PRINT запятая), CHR ¤ 8 (позиция назад) и CHR ¤ 13 (новая строка или ENTER) используются в операторе PRINT и управляют индикацией на экране. После CHR ¤ в значениях указанного кода может следовать точка с запятой, например,

70 PRINT «A»; CHR ¤ 6; «B»

Этот оператор выдаст индикацию на экране:

A B

Использовать управляющие коды CHR ¤ можно и иначе — выстраивая из них составную строку. Оператор

```
70 PRINT «A»+CHR ¤ 6+«B»
```

дает тот же эффект, что и приведенный ранее пример.

Коды от 16 до 23 управляют цветами и позициями. Каждый из них может использоваться в составной строке. После CHR ¤ 16 (INK управление) и CHR ¤ 17 (PAPER управление) должны следовать CHR ¤ с указанным кодом цвета (от 0 до 7), а CHR ¤ 18 — CHR ¤ 21 (FLASH, BRIGHT, INVERSE и OVER управление) должны применяться вместе с CHR ¤ 0 или CHR ¤ 1.

Команда

```
PRINT CHR ¤ 16+CHR ¤ 2+CHR ¤ 17+CHR ¤ 18 ¤ 6+CHR ¤
+CHR ¤ 1+«ЭРУДИТ»
```

сындицирует на экране надпись «ЭРУДИТ», мерцающую красным и желтым цветами.

Как и в случае, рассмотренном ранее, каждый знак плюс может быть заменен точкой с запятой.

После CHR 2 (AT управление) должны следовать два ключевых слова CHR со значениями, указывающими номера строки и столбца.

Команда

```
PRINT CHR ¤ 22+CHR 11+CHR ¤ 16+CHR¤ 67
```

отпечатает в середине экрана «С».

После CHR ¤ 23 (TAB управление) также должны следовать два ключевых слова CHR ¤ со значениями: первое указывает позицию TAB, а второе обычно равно 0.

Команда

```
PRINT CHR ¤ 23+CHR ¤ 16+CHR ¤ 0+CHR¤ 67
```

отпечатает в середине экрана «С».

Помните, что можно использовать только эти управляющие коды. Используя PRINT CHR ¤ со значением кода ключевого слова бо́льшим, чем 164, обычно индицируем ключевое слово, но оно не выполняется.

Формат

```
CHR¤ целочисл-конст (;) (+)
CHR¤ целочисл-перем (;) (+)
CHR¤ (целочисл-выр) (;) (+)
```

CIRCLE Окружность
CIRCLE

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT H

Оператор/команда

CIRCLE чертит на экране окружность.

Как использовать CIRCLE

После CIRCLE указываются три цифровые величины, разделенные запятыми, например,

20 CIRCLE X, Y, R

Каждое из них округляется до ближайшего целого числа, CIRCLE чертит окружность в сетке графики высокой разрешающей способности чернилами заданного цвета. Первые две величины (X,Y) указывают координаты центра, а третья величина (R) определяет длину луча. Эти величины должны быть такими, чтобы окружность могла поместиться в рабочей части экрана.

CIRCLE, как и PLOT или DRAW, может управлять операторами или командами определения цвета или может иметь эти операторы внутри.

Пример

Команда
CIRCLE 128, 88, 85

чертит окружность почти во весь экран.

Формат

CIRCLE (операт;) целочисл-выр, целочисл-выр, целочисл-выр

CLEAR Чистить
CLEAR

Набор на клавиатуре

X

Оператор/команда

CLEAR стирает значения всех существующих переменных, освобождая всю память до конечного адреса системной зоны Бейсика, хранящегося в ячейке RAMTOP CLEAR также может быть использован для определения RAMTOP.

Как использовать CLEAR

CLEAR используется как непосредственная команда и как оператор в программе. Никаких параметров указывать не нужно, например,

```
90 CLEAR
```

В этом случае CLEAR стирает все присвоенные переменным значения, включая и массивы. Он также выполняет действия команд CLS и RESTORE, очищая экран и устанавливая указатель данных около первого элемента списка данных. Кроме того, первая графическая точка, используемая в операторе DRAW, ставится в левом нижнем углу экрана, и очищается стек GO SUB.

Помните, что, заново описывая уже существующие массивы, не надо использовать CLEAR, поскольку DIM уничтожает уже существующий массив с таким же именем.

Также учтите, что RUN выполняет действие CLEAR.

CLEAR и RAMTOP

В реальном случае после CLEAR может быть указана цифровая величина, например,

```
CLEAR 65317
```

В этом случае CLEAR выполняет все то, что было указано ранее, кроме того, определяет в ячейке RAMTOP, хранящей конечный адрес системной зоны Бейсика, указанное значение. Включив питание или нажав кнопку RESET, «ЭРУДИТ» автоматически определяет значение RAMTOP, равное 65367, т. е. в конце памяти остается только зона, зарезервированная для графики, устанавливаемой потребителем (адрес последней ячейки памяти 65535). Команда NEW очищает память до адреса, указанного RAMTOP, поэтому, определяя с помощью CLEAR наименьший адрес RAMTOP (в данном

вают, то с помощью CONTINUE ее можно снова запустить с места остановки. Если программа останавливается из-за ошибки, то ее нужно исправить перед тем, как использовать CONTINUE.

Как использовать CONTINUE

При остановке программы CONTINUE используется как непосредственная команда. Не надо указывать никаких параметров. После использования CONTINUE программа продолжается с того оператора, выполняя который она остановилась.

Если остановка произошла из-за ошибки, то перед употреблением CONTINUE с помощью другой команды можно исправить ошибку. Если программа остановилась, выполняя оператор STOP, и выдала сообщение 9 или была остановлена нажатием клавиши BREAK (сообщение L), то CONTINUE продолжает программу со следующего оператора.

Если CONTINUE используется для продолжения непосредственной команды, то возможны три случая:

непосредственная команда была остановлена во время выполнения своего первого оператора. Теперь CONTINUE зациклится, т. е. будет снова и снова исполнять сама себя. Не будет индикации никакого сообщения, но управление может быть возвращено нажатием клавиши BREAK;

непосредственная команда была остановлена во время выполнения своего второго оператора. Тогда CONTINUE выдаст сообщение 0;

непосредственная команда была остановлена во время выполнения ее третьего или любого последующего оператора. Тогда CONTINUE выдаст сообщение N.

Формат

CONTINUE

COPY Копировать
COPY

Набор на клавиатуре

Z

Команда

COPY дает указание печатающему устройству «ЭРУ-

ДИТА» сделать копию информации, которая индицируется на экране телевизора.

Как использовать COPY

COPY используется как непосредственная команда тогда, когда программа полностью выполнена или остановлена. Никаких параметров указывать не надо. После COPY, если не подсоединено печатающее устройство, печатаются первые 22 строки с индикацией их на экране. Помните, что все точки любого цвета чернил печатаются черным цветом, а точки любого цвета бумаги не печатаются вообще.

Печатание может быть остановлено клавишей BREAK.

Если на экране есть листинг программы, то его можно отпечатать, используя COPY, если этот листинг был создан с применением команды или оператора LIST. Помните, что «автоматический» листинг, когда после выполнения программы или ее остановки нажата клавиша ENTER, не может быть отпечатан при помощи COPY.

Формат

COPY

COS Косинус
COSine

Набор на клавиатуре

EXTEND MODE
W

Функция

COS вычисляет косинус угла.

Как использовать COS

После COS указывается цифровая величина, например,

90 LET A=COS B

Выражение должно быть заключено в скобки. Указываемая после COS величина — это угол, выраженный в радианах. Результат функции COS — косинус указанного угла. Градусы могут быть превращены в радианы умножением на PI/180.

Помните, что результатом функции COS является отрицательный угол от 90 до 270 градусов и положительный угол от 0 до 90 и от 270 до 360 градусов.

Пример

Команда
PRINT COS (60* PI/180)

индицирует на экране 0.5 — косинус угла в 60 градусов.

Формат

COS цифр-выр
COS цифр-перем
COS (цифр-выр)

D A T A Данные
DATA

Набор на клавиатуре

EXTEND MODE
D

Оператор

DATA устанавливает список элементов данных внутри программы. Элементы могут быть цифровые или символьные. Каждый элемент данных присваивается переменной при помощи оператора READ.

Присвоение осуществляется в том порядке, в каком данные записаны в программе, но оператором RESTORE можно установить начало присвоения с любого оператора DATA.

Как использовать DATA

DATA может быть использован только в программе как оператор. Обычно после него указывается список цифровых и символьных констант, разделенных между собой запятыми, например,

50 DATA 1200, «ВЕНЕРА», 6790, «МАРС»

После этого каждая константа присваивается переменной оператором READ, который считывает данные оператора DATA. Операторы DATA могут быть в любом месте программы. Число констант,

их вид (цифровая или символьная) и порядок расположения должны соответствовать числу и виду переменных и их последовательности в операторе READ. Если список данных очень длинный, то он может быть разбит на несколько операторов DATA, следующих один за другим.

Пример

Представленная ниже программа

```
10 FOR N=1 TO 2
20 READ A, B$
30 PRINT B$ , A; «КМ»
40 NEXT N
50 DATA 12100, «ВЕНЕРА», 6790, «МАРС»
```

индицирует на экране диаметр планет.

```
ВЕНЕРА   12100 КМ
МАРС      6790 КМ
```

Использование DATA с переменными

Элементы данных в операторе DATA могут быть цифровыми или символьными переменными, или выражениями, но переменным перед этим должны быть присвоены какие-нибудь значения. В представленном выше примере оператор DATA может быть заменен следующим:

```
50 DATA S, P$, S—5310, «МАРС»
```

Если перед этим переменной S присваиваем значение 12100, а переменной P$ — «ВЕНЕРА», то на экране получим то же изображение, как и в предыдущем примере.

LOAD/SAVE/VERIFY DATA

DATA также может использоваться вместе с LOAD, SAVE и VERIFY при записи массивов на магнитную ленту. Об этом смотрите в соответствующих описаниях.

Формат

```
DATA цифр-выр (, цифр-выр) (, симв-выр)
DATA симв-выр (, цифр-выр) (, симв-выр)
```

DEF FN Определить функцию
DEFine FuNction

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT 1

Оператор

DEF FN позволяет потребителю установить свою функцию, которой нет среди ключевых слов «ЭРУДИТА». Можно передать функции множество параметров оператором FN, вызывающим функцию, результат которой может быть или цифровым или символьным.

Как использовать DEF FN

DEF FN может использоваться только как оператор в программе. Если определяется цифровая функция, то после DEF FN употребляются одна буква, а потом — одна или несколько цифровых переменных, разделенных между собой запятыми и заключенных в скобки, например, DEF FN (A, B). Далее следуют знак равенства и цифровое выражение, в котором применяются указанные переменные, например,

```
100 DEF FN(A, B)=SQR(A↑2+B↑2)
```

Буква, следующая после DEF FN (в примере — S), есть имя определяемой функции. Заглавные и строчные буквы не разделяются.

Оператор DEF FN может находиться в любом месте программы.

Для вызова установленной функции используется оператор FN. После него сразу указываются буква имени функции и далее список цифровых величин, в котором величины разделены между собой запятыми и заключены в скобки, например,

```
50 PRINT FNS (7, 2)
```

Величины, записанные в скобках, передаются переменным функции в таком порядке, в каком они записаны в операторе DEF FN. В нашем случае переменной A присваивается значение 7, а B — 2. Результат функции FN — значение, рассчитанное по выражению.

После DEF FN можно указывать только букву имени и пару скобок, например,

```
100 DEF FN A ( )=SQR(A+5.5)
```

Во время вызова функции с помощью FN ей передается значение переходной переменной (в примере — A).

DEF FN и строки

Операторы DEF FN и FN также могут использоваться для определения символьных функций и их вызова. В этом случае имя функции одна буква, после которой следует символ «¤ », а в списке переменных оператора хоть одна переменная должна быть символьной, т. е. имя должно быть обозначено буквой с символом «¤ ».

Для определения функции используется символьное выражение. например,

```
100 DEF FN A¤ (B¤,J,I) = B¤ (J TO 1)
```

В операторе FN должны быть указаны имя функции, а далее символьная величина вместе с другими параметрами, которые надо передать функции. В нашем случае команда:

```
PRINT FN A¤ («ПРОСТОТА», 4, 6)
```

индицирует на экране

```
СТО
```

Формат

```
DEF FN буква ((буква) (, буква)) = цифр-выр
DEF FN буква¤ ((буква¤) (, буква) (, буква) (, буква ¤) =
=симв-выр
FN буква ((цифр-выр) (, цифр-выр))
FN буква ¤ ((симв-выр) (, цифр-выр) (, цифр-выр) (, симв-выр))
```

DIM — Определение объема
DIMension

Набор на клавиатуре

D

Оператор

DIM используется для описания объема массивов. Массив — это список переменных (цифровых или символьных), имеющих одно имя. Эти переменные или элементы массива отличаются индексами — величинами, которые указывают место элемента в массиве.

Как использовать DIM

DIM используется в программе как оператор. После него следуют одна буква, которая является именем массива, а далее — одна или несколько цифровых величин, разделенных между собой запятыми и заключенных в скобки, например,

```
60 DIM A(15)
70 DIM B(10,3)
```

В первом случае создается одномерный цифровой массив, имеющий 15 элементов с индексами от 1 до 15. Массив имеет имя A, а переменные с индексами находятся в пределах от A(1) до A(15). Любой ранее существовавший массив с таким именем уничтожается, а всем переменным массива присваиваются нулевые значения. Помните, что имена массивов нужно писать латинскими буквами.

Число величин, указанных в скобках, — это число измерений создаваемого массива. Во втором примере создастся двухмерный массив, имеющий 30 элементов: одно измерение массива — 10 элементов, другое — 3. Элементы этого массива нумеруются от B(1,1) до B(10,3).

Можно создавать массивы с любым количеством измерений.

