ZX - РЕВЮ
N11,12 1993

"ИНФОРКОМ" 121019, Москва, Г-19, а/я 16

Вниманию читателей!

Мы продолжаем публиковать адреса пунктов, в которых можно приобрести наши материалы.
г. МОСКВА, ул. Новый Арбат, д. 2. 19-е отделение связи, 1-ый этаж операционного зала.
г. МОСКВА, радиорынок в Митино, проезд до ст. метро Тушинская или поездом до пл. Трикотажная (Рижское направл.). Суббота, воскр. 9 - 14. Киоск N J-37.
г. МОСКВА, радиорынок в Царицыно, проезд до ст. метро Царицыно. Суббота, воскр, 9 - 14.
г. БЕЛГОРОД, магазин "РАДИОТОВАРЫ", ул. Ленина, 32.
г. БЕЛГОРОД, Октябрьская 84, кв. 103, "СТУДИЯ КОМПЬЮТЕР".
г. ВЛАДИВОСТОК, Океанский проспект, 140, магазин "ПАНОРАМА". Проезд трамваем до ост. Некрасовская.
г. ВОРОНЕЖ, студия компьютерных игр SAN-SAN, магазин-салон "ЭЛЕКТРОНИКА", тел. 14-00-73.
г. ДНЕПРОПЕТРОВСК, ул. Шевченко, 34. фирма "ЭКОС".
г. ЕКАТЕРИНБУРГ, магазин "СПЕКТРУМ", Главный проспект, 99.
г. ИЖЕВСК, радиорынок "Козий парк".
г. КЕМЕРОВО, магазин "ТЕХНИЧЕСКАЯ КНИГА", ул. Весенняя, 24.
г. КЕМЕРОВО, магазин "ОРБИТА", пр. Ленина, 133.
г. КИРОВ, "Дом науки и техн.", магазин-салон "МАРС",ул. Производственная, Д.27.
г. КРАСНОЯРСК, радиорынок, проезд до ост. "Затон", субб, вскр.
г. НАБЕРЕЖНЫЕ ЧЕЛНЫ, Татарстан, Новый Город, салон-магазин "ПРИНТЕР".
г. НИЖ. НОВГОРОД, ИМА "Ф-ПЛЮС". Магазин "ФОТОЛЮБИТЕЛЬ", ул. Горького, 146.
г. ОРЕНБУРГ, магазин "ВОЕННАЯ КНИГА", ул. Советская.
г. ПРОКОПЬЕВСК, Кемеровской об. Тырган, магазин "ОДЕЖДА", фирма "РОНЭТ".
г. РЫБИНСК, ул. Гоголя, 5. ТТЦ "ГНОМ".
г. ХАБАРОВСК, ул. Запарина 65, магазин "ФИЛАТЕЛИЯ".
г. ЧЕБОКСАРЫ, маг. "ЭКСПРЕСС". НПК фирма "НОВА", ул. Привокзальная, Д.6.
г. ЯРОСЛАВЛЬ, магазин "РАДИОСПОРТ-ТУРИЗМ", Ленинградский пр-т.

СПЕКТРУМ В ШКОЛЕ

В последнее время наши статьи из рубрики "Спектрум в школе" предназначались в основном для старшеклассников. Но ведь если в Вашей школе есть кабинет информатики, то, наверное, и малышам было бы интересно в него заглянуть.

Предельно простая программа, которую мы приводим сегодня, послужит для развлечения и приобщения к компьютеру тех, кто только-только учится считать до десяти.

Суть программы состоит в том, что на экране на некоторое время изображаются несколько фигурок и малыш должен сосчитать, сколько их и нажать соответствующую клавишу на компьютере. В качестве фигурок мы приняли любимый детьми "черепашек-ниндзя", изобразив их с помощью четырех символов UDG-графики. Несмотря на свою очевидную простоту, эта программа может иметь вполне реальный успех, если ей сделать приличный художественный дизайн, ввести в нее элементы мультипликации и красиво озвучить. Так что дерзайте, все в Ваших руках.

Конструкция одного "героя" сделана из четырех символов UDG и имеет размер 16X16 пикселов. Символы располагаются следующим образом:

```
[A]  [B]
[C]  [D]
```

Для того, чтобы отличить символы UDG в программе от обычных символов, мы взяли их в квадратные скобки. Вам, конечно, никаких скобок набирать при наборе программы не

надо (надо только перейти в графический режим - CAPS SHIFT + 9 - курсор "G").

```
10 REM **СОСЧИТАЙКА**
20 LET wait = 310
25 LET povtor=140
30 REM Ввод символов графики пользователя.
40 FOR n=0 TO 31
50 READ a
60 POKE USR "a"+n,a
70 NEXT n
80 DATA 1,1,3,15
81 DATA 31,59,63,15
82 DATA 128,129,134,248
83 DATA 252,198,194,124
84 DATA 55,70,134,12
85 DATA 12,12,24,16
86 DATA 224, 112,48, 48
87 DATA 96, 96, 112,0
90 REM Запустите набранные строки командой RUN, после чего можете продолжать набор
      программы.
100 LET a$="[AB]"
110 LET b$="[CD]"
120 DIM x(10): DIM y(10)
130 BORDER 1: PAPER 7: INK 2: BRIGHT 1
140 REM povtor
150 CLS
160 LET count=1+INT(9*RND)
170 REM Расположение черепашек
180 LET x(1)=2*INT(16*RND)
190 LET y(1)=2*INT(10*RND)
200 FOR n=2 TO count
210 LET x(n)=2*INT(16*RND)
220 LET y(n)=2*INT(10*RND)
230 FOR k=1 TO n-1
240 IF x=(n)=x(k) AND y(n)=y(k) THEN LET n=n-1: REM Если у нас случайно совпали координаты
      позиции печати двух черепашек, то играющий на экране увидит только одну и потому этот
      вариант надо исключать.
250 NEXT k: NEXT n
260 REM Печать "черепашек"
270 FOE n=1 TO count
280 PRINT AT y(n),x(n); a$
282 LET y(n) = y(n)+1
285 PRINT AT y(n),x(n); b$
290 NEXT n
300 PRINT INK 9; AT 20,10;"???????????"
310 REM wait
320 IF INKEY$="" THEN GO TO wait: REM здесь между кавычками не должно быть пробела
330 IF INKEY$ <> STR$ count THEN BEEP .5,0: BEEP 1,-24: GO TO wait
340 REM Правильный ответ
350 FLASH 1: CLS
360 FOR n=1 TO 10
370 BEEP .33,12
380 BEEP .33,24
390 NEXT n
400 FLASH 0
410 GO TO povtor
```

SINCLAIR LOGO

(Окончание)

Начало см. на с. 69-74, 90-96, 134 - 138, 180 - 186.

ГЛАВА 9. ВВОД И ВЫВОД ИНФОРМАЦИИ

До сих пор мы рассматривали только один метод ввода информации в компьютер - READLIST и (если не учитывать черепашью графику) только один метод вывода информации - PRINT. В этой главе мы рассмотрим некоторые методы, которые дадут программисту новые возможности для ввода/вывода, хотя и ценой более сложного программирования.

READCHAR ждет, когда пользователь нажмет какую-либо клавишу, принимает один символ с клавиатуры и выдает его в качестве выходного результата. После нажатия клавиши не надо нажимать ENTER, принятый символ не изображается на экране, ввод символа не сопровождается изображением "приглашения". Попробуйте:

```
MAKE "CHARACTER READCHAR
```

Нажмите ENTER - система ожидает нажатия любой клавиши. Нажмите любую клавишу. На экране появится приглашение "?" - это означает, что ЛОГО-систена вернулась в командный режим.

Вы можете проверить, что за символ был введен:

```
PRINT :CHARACTER
```

Это обычный прием, когда надо узнать, что за клавиша была нажата. Вот, например, программа, с помощью которой Вы можете управлять движением черепашки по экрану.

```
TO DRAW
   MAKE "CHARACTER READCHAR
   IF :CHARACTER = "F [FORWARD 5]
   IF :CHARACTER = "B [BACK 5]
   IF :CHARACTER = "R [RIGHT 10]
   IF :CHARACTER = "L [LEFT 10]
   IF :CHARACTER = "S [STOP]
   DRAW
END
```

Функция READCHAR бывает очень полезной, когда вместо символов Вы работаете с их кодами. Коды символов с 0 по 127 называют кодами ASCII (American Standard Code for Information Interchange). Коды с 0 по 31 используются компьютером в качестве управляющих при операциях ввода/вывода (например для перемещения курсора) и не являются печатными, т.е. их нельзя изобразить на экране, хотя они и влияют на то, как изображаются другие (печатные) коды. Коды с 32-го по 127 - печатные символы (буквы, цифры, знаки препинания). Дополнительно к стандарту ASCII "Спектрум" имеет еще ряд кодов для символов.

Коды 123 - 143 представляют символы блочной графики, а символы 144 - 164 - это символы графики пользователя.

Вы можете узнать, какому символу какой код соответствует с помощью следующей несложной программы.

```
TO SHOWASCII
   MAKE "CHARACTER READCHAR
   PRINT ASCII :CHARACTER
   SHOWASCII
END
```

Создав такую процедуру, Вы можете дать команду SHOWASCII и, нажав любую клавишу, увидеть на экране код этого символа.

Противоположное действие имеет команда CHAR. Она переводит код от 32 до 143 в соответствующий графический символ.

Так, PRINT CHAR 65 даст:

```
A
```

Можно использовать команду CHAR совместно с командой ASCII. Нижеприведенная

процедура будет шифровать ту информацию, которую Вы введете с клавиатуры.

```
TO CONFUSE
   MAKE "CODE ASCII READCHAR
   IF :CODE = 13 [STOP]
   TYPE CHAR (:CODE + 1)
   CONFUSE
END
```

Процедура прекратит свою работу, когда будет нажата клавиша ENTER (ее код = 13).

Если Вам не надо ожидать нажатия клавиши, то Вы можете использовать функцию KEYP. Она проверяет факт нажатия какой-либо клавиши и выдает TRUE или FALSE в зависимости от того, имело или нет место такое событие.