Элементы массива идентифицируются следующим образом: указываются имя массива, а затем одна или несколько величин, заключенных в скобки, например:

```
90 PRINT  A(I)
100 PRINT B(8,J)
```

DIM и символьные массивы

Оператор DIM используется также и для определения символьных массивов, только в этом случае именем массива является буква с символом «$». Кроме того, в скобках должна быть указана еще одна величина, определяющая длину каждой строки, например,

```
60 DIM A$ (15,4)
70 DIM B$(10,3,6)
```

Первый оператор создает массив с 15 элементами, каждый из которых — строка длиной в 4 символа. Именам переменных с индексами от A$ (1) до A$ (15) присваиваются пустые строки. Любой существовавший ранее массив с этим уничтожается. И еще одна особенность: в этом случае не может существовать символьная переменная с таким же именем, какое дано символьному массиву.

Второй оператор создает двухмерный массив, имеющий 30 эле-

ментов: одно измерение массива — 10 элементов, другое — 3. Все элементы длиной в 6 символов.

Когда элементам символьного массива присваиваются символьные величины, то, если присваивается строка короче заданной длины, в конце прибавляются пробелы, а если длиннее — она урезается до нужного числа символов. Элементы символьного массива идентифицируются так: указывается имя массива, а затем в скобках одна или несколько цифровых величин, указывающих индексы. Например, A$ (2) элемент может быть «МЕТР», а элемент B$ (10,2) «ГРАДУС». Тем не менее может быть указана дополнительная величина, которая выделяет из строки один символ. В наших примерах A$ (2,2) будет «Е» (второй символ в строке «МЕТР»), а B$ (10,2,5) — «У».

Символ-массивы нулевого измерения

Можно создать символ-массив нулевого измерения, указывая в скобках только одну величину, например:

```
20 DIM C$ (11)
```

Этот массив имеет только один элемент — C$, длина которого фиксирована (в нашем случае 11 символов).

Формат

```
DIM буква (цифр-выр(, цифр-выр))
DIM буква$ (цифр-выр(, цифр-выр))
```

DRAW — Чертить линию

DRAW

Набор на клавиатуре

W

Оператор/команда

DRAW используется для вычерчивания на экране прямых и кривых линий.

Как использовать DRAW

Обычно DRAW используется в программе как оператор. Если

чертится прямая линия, то после DRAW указываются две величины, разделенные запятой, например,

```
70 DRAW X,Y
```

В этом случае на сетке высокой разрешающей способности чертится прямая линия, начинающаяся с точки, указанной в предыдущем графическом операторе: это или точка, сындицированная посредством PLOT, или точка, до которой чертил линию предыдущий оператор DRAW. Оба числа, указанные после DRAW, при необходимости округляются до ближайших целых чисел. Они указывают, до какого места надо чертить линию. Однако это не координаты конечной точки: первая величина (X) — это длина линии (в точках) по оси X координат, а вторая (Y) — по оси Y координат, т. е. изменения координат конечной точки относительно начальной точки. Эти величины могут быть отрицательными, если линия чертится влево или вниз, а конечная точка обязательно должна быть в рабочей части.

Если перед этим не было никакого графического оператора, то DRAW чертит линию с нижнего левого угла (координаты 0, 0).

Цвета линий, вычерчиваемых DRAW, можно определить с помощью операторов или команд управления цветом. Как в PLOT или CIRCLE операторах, внутри DRAW могут быть размещены некоторые другие операторы.

Вычерчивание кривых с помощью DRAW

Если надо начертить кривую — часть окружности, то после DRAW может быть указана и третья величина, например,

```
70 DRAW X, Y, Z
```

В этом случае вместо прямой линии чертится дуга, степень выгиба которой определяет третья величина (Z). Это угол в радианах, который получается при соединении концов дуги с центром окружности, частью которой является эта дуга.

Если угол положительный, то дуга чертится против часовой стрелки, а если отрицательный — то по часовой стрелке, т. е. дуга будет выгнута в обратную сторону.

Пример

Далее представлена программа для вычерчивания ромба

```
10 PLOT 120,140
20 DRAW 80,—50
```

```
30 DRAW —80,—50
40 DRAW —80,50
50 DRAW 80,50
```

Если в конце оператора добавите «1», то края ромба вогнутся вовнутрь, а если «—1», то выгнутся наружу.

Формат

DRAW (операт);целочисл-выр, целочисл-выр(целочисл-выр)

EXP — Экспоненциальная функция
EXPonent

Набор на клавиатуре

EXTEND MODE
X

Функция

EXP — это математическая функция, которая возводит число в указанную степень.

Как использовать EXP

После EXP указывается цифровая величина, например,

```
90 LET A=EXP B
```

Выражение должно быть заключено в скобки. EXP возводит число в степень, которая указана как аргумент (B).

Пример

Команда
PRINT EXP 1

сындицирует на экране 2.7182818, т. е. значение числа e.

Формат

EXP цифр-конст
EXP цифр-перем
EXP (цифр-выр)

FLASH — Мерцание
FLASH

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT V

Оператор/команда

FLASH заставляет позицию символа на экране мерцать, т. е. менять цвета чернил и бумаги с постоянной частотой. Подробнее об этом описано во второй главе.

Как использовать FLASH

FLASH можно использовать как непосредственную команду, но обычно она используется в качестве оператора в программе. После FLASH указывается цифровая величина, например,

```
30 FLASH 1
```

Указанная величина округляется до ближайшего целого числа и может иметь значения 0, 1 или 8. FLASH 1 вызывает мерцание символов, индицируемых посредством PRINT или INPUT. Оператор FLASH 8 оставляет в позициях символов прежние управляющие значения мерцания. Оператор FLASH 0 уничтожает действие предыдущих операторов FLASH 1 и FLASH 8, поэтому все индицируемые после этого символы не мерцают.

Оператор FLASH также может быть размещен внутри операторов, управляющих индикацией на экране: PRINT, INPUT, PLOT, DRAW и CIRCLE. В этом случае FLASH указывается после ключевого слова, но перед данными и параметрами индикации. Сразу после FLASH указывается ранее описанная цифровая величина, после которой следует точка с запятой, например,

```
90 PRINT FLASH 1; INK 3; PAPER 7; «ЭРУДИТ»
```

В данном случае воздействие FLASH носит ограниченный характер и служит лишь для индикации символов, точек и линий данного оператора.

Учтите, что FLASH 1 заставляет мерцать позицию всего символа (8×8), даже в том случае, если высвечивается только одна точка.

Формат

FLSH целочисл-выр (;)

FN FuNction
Функция

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT 2

Функция

FN вызывает установленную потребителем функцию. Она всегда используется вместе с оператором DEF FN, который определяет эту функцию.

Как использовать FN

Если вызывается цифровая функция, то после FN указываются буква, а за ней пара скобок. Если функции надо передать какие-либо параметры, то они указываются в скобках и разделяются запятыми, например,

```
100 LET X=FN S 7,2)
```

Параметры 7 и 2 передаются функции, имя которой S. Результат функции с помощью оператора LET присваивается переменной X. Даже если не надо передавать никаких параметров, скобки все равно должны быть, например,

```
200 PRINT FN S ( )
```

В этом случае функция использует значения переменных, которые в данный момент им присвоены.

Таким же образом FN вызывает и символьную функцию, только с буквой имени нужно еще указать символ « » (смотрите описание DEF FN).

FN не может использоваться рекурсивно, т. е. вызывать саму себя.

Формат

```
FN буква ( (цифр-выр) (, цифр-выр))
FN буква   ((симв-выр) (, цифр-выр) (, цифр-выр) (, симв-выр))
```

FOR — От FOR

Набор на клавиатуре

F

Оператор/команда

FOR всегда вместе с ключевыми словами TO и NEXT позволяет организовать многократное исполнение фрагмента программы, заключенного между ними.

Как использовать FOR

FOR с TO составляют один оператор. После FOR указываются буква, знак равенства, а затем две цифровые величины, разделенные ключевым словом TO, например,

```
80 FOR I=1 TO 10
```

Буква I — имя переменной цикла.

Далее следуют операторы, которые необходимо повторять. В своих выражениях они могут использовать указанную переменную цикла I. Цикл кончается оператором NEXT, в котором после ключевого слова NEXT указывается имя переменной цикла, например,

```
120 NEXT I
```

Начав действовать, оператор FOR уничтожает любую переменную, имеющую такое же имя, как и переменная цикла, и присваивает переменной цикла начальное значение, которое указано после знака равенства перед TO (в нашем случае I). Далее выполняются следующие операторы, пока не достигается оператор NEXT. Теперь значение меньше или равно пределу — величине, указанной после TO (в нашем случае — 10), программа возвращается к оператору FOR, и цикл FOR NEXT повторяется еще раз. Если переменная цикла достигает значения, превышающего предельное, то цикл кончается, и программа переходит к оператору, следующему после NEXT.

В приведенном примере цикл повторяется 10 раз, а переменная цикла получает значения от 1 до 10. По окончании цикла ее значение равно 11. Помните, что «ЭРУДИТ» не различает заглавные и строчные буквы, употребляемые в качестве имени переменной цикла.

ИСПОЛЬЗОВАНИЕ КЛЮЧЕВОГО СЛОВА

STEP В ЦИКЛЕ FOR NEXT

К оператору FOR может быть присоединено ключевое слово STEP. Оно дает возможность увеличить переменную цикла на какое-то значение, отличное от 1, или ее не увеличивать, а уменьшать

STEP указывается после предельного значения, а за ним указывается еще одна величина, например,

```
80 FOR I=1 TO 10 STEP 1.5
```

В этом случае переменная цикла каждый раз увеличивается на значение, указанное после STEP — на шаг, который в данном примере 1.5, пока не станет больше предельного значения. Переменная цикла получает значения 1, 2.5, 4, 5.5, 7, 8.5 и 10. По окончании цикла — 11.5.

Отрицательное значение шага каждый раз уменьшает значение переменной цикла. В этом случае начальное значение должно быть больше предельного. Цикл кончается тогда, когда значение переменной цикла становится меньше предельного, например,

```
80 FOR I=10 TO 1 STEP —1
```

Значение переменной I уменьшается от 10 до 1, а по окончании цикла равно 0.

Циклы, вложенные один в другой

Один или несколько циклов FOR NEXT могут быть вложены один в другой. Так можно вкладывать сколько угодно циклов, только порядок указания переменных цикла в операторах NEXT должен строго соответствовать обратной очередности операторов FOR.

Формат

FOR буква-цифр-выр TO цифр-выр (STEP цифр-выр)

NEXT буква

GO SUB — Перейти к программе
GO to SUBroutine

Набор на клавиатуре

H

Оператор/команда

GO SUB указывает программе на переход к выполнению подпрограммы. Подпрограмма — это отдельная часть программы. Это целесообразно, если та же подпрограмма в программе нужна несколько раз.

Как использовать GO SUB

GO SUB может использоваться как оператор или как непосредственная команда. После GO SUB указывается цифровая величина, например,

```
GO SUB 1000
```

Во время исполнения эта величина округляется до ближайшего целого числа, и программа переходит к строке, на номер которой указывает эта величина (в нашем случае — к строке 1000). Если используемая величина является переменной или выражением, то программа может сама рассчитывать, к какой подпрограмме перейти. Если строки с указанным номером нет, то программа все равно переходит к ближайшей строке с большим номером.

Программа обязательно должна оканчиваться ключевым словом RETURN. Тогда программа, после выполнения подпрограммы, возвращается к оператору, следующему после GO SUB. Подпрограмма может вызывать другую подпрограмму и т. д. В этом случае RETURN возвращает программу обратно к оператору, следующему после последнего исполненного оператора GO SUB.

Стек GO SUB

Когда выполняется оператор GO SUB, номер его строки помещается в стеке GO SUB, находящемся в памяти. Если перед RETURN исполнены два или несколько операторов GO SUB, то номера их строк записываются в стеке один за другим так, чтобы номер последнего исполненного оператора GO SUB был вверху стека. Оператор RETURN всегда берет номер строки сверху из стека и возвращает программу именно к этой строке.

Помните, что ошибка «4 НЕ ХВАТАЕТ ПАМЯТИ» может появиться в случае, если в программе недостаточно операторов RETURN.

Формат

```
GO SUB целочисл-выр
```

GO TO	Переход к GO TO

Набор на клавиатуре

G

Оператор/команда

GO TO указывает программе на переход к указанной строке.

Как использовать GO TO

GO TO может использоваться как непосредственная команда при запуске программы со строки с указанным номером, не очищая при этом экрана. После GO TO указывается цифровая величина, например,

```
80 GO TO 200
```

Во время исполнения указанная величина округляется до ближайшего целого числа, и программа переходит к строке, номер которой указывает эта величина. Если используемая величина является переменной или выражением, то программа может сама рассчитать, на какую строку перейти. Учтите, если строки с указанным номером нет, то программа все равно переходит к первой строке с большим номером.

Формат

GO TO целочисл-выр

IF — Если IF

Набор на клавиатуре

V

Оператор/команда

IF всегда используется с ключевым словом THEN, когда надо решить, как дальше выполнять программу. Компьютер выполняет это, проверив какое-нибудь условие: истинно оно или ложно. Если условие истинно, то выполняются одни действия, если нет — другие.

Как использовать IF THEN

После IF указывается цифровая величина или условие. Далее следует THEN и один или несколько операторов Бейсика, например,

```
30 IF A THEN GO TO 200
110 IF B$ = «НЕТ» THEN PRINT «КОНЕЦ» : STOP
```

Константа, переменная (А) или выражение признаются истинными, если они имеют ненулевое значение. Тогда выполняются операторы, следующие в той же самой строке за THEN. Далее программа переходит к следующей строке.

Если значение константы, переменной или выражения являются нулевыми, то они считаются ложными. Тогда операторы, следующие за THEN, не исполняются — программа сразу переходит к следующей строке.

В приводимом примере программа не переходит на 200 строку, если значение А будет равно 0.

Во втором примере после IF идет проверка условий. Если условие истинное, т. е. В ¤ =«НЕТ», то выполняются операторы, следующие после THEN. Если условие ложное, то программа сразу переходит к следующей строке. В приведенном примере, если символьная В ¤ имеет значение «НЕТ», то появится индикация записи «КОНЕЦ», и программа остановится. Если В ¤ имеет другое значение, то программа будет продолжаться с другой строки.

Для истинного условия «ЭРУДИТ» устанавливает значение, равное 1, а для ложного — 0. Любое ненулевое значение она принимает, как истинное, а нулевое — как ложное. Условие может быть присвоено переменной с помощью следующего оператора:

```
20 LET A=B ¤ =«НЕТ»
```

Обратите внимание на то, что в отличие от других вариантов Бейсика здесь перед GO TO нельзя пропустить THEN.

Формат

IF цифр-выр THEN операт (: операт)
IF условие THEN операт (: операт)

IN B
IN

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT X

Функция

IN фиксирует состояние клавиатуры или другого устройства ввода или вывода. Она считывает байт из порта с указан-

ным адресом. Этот байт указывает состояние устройства, присоединенного к указанному порту.

Как использовать IN

После IN указывается цифровая величина, например,

100 LET A=IN B,

которая должна быть в пределах от 0 до 65535. Она указывает адрес порта, из которого должен быть считан байт. Тогда результат функции IN, который присваивается оператором LET переменной A, будет байт, считанный из порта, указанного переменной B.

Формат

IN цифр-конст
IN цифр-перем
IN (цифр-выр)

INK — Чернила
INK

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT X

Оператор/команда

INK устанавливает основной цвет, в который окрашиваются индицируемые символы, точки или линии.

Как использовать INK

INK может использоваться как непосредственная команда, но обычно он используется как оператор в программе. После INK указывается цифровая величина, например,

90 INK A

Указанная величина округляется до ближайшего целого числа и может быть в пределах от 0 до 9. Она устанавливает следующие основные цвета (чернил):

0 черный,

1 синий,
2 красный,
3 пурпурный,
4 зеленый,
5 голубой,
6 желтый,
7 белый,
8 прозрачный,
9 контрастный (черный или белый).

Оператор INK 8 указывает, что в позиции экрана оставляется указанный ранее цвет чернил. Оператор INK 9 делает цвет чернил черным или белым, в зависимости от цвета фона (бумаги) так, чтобы индицируемый символ был бы ярче виден.

Общее и частное определение цвета чернил

Когда INK используется как отдельный оператор, определение цвета является общим для всех последующих операторов индикации. INK также может быть размещен в каком-либо из операторов индикации: PRINT, INPUT, PLOT, DRAW или CIRCLE. В этом случае INK указывается после ключевого слова, но перед данными или параметрами индикации. Сразу после INK указывается описанная ранее цифровая величина, после которой следует точка с запятой, например,

```
90 PLOT INK 2; 128, 88
```

В данном случае воздействие INK носит ограниченный характер и служит лишь для индикации символов, точек и линий данного оператора.

Формат

INK целочисл-выр(;)

INKEY ¤	Ввести строку клавиши INput KEY string

Набор на клавиатуре

EXTEND MODE
N

Функция

INKEY ¤ используется для определения нажатой клавиши.

Как использовать INKEY$

После INKEY$ не нужно указывать никакого параметра. Она в основном используется тогда, когда символьной переменной надо присвоить символ клавиатуры, или если надо проверить нужная ли клавиша символа нажата, например,

```
60 LET A$ =INKEY $
TO IF INKEY$ =«D» THEN GO TO 170
```

Результат функции INKEY$ — символ, который во время выполнения функции вводится с клавиатуры. Если в этот момент не нажата ни одна клавиша, то результат — пустая строка (« »). Учтите, что INKEY$ различает заглавные и строчные буквы, а также другие символы, набранные какими-нибудь клавишами SHIFT или без них. Если для Вас не важно, какой символ клавиши набран, а важно только какая клавиша нажата, то используйте функцию IN.

В отличие от INPUT, INKEY$ не ждет, пока будет нажата клавиша. Она сразу же переходит к следующему оператору. Если надо ждать нажатия клавиши, то обычно INKEY$ размещается в цикле, который повторяет ее до тех пор, пока не будет нажата клавиша.

Пример

Представленная далее строка прерывает выполнение программы до тех пор, пока не будет нажата клавиша «Т» (без CAPS LOCK или CAPS SHIFT):

```
90 IF INKEY$ <> «T» THEN GO TO 90
```

Формат

INKEY $

INPUT Ввести
INPUT

Набор на клавиатуре

I

Оператор/команда

_____ INPUT позволяет вводить данные во время выполнения программы.

Как использовать INPUT

INPUT обычно используется как оператор в программе. Характер его применения очень похож на характер применения оператора PRINT. В простейшем случае после INPUT указывается цифровая или символьная переменная, например,

```
80 INPUT A
90 INPUT B$
```

В это время компьютер ждет, пока будет введено число или строка. Вводимое значение индицируется в начале нижней строки. После нажатия ENTER введенное значение присваивается указанной переменной, и программа продолжается дальше.

В операторе INPUT может быть указано и больше переменных, а также он может сындицировать вспомогательные комментарии. Это делается таким же образом, как и в операторе PRINT, используя в начале и в конце комментариев кавычки, а для разделения элементов оператора — запятые или точки с запятыми. Операторы индицирования такие, как INK, FLASH или PAPER могут быть размещены в операторе INPUT, например,

```
100 INPUT INK 4; «ВАШЕ ИМЯ?»;
V$,(«СКОЛЬКО ВАМ, «+V$ +», ЛЕТ?»);M
```

Обратите внимание на различия в операторе PRINT. Все переменные и выражения, которые создают комментарии, должны быть заключены в скобки. Оператор INPUT ждет, пока будет введено одно значение, и только после ввода индицирует комментарии, находящиеся перед следующей переменной и т. д. Индикация начинается в нижней строке, а затем, если требуется больше строк, предыдущие перемещаются вверх.

Ключевое слово AT в операторе INPUT применяется так же, как и в PRINT. Указав AT 0,0, получим индикацию в начале второй строки внизу. Если для индикации требуется больше двух строк, то ранее индицированные строки перемещаются вверх.

Как остановить выполнение INPUT

Если оператор INPUT указан цифровыми величинами, то остановить программу можно путем введения ключевого слова STOP. Если указана символьная переменная, то остановить программу можно, уничтожив первые кавычки, а потом ввести STOP.

Использование INPUT с LINE

Оператор INPUT LINE может использоваться только с сим-

вольными переменными. Обычно INPUT с символьной переменной, ожидая вводимой строки, индицирует двойные кавычки. Когда строка вводится с клавиатуры, она появляется на экране, заключенная в эти кавычки. Если применять INPUT LINE с именем символьной переменной, то индикации кавычек не будет. Если нужны вспомогательные комментарии, то они должны быть указаны между INPUT и LINE, например,

100 INPUT «КАК ВАШЕ ИМЯ?» LINE V$

Формат

INPUT (комментарии) (;) (,) (') цифр-перем
INPUT (комментарии) (;) (,) (') симв-перем
INPUT (комментарии) (;) (,) (') LINE симв-перем
(комментарии) = (симв-конст) (симв-выр))
(AT целочисл-выр, целочисл-выр)
(операт) (;) (,) (')

INT — Целое число
INTeger

Набор на клавиатуре

EXTEND MODE
R

Функция

INT обращает числа, имеющие дробную часть, в целые числа.

Как использовать INT

После INT указывается цифровая величина, например,

30 LET A=INT B

Выражение должно быть заключено в скобки. Результат функции INT — значение, округленное до ближайшего наименьшего целого числа. Для отрицательных чисел меньшим является число, имеющее большее абсолютное значение.

Пример

Команда
PRINT INT 98.65, INT —4.95

сындицирует на экране
98 —5

Формат

INT цифр-конст
INT цифр-перем
INT (цифр-выр)

INVERSE Обратный
INVERSE

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT M

Оператор/команда

INVERSE меняет местами цвета позиций символа, т. е. цвет чернил становится цветом бумаги и наоборот.

Как использовать INVERSE

Обычно INVERSE используется в программе как оператор. После него указывается цифровая величина, например,

30 INVERSE 1

Указанная величина округляется до ближайшего целого числа и может иметь значения 0 или 1. Оператор INVERSE 1 меняет местами цвета чернил и бумаги у всех последующих символов, которые индицируются с PRINT или INPUT, а INVERSE 0 возвращает первоначальные цвета.

Помните, что INVERSE так же, как и INK, может быть размещен в операторах индикации. Тем не менее, используя оператор INVERSE с операторами CIRCLE, PLOT и DRAW, оператор INVERSE 1 индицирует точку или чертит линию цвета бумаги, но ее не видно.

Формат

INVERSE целочисл-выр

LEN Длина строки
LENgth of string

Набор на клавиатуре

EXTEND MODE
K

Функция

LEN устанавливает длину символьной строки.

Как использовать LEN

После LEN указывается символьная величина, например,

90 LET A=LEN B$

Выражение должно быть заключено в скобки. Результат LEN — значение, указывающее число символов в строке.

Пример

Далее представленная строка программы принимает только те строки, длина которых не превышает 11 символов:

100 INPUT B$: IF LEN B$>11 THEN GO TO 100

Формат

LEN симв-конст
LEN симв-перем
LEN(симв-выр)

LET — Пусть
LET

Набор на клавиатуре

L

Оператор/команда

LET применяется, когда переменной надо присвоить значение. В операторе присвоения в Бейсике «ЭРУДИТА» ключевое слово LET пропускать нельзя.

Как использовать LET

Обычно LET используется как оператор в программе, но может быть использован и как непосредственная команда. После LET указываются цифровая или символьная переменная, далее знак равенства, например,

```
30 LET A=A/8
90 LET B$ ="ЭРУДИТ"
```

Во время исполнения указанная величина присваивается переменной.

Помните, что значения обыкновенных переменных не описываются, пока они не установлены операторами LET, READ или INPUT. Тем не менее во время определения массива с помощью DIM переменным массива присваиваются нулевые значения (или пустые строки).

Формат

LET цифр-перем=цифр-выр
LET симв-перм=симв-выр

LINE — Строка
LINE

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT 3

Смотрите описание SAVE.

LIST — Список
LIST

Набор на клавиатуре

K

Команда/оператор

LIST создает листинг программы, которая в данный момент находится в памяти компьютера.

Как использовать LIST

LIST обычно используется как непосредственная команда или как оператор в программе. Если нужен листинг всей программы, то ключевое слово LIST используется без каких-либо параметров. После непосредственной команды LIST на экране появляется первая страница листинга. При нажатии какой-нибудь клавиши (за исключением N, пробел, STOP и BREAK), сдвигая предыдущую страницу вверх, получаем новую страницу.

После LIST может быть указана любая цифровая величина — номер страницы, например,

LIST 80

Указанная величина округляется до ближайшего целого числа, и листинг начинается с указанной строки. Если строки с указанным номером нет, то листинг начинается с первой строки с большим номером.

Формат

LIST (целочисл-выр)

LL IST — Список с печатающим устройством
Line printor LIST

Набор на клавиатуре

EXTEND MODE
V

Команда/оператор

LL IST печатает на печатающем устройстве листинг программы, которая в данный момент находится в памяти компьютера.

Как использовать LL IST

LL IST используется в тех случаях, что и LIST (подробнее смотрите в описании LIST). Помните, что во время печатания листинга, тот листинг, который находится на экране, не меняется.

Формат

LLIST целочисл-выр

LN	Натуральный логарифм Logariphm (Natural)

Набор на клавиатуре

EXTEND MODE
Z

Функция

LN подсчитывает натуральный логарифм указанной величины (с основанием e). Функция LN является обратной к функции EXP.

Как использовать LN

После LN указывается цифровая величина, например,

90 LET A=LN B

Выражение должно быть заключено в скобки. Указанная величина должна быть больше нуля. Тогда результат функции LN — натуральный логарифм указанной величины.

Формат

LN цифр-конст
LN цифр-перем
LN (цифр-выр)

LOAD	Загрузить LOAD

Набор на клавиатуре

J

Команда/оператор

LOAD считывает программу с магнитной ленты и записывает ее в память компьютера.

Как использовать LOAD

Обычно LOAD используется как непосредственная команда или как оператор в программе. После LOAD указывается имя файла (являющееся символьной величиной), длиной до десяти символов, например,

LOAD «ИМЯ»

Во время выполнения LOAD существующая в памяти компьютера программа и ее переменные уничтожаются. «ЭРУДИТ» ищет на магнитной ленте программу с указанным именем, а обнаружив ее, записывает в память. Помните, что компьютер различает в именах программы русские и латинские буквы.

Если после LOAD указывается пустая строка, как в команде

LOAD, "",

то «ЭРУДИТ» считывает первую найденную программу.

Формат

LOAD симв-выр

LOAD CODE Загрузить коды
LOAD CODE

Набор на клавиатуре

J
EXTEND MODE
I

Команда/оператор

LOAD CODE используется для заполнения зоны информацией, считанной с магнитной ленты. Считываемая информация — это последовательность байтов, которые записываются в память по указанным адресам. Команда LOAD CODE может использоваться, например, для заполнения экранной памяти или записи информации о созданных потребителем графических символах и т. п.

Как использовать LOAD CODE

LOAD CODE может использоваться как непосредственная

команда или как оператор в программе. После LOAD указываются имя файла, а затем ключевое слово CODE, например,

LOAD «ДАННЫЕ» CODE

Имя файла, следующее после LOAD, — это имя считываемой информации, и к нему применимо все то, что сказано об именах программ (смотрите LOAD). Во время исполнения компьютер ищет информацию с указанным именем, а обнаружив ее, выдает на экране индикацию записи «БАЙТЫ:», после которой следует имя.

Затем «ЭРУДИТ» считывает байты с магнитной ленты и записывает в память по тем адресам, по которым они хранились до записи на ленту. Любая находящаяся там информация уничтожается.

После CODE может быть указана одна или две цифровые величины, разделенные запятой, например,

LOAD «ГРАФИКА» CODE 65368, 168

Указанные величины округляются до ближайшего целого числа. Первая величина определяет начальный адрес (в примере — 65368 — начало зоны графики, установленной потребителем), по которому будет записана информация, а вторая — число считываемых байтов (168 байтов — 21 графический символ). Если это число указано неверно, то выдается сообщение об ошибке при считывании с ленты. Если после CODE указывается только одна величина, то она устанавливает начальный адрес зоны памяти, а считываются и записываются в память все байты, записанные на ленту с указанным именем.