Вот еще одна программа для черепашки, отличающаяся тем, что между нажатиями клавиш черепашка продолжает свое движение.

```
TO DRAW2
   IF KEYP [DOIT]
   FORWARD 5 DRAW2
END
TO DOIT
   MAKE "CHARACTER READCHAR
   IF :CHARACTER ="M [RIGHT 90]
   IF :CHARACTER ="Z [LEFT 90]
END
```

Клавишами M и Z Вы теперь можете управлять движением черепашки по экрану.

Символы от 144 до 164 включительно могут задаваться непосредственно пользователем. О том, как это сделать, рассказано в инструкции к компьютеру (и в книгах ИНФОРКОМа). Но для того, чтобы задать эти символы, в определенные адреса памяти компьютера надо внести свои значения. В языке ЛОГО для этой цели служит команда .DEPOSIT. Но будьте осторожны при работе с этой командой. Она напрямую меняет содержимое ячеек памяти компьютера. ЛОГО-система не в состоянии проверить правильность Ваших действий и, если Вы станете использовать эту команду без четкого понимания того, что и зачем Вы делаете, есть большая вероятность того, что Вы можете разрушить созданные Вами и хранящиеся в памяти компьютера процедуры.

Пара нижеприведенных процедур демонстрирует задание нового символа графики пользователя путем помещения в память компьютера (начиная с адреса 64215) списка из восьми чисел, которые и определяют конструкцию нового символа.

```
TO DEFCHAR :CH :ALIST
   IF OR NOT NUMBERP :CH NOT LISTP :ALIST [STOP]
   IF OR :CH<144 :CH>164 [STOP]
   DODEF :ALIST 0
END
TO DODEF :ALIST :COUNT
   IF EMPTYP :ALIST [STOP]
   .DEPOSIT 64216 + :CH*8 + :COUNT FIRST :ALIST
   DODEF BUTFIRST :ALIST :COUNT +1
END
```

Для того, чтобы определить, какими должны быть эти восемь чисел, задающие конструкцию символа, Вам придется почитать соответствующую литературу или поэкспериментировать. Так, например, следующая группа чисел задаст символ 144 в виде стрелки, направленной вниз:

```
DEFCHAR 144 [16 16 16 16 146 84 56 16]
```

Для управления выводом текста на экран компьютера тоже существует ряд полезных процедур и мы последовательно их рассмотрим.

SETCURSOR - принимает в качестве входных параметров список из двух чисел и помещает курсор в соответствующую позицию экрана. С этой позиции и начнется в последующем печать выводимого текста. Первое число списка указывает на номер экранного столбца (от 0 до 31), а второе число - на номер экранной строки (от 0 до 21). Левому верхнему углу соответствуют координаты (0,0).

SETTC (SET Text Color) тоже принимает на входе два числа. Первое число задает цвет

фона (аналог оператора БЕЙСИКА PAPER), а второе число задает цвет символов (аналог INK). Обычная ситуация для режима печати черным по белому - [7,0]. Вы можете поэкспериментировать со следующей процедурой:

```
TO TESTCOL
   TEXTSCREEN
   SETCUR [12 10]
   SETTC [1 6]
   PRINT [HELLO THERE]
   MAKE "G READCHAR
   SETTC [7 0]
END
```

В последних строках организовано ожидание нажатия клавиши для восстановления нормальных цветов. Параметр яркости фона, на котором печатаются символы, может задаваться командой BRIGHT (как и в БЕЙСИКе). Команда BRIGHT должна иметь при себе один параметр, который может быть равен 0 или 1. BRIGHT 1 - повышенная яркость, BRIGHT 0 - обычный режим.

Команда INVERSE включает режим печати символов в инвертированном виде, т.е. цвет символов становится цветом фона и наоборот.

Команда FLASH включает режим мигания символов, когда попеременно цвета символов и фона меняются местами.

Команда NORMAL отбивает эффект команд BRIGHT, INVERSE, FLASH и восстанавливает исходный режим печати текста.

Для тех, кто хочет посмотреть эти команды в работе, предлагается следующий пример:

```
TO FANCYNAME
   PRINT [WHAT IS YOUR NAME?]
   MAKE "NAME READLIST
   TEXTSCREEN
   BRIGHT 1
   SHOWNAME
   SETTC [7 0]
   SETCUR [0 21]
END

TO SHOWNAME
   IF KEYP [STOP]
   MAKE "X RANDOM 31
   MAKE "Y RANDOM 21
   MAKE "BG RANDOM 8
   MAKE "FG RNDCOL
   SETTC SENTENCE :BG :FG
   SETCUR SENTENCE :X :Y
   TYPE :NAME
   SHOWNAME
END

TO RNDCOL
   MAKE "FG RANDOM 8
   IF :FG=:BG [OUTPUT RNDCOL] [OUTPUT :FG]
END
```

Здесь процедура RNDCOL не только генерирует случайное число для установки цвета печатаемого сообщения, но еще и проверяет, не совпадает ли выбранный цвет символов с ранее установленным цветом фона и, если это так, то вызов RNDCOL повторяется до тех пор, пока эти цвета не станут разными.

А вот еще один пример для демонстрации приведенной выше техники. Группа процедур обрабатывает результаты матчей небольшой футбольной лиги. Главная процедура организует списки и печатает в цвете заголовки будущей таблицы, затем вызывается процедура HEADCOL, которая печатает заголовки столбцов, а потом процедура DOTEAM, обрабатывающая результаты для каждой команды.

```
TO LEAGUE
```

```
   MAKE "TEAMS [SPARTAK DINAMO AVANGARD PROGRESS]
   MAKE "COLS [P W D L PTS]
   TEXTSCREEN
   SETCURSOR [9 0]
   SETTC [5 0]
   TYPE [LEAGUE TABLE]
   SETCURSOR [0 2]
   SETTC [7 3]
   TYPE "TEAM
   HEADCOL 1
   SETTC [7 0]
   DOTEAM 1
   PRINT "
END
```

Процедура HEADCOL выбирает данные из списка заголовков колонок и печатает эти заголовки в нужной позиции экрана с помощью команды SETCURSOR. Работа продолжается, пока все пять заголовков не будут распечатаны.

```
TO HEADCOL :COL
   IF :COL>5 STOP
   SETCURSOR SENTENCE (3*:COL + 11] 2
   TYPE ITEM :COL :COLS
   HEAADCOL :COL + 1
END
```

Процедура DOTEAM определяет номер строки, в которой печатаются результаты для данной команды (он зависит от номера команды в списке команд). Она принимает из списка название команды, а затем принимает от пользователя с клавиатуры количество сыгранных игр (P), количество побед (W) и ничьих (D). При этом она использует еще одну процедуру - GETNUM.

Первыми двумя входными параметрами GETNUM являются координаты позиции экрана, в которой должна состояться печать. Третий входной параметр поначалу - пустое слово, оно будет содержать число, введенное пользователем.

Количество проигрышей команды (L) и набранные ею очки (PTS) вводить не надо, они могут быть рассчитаны самой программой. Предполагается, что победа оценивается двумя очками, а ничья - одним очком.

```
TO DOTEAM :TEAMNO
   IF :TEAMNO COUNT :TEAMS [STOP]
   MAKE "LINE 2*:TEMNO + 2
   SETCURSOR SENTENCE 0 :LINE
   TYPE ITEM :TEAMNO :TEAMS
   MAKE "PLAYED GETNUM 14 :LINE "
   MAKE "WON GETNUM 17 :LINE "
   MAKE "DRAWN GETNUM 20 :LINE "
   MAKE "LOST :PLAYED - :WON - :DRAWN
   SETCURSOR SENTENCE 23 :LINE
   TYPE :LOST
   MAKE "POINTS 2*:WON + :DRAWN
   SETCURSOR SENTENCE 25 :LINE
   TYPE :POINTS
   DOTEAM :TEAMNO + 1
END
```

Прежде, чем рассмотреть процедуру GETNUM, мы рассмотрим еще одну связанную с ней процедуру GETDIG. Она принимает цифру с клавиатуры, а возвращает и эту цифру и ее код ASCII. Она передает также коды ENTER (13) и DELETE (12), если соответствующие клавиши нажаты. Если же нажата любая другая клавиша, процедура ее игнорирует и вызов повторяется. Цифрам от нуля до 9 соответствуют коды ASCII от 48 до 57.

```
TO GETDIG
   MAKE "DIG READCHAR
   MAKE "CODE ASCII :DIG
   IF OR :CODE = 13 :CODE = 12 [STOP]
   IF OR :CODE<48 :CODE>57 [GETDIG]
END
```

Теперь рассмотрим процедуру GETNUM. Начинается она довольно странно. Сначала устанавливается курсор в позицию, соответствующую ее первым двум входным параметрам, после чего печатается ее третий параметр, хоть он и пустой при первом вызове. Это нужно потому, что процедура работает рекуррентно и в процессе ее работы третий параметр будет изменяться.

В качестве курсора для печати используем мигающий вопросительный знак. Это легко реализуется следующими строками:

```
FLASH
TYPE *?
NORMAL
```

Цифра, введенная пользователем, поставляется из процедуры GETDIG. Оттуда же могут поступить и коды ENTER или DELETE. При появлении этих кодов следует убрать вопросительный знак из текущей позиции, в противном случае он продолжает мигать. Удаление вопросительного знака исполняется печатью на его месте пробела (символ 32). Могут быть проблемы с точным определением местоположения вопросительного знака. Самый лучший способ - пойти к началу вводимого числа (мы знаем где это начало) и снова напечатать число, а за ним уже пробел - на месте вопросительного знака.

```
SETCURSOR SENTENCE :X :Y
TYPE NO
TYPE CHAR 32
```

Процедура GETDIG может выдать три возможных результата: была нажата цифра с кодом от 48 до 57, либо клавиша ENTER (код 13) либо DELETE (код 12).

Если это цифра, то ее просто надо напечатать, добавив к ранее напечатанным цифрам вводимого числа. В нижеследующем примере предполагается, что числа имеют не более двух знаков и при попытке ввести третью цифру она игнорируется.

```
IF AND :CODE>13 (COUNT :NO)<2 MAKE "NO WORD :NO :DIG
```

Если нажата клавиша ENTER и если число не пустое, то мы можем напечатать это число.