Как записывать байты на ленту, смотрите в описании SAVE CODE.

Формат

LOAD симв-выр CODE (целочисл-выр) (, целочисл-выр)

LOAD DATA — Загрузить данные
LOAD DATA

Набор на клавиатуре

J
EXTEND MODE
D

Оператор/команда

LOAD DATA используется для считывания массивов с магнитной ленты и записи их в память компьютера. На ленту массивы записываются с помощью ключевого слова SAVE DATA.

Как использовать LOAD DATA

LOAD DATA может использоваться в программе как оператор или как непосредственная команда. После LOAD указываются имя файла, а затем ключевое слово DATA и буква или буква со знаком и, наконец, пара пустых скобок, например,

```
300 LOAD «ЧИСЛА» DATA S ( )
350 LOAD «ИМЕНА» DATA V¤ ( )
```

Имя файла, указываемое после LOAD, — это имя массива на ленте, и к нему применимо все то, что сказано об именах программ (смотрите описание LOAD). Буква или буква со знаком ¤, указанная после DATA, — это имя массива, присваиваемое считанному и записанному в память массиву.

Во время выполнения компьютер ищет на ленте массив с указанным именем, а обнаружив его, индицирует на экране надпись «ЧИСЛОВОЙ МАССИВ:» или «СИМВОЛЬНЫЙ МАССИВ:», после которого следует имя. Далее «ЭРУДИТ» считывает массив и записывает его в память. Любой массив, находившийся в этот момент в памяти с тем же именем, уничтожается, и создается новый массив по данным с ленты.

Помните, что если считывается символьный массив, то уничтожается также и простая символьная переменная, имеющая такое же имя.

Формат

LOAD симв-выр DATA буква (¤) ( )

LOAD SCREEN ¤ — Загрузить экран
LOAD SCREEN

Набор на клавиатуре

J
EXTEND MODE
SYMBOL SHIFT K

Оператор/команда

LOAD SCREEN ✲ позволяет непосредственно с магнитной ленты считать и записать в память индицируемую информацию. Информация записывается непосредственно в зону экранной памяти.

Как использовать LOAD SCREEN ✲

Ключевые слова LOAD SCREEN ✲ могут использоваться в программе как оператор или как непосредственная команда. После LOAD указывается имя файла, а затем ключевое слово SCREEN ✲, например,

LOAD «ИЗОБРАЖЕНИЕ» SCREEN ✲

Имя файла, следующее после LOAD, — это имя экранной информации на ленте. К нему применимо все, что сказано об именах программ (смотрите описание LOAD). «ЭРУДИТ» ищет информацию с указанным именем, а обнаружив ее, считывает и записывает в экранную память: сначала информацию изображения, потом — информацию о цвете, мерцании и яркости. Во время считывания на экране медленно создается новое изображение.

Как записать индицируемую на экране информацию на магнитофонную ленту, смотрите в описании SAVE SCREEN ✲.

Формат

LOAD симв-выр SCREEN ✲

LPRINT — Печать на печатающем устройстве
Line printer PRINT

Набор на клавиатуре

EXTEND MODE
C

Оператор/команда

LPRINT дает указание печатающему устройству «ЭРУДИТА» печатать элементы данных таким образом, как PRINT индицирует их на экране.

Как использовать LPRINT

LPRINT может использоваться как оператор в программе или

как непосредственная команда. После LPRINT указываются элементы печатаемых данных, которые между собой разделены запятыми, точкой с запятой или апострофами, например,

```
90 LPRINT «НОМЕР»; N' «ИМЯ»; V¤, «ВОЗРАСТ»; A
```

Во время выполнения элементы данных печатаются таким же форматом, каким они индицируются на экране оператором PRINT. В составе оператора или команд LPRINT могут быть размещены также операторы TAB, некоторые символы управления CHR ¤, операторы INVERSE и OVER — аналогично случаю PRINT. Так же может использоваться оператор AT, но в этом случае номер строки игнорируется, а элементы данных печатаются в той же строке с указанного столбца.

Формат

LPRINT (TAB целочисл-выр;) (AT целочисл-выр, целочисл-выр;)
(CHR (целочисл-выр);) (операт;)
(цифр-выр) (симв-выр) (;) (,) (')

MERGE — Соединить

MERGE

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT T

Оператор/команда

MERGE позволяет соединить две программы в одну.

Как использовать MERGE

MERGE может использоваться как оператор в программе или как непосредственная команда. После MERGE указывается имя файла, например,

```
200 MERGE «ИГРА»
```

Имя файла — это имя программы, которая должна быть соединена с программой, находящейся в данный момент в памяти компьютера. К этому имени применимо все, что было сказано об име-

нах программ (смотрите описание LOAD). Во время выполнения MERGE считывает с магнитной ленты и записывает в память указанную программу, не уничтожая имеющуюся в памяти программу. Новая программа уничтожает строки с теми самыми номерами, а также переменные с теми же именами.

Формат

MERGE симв-выр

NEW — Новый NEW

Набор на клавиатуре

A

Команда/оператор

NEW очищает зону памяти Бейсика до адреса, который хранится в ячейке RAMTOP («верх памяти»), уничтожая любую находящуюся в этой зоне программу.

Как использовать NEW

NEW обычно используется как непосредственная команда или как оператор в программе. Ключевое слово NEW используется самостоятельно без каких-либо параметров. Во время выполнения уничтожаются программа и ее переменные, находящиеся в памяти до указанного RAMTOP адреса. При этом не уничтожаются только устанавливаемые потребителем графические символы, поскольку они хранятся в самом конце памяти, обычно за адресом, указанным RAMTOP.

Формат

NEW

NEXT — Следующий NEXT

Набор на клавиатуре

N

Оператор/команда

NEXT всегда используется с ключевым словом FOR, когда надо создать циклы FOR NEXT.

Как использовать NEXT

Обычно NEXT используется в программе как оператор, заканчивающий цикл FOR NEXT. После NEXT указывается буква — имя переменной цикла, например

100 NEXT I

В Бейсике «Эрудита» имя переменной цикла должно быть обязательно указано.

Подробнее о цикле FOR NEXT смотрите в описании FOR.

Формат

NEXT буква

NOT Нет NOT

Набор на клавиатуре

SYMBOL SHIFT S

Логическая операция/функция

NOT в большинстве случаев используется при изменении значения условия на противоположное, т. е. истинное условие становится ложным и наоборот.

Как использовать NOT

После NOT указывается условие или цифровая величина, например

```
80 IF NOT A=B—C THEN PRINT «ПЛОХО»
90 LET ПЕРЕМ=A=B—C : IF NOT ПЕРЕМ THEN PRINT
   «ПЛОХО»
```

Когда после NOT указывается условие (в примере A=B—C), то компьютер сначала устанавливает значение условия, равное 1, если оно истинно, или 0, если оно ложно. Дальше NOT действует как функция, изменяющая значение условия на противоположное.

Помните, что условие должно быть заключено в скобки, если в нем используется логический оператор AND или OR. Если после NOT указывается цифровая величина, то результатом NOT является 0, если указанная величина является цифровой величиной, не равной 0 или —1, если величина равна 0.

В данных примерах, если A=B—C или переменная ПЕРЕМ будет иметь значение, равное 0, то компьютер выдаст индикацию записи «ПЛОХО».

Формат

NOT условие
NOT цифр-выр

OR Или
OR

Набор на клавиатуре

SYMBOL SHIFT V

Логическая операция/функция

OR используется как логическая операция проверки комбинации условий. Если одно или несколько условий истинны, то истинна и вся комбинация. Как функция OR используется при выполнении двоичных операций с двумя цифровыми величинами.

Как использовать OR

Как логическая операция OR соединяет два условия в операторе, в котором проверяется правильность условий комбинации, например,

80 IF INKEY$ = «G» OR INKEY$ = «Г» THEN GO TO 200

Если одно или оба условия истинны, то истинна и вся комбинация. В данном примере какое-либо условие из двух становится истинным, если нажимается клавиша «G», а вся комбинация становится истинной независимо от того, были ли нажаты клавиши CAPS SHIFT или CAPS LOCK, и программа переходит на 200 строку.

OR как функция.

«Эрудит» присваивает цифровое значение, равное 1, истинному условию и 0 — ложному. Все ненулевые значения компьютер принимает как истинные, а нулевые — как ложные. Но перед OR можно указать цифровые величины, например,

```
80 LET A=B OR C
```

Переменной A присваивается значение, равное 1, если переменная C имеет нулевое значение, или A присваивается значение переменной B, если C имеет нулевое значение.

Помните, что «Эрудит» не выполняет логические действия с двумя цифровыми величинами по типовым таблицам проверки истинности.

ФОРМАТ

условие OR условие
цифр-выр OR цифр-выр

OUT — Наружу
OUT

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT 0

Оператор/команда

OUT посылает байт в порт с указанным адресом.

Как использовать OUT

Ключевое слово может использоваться как оператор в программе или как непосредственная команда. После OUT указываются две цифровые величины, разделенные запятыми, например,

```
80 OUT 254, 6
```

Обе величины округляются до ближайшего целого числа. Первая величина может быть в пределах от 0 до 65535 и указывает адрес порта, вторая — от 0 до 255 и указывает значение байта, посылаемого в порт.

Для порта, адрес которого 254, младшие три бита устанав-

ливают цвет контура: в данном примере устанавливается контур желтого цвета. Четвертый бит этого порта (бит 3) управляет выходом компьютера на магнитофон, пятый бит (бит 4) — громкоговорителем. Порт с адресом 251 управляет печатающим устройством, а порты 239 и 247 и упомянутые биты 254 порта используются совместно с другими периферийными устройствами.

Формат

OUT целочисл-выр, целочисл-выр

OVER На, над
OVER

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT N

Оператор/команда

OVER используется тогда, когда надо индицировать символ, не уничтожая символа, находившегося в этой позиции, а также когда надо индицировать точку или начертить линию цвета бумаги, но не чернил.

Как использовать OVER

Ключевое слово OVER обычно используется как оператор в программе. После него указывается цифровая величина, например,

90 OVER 1

Указанная величина округляется до ближайшего целого числа и может иметь значение, равное 0, или 1. Оператор OVER 0, который действует и по умолчанию, учитывает в индицируемой позиции ранее сындицированный символ. Оператор OVER 1 индицирует символ, не уничтожая ранее сындицированный в этой позиции символ, и таким образом получается комбинация символов.

Оператор OVER может быть размещен в операторе PRINT или INPUT, так же, как и INK. Тогда его влияние носит местный характер, т. е. применяется к индицируемым символам только

этих операторов индикации. Например, представляемый далее оператор отпечатает слово «ЭРУДИТ» и его подчеркнет:

```
90 PRINT AT 10, 12; «ЭРУДИТ»: OVER 1; AT 10, 12: «_____»
```

Но помните, что те пиксели, которые в обоих символах должны быть цвета чернил, в комбинированном символе печатаются цвета бумаги.

OVER в графике высокой разрешающей способности

Оператор OVER также используется с операторами PLOT, DRAW и CIRCLE. Если использовать оператор OVER 1, линии цвета бумаги вычерчиваются в тех местах, где они пересекают линии или символы цвета чернил. Если еще раз оператором OVER 1 начертим линии в том же месте или там же сындицируем точку, то они исчезнут — станут цвета бумаги.

Формат

OVER целочисл-перем

PAPER — Бумага
PAPER

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT C

Оператор/команда

PAPER используется для установления цвета бумаги или фона рабочей части экрана. Можно установить цвет для всей рабочей части экрана или только для той позиции, в которой находятся индицируемый символ, точка или вычерчиваемая линия.

Как использовать PAPER

Ключевое слово PAPER может использоваться как оператор в программе и как непосредственная команда. После него указывается цифровая величина, например,

```
70 PAPER A
```

Указанная величина округляется до ближайшего целого числа и должна находиться в пределах от 0 до 9. Цвета, определяемые этими кодами, являются теми же, как и в случае оператора INK. Определение цвета может быть местного или общего характера. В случае местного характера оператор PAPER так же, как и INK, размещается внутри оператора индикации. Подробнее об этом смотрите в описании оператора INK.

Если символы индицируются после применения оператора PAPER, то независимо от характера определения (общего или местного) выбранный цвет бумаги устанавливается только в тех позициях, в которых печатаются символы. Это верно и для случая индикации точек и вычерчиваний линий, когда определение носит местный характер.

Если хотим окрасить в цвет бумаги всю рабочую часть экрана, то после действия оператора PAPER, носящего общий характер, нужно оператором CLS очистить экран.

Формат

PAPER целочисл-выр(:)

PAUSE Пауза
PAUSE

Набор на клавиатуре

M

Оператор/команда

PAUSE используется тогда, когда надо остановить программу на ограниченное или неограниченное время.

Как используется PAUSE

Ключевое слово PAUSE обычно используется как оператор в программе. После него указывается цифровая величина, например,

90 PAUSE 50

Указанная величина округляется до ближайшего целого числа и может находиться в пределах от 0 до 65535. Она определя-

ет задержку программы: 1 соответствует 1/50 доли секунды. Таким образом, вышеприведенный пример дает задержку, равную одной секунде.

Помните, что пауза может быть прервана нажатием любой клавиши, а оператор PAUSE 0 определяет неограниченную паузу, которая длится до нажатия любой клавиши.

Формат

PAUSE целочисл-выр

PEEK Посмотреть, взглянуть
PEEK

Набор на клавиатуре

EXTEND MODE
0

Функция

PEEK считывает из памяти байт, записанный по указанному адресу.

Как использовать PEEK

После ключевого слова PEEK указывается цифровая величина, например,

90 LET A-PEEK (256 * B)

Выражение должно быть заключено в скобки. Указанная величина округляется до ближайшего целого числа и может находиться в пределах от 0 до 65535 — это адрес ячейки памяти. Результат функции PEEK (который в примере присваивается переменной A) — значение считанного по указанному адресу байта (с допустимыми пределами от 0 до 255).

Пример.

На телевизионном экране за секунду создается 50 кадров. С момента включения «Эрудита» число созданных кадров хранится в ячейках памяти, адреса которых 23672, 23673, 23674. Содержание этих ячеек, считанное функцией PEEK, можно использовать для подсчета реального времени. Представленная далее строка программы индицирует на экране время в секундах с момента включения «Эрудита»:

60 PRINT (PEEK 23672+256×PEEK 23673+65536×PEEK 23674)/50

Формат

PEEK целочисл-выр
PEEK целочисл-перем
PEEK (целочисл-выр)

PI — Число «пи»
PI

Набор на клавиатуре

EXTEND MODE
M

Функция

Результат функции PI — значение математической константы, используемое в расчетах.

Как использовать PI

При подсчете функции PI не указываются никакие величины, например,

80 DPAW 30, 20, PI

Результат функции PI — значение, равное 3,1415927

Формат

PI

PLOT — Обозначить
PLOT

Набор на клавиатуре

Q

Оператор/команда

PLOT используется в графике высокой раз-

решающей способности, когда надо сындицировать в указанном месте экрана пиксель (точку) цвета чернил.

Как использовать PLOT

Ключевое слово PLOT используется как оператор в программе или как непосредственная команда. После него обычно указываются две цифровые величины, разделенные запятой, например,

```
20 PLOT 100, 90
```

Обе указанные величины округляются до ближайших целых чисел. Первая величина может быть в пределах от 0 до 255, и указывает горизонтальную или X-координату пикселя. Вторая величина может быть в пределах от 0 до 175 и указывает вертикальную Y-координату пикселя. Тогда в указанной позиции индицируется пиксель цвета чернил.

Обратите внимание на то, какое влияние на оператора PLOT имеют операторы и команды управления цветом. Если после оператора OVER 1 сындицировать пиксель еще раз в той же позиции, то цвет поменяется с цвета чернил на цвет бумаги. После оператора INVERSE 1 индицируется пиксель цвета бумаги. После оператора BRIGHT 1 или FLASH 1 цвета становятся ярче или начинает мерцать вся позиция символа графики низкой разрешающей способности, в которой индицируется точка.

Эти четыре ключевые слова, а также INK и PAPER могут быть размещены внутри оператора PLOT, как и в случае оператора PRINT, например,

```
100 PLOT INK 4; X, Y
```

В этом случае воздействие имеет местный характер и приемлемо только к индицируемому пикселю данного оператора. Размещенный оператор PAPER меняет цвет всей позиции, в которой находится пиксель, в цвет бумаги. а INK — меняет цвет чернил.

Также помните, что PLOT определяет начальную точку линии, вычерчиваемой следующим за ним оператором DRAW.

Формат

PLOT (операт:) целочисл-выр, целочисл-выр

POINT	Точка POINT

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT 8

Функция

POINT используется, когда надо определить: имеет ли указанный пиксель цвет чернил или он цвета бумаги. Но когда именно цвет, указанный пикселем, POINT не устанавливает.

Как использовать POINT

После POINT указываются две цифровые величины, разделенные запятыми и заключенные в скобки, например,

100 IF POINT (X,Y) = 0 THEN GO TO 1000

Указанные величины округляются до ближайшего целого числа. Первая величина может быть в пределах от 0 до 255 и указывает горизонтальную X-координату пикселя. Вторая величина может быть в пределах от 0 до 175 и указывает вертикальную Y-координату пикселя. Таким образом, результатом POINT является 1, если в указанной позиции пиксель цвета чернил, или — 0, если пиксель цвета бумаги.

Формат

POINT (целочисл-выр, целочисл-выр)

POKE — Втиснуть, втолкнуть

POKE

Набор на клавиатуре

O

Оператор/команда

POKE используется, когда в память надо записать новые байты по указанному адресу. Они обычно вписываются тогда, когда надо выполнить действия, которые не могут выполнить ключевые слова Бейсика.

Как использовать POKE

Ключевое слово POKE используется для оператора в программе или как непосредственная команда. После него указываются две цифровые величины, разделенные запятой, например,

POKE 23609, 255

Указанные величины округляются до ближайшего целого числа. Первая величина может быть в пределах от 16384 до 65535 — это адрес оперативной памяти. Вторая величина может быть в пределах от 0 до 255 — это значение байта, которое надо записать по указанному адресу.

В данном примере в ячейку оперативной памяти с адресом 23609 записывается байт, имеющий значение 255. Записанный в эту ячейку байт управляет звуковым сигналом, генерируемым в момент нажатия клавиши. Значение 255 устанавливает длительное жужжание вместо короткого пиканья. Другие значения байта устанавливают более короткие звуковые сигналы.

Формат

POKE целочисл-выр, целочисл-выр

PRINT — Печатать
PRINT

Набор на клавиатуре

P

Оператор/команда

PRINT индицирует данные на экране телевизора. Данные — это символ или последовательность символов. В операторе PRINT также могут быть размещены другие ключевые слова, определяющие место на экране и цвет индицируемых данных.

Как использовать PRINT

Ключевое слово PRINT может использоваться самостоятельно без всяких параметров или с указанными после него индицируемыми данными: любыми числовыми или символьными выражениями или их комбинациями.

Когда оператор PRINT используется с данными, то отдельные элементы данных должны быть разделены точками с запятой, запятыми или апострофами.

Некоторые ключевые слова (CHR$, TAB, AT, INK, PAPER, FLASH, BRIGHT, INVERSE и OVER) могут быть размещены между ключевыми словами PRINT и данными. После ключевых слов обязательно должна следовать точка с запятой.

PRINT и строки

Оператор PRINT без всяких данных или с указанной пустой строкой («») индицирует на экране пустую строку и передвигает курсор в начало следующей строки.

PRINT с указанной символикой константой (любые символы в кавычках) индицируют символы так, как они указаны в кавычках. Например,

команда
PRINT «01.04.1990»
сындицирует на экране 01.04.1990

PRINT с указанной символьной переменной или символьным выражением индицирует принадлежащие им строки.

PRINT и числа

Оператор PRINT с указанным цифровым выражением индицирует числовое значение этого выражения. Числа индицируются в десятичном формате, указывая до восьми значащих цифр. Нули в конце числа после десятичной точки не индицируются.

Очень большие и очень малые числа индицируются в другом формате с плавающей запятой. В этом формате индицируемое число складывается из двух частей, разделенных буквой «E». Первая его часть — мантисса, которую надо умножить на 10, Возведенное в степень, указанную второй частью числа — порядком. Например,

9.8765432E-8 соответствует $9.8765432 * 10^{-8}$

Определение формата печати с разделительными знаками

Если в операторе PRINT элементы данных разделены точками с запятой, то они индицируют на экране один за другим без промежутков между ними. Например,

команда
PRINT 1; 2; 3

сындицирует на экране
123

Если элементы данных разделены запятыми, то они начинают индицироваться или с начала строки или с ее середины в зависимости от того, в какой позиции индицируется первый элемент данных. Например,

команда
PRINT 1, 2, 3
сындицирует на экране
1 2
3

Если элементы данных разделены апострофами, то элемент данных, находящихся после апострофа, печатается с начала следующей строки. Например,

команда
PRINT 1 2 3
сындицирует на экране
1
2
3

Если оператор или команда PRINT заканчивается точкой с запятой, запятой или апострофом, то они свое воздействие сохраняют и на элементы данных, которые печатаются со следующим оператором PRINT.

PRINT и другие ключевые слова

После PRINT может быть указано ключевое слово TAB и цифровая величина, далее точка с запятой и элемент данных, например,

90 PRINT TAB S; A

Указанная после TAB величина округляется до ближайшего целого числа, затем делится на 32, а остаток, значение которого может быть в пределах от 0 до 31, — это позиции этого или следующего столбца строки, с которого печатаются дальше указанные данные.

После PRINT может быть указано ключевое слово AT с двумя цифровыми величинами, разделенными запятыми, далее точка с запятой и элемент данных, например,

80 PRINT AT E, S; «Эрудит»

Первая величина (в примере E) может быть в пределах от 0 до 21 и указывает номер строки, в которой будут индицироваться данные. Вторая величина (S) может быть в пределах от 0 до 31 и указывает номер позиции столбца, с которого будут индицироваться данные. Обе величины округляются до ближайших целых чисел.

Команда PRINT AT 11, 16; «*» выдает в середине экрана индикацию звездочки.

После PRINT также может быть указано одно или несколько ключевых слов CHR$. Подробнее об этом смотрите в описании CHR$.

PRINT и ключевые слова управления цветом

Цвета данных, индицируемых вместе с оператором PRINT, определяются командами или операторами управления цветом: INK, PAPER, FLASH, BRIGHT, INVERSE и OVER, которые могут быть размещены внутри оператора PRINT. Они указываются перед элементом данных, и после каждого из них должна следовать точка с запятой, например,

```
70 PRINT AT 9, 12; INK 4; FLASH 1; «Эрудит»
```

В этом случае атрибуты, указанные перед элементом данных, оказываются местным воздействием и используются для индикации только этого элемента данных.

В операторе PRINT также можно использовать местные коды управления цветом, вводимые вместе с данными (см. рис. 2.22).

Формат

PRINT (TAB целочисл-выр;) (AT целочисл-выр, целочисл-выр;) (CHR$ (целочисл-выр);) (оператор:) (цифр-выр) (сим-выр) (:) (,) (')

RANDOMIZE Рандомизировать

RANDOMIZE

Набор на клавиатуре

T

Оператор/команда

RANDOMIZE, который на клавиатуре сок-

ращенно записан как RAND, используется вместе с RND, когда надо прогенерировать случайную или псевдослучайную последовательность числа.

Как использовать RANDOMIZE

Ключевое слово RANDOMIZE используется в программе как оператор или как непосредственная команда. После него может быть (но не обязательно) указана цифровая величина, например,

```
RANDOMIZE
90 RANDOMIZE 1
```

Указанная величина округляется до ближайшего целого числа и может находиться в пределах от 0 до 65535. Значение, большее, чем 0, присваивается системой переменной SEED и в зависимости от нее функция RND всегда генерирует ту же самую последовательность числа.

Если после RANDOMIZE не указана никакая величина или указан 0, то переменной SEED присваивается значение другой системной переменной FRAMES. Эта переменная подсчитывает число развернутых на телевизионном экране кадров с момента включения компьютера. В этом случае последовательность числа, генерируемая функцией RND, является чисто случайной, т. к. значение переменной SEED меняется 50 раз в секунду.

Если RANDOMIZE не используется, то функция RND начинает генерировать ту же самую последовательность числа с момента включения компьютера или с момента нажатия кнопки сброса, или с момента выполнения команды NEW.

Формат
RANDOMIZE (цифр-выр)

READ — Читать

READ

Набор на клавиатуре

EXTEND MODE
A

Оператор/команда

READ используется вместе с DATA, когда

надо присвоить значения переменным, записанным в операторе DATA.

Как использовать READ

Ключевое слово READ обычно используется в программе как оператор. После него указывается одна или несколько цифровых или символичных переменных, разделенных запятыми, например,

60 READ A, B$

Когда оператор READ выполняется первый раз, из начала списка элементов DATA берется столько значений, сколько переменных в операторе READ, и эти значения присваиваются переменным. В следующий раз берутся следующие переменные из списка DATA и т. д. Подробнее об этом смотрите в описании DATA.

Формат

READ цифр-перем (,цифр-перем) (,симв-перем)
READ симв-перем (,цифр-перем) (,симв-перем)

REM Замечание
REMark

Набор на клавиатуре

E

———

Оператор

REM используется, когда в программу надо вставить комментарии, которые поясняют действие программы, указывают названия подпрограмм и т. п. На выполнение программы комментарии влияния не оказывают, они видны только в листинге.

Как использовать REM

Оператор REM создает в программе или отдельную строку, или является последним оператором в строке. После ключевого слова REM может следовать комментарий, состоящий из любых символов, например,

90 LET M=M+1: REM M — это минуты

Когда компьютер встречает оператора REM, он игнорирует все, что находится в этой строке после REM.

Формат

REM любые символы

RESTORE

Восстановить
RESTORE

Набор на клавиатуре

EXTEND MODE
S

Оператор/команда

RESTORE используется совместно с операторами READ и DATA, когда надо считать с помощью READ из списка DATA данные не по порядку, а начиная с определенного места.

Как использовать RESTORE

Ключевое слово RESTORE обычно используется в программе как оператор. После него может быть (но не обязательно) указана цифровая величина, например,

100 RESTORE 900

Указанная величина округляется до ближайшего целого числа. Это номер строки программы, в которой есть оператор DATA. Идущий после RESTORE оператор READ будет брать данные для присваивания, начиная именно с этого места. Если строки с указанным номером нет вообще или в указанной строке нет оператора DATA, то READ будет брать данные из первого оператора DATA, который находится в строке с большим номером.

Если после RESTORE указан 0 или вообще не указана никакая величина, то следующий оператор READ будет брать данные из самого первого оператора DATA в программе.

Формат

RESTORE (целочисл-выр)

RETURN Возвратиться
RETURN

Набор на клавиатуре

Y

Оператор/команда

RETURN обычно используется как оператор в программе.

Оно указывается в конце подпрограмм без всяких величин, например,

2000 RETURN

Во время исполнения программа переходит к оператору, следующему за последним исполненным оператором GOSUB.

Формат

RETURN

RND Случайное число
RaNDom number

Набор на клавиатуре

EXTEND MODE
T

Функция

RND применяется, когда требуется генерировать случайное число.

Как использовать RND

После ключевого слова RND никакая величина не указывается, например,

90 LET A=RND

Результат функции RND — случайное число, больше 0, но меньше 1.

Функция RND генерирует ту же последовательность с момента включения «Эрудита» или после нажатия кнопки сброса, или после выполнения команды NEW. Последовательность генерируется путем деления числа 75, возведенного в различные степени (в первую, вторую, третью и т. д.), на 65537, затем отнимается от остатка 1, и этот результат делится на 65536.

Если требуется другая последовательность случайных или псевдослучайных чисел, то перед RND напишите оператор RANDOMIZE.