Если нажата клавиша DELETE, то мы должны удалить последний символ числа, используя BUTLAST. Но и в этом случае сначала надо удостовериться, что число не является пустым словом. И если Вы рассмотрите нижеприведенную процедуру, то увидите в ней несколько необычную строку:

```
IF EMPTYP :NO [MAKE "NO"]
```

Кажется, что в ней нет необходимости, но это не так. Дело в том, что когда мы применяем BUTLAST к слову, в котором есть только один символ, то рассчитываем получить пустое слово, но вместо него получаем пустой список, с которым команда WORD, помещающая очередную цифру в число не сможет работать. Эта странная операция конвертирует пустой список в пустое слово,

```
TO GETNUM :X :Y :NO
   SETCURSOR SENTENCE :X :Y
   TYPE :NO
   FLASH
   TYPE "?
   NORMAL
   GETDIG
   SETCURSOR SENTENCE :X :Y
   TYPE :NO
   TYPE CHAR 32
   IF AND :CODE = 13 (NOT EMPTYP :NO) [OUTPUT :NO]
   IF AND :CODE = 12 (NOT EMPTYP :NO) [MAKE "NO BUTLAST :NO]
   IF EMPTYP :NO [MAKE "NO"] IF AND :CODE>13 (COUNT :NO)<2 MAKE "NO WORD :NO :DIG
   OUTPUT GETNUM :X :Y :NO
END
```

Программу можно улучшать и дальше. Можно предусмотреть, чтобы пользователь не смог ввести число побед больше, чем было сыграно игр, в результате чего число поражений станет отрицательным. Совершенствованию нет предела, но сама техника Вам теперь должна быть понятна.

ЗВУК.

"Спектрум" может выдавать звуковые сигналы. Это делается с помощью команды SOUND, после которой идет список из двух чисел. Первое число выражает длительность звучания ноты в секундах и должно быть в диапазоне от 0 до 10.5. Второе число - высота ноты, измеренная в полутонах относительно ноты "до" первой октавы. Это число должно быть в диапазоне от -60 до +69. Следующая процедура превратит клавиатуру Вашего компьютера в электроорган, но играть на нем весьма сложно.

```
TO KEYBOARD
   MAKE "KEY (ASCII READCHAR)-65
   IF AND :KEY>-59: KEY<70
   [SOUND SENTENCE 0.5 :KEY]
   KEYBOARD
END
```

Сыграть список нот равной длительности Вы можете с помощью процедуры:

```
TO PLAY :ALIST
   IF EMPTYP :ALIST [STOP]
   SOUND SENTENCE 1 FIRST :ALIST
   PLAY BUTFRST :ALIST
END
```

Полутон - это звуковой интервал между двумя соседними (белыми или черными) клавишами на фортепиано. Если Вы не знаете, как складывается музыкальный ряд из полутонов, то попробуйте сыграть следующую гамму "до мажор".

```
PLAY [0 2 4 5 7 9 11 12]
```

К сожалению, способ задания звуков с помощью команды SOUND очень далек от общепринятой нотной нотации. Впрочем, можно написать процедуру, которая сделает за нас необходимое преобразование. Для этого служит следующая приведенная нами процедура. Первый параметр, который она ожидает, задает темп музыкальной мелодии. Это продолжительность звучания (в секундах) ноты, длительность которой принята нами за единицу. Затем процедура принимает список нот и размещает их в предварительно созданном списке SLIST, после чего исполняется мелодия.

```
TO MUSIC
   MAKE "NOTELIST [DO DO# RE RE# MI FA FA# SO SO# LA LA# SI DO']
   MAKE "SLIST[]
   PRINT [TEMP?]
   PRINT [ВВЕДИТЕ НОТЫ ПО ОДНОЙ, СНАЧАЛА ДЛИТЕЛЬНОСТЬ]
   GETMUS
   PLAY :SLIST
END
```

Процедура GETMUS принимает с клавиатуры список из двух чисел.

Первое число - длительность ноты. Предполагается, что самая короткая из них имеет длину 1, а длительность прочих - 1, 2, 3,... и т.д. Реальная же длительность (в секундах) будет зависеть от введенного Вами параметра TEMP.

Высоту ноты будет определять ее название - Do, Re, Mi и т.д.

Процедуре GETMUS нужна еще одна вспомогательная процедура FIND, которая найдет объект в списке. Первые два входных параметра для процедуры FIND - это имена объекта и списка, а третий параметр - стартовый номер, с которого начинается отсчет в списке. Если мы хотим, чтобы первой нотой была "ДО" первой октавы, то положим этот параметр равным нулю.

```
TO GETMUS
   MAKE "NIOTE READLIST
   IF EMPTYP :NOTE [STOP]
   MAKE "DUR :TEMP*FIRST :NOTE
   MAKE "PITCH FIND LAST :NOTE :NOTELIST 0
   MAKE "SLIST LPUT LIST :DUR :PITCH :SLIST
   GETMUS
END

TO FIND :OBJECT :ALIST :N
   IF EMPTYP :ALIST [OUTPUT 0]
```

```
    IF :OBJECT = FIRST :ALIST
       [OUTPUT :N]
    OUTPUT FIND :OBJECT BUTFIRST :ALIST :N + 1
END
```

И, наконец, само исполнение музыки реализует процедура PLAY. Она отличается от ранее приведенной процедуры с тем же именем PLAY только тем, что полагает каждый элемент своего входного списка тоже списком, состоящим из двух элементов, которые затем передаются команде SOUND).

```
TO PLAY :ALIST
   IF EMPTYP :ALIST [STOP]
   SOUND FIRST :ALIST
   PLAY BUTFIRST :ALIST
END
```

Введя все процедуры, попробуйте программу в деле:

```
   TEMP?
?0.3
   ВВЕДИТЕ НОТЫ ПО ОДНОЙ, СНАЧАЛА ДЛИТЕЛЬНОСТЬ
?2SO
?4DO'
?2SI
?2LA
?4SO
?2LA
?1SI
?1DO'
?2MI
?2MI
?2FA
?2RE
?6DO
```

Когда закончите, нажмите ENTER и после короткой паузы музыка заиграет. Один раз введя мелодию, Вы сможете воспроизвести ее всякий раз, как захотите с помощью команды:

```
PLAY :SLIST
```

ГЛАВА 10. ЕСЛИ ЧТО-ТО НЕ ПОЛУЧАЕТСЯ

Мы начнем с того, что скажем несколько успокоительных слов тем, кто только что понял, что написанная им процедура не работают. Главное, что вам нужно знать о программировании и о компьютерах - это то, что неработоспособность - это естественное состояние любой компьютерной программы.

Неопытного программиста легко отличить от опытного. Новичок неприятно удивляется, когда только что написанная им программа не желает работать с первого раза. Опытные программист всегда удивляется, когда она с первого раза работает как надо и начинает искать в ней скрытые подвохи.

Можно сказать, что при программировании на языке ЛОГО Вас ждут два разных типа ошибок. Если Вы сделали ошибку при написания команд и ЛОГО не понимает, что Вы хотите сделать, Вы получите сообщение об ошибке. В других случаях ЛОГО понимает, что Вы хотели сделать и выполняет Ваши команды, но результат Вы получаете совсем не тот, который Вам необходим.

Методика отыскания и исправления ошибок и в том и в другом случае очень похожа. Хотя первый тип ошибок проще поддается исправлению, поскольку ЛОГО-система сама уже сказала Вам в своем сообщении что ей не нравится и в какой процедуре она наткнулась на препятствие. В этом смысле ЛОГО удобнее многих другие языков программирования, поскольку его сообщения как правило более осмысленны и содержательны, чем в других языках. Тем не менее, все-таки эти сообщения были написаны создателями языка задолго до того, как Вы начали писать свою программу. Им как бы пришлось "предугадать" Ваши возможные ошибки. Это не всегда можно сделать точно. Потому не удивляйтесь, если эти

сообщения будут не абсолютно точны в определении Ваших ошибок.

Делу может помочь краткий обзор некоторых наиболее часто встречающихся ошибок и сообщений. Например:

```
1. *You don't say what to do with ABCD*
(Вы не указали, что делать с ABCD)
```

Такое сообщение легко объяснить. Помните, что всякая операция, выдающая какой-то результат, слева должна иметь имя команды, которая примет этот результат. В нашем примере операция произвела слово ABCB и либо отсутствует команда, которая примет этот результат в качестве входного параметра, либо она есть, но это не тот входной параметр, который для нее приемлем.

Рассмотрим пример. Предположим, что некоторая процедура SAYABCD выдает в качестве результата ABCD и Вы запишете такую строку:

```
PRINT "RESULT SAYABCD
```

В результате Вы получите приведенное выше сообщение об ошибке. Дело в том, что команда PRINT ожидает только один параметр в качестве входного и он здесь есть - "RESULT. Никакого указания, что делать с ABCD, здесь нет. Чтобы исправить ошибку, очевидно надо использовать команду SENTENCE, дабы команда PRINT могла распечатывать несколько данных.

```
2. *Not enough inputs to ABCD*
(недостает входных параметров для ABCD)
```

Если Вы рассмотрите определение процедуры ABCD, то должны обнаружить, что она ожидает большое число входных параметров, чем Вы ей дали. Может быть, это будет выглядеть и неочевидным, но метод анализа строк ЛОГО, который мы привели в Главе 3 Вам должен помочь.

```
3. *XYZ doesn't output to ABCD*
(XYZ не выдает данные для ABCD
```

Эта ошибка возникает тогда, когда операция ABCD применяется к некоторой команде XYZ (не являющейся операцией), которая не выдает выходного результата, необходимого для успешной работы ABCD.

```
4. *Too many inside parenthesis*
(слишком много скобок).
```

Эта ошибка происходит, когда Ваш список имеет слишком много внутренних круглых скобок. В большинстве случаев присутствие такой ошибки может так напугать программиста, что он вообще будет стараться обходиться без таких скобок, хотя это тоже излишняя крайность.

```
5. "XYZ doesn't like ABCDE as input*
(XYZ не принимает ABCDE в качестве входного параметра).
```

Эта ошибка может происходить в двух случаях. Либо результат ABCDE неприемлем для XYZ, т.к. он имеет не тот тип (это, например, число, а должен быть список):

```
SETCUR 10 15
```

Либо результат имеет правильный тип, но не находится в допустимых пределах:

```
SETCUR [10 45]
```

При появлении такой ошибки Вам следует внимательно вручную проверить что именно Ваша процедуру ожидает в качестве входного параметра.

Сообщение об этой ошибке может иметь весьма обескураживающий вид, например:

```
*XYZ doesn't like as input*
```

Здесь выпало слово ABCDE - это потому, что результат ABCDE является в данном случае пустым словом.

```
6. *Not enough space to proceed*
(Не хватает рабочего пространства).
```

Это сообщение о том, что наступила нехватка оперативной памяти компьютера для продолжения дальнейших вычислений. В такой ситуации можно попробовать принять

предохранительные меры. Один из приемов - запустить процедуру RECYCLE. Эта команда выполнит то, что на компьютерном жаргоне называют "уборкой мусора". Она разыщет области памяти, выделенные в свое время для каких-то операций, в которых более нет необходимости и освободит дополнительный объем памяти. Может быть, эта мера и не приведет к какому-то положительному результату, поскольку время от времени система сама производит "уборку мусора" автоматически.

Другой, более радикальный прием - удалить из памяти те процедуры, которые Вам более не нужны.

Но команду RECYCLE можно использовать и для повышения быстродействия Ваших программ. Поскольку автоматическая "уборка мусора" может занимать некоторое время, то работа программы может быть задержана. В разных частях программы, в разных процедурах у Вас могут быть разные требования по скорости. Поэтому имеет смысл самому поставить RECYCLE в тех местах, где Вы можете позволить себе ожидание, а не дожидаться, пока автоматика начнет заниматься этим делом в тех местах, где для Вас это не удобно.

Теперь мы рассмотрим ряд приемов, с помощью которых отыскивают и устраняют ошибки в программах. В качестве примера мы вернемся к одной из ранее рассмотренных процедур - это процедура GETNUM из Главы 9. Процедура предназначена для ввода двузначных чисел и изображения мигающего курсора в экранной позиции с координатами x и y. Процедура GETNUM использует в работе процедуру GETDIG, рассмотренную в предыдущей главе.

```
TO GETDIG
   MAKE "DIG READCHAR
   MAKE "CODE ASCII :DIG
   IF OR :CODE=13 :CODE=12[STOP]
   IF OR :CODE<48 :CODE>57[GETDIG]
END
```

Испытание процедуры GETDIG прошло нормально. При этом нажимались различные цифровые клавиши и процедура правильно распечатывала значения DIG и CODE.

После этого испытания была набрана (так, как показано ниже) процедура GETNUM.

```
TO GETNUM :X :Y :NO
   SETCUR SE :X :Y
   TYPE :NO
   FLASH
   TYPE "?
   NORMAL
   GETDIG
   IF AND :CODE=13 NOT EMPTYP :NO
      [OUTPUT :NO]
   IF AND :CODE=12 NOT EMPTYP :NO
      [MAKE "NO BUTLAST :NO]
   IF AND :CODE>13 COUNT :NO<2
      [MAKE "NO WORD :NO :DIG]
   OUTPUT GETNUM :X :Y :NO
END
```

Может быть Вам будет интересно набрать эту процедуру и проследить этапы отладки.

Первое тестирование процедуры выполнялось командой:

```
PRINT GETNUM 10 10 "
```

Все прошло нормально и мигающий вопросительный знак был напечатан в позиции с координатами [10 10]. Но при попытке нажать клавишу 3 появилось сообщение об ошибке:

```
5 is not TRUE or FALSE in GETNUM
```

(5 не имеет значения "ИСТИНА" или "ЛОЖЬ" в процедуре GETNUM).

Единственно, где в процедуре GETNUM могут участвовать логические значения TRUE или FALSE - это три строки IF... Неясно только, какая же из трех работает неправильно.

Для выяснения этого вопроса в процедуру GETNUM были встроены три отладочные строки печати после каждой из строк IF.

```
PRINT "I1
PRINT "I2
PRINT "I3
```

После этого вновь была нажата клавиша "3" и был получен следующий результат:

```
I1
I2
5 is not TRUE or FALSE in GETNUM
```

Теперь стало ясно, что первые две строки IF проходят нормально и неприятности происходят в третьей строке. Три вспомогательные отладочные строки PRINT были удалены, но перед последним IF были поставлены две новые контрольные строки:

```
PRINT :CODE
PRINT COUNT :NO
```

Процедура была запущена еще раз и вновь нажата клавиша "3". Результат был таким:

```
51
0
5 is not TRUE or FALSE in GETNUM
```

Итак, 51 - это код символа "3", а 0 - это количество символов в переменной NO (в этот момент времени). Так откуда же появилось число 5? Для проверки были опять изменены две строки контрольной печати.

```
PRINT :CODE>13
PRINT COUNT :NO<2
```

Вторая строка дала неожиданный результат:

```
< does not like as input
```

Обратим внимание на то, что единственный случай, когда команда "<" может получить пустое слово в качестве входного параметра, может быть только тогда, когда вместо COUNT :NO будет стоять просто :NO. Итак, может быть все дело просто в порядке исполнения операций в этой строке?

Тогда решение будет в использовании круглых скобок так, чтобы сначала выполнялась функция COUNT :NO, а потом ее результат сравнивался бы с числом 2.

```
IF AND :CODE>13 (COUNT :NO)<2
   [MAKE "NO WORD :NO :DIG]
```

Итак, что же произошло? Сначала почему-то при сравнении пустого слова :NO и числа 2 прошел результат FALSE (видимо, это некорректное сравнение), а потом COUNT FALSE дал число 5 (по количеству символов в слове FALSE), которое конечно не может служить корректным вводом для последующей операции AND.

Теперь процедура работает нормально? Нажав клавиши "3" и "2", мы получим на экране "32" и мигающий знак вопроса. После нажатия ENTER на экране появится число 32, как результат работы команды PRINT. Прочие тесты показывают, что прочие клавиши игнорируются и более двух цифр ввести в число не удается.

Но нам надо еще проверить работу DELETE. Запустим процедуру еще раз. Нажмем клавишу "2". На экране появится:

```
2?
```

Теперь нажмем DELETE и на экране появится:

```
??
```

Внимательное исследование показало, что левый знак вопроса появился после стирания цифры 2 на ее месте, там ему и положено быть, а правый знак вопроса - это тот, который остался там с прошлого раза - его не должно бы быть. Прежде чем стирать цифру 2 этот вопросительный знак должен быть погашен печатью поверх него пробела. Об этом мы говорили в предыдущей главе.

После исправления и этой ошибки нажатие на цифру "2" дает:

```
2?
```

А стирание - DELETE дает:

```
?
```

Отлично! Теперь для проверки нажмем на цифру "3" ... и получим новое сообщение об ошибке:

```
Word doesn't like [] as input in GETNUM
```

(команда WORD не принимает [] в качестве входного параметра в процедуре GETNUM)

Локализовать эту ошибку несложно, поскольку в процедуре GETNUM команда WORD встречается только один раз. Но сообщение вполне определенно указывает, что в качестве параметра команды WORD оказалось не слово, а пустой список []. Более того, мы

обнаружили, что проявляется эта ошибка только после использования DELETE и потому может быть не строка, в которой стоит WORD является причиной ошибки, а та строка, которая реализует операцию DELETE.

Теперь можно поэкспериментировать:

```
MAKE "NO 2
MAKE "NO BUTLAST :NO
```

И попробуйте дать команду

```
PRINT :NO
```

В качестве результата Вы получите пустую строку, что и следовало ожидать. А теперь попробуйте так:

```
MAKE "NO WORD :NO 3
```

и Вы получите сообщение об ошибке такое же, как в предыдущей процедуре. Теперь надо вспомнить об еще одной полезной команде вывода, о которой мы до сих пор не говорили. Мы рассказывали только о PRINT и TYPE, а есть еще команда SHOW. Она отличается тем, что показывает списки вместе с наружными скобками - очень полезная вещь для проверок.

```
SHOW :NO
```

дает:

```
[]
```

Вот и открыта причина необычного поведения процедуры. После применения BUTLAST к слову, содержащему только один символ, мы получили не пустое слово, а пустой список. Это фактически системная ошибка. Не исключено, что в той версии языка, с которой работаете Вы, эта ошибка уже ликвидирована, но для нас она послужила наглядным примером технологии отладки программ.

Способ обхода этой ошибки был нами продемонстрирован в предыдущей главе и здесь мы не будем повторяться.

Можно сделать первые выводы. Во-первых, отлаживайте свои процедуры порознь. ЛОГО любезно сообщает вам, в какой процедуре произошла ошибка, но первопричина может быть в другой процедуре. Например, причиной ошибки может быть неправильно переданный параметр, полученный из другой процедуры. Поэтому прежде чем подключать одну процедуру к другой, постарайтесь выполнить в ней все возможные проверки. Может быть, Вам потребуется написать специальные тестирующие процедуры для тестирования отдельных фрагментов будущей сложной структуры.

Например, если Вы пишете программу для карточной игры, то прежде чем подключать к комплексу процедуру, которая будет сдавать карты из колоды в случайном порядке, полезно написать тестирующую процедуру, которая будет просто распечатывать значения карт из колоды и убедиться, что они действительно идут в случайном порядке.

Если Вы убедитесь, что в этой процедуре все в порядке, то столкнувшись впоследствии с проблемами в вышестоящей процедуре, Вы сможете сказать себе "Здесь я все проверил, здесь ошибку можно не искать, будем искать ее во вновь добавленных фрагментах". Хотя, конечно полностью исключить вероятность ошибки удается не всегда.

Другой метод поиска ошибок - трассирование (трейсинг). Он состоит в прослеживании шаг за шагом того, что делает программа. Этот метод является наиболее важным в тех случаях, когда программа "зависает", т.е. она работает, но не выдает никаких результатов на экран. Возникает впечатление, что она "не работает".

Трейсинг можно делать двумя разными способами. Самый простой - сделать так, чтобы в начале каждой процедуры стоял оператор печати имени этой процедуры. Тогда даже при "зависании" программы Вы сможете увидеть, в какой процедуре или в каких процедурах это "зависание" произошло. Подобный трейсинг делать очень просто, а эффект он дает большой. Когда программа отлажена, то лишние строки печати можно убрать.

Другой прием трейсинга - состоит в прослеживании не процедур, а переменных. Это следующая линия атаки, когда Вы в принципе уже установили, в каких процедурах происходит сбой. В этом случае Вы в нескольких местах этих процедур ставите печать важных переменных (переменной) и по характеру их (ее) изменения выясняете конкретно точку, вызывающую зависание.