Случайные целые числа

Большинство операторов и функций «Эрудита»; к примеру INK и CHR$, округляют числа до ближайших целых чисел, поэтому функция RND может использоваться с ними непосредственно. Например, PAPER RND*7 генерирует случайный цвет бумаги. Если нужно заранее подготовить случайное число, то можно использовать следующий метод; случайное целое число от 1 до N дает выражение INT(RND*N)+1, а случайное число от 1 до N — INT(RND*N+0.5).

Формат

RND

RUN	Выполнять RUN

Набор на клавиатуре

R

Команда/оператор

RUN обычно начинает выполнение программы с первой строчки.

Как использовать RUN

Ключевое слово RUN может применяться как непосредственная команда или как оператор в программе. После него может быть (но не обязательно) указана цифровая величина, например,

RUN 40

Если никакая величина не указана, то программа выполняется с первой строки. Если величина указана, то она округляется до ближайшего целого числа, и программа выполняется со строки с этим номером. Если такой строки нет, то программа начинается с ближайшей имеющейся строки.

Запомните, что прежде чем запустить программу, команда RUN выполняет действие команды CLEAR, очищая память и уничтожая значения переменных. Если хотите этого избежать, то вместо RUN примените G0 T0 с указанием номера строки.

Если программа записана на ленту при помощи ключевого слова LINE, то при вводе ее в компьютер с ленты, она выполняется автоматически, и RUN не нужен.

Формат

RUN (целочисл-выр)

SAVE — Сохранить
SAVE

Набор на клавиатуре

S

Команда/оператор

SAVE используется, когда надо послать программу на магнитофон для записи на магнитную ленту.

Как использовать SAVE

Ключевое слово SAVE обычно используется как непосредственная команда, но может быть использовано и как оператор в программе. После него указывается имя файла, которое является символьной величиной (до десяти символов), например,

SAVE «ИМЯ»

Во время выполнения индицируется сообщение:

«ВКЛЮЧИТЬ МАГ., НАЖАТЬ КЛАВИШУ».

После нажатия какой-нибудь клавиши программа направля-

ется на кассетный магнитофон, а по окончании выдается сообщение

0 ВЫПОЛНЕНО, 0:1

Автоматический запуск

Чтобы записанная с ленты программа начала выполняться автоматически, записывать ее надо, используя сочетание ключевых слов SAVE и LINE. После имени программы надо указать ключевое слово LINE и цифровую величину, например,

SAVE «ИМЯ» LINE 1

Указанная после LINE цифровая величина округляется до ближайшего целого числа — это номер строки программы. После этого программа записывается на магнитную ленту аналогично случаю SAVE. Во время считывания с ленты, программа автоматически запускается с указанной строки, а если такая отсутствует — с ближайшей имеющейся.

Практически, если надо запустить программу с начала, используется сочетание LINE 1.

Формат

SAVE симв-выр (LINE целочисл-выр)

SAVE CODE Сохранить код

SAVE CODE

Набор на клавиатуре

S
EXTEND MODE
1

Команда/оператор

SAVE CODE, когда надо записать на ленту часть информации, посылает ее из памяти в кассетный магнитофон. Записанная информация может быть считана с ленты, если применить команду LOAD CODE.

Как использовать SAVE CODE

Ключевые слова SAVE CODE могут использоваться как непосредственная команда или как оператор в программе. После SAVE указывается имя файла, а за ним — ключевое слово CODE и две цифровые величины, разделенные запятой, например,

SAVE «КАРТИНА» CODE 16384, 6912

Имя файла может содержать до десяти символов. Обе величины, указанные после CODE, округляются до ближайшего целого числа. Первое число указывает начальный адрес в информационной памяти (в примере — 16384), второе число количество байтов, которые надо записать на ленту (6912). Во время выполнения информация посылает на кассетный магнитофон тем же способом, что и в случае с SAVE.

В данном примере информация записывается на ленту с экранной зоны памяти, т. е. записывается изображение, имеющееся на экране.

Формат

SAVE симв-выр CODE целочисл-выр, целочисл-выр

SAVE DATA Сохранить данные
SAVE DATA

Набор на клавиатуре

S
EXTEND MODE
D

Оператор/команда

_____ SAVE DATA записывает на ленту массив. Записанный массив может быть считан при помощи оператора LOAD DATA.

Как использовать SAVE DATA

Ключевые слова SAVE DATA используются как непосредственная команда или как оператор в программе. После SAVE

указываются имя файла, затем ключевое слово DATA и буква или буква с символом «⋇» и, наконец, пустые скобки, например,

```
300 SAVE «ЧИСЛА» DATA S ( )
350 SAVE «ИМЕНА» DATA V⋇ ( )
```

Имя файла массива может содержать до десяти символов. Буква или буква с символом «⋇», следующая за DATA, — это имя массива, который надо записать на ленту. Во время выполнения массив записывается на ленту тем же способом, что и в случае с SAVE.

Формат

SAVE симв-выр DATA буква (⋇) ( )

SAVE SCREEN ⋇ Сохранить экран
SAVE SCREEN

Набор на клавиатуре

S
EXTEND MODE
SYMBOL SHIFT K

Оператор/команда

SAVE SCREEN ⋇ записывает на ленту информацию, индицируемую на экране. Записанная информация может быть считана при помощи команды LOAD SCREEN ⋇

Как использовать SAVE SCREEN ⋇

Ключевое слово SAVE SCREEN ⋇ используется в программе как оператор или как непосредственная команда. После SAVE указываются имя файла, а затем ключевое слово SCREEN ⋇, например,

SAVE «ИМЯ» SCREEN ⋇

Имя файла может быть длиной до десяти символов. Во время выполнения на экране индицируется информация, посылаемая на магнитофон тем же способом, как и в случае с SAVE.

Формат

SAVE симв-выр SCREEN ¤

SCREEN ¤ Экран
SCREEN

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT K

Функция

SCREEN ¤ устанавливает, какой символ индицируется в указанной на экране позиции.

Как использовать SCREEN ¤

После ключевого слова SCREEN ¤ указываются две цифровые величины, разделенные запятой и заключенные в скобки, например,

100 IF SCREEN ¤ (E, S) = «0» THEN LET R = R + 1

Указанные величины округляются до ближайших целых чисел. Первая величина (в примере — E) может быть в пределах от 0 до 21 и указывает номер строки, в которой находится нужная позиция. Вторая величина (S) может быть в пределах от 0 до 32 и указывает номер столбца позиции. Результат функции SCREEN ¤ символьная константа, состоящая из символа, который индицируется в данной позиции. В данном примере переменная R будет увеличиваться на единицу только в том случае, если в позиции с координатами (E, S) будет индицироваться буква «О». Если в указанной позиции вообще не индицируется никакой символ, то SCREEN ¤ возвращает пустую строку (« »).

Запомните, что ключевое слово SCREEN ¤ может также использоваться вместе с SAVE и LOAD, когда надо запомнить на ленте или считать с нее информацию, индицируемую на экране. Подробнее об этом смотрите в описаниях SAVE SCREEN ¤ и LOAD SCREEN ¤

Формат

SCREEN ¤ (целочисл-выр, целочисл-выр)

SGN Знак
SiGN

Набор на клавиатуре

EXTEND MODE
F

Функция

SGN устанавливает, является ли число положительным, отрицательным или равным нулю.

Как использовать SGN

После ключевого слова SGN указывается цифровая величина, например,

90 LET A=SGN B

Выражение должно быть заключено в скобки. Результат функции SGN равен 1, если значение аргумента положительное; минус 1, если значение аргумента отрицательное и равно 0, если значение аргумента равно 0.

Формат

SGN цифр-конст
SGN цифр-перем
SGN (цифр-выр)

SIN Синус
SINe

Набор на клавиатуре

EXTEND MODE
Q

Функция

SIN рассчитывает синус угла.

Как использовать SIN

После ключевого слова SIN указывается цифровая величина, например,

```
90 LET A=SIN B
```

Выражение должно быть заключено в скобки. Величина, указанная после SIN, — угол, выраженный в радианах. Результат функции — синус указанного угла. Градусы могут быть переведены в радианы путем умножения их на PI/180.

Запомните: результат функции является положительным для углов от 0 до 180 градусов и отрицательным для углов от 180 до 360 градусов.

Пример
Команда
PRINT SIN (30 *PI/180)
сындицирует на экране синус угла 30 градусов — 0,5

Формат

SIN цифр-выр
SIN цифр-перем
SIN (цифр-выр)

SQR — Квадратный корень
SQare Root

Набор на клавиатуре

EXTEND MODE
H

Функция

SQR вычисляет квадратный корень.

Как использовать SQR

После ключевого слова SQR указывается цифровая величина, например,

```
90 LET A=SQR B
```

Выражение должно быть заключено в скобки. Указанная ве-

личина должна быть положительной. В таком случае результат функции SQR — квадратный корень, извлеченный из аргумента.

Формат

SQR цифр-выр
SQR цифр-перем
SQR (цифр-выр)

STEP — Шаг

STEP

Набор на клавиатуре

SYMBOL SHIFT D

Смотрите описание FOR

STOP — Остановить

STOP

Набор на клавиатуре

SYMBOL SHIFT A

Оператор/команда

STOP останавливает программу в указанной точке. Оператор STOP часто принимается в том случае, когда необходимо отделить основную программу от подпрограммы. Он употребляется также и при отладке программы.

Как используется STOP

Ключевое слово STOP обычно используется как оператор в программе. Нет необходимости указывать какие-либо величины, например,

```
200 STOP
```

Во время исполнения программа остановится, и будет выдано сообщение

```
9 ОПЕРАТОР STOP
```

и сообщение о номерах строки программы и оператора.

При отладке программы, после остановки можно индицировать или заменить значение переменных. После команды CONTINUE программа выполняется со следующим оператором.

Формат

STOP

STR ✱ Строка
STRing

Набор на клавиатуре

EXTEND MODE
Y

Функция

STR ✱ число заменяет строкой.

Как использовать STR✱

После ключевого слова STR ✱ указывается цифровая величина, например,

90 LET A✱ =STR ✱ B

Выражение должно быть заключено в скобки. Результат функции STR ✱ — символьная константа, соответствующая значению аргумента. Если в примере переменная B будет иметь значение 87, то переменной A ✱ будет присвоена строка «87».

Формат

STR ✱ цифр-конст
STR ✱ цифр-перем
STR ✱ (цифр-выр)

TAB Фиксировать
TAB

Набор на клавиатуре

EXTEND MODE
P

Смотрите описание LPRINT и PRINT

TAN — Тангенс
TANgent

Набор на клавиатуре

EXTEND MODE
E

Функция

TAN вычисляет тангенс угла.

Как использовать TAN

После ключевого слова TAN указывается цифровая величина, например,

90 LET A=TAN B

Выражение должно быть заключено в скобки. Указанная после TAN величина является углом, выраженным в радианах. Результат функции TAN — тангенс указанного угла. Градусы могут быть переведены в радианы при помощи умножения на PI/180.

Запомните, что результат функции положительный для угла от 0 до 90 и от 180 до 270 градусов и отрицательный для углов от 90 до 180 и от 270 до 360 градусов.

Формат

TAN цифр-конст
TAN цифр-перем
TAN (цифр-выр)

THEN — Тогда
THEN

Набор на клавиатуре

SYMBOL SHIFT G

Смотрите описание IF

TO — До TO

Набор на клавиатуре

SYMBOL SHIFT F

Функция

В Бейсике «Эрудит» ключевое слово TO имеет два различных значения. Оно используется вместе с FOR, когда надо составить цикл FOR NEXT (подробнее смотрите в описании FOR), а также для дробления строк, когда из строки надо сделать строки покороче.

Как использовать TO для дробления строк

Функция TO применяется для указания первого и последнего символа основной части строки, из которой создается новая строка. После какой-нибудь символьной величины открывается скобка, в которой указывается цифровая величина, которую можно и опустить, затем ключевое слово TO, еще одна цифровая величина, которую тоже можно опустить, затем скобка закрывается, например,

```
90 PRINT A$ (3 TO 8)
```

Символьное выражение тоже должно быть заключено в скобки. Указанная символьная величина (в примере — A$) — это основная строка, из которой взята часть для создания новой строки. Две цифровые величины указывают первый и последний символы отделяемой части строки. Во время выполнения программы функции TO создает новую строку (в нашем примере — начиная с третьего и кончая восьмым символом A$ строки).

Если не указана первая цифровая величина, то она автоматически становится равной 1. Если не указана вторая цифровая величина, то ей автоматически придается значение, равное числу символов в основной строке. Таким образом, первая цифровая

величина может быть опущена, если новая строка должна начинаться с первого символа основной строки. Вторая цифровая величина может быть опущена, если новая строка должна продолжаться до конца основной строки.

Формат

симв-конст ([цифр-выр] TO [цифр-выр])
симв-перем ([цифр-выр] TO [цифр-выр])
(симв-выр) ([цифр-выр] TO [цифр-выр])

USR — Подпрограмма потребителя
User SubRoutine

Набор на клавиатуре

EXTEND MODE
L

Функция

USR используется в случаях, когда надо вызвать подпрограмму, записанную машинными кодами и занесенную в память по указанному адресу. Она также может применяться для записи данных графики, устанавливаемой потребителем, в зарезервированное место в конце памяти.

USR и машинные коды

При применении подпрограмм, написанных машинными кодами, после ключевого слова USR указывается цифровая величина, например,

```
90 PRINT USR 60000
110 RANDOMIZE USR 60000
```

Выражение должно быть заключено в скобки. Указанная величина округляется до ближайшего целого числа — это начальный адрес памяти, с которого и была записана машинными кодами подпрограмма, находящаяся по этому адресу. Результат функции USR — значения, находящиеся в паре микропроцессорных регистров BC. Комбинация ключевых слов . . . . . содержание регистров BC.

USR и графика, устанавливаемая потребителем

При создании символов графики, устанавливаемой потребителем, ключевое слово USR применяется с POKE. В этом случае функция USR применяется для установления адреса, необходимого оператору POKE. После USR устанавливается символьная переменная или символьная константа, например,

```
90 POKE USR «B», 129
```

Указанный символ может быть любой буквой от «A» до «U». В этом случае функция USR устанавливает адрес одной из 21 секции памяти, зарезервированной для графики, устанавливаемой потребителем. Каждая секция имеет восемь адресов, по которым записываются восемь байтов, создающих один графический символ. Эти байты могут быть указаны в десятичном или в двоичном (смотрите BIN) формате.