Если Вы столкнетесь с тем, что трассирующие операторы ведут печать на экране так быстро, что не удается визуально отследить и понять, что делает программа, то в паре с операторами печати надо поставить и замедляющие отладочные операторы. Удобным таким оператором является READCHAR, который приостановит работу до тех пор, пока Вы не нажмете клавишу.

Вы можете написать и свою трассирующую процедуру, назвав ее например TRACE. При создании такой процедуры Вы можете использовать три новых примитива.

COPYDEF - выполняет новую копию процедуры

TEXT - конвертирует процедуру в текстовой список ее операторов. С этим списком может работать ЛОГО, как со списком, а не как с исполняемой процедурой.

DEFINE - выполняет обратное действие - конвертирует список инструкций в исполняемую процедуру.

Теперь мы можем создать трассирующую процедуру TRACE, которая будет перед тем, как исполнять очередную строку отлаживаемой процедуры, распечатывать ее содержимое на экране. Вызывается она так:

```
TRACE "PROC
```

А отключается

```
UNTRACE "PROC
```

PROC - имя отлаживаемой процедуры. Отладочные строки будут печататься только тогда, когда исполняется эта процедура PROC.

```
TO TRACE :APROC
   IF DEFINEDP WORD ". :APROC
      [PRINT SENTENCE :APROC [ALREADY TRACED] STOP]
   IF NOT DEFINEDP :APROC
      [PRINT SENTENCE :APROC [NOT A PROCEDURE] STOP]
   COPYDEF WORD ". :APROC :APROC
   MAKE :APROC TEXT :APROC
   MAKE ".P LIST FIRST THING :APROC SENTENCE "PRINT WORD CHAR 34 :APROC
   TREST BUTFIRST THING :APROC
   DEFFINE :APROC :.P
END
TO TREST :ALIST
   IF EMPTYP :ALIST [STOP] MAKE ".P LPUT LIST "PRINT
FIRST :ALIST :.P
   MAKE ".P LPUT FIRST :ALIST :.P
   MAKE ".P LPUT MAKE ". READCHAR :.P
   TREST BUTFIRST :ALIST END TO UNTRACE :APROC
   IF NO DEFINEDP WORD ". :AFROC [STOP]
   COPYDEF :APROC WORD ". :APROC
   ERASE WORD ". :APROC
END
```

Может быть Вы, если захотите, расширите эти отладочные процедуры, чтобы они давали еще больше информации. Придется поэкспериментировать. Может, правда, случиться так, что из-за слишком высокоразвитых отладочных процедур у Вас возникнут проблемы с нехваткой оперативной памяти компьютера. Но, к счастью, необходимость в таких отладочных процедурах почти не возникает. В реальной практике вполне достаточно тех общих принципов отладки, на которые мы уже указали.

Стюарт Николс.

ПРИМЕНЕНИЕ АССЕМБЛЕРА ДЛЯ СОЗДАНИЯ БЫСТРОРАБОТАЮЩИХ ПРОГРАММ

Перевод с английского Пашорина В.И.
Редактирование текста "ИНФОРКОМ".

Окончание.
Начало см. с. 9-12, 48-61, 97-103, 139-144, 187-189.

10. ПЕРЕВОД БЕЙСИК-ПРОГРАММЫ В МАШИННЫЕ КОДЫ

Если Вы дошли до этой главы с ясным пониманием всего прочитанного, то примите поздравления, теперь у Вас не должно быть проблем с программированием в машинных кодах.

В этой же главе мы разберем способ создания игровой программы полностью в машинных кодах, используя в качестве примера программу на БЕЙСИКе. Мы полагаем, что, Вы уже умеете программировать на этом языке.

Весьма важно перед созданием любой программы отработать ее алгоритм в виде структурной схемы. Это облегчит выполнение перевода BASIC-программы в машинные коды. Мы займемся переводом в машинные коды игровой БЕЙСИК-программы "CROSS", что позволит приобрести навыки для перевода прочих собственных программ.

Идея игровой программы "CROSS" состоит в том, чтобы управляя перемещением человечка вернуть его домой целым и невредимым, несмотря на обилие всевозможных препятствий (Лист._1).

Главная программа включает в себя инициализацию программных переменных, вывод на экран инструкции пользователю и цикл перемещения человечка по экрану с определением его текущего положения. Существует две точки выхода из этого цикла:

1. Если человечек сталкивается с препятствием, то осуществляется переход к процедуре HIT и проводится проверка оставшегося числа жизней. Если это число равно нулю, то игра заканчивается, в противном случае осуществляется переход к основному циклу главной программы.

2. Выход к процедуре HOME. Здесь производится проверка числа занятых домов, а затем увеличивается значение скорости перемещения и добавляется еще один паук-людоед. Из процедуры HOME возврат осуществляется к основному циклу главной программы.

БЕЙСИК-программа довольно проста, хотя для задания UDG, цвета и звука, а также попиксельного перемещения экрана используются фрагменты в машинных кодах, так как на БЕЙСИКе это выполнялось бы крайне медленно. Машинные коды вводятся в программу через оператор POKE. Хорошо разберитесь с работой БЕЙСИК-программы, потому что мы переходим к построчному переводу ее в машинные коды.

Листинг_1.

Программа CROSS на БЕЙСИКе (с сокращениями).

Внимание: Символы графики пользователя [UDG] в тексте программы взяты в квадратные скобки [...]. Они набираются в графическом режиме.

```
20 CLEAR 32243: GO TO 40
25 BEEP .01,b-a
30 PRINT OVER 1; PAPER 8; INK 8; AT a,y2; "[G]": RETURN
40 PRINT AT 11,5; "Please wait"
50...140 - запись кодов процедуры ROLL через цикл FOR...NEXT с помощью READ...DATA и POKE.
150 LET a=PEEK 23675 + 256*PEEK 23676
```

```
160 FOR b=a TO a+167: READ c: POKE b,c: NEXT c
180 DATA....  Данные по конструкции
..............  символов UDG-графики
270 DATA..... (168 байтов)
280 PRINT AT 11,3; "Do you want instructions?";AT 13,11; "(Y)es"; AT 15,11;"(n)o"
290 LET a$=INKEY$: IF a$="" THEN GO TO 290
295 IF a$<>"y" THEN GO TO 400
300 CLS: PRINT AT 0,11;"OBJECT": PRINT: PRINT "To guide a [G] across a road and a river,
      avoiding [AB] [CDE] [F] [GRRS] [TU]":
310 PRINT: PRINT "A [OP] patrols the central island.":
320 PRINT: PRINT "There are 4 HOMES to be filled. i.e. gaps in fence [NNN] [NNN]"
330 PRINT: PRINT "Once all 4 HOMES are filled the speed will increase, an extra spider is
      added and the HOMES will empty"
340 PRINT AT 18,9; "Press any Key"
350 PAUSE 0
370 CLS: PRINT AT 7,11; "CONTROLS": PRINT
375 PRINT "      !               <0>"; FLASH 1
380 PRINT AT 11,6; "1"; FLASH 0;" 2 3 4 5 6 7 8 ";FLASH 1; "9"; FLASH 0; " "; FLASH 1; "0";
      FLASH 0
385 PRINT AT 18,5; "Press any Key to PLAY"
390 PAUSE 0
400 BRIGHT 1: PAPER 5: BORDER 5: CLS
410 LET hi=0
420 PRINT PAPER 4; AT 10,0;"               [OP]               "
430 LET lives=9: LET score=0: LET home=0: POKE 32425,201: POKE 32450,201: POKE 32469,201
450 PRINT AT 0,0; PAPER 4; "[NNNN]"; PAPER 7; " ";
      PAPER 4; "[NNNNNN]"; PAPER 7;" ";
      PAPER 4; "[NNNNNNN]"; PAPER 7;" ";
      PAPER 4; "[NNNNNNN]"; PAPER 7;" ";
      PAPER 4; "[NNNN]"
455 IF home<>0 THEN GO TO 660
460 PRINT PAPER 4;  INK 5;          "--------------------------------"
470 PRINT                           "[QRRRS    QRRRS   QRRSB QRRS     ]"
480 PRINT INK 7;                    "[   MMM    MMMM     MMM    MM    ]"
490 PRINT INK 2;                    "[  TU    TU   TU   TU   TU       ]"
500 PRINT INK 7;                    "[MM M      MMMMM      MMM        ]"
510 PRINT INK 1;                    "[ RS    QRRS      QRRRRS     QRRR]"
520 PRINT INK 7;                    "[   MM  MMMMMMM     MMM          ]"
530 PRINT                           "[U      TU     TU      TU       T]"
540 PRINT PAPER 4;                  "[NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN]"
550 PRINT PAPER 0; INK 7 AT 11,0;   "[NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN]"
560 PRINT PAPER 0; INK 3            "[ AB  F  AB   CDCDE      FF      ]"
570 PRINT PAPER 0; INK 7            "                                  "
580 PRINT PAPER 0; INK 5            "[  AB  AB    AB    AB     AB     ]"
590 PRINT PAPER 0; INK 7            "=================================="
600 PRINT PAPER 0; INK 4            "[B    AB   AB    AB      A       ]"
610 PRINT PAPER 0; INK 7            "                                  "
620 PRINT PAPER 0; INK 6            "[ CD   AB    AB    FF       EDCDC]"
630 PRINT PAPER 4;                  "[NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN]"
640 PRINT PAPER 4;                  "                                  "
650 PRINT PAPER 1; INK 7 " SCORE ";AT 21,11;" MEN "; PAPER 5; INK 0; " LIVES "; PAPER 1; INK
      7; " HI SCORE "
660 LET X1=20: LET y1 = 16: LET x2=x1: LET y2=y1
670 PRINT PAPER 8; INK 8; AT x1,y1; " "
680 RANDOMIZE USR 32295
690 IF SCREEN$ (x2,y2)=" " THEN GO TO 880
700 LET a=x2: FOR b=25 TO 35: GO SUB 25: GO SUB 25: NEXT b
730 FOR a=x2 TO 20 STEP 2: GO SUB 25: GO SUB 25: NEXT a
740 LET lives=lives-1: PRINT AT 21,16; lives
750 LET x2 = 20
760 IF lives <> 0 THEN GO TO 680
770 IF hi >score THEN GO TO 790
780 LET hi = score: PRINT AT 21,27; hi
790 PRINT FLASH 1; PAPER 7; AT 12,0; "          GAME   OVER       "
800 PRINT AT 14,0;    "Another game?    (y)es     (n)o "
```

```
840 IF INKEY$ = "n" THEN RANDOMIZE USR 0
850 IF INKEY$ <> "y" THEN GO TO 840
860 PRINT PAPER 5; AT 21,7;"     ": GO TO 480
880 IF x2<>0 THEN GO TO 1050
890 PRINT PAPER 8; INK 8; AT x1,y1;" ": PRINT AT x2,y2;"[G]"
900 RESTORE 920: FOR a=1 TO 8: READ b,c
910 BEEP b,c: NEXT a
920 DATA .1,11,.1,11,.8,16,.05,11,.05,16,.05,11,.05,16,1,20
930 LET home=home+1: LET score=score+50: PRINT AT 21,7; score
950 IF home/4<>INT(home/4) THEN GO TO 660
960 IF home=4 THEN POKE 32425,0
970 IF home=8 THEN POKE 32450,0
980 IF home=12 THEN POKE 32469,0
985 IF home>36 THEN GO TO 450
990 LET a=RND*30
1000 LET a=a+1
1005 IF a>31 THEN LET a=0
1010 IF SCREEN$(10,a)="" THEN GO TO 1000
1020 IF SCREEN$(10,a+1) = "" THEN GO TO 1000
1030 PRINT PAPER 4; AT 10,a; "[OP]"
1035 RESTORE 920: FOR a=1 TO 8: READ b,c: BEEP b,c: NEXT a
1040 GO TO 450
1050 PRINT PAPER 8; INK 8; AT x2,y2; "[G]"
1060 LET x1=x2: LET y1=y2
1070 IF INKEY$ <> "1" THEN GO TO 1100
1080 BEEP .001,33
1090 LET x2=x2-2: LET score=score+5: PRINT AT 21,7; score
1100 LET y2=y2+(INKEY$="0" AND y2<>31)-(INKEY$="9" AND y2<>0)
1110 GO TO 670
```

СТРОКА 20. Эта строка не нужна в окончательной версии, так как она устанавливает верхнюю границу RAMTOP для БЕЙСИК-программы.

СТРОКИ 25 и 30. Это две строки подпрограммы, вызываемой в строках 700 и 730. Размещение ее в начальных строках необходимо для сокращения времени поиска при вызове этой подпрограммы, так как всегда поиск нужных подпрограмм начинается с начальных строк БЕЙСИКа. Машинные коды, соответствующие этим двум строкам, размещаются в участке памяти, где будет записана процедура HIT.

СТРОКА 40. Для того, чтобы ввести фрагменты программы в кодах с помощью POKE требуется время. Поэтому в БЕЙСИК-программе и введена эта строка с сообщением о необходимости немного подождать. В программе, написанной полностью в кодах, давать такое сообщение, конечно, нет необходимости.

СТРОКИ 50...140. Это строки для ввода через POKE процедуры "ROLL", обеспечивающей попиксельное смещение изображения на экране. Понятно, что для программы в машинных кодах эту операцию выполнять также не нужно, так как эти коды будут записаны в соответствующие адреса при загрузке программы в память (см. листинг процедуры ROLL в конце статьи).

СТРОКИ 150...270. Это первые строки, которые нуждаются в преобразовании для перехода к программе в машинных кодах. Здесь осуществляется запись в область UDG всех графических элементов пользователя, так как этот участок памяти занимает 158 байтов, то и в строках DATA записаны 168 чисел, определяющих форму UDG.

В дальнейшем, конечно, для задания UDG мы будем непосредственно записывать в участок памяти UDG значение этих 168 байтов. Сначала мы разместим данные, характеризующих форму UDG, начиная с адреса 24001, а затем перешлем их в участок памяти UDG, использовав команду LDIR. При этом в регистровую пару DE записывается начальный адрес участка UDG из системной переменной 5C7B (83675), а в регистровую пару HL начальный адрес данных, характеризующих форму UDG. Процедура пересылки, написанная в машинных кодах, размещается сразу же за данными 168 байтами. Вызов этой процедуры поэтому производится по команде RANDOMIZE USR 24169.

UDG-графика.

```
5DC1H
DEFB 0F 12 22 7F FF FF 28 10
DEFB 80 40 20 FE FE FF 28 10
DEFB 7F 7F 7F 7F 7F FF 15 08
DEFB FF FE FE FE FF FF 40 80
DEFB 00 F8 C4 C4 FE FE 28 10
DEFB 18 18 24 7E 3C 5A A5 42
DEFB 38 28 92 72 38 38 28 6C
DEFB 01 02 04 7F 7F FF 14 08
DEFB F0 48 44 FE FF FF 14 08
DEFB 00 1F 23 23 7F 7F 14 08
DEFB 7F 7F 7F 7F FF FF 02 01
DEFB FE FE FE FE FE FF A8 10
DEFB 10 29 C7 00 26 00 00 00
DEFB 00 44 FF 44 44 FF 44 00
DEFB 00 22 55 8F 97 A3 A0 00
DEFB 00 44 AA F1 E9 C5 05 00
DEFB 10 10 10 FE 3F IF 0F 07
DEFB 00 00 00 00 1E FF FF FF
DEFB 60 7C 54 78 7F FF FE 7C
DEFB 00 00 03 02 0F 3F FF 00
DEFB 05 0C 96 F0 E0 55 FF 00

5E69
24169   LD DE,(5C7B)
24173   LD HL,5DC1
24175   LD BC,00AB
24179   LDIR
24181   JR,5EA2
```

СТРОКА 280. Сообщение, выводимое на экран по команде PRINT в этой строке, можно выводить на экран, используя процедуру ОЗУ, находящуюся по адресу 3252 (203CH). Машинные коды этой процедуры записаны в участке 5EA2... 5EAF, а сам текст сообщения - в участке 5E77...5EA1 (Листинг-4).

Печать сообщения.

```
5E77H
DEFB 16 0B 03 44 5F 20 79 6F
DEFB 75 20 77 61 6E 74 20 69
DEFB 6E 73 74 72 75 63 74 69
DEFB 6F 6E 73 3F 16 0D 0B 28
DEFB 79 29 65 73 16 0F 0B 28
DEFB 6E 29 6F

5EA2
24225   LD A,02
24228   CALL 1601H
24231   LD DE,5E77H
24234   LD BC,002BH
24237   CALL 203CH
```

СТРОКИ 290 и 295. Эти строки необходимы, чтобы приостановить выполнение программы до тех пор, пока не будет нажата клавиша. Если будет нажата клавиша "y", то на экране появится инструкция по правилам игры. Если же будет нажата любая другая клавиша, то произойдет переход к адресу 608AH. Машинные коды, соответствующие этим строкам, записаны по адресу с 5EB0H по 5EBEH (Лист.5)

Проверка нажатия клавиши "y".

```
24240   HALT
24241   BIT 5,(IY+01)
24245   JR Z,-07
```

```
24247   LD A,(5C08H)
24250   CP 79
24252   JP NZ,608AH
```

СТРОКА 300:1 CLS - Перевод этой команды в машинные коды мы уже разбирали (Листинг-6).

Очистка экрана.

```
5EBFH
24255   LD A,02
24257   CALL 1601H
24260   CALL 0D6BH
24263   JP 5F73H
```

СТРОКИ 300:2...350. Для вывода сообщений на экран снова будем использовать процедуру вывода на экран последовательности символов ПЗУ, коды которых записаны в участок памяти с именем DATA. Коды символов первого сообщения находятся в памяти с 5ECAH по 5F72H, а второго сообщения 5F84H...5FF9H. Процедуры же вывода PRINT DATA находятся соответственно 5F73H...5F83H и 5FFAH... 6002H (Листинг_7).

В ячейках с 6003H по 600CH находятся машинные коды, реализующие команду PAUSE 0. Не забывайте, что бит 5 переменной 23611 (IY+1) должен быть сброшен (ожидание нажатия клавиши).

Печать сообщений, пауза.

```
5ECAH
DEFB 16 00 0B 4F 42 4A 45 43
DEFB 54 0D 0D 54 6F 20 67 75
DEFB 69 64 65 20 61 20 96 20
DEFB 61 63 72 6F 73 73 20 61
DEFB 20 72 6F 61 64 20 61 6E
DEFB 64 20 61 72 69 76 65 72
DEFB 2C 61 76 6F 69 64 69 6E
DEFB 67 20 90 91 20 92 93 94
DEFB 20 95 20 A0 A1 A1 A2 20
DEFB A3 A4 0D 0D 41 20 9E 9F
DEFB 20 70 61 74 72 6F 6C 73
DEFB 20 74 68 65 20 63 65 6E
DEFB 74 72 61 6C 20 69 73 6C
DEFB 61 6E 64 EE 0D 0D 54 68
DEFB 65 72 65 20 61 72 65 20
DEFB 34 20 48 4F 4D 45 53 20
DEFB 74 6F 20 62 65 20 66 69
DEFB 6C 6C 65 64 2E 20 69 2E
DEFB 65 2E 20 67 61 70 73 20
DEFB 69 6E 20 65 65 6E 63 65
DEFB 20 9D 9D 9D 20 9D 9D 9D
DEFB 0D

5F73H
24435   LD A,02
24437   CALL 1601H
24440   LD DE,5ECAH
24443   LD BC,00A9H
24446   CALL 203CH
24449   JP 5FFAH

5FB4H
DEFB 0D 4F 6E 63 65 20 61 6C
DEFB 6C 20 34 20 48 4F 4D 45
DEFB 53 20 61 72 65 20 66 69
DEFB 6C 6C 65 64 20 74 68 65
DEFB 20 73 70 65 65 64 20 77
```

```
DEFB 59 6C 6C 20 69 6E 63 72
DEFB 65 61 73 65 2C 20 61 6E
DEFB 20 65 78 74 72 61 20 73
DEFB 70 69 64 65 72 20 59 73
DEFB 20 61 64 64 65 64 20 61
DEFB 6E 64 20 74 68 65 20 48
DEFB 4F 4D 45 53 20 77 69 6C
DEFB 6C 65 6D 70 74 79 16 12
DEFB 09 50 72 65 73 73 20 61
DEFB 6E 79 20 6B 65 79

5FFAH
24570   LD DE,5F84H
24573   LD BC,0076H
24576   CALL 203CH
24579   RES 5,(IY+01)
24583   HALT
24584   BIT 5,(IY+01)
24568   JR Z,-07
```

СТРОКА 370: 1 CLS. В участке памяти с 600E по 6012 записаны машинные коды процедуры очистки экрана. Здесь применяется другой способ, основанный на применении процедуры ПЗУ 0E44H, которая очищает столько строк экрана (считая снизу), сколько установлено в регистре B(Листинг_8).