Формат

```
USR целочисл-конст
USR целочисл-перем
USR (целочисл-выр)
USR симв-конст
USR симв-перем
```

VAL — Значение VALue

Набор на клавиатуре

```
EXTEND MODE
J
```

Функция

VAL возвращает числовое значение строки, если символы строки — цифры.

Как использовать VAL

После ключевого слова VAL указывается символьная константа или символьная переменная, например,

```
90 LET A=VAL B
```

С символьной переменной снимаются кавычки, и получается цифровое значение. Тогда результат функции VAL — цифровая константа.

Примеры

Если в приведенном выше примере переменная B ¤ будет иметь значение «1998», то переменной A будет присвоено значение 1998.

Функция VAL также может рассчитать значение выражения. Например, пусть имеет программу

```
10 INPUT X
20 LET A ¤ =«X/4»
30 PRINT VAL A¤
```

Введем значение переменной X, равное 60. Тогда функция VAL, содержащаяся в операторе PRINT, сняв кавычки со значения переменной A ¤ , дает выражение X/4 и, вставив числовое значение X, рассчитывает ее. Оператор PRINT индицирует на экране 15.

Формат

VAL симв-конст
VAL симв-перем

VAL ¤ — Значение (строки)
VALue (string)

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT J

Функция

VAL ¤ преобразует строку в символьное выражение.

Как использовать VAL ¤

После ключевого слова VAL ¤ указывается символьная переменная, например,

```
100 PRINT VAL ¤ A ¤
```

Со значения символьной переменной снимаются кавычки, и тогда получается символьное выражение. В этом случае результат функции VAL ¤ — символьная константа, рассчитанная из полученного символьного выражения.

Пример

Опробуйте программу
10 INPUT X ¤
20 LET A ¤ =«««Супер»»+X ¤ »
30 PRINT VAL ¤ A ¤

Введите значение переменной X ¤, например, «ФИНАЛ». Тогда функция VAL ¤, содержащаяся в операторе PRINT, сняв кавычки со значения переменной A ¤, даст выражение «СУПЕР»+X ¤ и, вставив значение X ¤, образует строку «СУПЕРФИНАЛ», которую и индицирует на экране оператор PRINT.

Формат

VAL ¤ симв-перем

VERIFY Проверить
VERIFY

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT R

Команда/оператор

VERIFY проверяет правильность записи программы на ленту во время действия команды SAVE.

Как использовать VERIFY

Ключевое слово VERIFY обычно применяется как непосредственная команда точно так же, как и LOAD. После него указывается имя программы, например,

VERIFY «ИМЯ»

Когда запускается магнитная лента, имена всех обнаруженных программ индицируются на экране, а любая записанная на

ленту программа, имеющая указанное имя, сравнивается с программой, содержащейся в памяти компьютера. Если обе сравниваемые программы идентичны, то выдается сообщение:

```
0 ВЫПОЛНЕНО, 0:1
```

VERIFY CODE и VERIFY DATA

Сочетание ключевых слов VERIFY CODE может применяться точно так же, как и LOAD CODE, в случае, если надо проверить правильность записанной на ленту части информации памяти. Ключевые слова VERIFY DATA применяются аналогично LOAD DATA в случае проверки правильности записи на ленту массива. Подробнее смотрите описание LOAD CODE и LOAD DATA.

Формат

```
VERIFY симв-выр
VERIFY симв-выр CODE (целочисл-выр) (,целочисл-выр)
VERIFY симв-выр DATA буква ( ¤ ) ( )
```

VERIFY CODE Проверить коды
VERIFY CODE

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT R
EXTEND MODE
I

См. описание VERIFY

VERIFY DATA Проверить данные
VERIFY DATA

Набор на клавиатуре

EXTEND MODE
SYMBOL SHIFT R
EXTEND MODE
D

См. описание VERIFY

3.3. СООБЩЕНИЯ, ВЫДАВАЕМЫЕ «ЭРУДИТОМ»

Некоторые синтаксические ошибки, допущенные в программе, «Эрудит» выявляет уже во время ввода программы. При попытке ввести строку с явной ошибкой, компьютер индицирует рядом с ошибкой мигающий знак вопроса. Однако иногда ошибку можно выявить только во время выполнения программы. Это случается, например, когда переполняется память компьютера, когда в программе встречается переменная, которой не присвоено еще никакого значения, или когда в процессе расчетов выражения получаются чрезмерно большие числа и т. п. В таких случаях «Эрудит» прерывает выполнение программы.

Прервав выполнение программы, компьютер в нижней части экрана индицирует сообщение, что происходит не только в случае обнаружения ошибки. Сообщение выдается и в случае успешного выполнения программы или команды.

В начале каждого сообщения выдается его кодовый номер (цифра или буква), затем краткое описание причины остановки, а затем номер строки программы и порядковый номер оператора в этой строке, при выполнении которого произошла остановка. Если остановка произошла во время выполнения программы, в сообщении указывается строка с номером 0. Оператор, находящийся в начале строки, имеет номер 1, следующий оператор, указанный после двоеточия или после ключевого слова THEN, имеет номер 2 и т. д. Команда CONTINUE обычно продолжает выполнение программы с оператора, указанного в сообщении.

Ниже приводится перечень сообщений с краткими пояснениями:

0 ВЫПОЛНЕНО

Успешное выполнение или попытка перехода на строку, номер которой больше любой из существующих. Команда CONTINUE игнорирует это сообщение и продолжает выполнять программу с оператора, указанного в предыдущем сообщении.

1 NEXT БЕЗ FOR

В программе встречен оператор (или команда) NEXT, перед которой отсутствует соответствующий оператор FOR, а в программе существует переменная, имеющая то же имя, что и переменная цикла.

2 ПЕРЕМЕННОЙ НЕТ

Использована простая переменная, которой предварительно не

присвоено никакого значения (с помощью оператора LET и READ или INPUT или при считывании с ленты) или в NEXT операторе (команде) использована переменная цикла, не определенная предварительно оператором FOR, или индексирована переменная (элемент массива), использованная перед описанием массива с помощью оператора DIM или перед считыванием массива с ленты.

3 НЕПРАВИЛЬНЫЙ ИНДЕКС

Индекс вышел за описанные границы массива.

4 НЕ ХВАТАЕТ ПАМЯТИ

В памяти имеется недостаток места для полного выполнения оператора или команды.

5 ЗА ПРЕДЕЛАМИ ЭКРАНА

Оператор (или команда) INPUT создал больше 23 строк в нижней части экрана или в комбинации ключевых слов PRINT AT был указан номер строки, равный или больше 22.

6 ОЧЕНЬ БОЛЬШОЕ ЧИСЛО

Компьютер пытается получить число больше 10^{38}.

7 RETURN БЕЗ GOSUB

Количество операторов RETURN по крайней мере на единицу больше, чем количество выполненных операторов GOSUB.

9 ОПЕРАТОР STOP

Программа остановлена оператором STOP. Команда CONTINUE продолжит выполнение программы со следующего оператора.

A ПЛОХОЙ АРГУМЕНТ

Функции был указан неверный аргумент или его значение.

B БОЛЬШОЕ ЦЕЛОЕ ЧИСЛО

Цифровое значение округлено до ближайшего числа, которое получилось больше допустимого.

C ОШИБКА В БЕЙСИКЕ

Текст символьного аргумента не является правильным выражением.

D BREAK: CONT ПОВТОРИТ

Была нажата клавиша BREAK. Команда CONTINUE повторяет оператор, при выполнении которого компьютер остановится.

E ЗА ПРЕДЕЛАМИ DATA

Оператор (или команда) READ пытается считывать данные, в то время, как список данных DATA уже закончился.

F ПЛОХОЕ ИМЯ ФАЙЛА

В команде SAVE указано имя, в котором более десяти символов.

G НЕТ МЕСТА ДЛЯ СТРОКИ

В памяти не осталось достаточно места для размещения новой строки программы.

H STOP В INPUT'Е

Во время действия оператора INPUT вводимые данные начались с ключевого слова STOP или во время действия оператора INPUT LINE была нажата клавиша STOP. Команда CONTINUE повторяет операторы INPUT.

I FOR БЕЗ NEXT

Цикл FOR NEXT не может выполняться, т. к. значение предела или шага указано неверно (например, FOR 1=7 TO 1, не применяя STEP), а соответствующий оператор NEXT не найден.

K НЕПРАВИЛЬНЫЙ ЦВЕТ

Недопустимые значения в операторах (или командах) INK, PAPER, FLASH, BRIGHT, INVERSE или OVER.

L BREAK В ПРОГРАММЕ

Была нажата клавиша BREAK. В сообщении указан последний выполненный оператор. Команда CONTINUE продолжает выполнение программы со следующего оператора.

M ПЛОХОЙ RAMTOP

Значение, которое хотим присвоить системной переменной RAMTOP, слишком велико или слишком мало.

N НЕТ ОПЕРАТОРА

Попытка перехода к несуществующему оператору.

P FN БЕЗ DEF

Применен оператор FN без соответствующего DEF FN оператора.

Q ОШИБКА ПАРАМЕТРА

В операторе FN неверно указано количество параметров, которые надо передать функции, или один из указанных параметров не того типа (строка вместо числа или наоборот).

R ОШИБКА ЗАГРУЗКИ

Не состоявшееся по какой-либо причине считывание с ленты, объединение двух программ или верификация.

3.4. СЕТКИ ГРАФИКИ ВЫСОКОЙ И НИЗКОЙ РАЗРЕШАЮЩЕЙ СПОСОБНОСТИ

Нижеприведенная сетка указывает координаты позиции графики низкой разрешающей способности и пикселей графики высокой разрешающей способности. Сетка графики низкой разрешающей способности состоит из двух частей: основной части экрана (строки от 0 до 21) и двух нижних строк. В основной части экрана координаты символа можно установить при помощи ключевых слов PRINT AT, а в нижней части — с помощью INPUT AT. Координаты этой сетки указаны по верхнему и левому краям сетки.

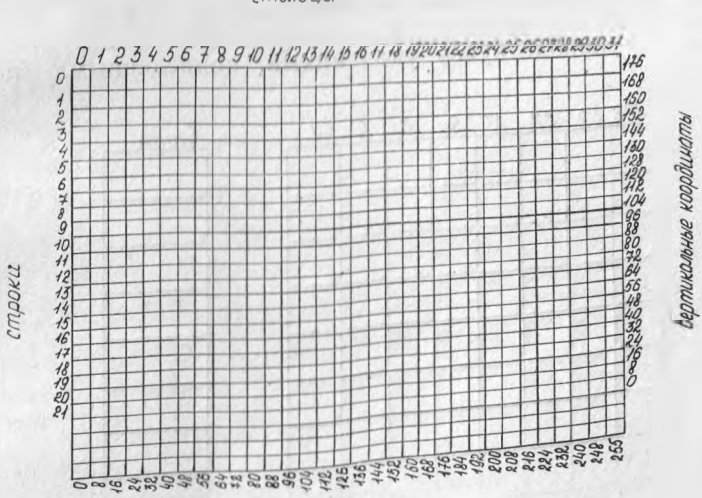


Рис. 26

Сетка графики высокой разрешающей способности занимает только основную часть экрана. Эта сетка применяется для графики, которую создают операторы (или команда) PLOT, DRAW и CIRCLE. Координаты сетки указаны по нижнему и правому краям сетки.

3.5. СЕТКА ГРАФИКИ, УСТАНАВЛИВАЕМОЙ ПОТРЕБИТЕЛЕМ

Символ графики, устанавливаемый потребителем, складывается из 64 пикселей — 8 строк по 8 пикселей. Информация о символе может быть записана в компьютерную память указанием восьми двоичных или восьми десятичных чисел. Первый способ проще: двоичное число, указанное после ключевого слова BIN, точно отражает 8 пикселей символа, находящихся в одной строке — 1 соответствует пикселю цвета чернил, 0 — бумаги. Однако запись в десятичных числах в несколько раз короче, чем двоичных. Во время ввода программы экономится время, но возникает проблема, как превратить двоичные числа в десятичные.

Выполнить это вам поможет предлагаемая ниже сетка (см. рис 3.2). Карандашом заштрихуйте те квадратики, которые в данном символе должны иметь цвет чернил. Затем просуммируйте значения затушеванных квадратиков каждой строки. Получите восемь десятичных чисел. Далее полученные числа введите в компьютер, руководствуясь указаниями, приведенными на стр. 68—72.

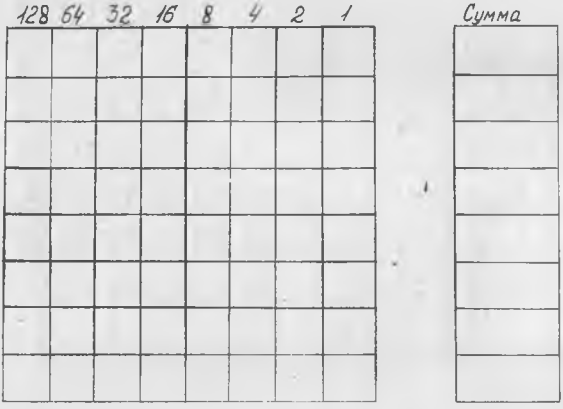


Рис. 27

3.6. КРАТКОЕ ОПИСАНИЕ БЕЙСИКА «ЭРУДИТА»

Ниже приводится описание команд, операторов и функций Бейсика «Эрудит». Деление на команды и операторы условно в зависимости от того, где указанные ключевые слова чаще употребляются. Большинство команд могут использоваться как операторы и наоборот, т. е. большинство операторов применяются как команды, а все три логические операции могут применяться как функции. Существуют еще шесть команд управления файлами микродрайвов: CAT, CLOSE, ERASE, FORMAT, MOVE и OPEN#, но они в этой книге не описаны.