Очистка экрана.

```
600EH
24590   LD B,18H
2459E   CALL 0E44H
24595   JP 6076H
```

СТРОКИ 370:2...390. Коды символов, выводимых на экран, записаны в участке 6016H... 6075H, а сама же процедура печати находится в адресах - 6076H... 607EH. Здесь опять же имеются машинные коды, реализующие команду PAUSE 0 (Листинг_9).

Печать сообщений, пауза.

```
6015H
DEFB 16 07 0B 43 4F 4E 54 52
DEFB 4F 4C 53 0D 0D 20 20 20
DEFB 20 20 20 5E 20 20 20 20
DEFB 20 20 20 20 20 20 20 20
DEFB 20 20 20 3C 30 3E 12 01
DEFB 16 0B 06 31 12 00 20 32
DEFB 20 33 20 34 20 35 20 36
DEFB 20 37 20 38 30 12 01 39
DEFB 12 00 20 12 01 30 12 00
DEFB 16 12 05 50 72 65 73 73
DEFB 20 61 6E 79 20 6B 65 79
DEFB 20 74 6F 20 50 4C 41 59

6076H
24694   LD DE,6016H
24897   LD BC,0060H
24700   CALL 203CH
24703   RES 5,(IY+01)
24707   HALT
24708   BIT 5,(IY+01)
24712   JR Z,-07
```

Строка 400. В первой главе обсуждался порядок перевода всех операторов этой строки в машинные коды. Вызов процедуры CLS здесь необходим для установки на экране соответствующих атрибутов, как и в БЕЙСИК-программе. Машинные коды записаны в

участке 608AH...609BH.

Установка атрибутов

```
608AH
24714   LD A,05
24716   CALL 229BH
24719   LD A,68H
24721   LD (5C8D),A
24724   LD A,02
24725   CALL 1601H
```

СТРОКА 410. В этой строке переменной hi, являющейся показателем достигнутого уровня, присваивается нулевое начальное значение. Полагая, что значение этой переменной не может быть больше 65535, выделим в буфере принтера 2 байта для хранения значения этой переменной. Соответствующие этому оператору машинные коды записаны в участке памяти 609CH ...60A1H.

Инициализация переменной hi.

```
609CH
24732   LD HL,0000
24735   LD (5B00H),HL
24738   JR +25H
```

СТРОКА 420. Коды символов сообщения, выводимого на экран, записаны в адресах 60A4H...60C8H. Процедура печати - 60C9H...60D6H.

Печать строки.

```
60A4H
DEFB 11 04 16 0A 00 20 20 20
DEFB 20 20 20 20 20 20 20 20
DEFB 20 20 20 20 9E 9F 20 20
DEFB 20 20 20 20 20 20 20 20
DEFB 20 20 20 20 20

60C9H
24777   LD A,02
24779   CALL 1601H
24782   LD DE,60A4H
24785   LD BC,0025H
24788   CALL 203CH
```

СТРОКА 430. В этой строке определяются исходные значения количества "жизней", занятых домов и достигнутого уровня. Эти значения также хранятся в буфере принтера по следующим адресам:

количество "жизней" - 5B02
достигнутый уровень - 5B03/4
количество домов - 5B05

Машинные коды, соответствующие этой строке, см. 60D7H...60E5H.

Инициализация переменных.

```
60D7H
24791   LD A,09
24793   LD(5B02H),A
24796   LD HL,0000
24799   LD (5B03H),HL
24802   XOR A
24803   LD (5B05),A
```

СТРОКА 440. Операторы этой строки обнуляют скорость перемещения и устанавливают команды RET в соответствующих ячейках подпрограммы LINE ROLL.

Машинные коды см. 60E6H…60F3H.

Коррекция процедуры ROLL.

```
60E6H
24806  LD A,C9H
24808  LD (7EA9H),A
24811  LD (7EC2H),A
24814  LD (7ED5H),A
24817  JP 6129H
```

СТРОКА 450. Коды символов этой строки и процедура печати:

Печать строки.

```
60F4H
DEFB 16 00 00 11 04 9D 9D 9D
DEFB 9D 11 07 20 11 04 9D 9D
DEFB 9D 9D 9D 9D 11 07 20 11
DEFB 04 9D 9D 9D 9D 9D 9D 9D
DEFB 11 07 20 11 04 9D 9D 9D
DEFB 9D 9D 9D 9D 11 07 20 11
DEFB 04 9D 9D 9D 9D

6129H
24873   LD A,
24875   CALL 1601H
24878   LD DE,60F4H
24881   LD BC,0035
24884   CALL 203CH
```

СТРОКА 455. В этой строке проверяется количество домов и, если оно равно нулю, то осуществляется перевод к строке 660. Для реализации этой строки в машинных кодах необходимо извлечь содержимое ячейки 5B05H, где записано количество домов, записать это значение в регистр A и использовать для проверки на нулевое значение команду AND A. Если флаг нуля (ZERO) после этой операции не будет установлен, то осуществляется переход к процедуре PRINT DISPLAY. Если же этот флаг не включен, то эта процедура пропускается и начинается игра. Машинные коды, соответствующие этой строке, записаны в ячейках 6137H…6140H.

Проверка переменной home.

```
6137H
24687   LD A,(5B05H)
24890   AND A
24891   JP NZ,641EH
24894   JP 63D2H
```

СТРОКИ 460…640. Машинные коды, соответствующие этим строкам, занимают большой объем памяти. Коды всех символов сообщений этих строк записаны в участке памяти 6141H… 63D1H, а процедура печати - 63D2H…63DFH.

Инициализация экрана.

```
6141H
DEFB 11 04 10 05 8C 8C 8C 8C   DEFB 9C 9C 20 20 20 20 20 9C   DEFB 9C 9C 20 20 20 20 10 01
DEFB 8C 8C 8C 8C 8C 8C 8C 8C   DEFB 9C 9C 20 20 20 20 20 20   DEFB A1 A2 20 20 20 20 20 00
DEFB 8C 8C 8C 8C 8C 8C 8C 8C   DEFB 9C 10 02 20 20 A3 A4 20   DEFB A1 A1 A2 20 20 20 20 20
DEFB 8C 8C 8C 8C 11 05 10 00   DEFB 20 20 20 20 A3 A4 20 20   DEFB 20 A0 A1 A1 A1 A1 AE 20
DEFB 20 A0 A1 A1 A2 20 20 20   DEFB 20 20 20 A3 A4 20 20 20   DEFB 20 20 20 20 A0 A1 A1 A1
DEFB A0 A1 A1 A1 A2 20 20 20   DEFB 20 A3 A4 20 20 20 20 20   DEFB 10 07 20 20 20 9C 9C 20
DEFB 20 A0 A1 A1 A3 20 20 20   DEFB 20 A3 A4 20 10 07 9C 9C   DEFB 20 9C 9C 9C 9C 9C 20 20
DEFB A0 A1 A1 A2 20 20 20 20   DEFB 20 20 20 9C 20 20 20 20   DEFB 20 9C 9C 20 E0 20 20 20
DEFB 10 07 20 20 20 9C 9C 9C   DEFB 20 20 9C 9C 9C 9C 9C 20   DEFB 20 9C 9C 9C 20 20 20 80
DEFB 20 20 20 20 20 9C 9C 9C   DEFB 20 20 20 20 20 20 20 9C   DEFB 20 20 10 00 A4 20 20 20
```

```
DEFB 20 20 20 20 20 20 A3 A4    DEFB 07 2D 2D 2D 2D 2D 3D 2D    DEFB 07 2D 2D 2D 2D 2D 2D 2D
DEFB 20 20 20 20 20 A3 A4 20    DEFB 2D 2D 2D 2D 2D 2D 2D 2D    DEFB 2D 2D 2D 2D 2D 2D 2D 2D
DEFB 20 20 20 20 20 A3 A4 20    DEFB 2D 2D 2D 2D 2D 2D 2D 2D    DEFB 2D 2D 2D 2D 2D ED 2D 2D
DEFB 20 20 20 A3 11 04 9D 9D    DEFB 2D 2D 2D 2D 2D 2D 2D 2D    DEFB 2D 2D 2D 2D 2D 2D 2D 2D
DEFB 9D 9D 9D 9D 9D 9D 9D 9D    DEFB 2D 10 05 20 90 92 20 20    DEFB 2D 10 06 9A 9B 20 20 20
DEFB 9D 9D 9D 9D 9D 9D 9D 9D    DEFB 20 20 90 91 20 20 20 E0    DEFB 97 98 20 20 97 98 20 95
DEFB 9D 9D 9D 9D 9D 9D 9D 9D    DEFB 90 91 20 20 90 91 20 20    DEFB 20 20 95 20 20 99 9A 9B
DEFB 9D 9D 9D 9D 9D 9D 11 00    DEFB 20 20 20 20 20 20 90 91    DEFB 9A 9B 20 20 20 95 20 20
DEFB 10 07 16 0B 00 9D 9D 9D    DEFB 20 20 20 10 07 3D 3D 3D    DEFB 20 20 99 11 04 10 00 9D
DEFB 9D 9D 9D 9D 9D 9D 9D 9D    DEFB 3D 3D 3D 3D 3D 3D 3D 3D    DEFB 9D 9D 9D 9D 9D 9D 9D 9D
DEFB 9D 9D 9D 9D 9D 9D 9D 9D    DEFB 3D 3D 3D 3D 3D 3D 3D 3D    DEFB 9D 9D 9D 9D 9D 9D 9D 9D
DEFB 9D 9D 9D 9D 9D 9D 9D 9D    DEFB 3D 3D 3D 3D 3D 3D 3D 3D    DEFB 9D 9D 9D 9D 9D 9D 9D 9D
DEFB 9D 9D 9D 9D 9D 10 03 80    DEFB 3D 3D 3D 3D 3D 10 04 98    DEFB 9D 9D 9D 9D 9D 9D 9D 11
DEFB 90 91 20 95 20 20 90 91    DEFB 20 20 20 20 20 97 98 20    DEFB 04 20 20 20 20 20 20 20
DEFB 20 20 92 93 92 93 94 20    DEFB 20 20 97 98 20 20 20 20    DEFB 20 20 20 20 20 20 20 20
DEFB 95 20 95 20 20 20 92 93    DEFB 20 20 20 97 98 20 20 20    DEFB 20 20 20 20 20 20 20 20
DEFB 94 20 20 20 20 20 20 10    DEFB 20 20 20 20 20 20 97 10    DEFB 20 20 20 20 20 20 20 20

63D2H
25554   LD A,02
25556   CALL 1601H
25559   LD DE,6141H
25562   LD BC,0291H
25565   CALL 203CH
```

СТРОКА 650. Для реализации этой строки в машинных кодах необходимо предварительно преобразовать число, записанное в ячейке 5B02H. Для этого оно извлекается из этой ячейки, суммируется с числом 48 и результат записывается в ячейку 6402. Машинные коды этой операции 63E0H...63E9H. Коды символов сообщения для вывода на экран записаны в 63EAH...640FH, а процедура печати - 6410H...641EH.