Все команды, операторы и функции расположены в алфавитном порядке. Ключевые слова, вводимые одним нажатием клавиши (предварительно установив нижний регистр), отпечатаны заглавными буквами. Команды, атрибуты конструкции оператора или функции, заключенные в квадратные скобки, могут быть опущены или могут повторяться. Более подробное описание ключевых слов Бейсика приведено в третьей главе книги.

3.6.1. Команда БЕЙСИКА

CONTINUE

Запустить программу с места остановки для дальнейшего выполнения.

COPY

Отпечатать на печатающем устройстве копию информации, индицируемой в данный момент на экране телевизора.

LIST (номер строки)

Создать на экране телевизора листинг программы из памяти компьютера. Если указан номер строки, листинг начать с нее.

LLIST (номер строки)

Отпечатать на печатающем устройстве листинг программы, содержащейся в памяти компьютера. Если указан номер строки, листинг начать с нее.

LOAD имя файла

Считать программу с указанным именем с магнитной ленты и записать в память компьютера, а в некоторых случаях еще и запустить программу.

LOAD имя файла CODE (начальный адрес) (,число байтов)

Считать последовательность байтов с указанным именем с магнитной ленты и записать в память компьютера. Если указан начальный адрес памяти, то запись байтов в память начать с этого адреса, а если указано число байтов, то в память записать только указанное число байтов.

LOAD имя файла DATA имя массива ()

Считать массив с указанным именем файла с магнитной ленты и записать в память компьютера. Присвоить массиву имя, указанное после ключевого слова DATA.

LOAD имя файла SCREEN ×

Считать экранную информацию с указанным именем с магнитной ленты и записать в экранную память компьютера.

MERGE имя файла

Считать программу с указанным именем с магнитной ленты и объединить ее с программой, содержащейся в памяти компьютера.

NEW

Очистить память компьютера, уничтожить содержащиеся в ней программы, переменные и массивы.

RUN (номер строки)

Начать выполнение содержащейся в памяти компьютера программы с указанной строки. Если строка не указана, начать с первой.

SAVE имя файла [LINE номер строки]

Записать программу на магнитную ленту, присвоив ей указанное имя. Если применено ключевое слово LINE с указанным номером строки, то после считывания при помощи команды LOAD, начать выполнение программы с указанной строки.

SAVE имя файла CODE (начальный адрес) (,число байтов)

Записать последовательность байтов на магнитную ленту, присвоив ей указанное имя. После ключевого слова CODE указываются начальный адрес последовательности байтов в памяти компьютера и число записываемых байтов.

SAVE имя файлов DATA имя массива ()

Записать массив, имя которого указано после ключевого слова DATA, на магнитную ленту, присвоив указанное имя файла.

SAVE имя файла SCREEN *

Записать информацию экранной памяти на магнитную ленту, присвоить ей указанное имя.

VERIFY имя файла

Считать программу с указанным именем с магнитной ленты и сравнить с программой, содержащейся в памяти компьютера.

VERIFY имя файла CODE [начальный адрес] [,число байтов]

Считать последовательность байтов с указанным именем с магнитной ленты и сравнить с последовательностью байтов, содержащейся в памяти компьютера. Могут быть также указаны начальный адрес памяти и число сравнительных байтов.

VERIFY имя файла DATA имя массива ()

Считать массив с указанным именем с магнитной ленты и сравнить с массивом, содержащимся в памяти компьютера, имя которого узакано после ключевого слова DATA.

3.6.2. Операторы БЕЙСИКА

BEEP длительность, высота тона

Сформировать звуковой сигнал с указанными длительностью и высотой тона.

BORDER цвет

Установить цвет контура телевизионного экрана.

BRIGHT код яркости (;)

Установить яркость цвета индицируемых сигналов.

CIRCLE (оператор установки цвета:) X координата, Y координата, радиус

Начертить на экране телевизора окружность.

CLEAR (адрес)

Стереть значения всех переменных и массивов и выполнить действия оператора CLS и RESTORE. Если указан адрес, тогда

изменить последний (конечный) адрес системной зоны Бейсика на указанный адрес.

CLS

Очистить экран телевизора.

DATA элемент данных (,элемент данных)

Установить список элементов данных внутри программы.

DEF FN имя функции ([имя переменной] [,имя переменной]) =выражение

Установить функцию потребителя. В выражении, характеризующем функцию, могут быть использованы фиктивные переменные, имена которых указаны в левой части равенства между скобками.

DIM имя массива (индекс [,индекс] [,длина])

Описать массив, т. е. выделить ему место в памяти, установить число его измерений и максимальное значение индексов в каждом измерении. Если массив символьный, установить и длину строк массива.

DRAW (оператор установления цветов;) изменение по X координате, изменение по Y координате [, угол]

Начертить на экране телевизора прямую линию или, если указан угол, дугу.

FLASH код мигания (;)

Установить или уничтожить мигание индицируемых символов.

FOR имя переменной цикла-начальное значение TO предел [STEP шаг]

Начало цикла FOR NEXT.

GO SUB номер строки

Передать управление подпрограмме в указанную строку.

GO TO номер строки

Переход к заданному номеру строки.

IF условие THEN оператор (: оператор)

Выполнить один или несколько операторов, следующих после ключевого слова THEN, если условие верное. В противном случае эти операторы пропустить.

INK цвет [;]

Установить основной цвет (чернил) на экране телевизора.

INPUT (комментарии) (;) (,) (') имя переменной (комментарии) = (символьная константа) (символьное выражение) (функция AT) (Оператор установления цвета) (;) (,) (')

Ввести во время управления программы данные с клавиатуры и присвоить их указанным переменным.

INVERSE код инверсного или прямого изображения (;)

Установить инверсные или воспроизвести прямые цвета индицируемых символов.

LET имя переменной-выражение

Присвоить указанной переменной значение выражения.

LPRINT (функция TAB;) (функция AT;) (функция CHR ж ;) (оператор установления цвета;) выражение (;) (,) (')

Отпечатать на печатающем устройстве указанные элементы данных таким же образом, как оператор PRINT индицирует их на экране.

NEXT имя переменной цикла

Конец цикла FOR NEXT.

OUT адрес порта, значение байта

Направить байт информации в указанный порт вывода.

OVER код управления (;)

Синдицировать символ, точку или начертить линию, не уничтожая уже находящийся в этой позиции символ, точку или линию.

PAPER цвет (;)

Установить цвет фона (бумаги) на экране телевизора.

PAUSE длительность

Остановить программу на ограниченный или неограниченный интервал времени.

PLOT (оператор установки цветов;) X координата, Y координата

Синдицировать на экране телевизора точку.

POKE адрес, значение байта

Записать байт информации в память по указанному адресу.

PRINT (функция TAB;) (функция AT;) (функция CHR ;) (оператор установки цвета;) выражение (;) (,) (')

Синдицировать на экране телевизора указанные элементы данных.

RANDOMIZE (цифровая величина)

Установить генерирование последовательности случайных или псевдослучайных чисел при помощи функции RND.

READ имя переменной (, имя переменной)

Присвоить указанным переменным значения, взятые из списка элементов данных, составленного при помощи оператора DATA.

REM любой текст

Составить комментарии, поясняющие действие программы. На выполнение программы влияния не оказывает.

RESTORE (номер строки)

Установить указатель списка элементов данных при операторе DATA, находящемся в указанной строке. Если строка не указана, то при первом операторе DATA.

RETURN

Вернуть управление из подпрограммы оператору, следующему за последним выполненным оператором GO SUB в основной программе.

STOP

Остановить выполнение программы.

3.6.3. ФУНКЦИИ БЕЙСИКА

ABS цифровая величина

Вычислить абсолютное значение указанной цифровой величины. Цифровое значение должно быть заключено в скобки.

ACS цифровая величина

Вычислить арккосинус указанной цифровой величины. Цифровое значение должно быть заключено в скобки.

ASN цифровая величина

Вычислить арксинус указанной цифровой величины. Цифровое выражение должно быть заключено в скобки.

AT строка, столбец

Установить позицию выводимого на экран телевизора символа.

ATN цифровая величина

Вычислить арктангенс указанной цифровой величины. Цифровое выражение должно быть заключено в скобки.

ATTR (строка, столбец)

Установить атрибуты указанной на экране телевизора позиции символа, т. е. цвета чернил и бумаги, наличие яркости и бумаги.

BIN двоичное число

Перевести двоичное число в десятичное.

CHR ✕ цифровая величина (;) (+)

Установить символ, соответствующий указанному цифровому коду. Цифровое выражение должно быть заключено в скобки.

CODE символьная величина

Установить цифровой код указанного символа. Символьное выражение должно быть заключено в скобки.

COS цифровая величина

Вычислить значение косинуса, указанной цифровой величины. Цифровое выражение должно быть заключено в скобки.

EXP цифровая величина

Возвести постоянную e (2,7182818) в степень, которая равна указанной цифровой величине. Цифровое выражение должно быть заключено в скобки.

FN название функции ([значение] [,значение])

Вызвать функцию, установленную потребителем, которая была описана оператором DEF FN. Присвоить фиктивным переменным функции реальные значения, которые заключены в скобки.

IN цифровая величина

Считать байт из вводимого порта, адрес которого указывает цифровая величина. Цифровое выражение должно быть заключено в скобки.

INKEY ¤

Выдать информацию о том, какая клавиша клавиатуры нажата в настоящий момент.

INT цифровая величина

Установить наибольшее целое число, не превышающее указанную цифровую величину. Цифровое выражение должно быть заключено в скобки.

LEN символьная величина

Установить в указанной символьной величине число символов. Символьное выражение должно быть заключено в скобки.

LN цифровая величина

Вычислить натуральный логарифм указанной цифровой величины. Цифровое выражение должно быть заключено в скобки.

PEEK цифровая величина

Считать байт из ячейки памяти, адрес которой указывает цифровая величина. Цифровое выражение должно быть заключено в скобки.

PI

Установить числовое значение постоянной пи (3,14...).

POINT (X координата, Y координата)

Установить, в какой цвет окрашена точка с указанными координатами на экране телевизора.

RND

Сформировать случайное число, меньше 1, но не меньше 0.

SCREEN ¤ (строка, столбец)

Установить, какой символ будет индицироваться в указанной позиции экрана.

SGN цифровая величина

Установить знак указанной цифровой величины. Цифровое выражение должно быть заключено в скобки.

SIN цифровая величина

Вычислить синус указанной цифровой величины. Цифровое выражение должно быть заключено в скобки.

SQR цифровая величина

Вычислить квадратный корень указанной цифровой величины. Цифровое выражение должно быть заключено в скобки.

STR ¤ цифровая величина

Заменить указанную цифровую величину строкой символа. Цифровое выражение должно быть заключено в скобки.

TAB столбец:

Установить позицию выводимых символов в текущей или следующей строке экрана телевизора.

TAN цифровая величина

Вычислить тангенс указанной цифровой величины. Цифровое значение должно быть заключено в скобки.

Начальная строка или ее индентификатор ([номер первого символа в начальной строке] TO [номер последнего символа в начальной строке])

Сформировать новую строку, которая была бы частью начальной строки. Для новой строки следует выбирать символы из начальной строки, начиная и заканчивая символами, номера которых указаны по обеим сторонам ключевого слова TO. Символьное выражение должно быть заключено в скобки.

USR цифровая величина

Вызвать подпрограмму, написанную машинным кодом, которая хранится в памяти, начиная с адреса, который указывается цифровой величиной. Цифровое выражение должно быть заключено в скобки.

USR символьная постоянная или переменная

Установить начальный адрес записанных в память кодов символа устанавливаемой потребителем графики.

VAL символьная постоянная или переменная

Заменить указанную строку с цифровым значением на число.

VAL $ название символьной переменной

Заменить указанную символьную строку символьным выражением.

3.6.4. ЛОГИЧЕСКИЕ ОПЕРАЦИИ БЕЙСИКА

условие AND условие

Соединить два условия в одну комбинацию. Комбинация справедлива только тогда, когда справедливы оба условия.

NOT условие

Заменить значение условия на противоположное.

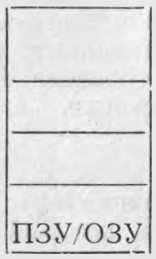
условие OR условие

Соединить два условия в одну комбинацию. Комбинация справедлива тогда, когда справедливо одно из условий.

Глава 4

ДОПОЛНИТЕЛЬНЫЕ ВОЗМОЖНОСТИ ВАШЕГО «ЭРУДИТА»

4.1. РАСПРЕДЕЛЕНИЕ ПАМЯТИ



RAMTOP (вершина памяти) ОЗУ пользователя

Служебные ячейки Бейсика

Экранная область

Отключаемое ПЗУ или ОЗУ пользователя Бейсик «Эрудит»

Общий объем памяти «Эрудита» 64 кБайт. Из них первые 16 кБайт занимает системное ПЗУ с языком программирования Бейсик. Около 6 кБайт отводится под экранную область и не-

сколько сотен под служебные ячейки Бейсика, необходимые для его правильного функционирования. Остальная память отводится под программы пользователя.

4.2. РАСШИРЕНИЕ ПАМЯТИ

Пользователь при желании может отключать системное ПЗУ и использовать 16 кБайт ОЗУ для своих программ. Предупреждение: отключив ПЗУ, вы лишаетесь языка программирования Бейсик и должны сами заботиться о том, чтобы компьютер понимал ваши команды.

Регистр переключения адресуется как внешнее устройство с адресом 0. Его можно переключить командой Бейсика OUT 0,9. Возврат в прежнее состояние — записать в регистр управляющее слово 0.(OUT 0,0). Системное программное обеспечение работу с регистром не поддерживает. В качестве примера на кассете дана программа «МОНИТОР 82», использующая освобожденное адресное пространство.

Персональный бытовой компьютер

«ЭРУДИТ-002»

Сдано в набор 15.08.91. Подписано в печать 10.X.91. Формат 60×84/16.
Печать высокая. Гарнитура литературная. Объем 10,5 п. л. Тираж 3000 экз.
Заказ 1089. Цена договорная.

Типография «Моряк», Одесса, ул. Ленина, 26

ДЛЯ ЗАМЕТОК

ДЛЯ ЗАМЕТОК