Инициализация строки результата

```
63E0H
25568   LD A,(5B02H)
25571   ADD A,30
25573   LD (6401H),A
25576   JR +26

63EAH
DEFB 11 01 10 07 20 53 43 4F
DEFB 52 45 20 16 15 0B 20 4D
DEFB 45 4E 20 11 05 10 00 39
DEFB 11 01 10 07 20 48 49 2D
DEFB 53 43 4F 52 45 20

6410H
25616   LD A,02
25618   CALL 1601H
25621   LD DE,63E1H
25624   LD BC,0026H
25627   CALL 203CH
```

СТРОКА 660. В этой строке определяются исходные координаты положения человечка на экране x1 и y1, а также координаты его нового положения x2 и y2. Для программы в машинных кодах эти координаты хранятся также в буфере принтера:

x1 - 5B06H x2 - 5B08H

y1 - 5B07H y2 - 5B09H

Для записи в буфер принтера координат лучше использовать шестнадцатиричную систему счисления, тогда процедура записи будет иметь следующий вид:

Инициализация координат героя.

```
641FH
25630   LD HL,1014H
25633   LD (5B06H),HL
25636   LD (5B08H),HL
```

Из этого примера понятно, что использовать десятичную систему счисления для записи координат в регистровую пару HL не удобно, т.к. трудно будет представить положение объекта на экране.

СТРОКА 670. Поскольку здесь необходимо выводить на экран только один символ, то лучше использовать команду RST 10H.

Печать героя.

```
6427H
25639   LD A,02
25641   CALL 1601H
25644   LD A,11H
25646   RST 10H
25647   LD A,08
25649   RST 10H
25650   LD A,10H
25652   RST 10H
25653   LD A,08
25655   RST 10H
25656   LD A,16H
25658   RST 10H
25659   LD A,(5B06H)
25662   RST 10H
25663   LD A,(5B07H)
25666   RST 10H
25667   LD A,20H
25669   RST 10H
```

СТРОКА 680. В этой строке производится вызов процедуры ROLL, записанной в машинных кодах с адреса 7E27H. Код, соответствующий этой строке:

Вызов процедуры ROLL.

```
6446H
25670  CALL 7E27H
```

СТРОКА 690. в этой строке проверяется состояние символьной ячейки с координатами x2, y2. Для реализации этой строки в машинных кодах будем использовать процедуру SCREEN$, описанную в главе 8, передающую значение параметров исследуемой ячейки на стек калькулятора и затем в регистры A,B,C,D,E. Для определения состояния ячейки используем:

```
LD A,(DE)
CP 20H
```

Помните, что для продолжения программы, после использования калькуляторного стека, необходимо очистить рабочий участок. Если в исследуемой ячейке не пробел, то осуществляется переход к процедуре HIT, которая начинается с адреса 6548H. В БЕЙСИК-программе соответственно переход к строке 880. Машинные коды, соответствующие этой строке записаны в участке 6449H…645DH.

Проверка местоположения героя.

```
6449H
25673   LD BC,(5B08H)
25677   CALL 2538H
25680   CALL 2BF1H
25683   LD A,(DE)
25684   PUSH AF
25685   CALL 16BFH
```

```
25688   POP AF
25689   CP 20H
25691   JP NZ,6548H
```

СТРОКА 880. В этой строке проверяется: достиг ли человечек вершины экрана, т.е. дошел ли он до дома. В машинных кодах для этого записываем значение координаты x2 в регистр A:

```
LD A,(5B08)
```

и сравниваем это значение с нулем. Если это значение равно нулю, то выполняется переход к процедуре HOME, которая начинается с адреса 66A6H. В Бейсик-программе соответственно осуществляется переход к строке 1050. Машинные коды, соответствующие этой строке 645EH...6465H.

Проверка достижения "дома"

```
645EH
25694   LD A,(5B08H)
25697   CP 00
25699   JP Z,66A6H
```

СТРОКА 1050. Здесь также для вывода на экран одного символа в позицию с координатами x2, y2 лучше использовать команду RST 10H, как и в строке 670. Машинные коды - 6466H... 6484H.

Печать героя в новой координате

```
6466H
25702   LD A,02
25704   CALL 1601H
25707   LD A,11
25709   RST 10H
25710   LD A,08
25712   RST 10H
25713   LD A,10H
25715   RST 10H
25716   LD A,08
25718   RST 10H
25719   LD A,16H
25721   RST 10H
25722   LD A,(5B08H)
25725   RST 10H
25726   LD A,(5B09H)
25729   RST 10H
25730   LD A,96H
25733   RST 10H
```

СТРОКА 1060. Для реализации этой строки в машинных кодах достаточно переписать значение ячеек 5B08 и 5B09 в ячейки 5B06 и 5B07 соответственно.

Запоминание новой координаты.

```
6485H
25733   LD HL,(5B08H)
25736   LD (5B06H),HL
```

СТРОКА 1070. Чтобы улучшить работу програлмы в машинных кодах для проверки состояния клавиатуры и контроля нажатия клавиши лучше использовать не системную переменную LAST KEY, а непосредственно команду IN A,(C). Причем, чтобы проверить нажата ли клавиша "1", достаточно проверить состояние только клавиш "1"..."5" так, как это описано в главе 5. Машинные коды - 648BH...6493H.

Проверка нажатия клавиши "1".

```
648BH
```

```
25739   LD BC,F77EH
25742   IN A,(C)
25744   BIT 0,A
25746   JR NZ,+1C
```

СТРОКА 1080. Поскольку длительность и тональность звукового сигнала принципиального значения не имеют, то для реализации этой строки в машинных кодах достаточно только вызвать процедуру BEEP из ПЗУ - CALL 03B5H.

Генерация звукового сигнала.

```
6494H
25748   LD HL,0032H
25751   LD DE,0005H
25754   CALL 03B5H
```

СТРОКА 1090. Для записи этой строки в машинных кодах необходимо извлечь содержимое ячейки 5B08 и вычесть из этого значения число 2, а результат вновь записать в эту же ячейку. Значение текущего счета, записанное в ячейках 5B03/4, увеличивается на 5 и вновь записывается в эти ячейки. Машинные коды этих операций - 649DH...64AFH. Для вывода на экран текущего значения счета используются процедуры STACK BC и PRINT VALUE ON STACK, но первоначально определяются параметры PRINT AT. Машинные коды - 64B1H...64C8H.

Коррекция счета (score).

```
649D2H
25757   LD A,(5B08H)
25760   DEC A
25761   DEC A
25763   LD (5B08H),A
25765   LD HL,(5B03H)
25768   INC HL
25769   INC HL
25770   INC HL
25771   INC HL
25772   INC HL
25773   LD (5B03H),HL
25776   NOP
25777   LD A,02
25779   CALL 1601H
25782   LD A,16H
25785   LD A,15H
25787   RST 10H
25788   LD A,07
25790   RST 10H
25791   LD BC,(5B03H)
25795   CALL 2D2BH
25798   CALL 2DE3H
```

СТРОКА 1100. Опять же, для чтения состояния клавиатуры и установки значения y2 используем команду IN A,(C). В программе в машинных кодах добавлена процедура, которой нет в БЕЙСИК программе. В участке памяти 64F3H...64F4H записаны машинные коды, которые составляют процедуру проверки нажатия клавиши "6". Если эта клавиша нажата, то осуществляется выход из программы. На этом главный цикл программы заканчивается.

Если теперь Вы запустите эту программу, то перемещение человечка по экрану будет настолько быстрым, что Вы не будете его замечать. Поэтому необходимо в главный цикл программы в машинных кодах включить замедляющую процедуру.

Этот же эффект можно получить, если в главный цикл включить дополнительно процедуру музыкального сопровождения перемещения. Машинные коды этой процедуры 64F5H...650BH. Завершающей командой главного цикла является команда JP 6427,

обеспечивающая переход на начало главного цикла. В БЕЙСИК-программе это выполняется переходом к строке 670.

Опрос клавиатуры, музыка.

```
64C9H
25801   LD BC,EFFEH
25804   IN A,(C)
25806   BIT 0,A
25808   JR NZ,+0EH
25810   LD A,(5B09H)
25813   CP 1F
25815   JR Z,+01
25817   INC A
25818   LD (5B09H),A
25831   JP 64F5H
25834   BIT 1,A
25836   JR NZ,02
25838   LD A,(5B09H)
25831   CP 00
25833   JR Z,+01
25835   DEC A
25836   LD (5B09H),A
25839   JP 64F5H

64F3H
25843   BIT 4,A
25844   RET Z

64F5H
25845   LD B,0A
25847   LD DE,0078H
25850   LD HL,0001H
25853   PUSH HL
25854   PUSH DE
25855   PUSH BC
25856   CALL 03B5
25859   POP BC
25860   POP DE
25861   POP HL
25883   INC A
25864   LD L,A
25865   NOP
25866   DJNZ -0F
25868   JP 6427
```

Процедура HIT

СТРОКА 700. В этой строке вводится новая переменная a = x2. Для того, чтобы сохранить значение этой переменной, будем использовать свободную ячейку 5CB0H из области системных переменных. Переменная b цикла FOR/NEXT записывается в регистр B:

```
LD B,19H (25D)
```

и в ячейку 5CB1H для использования в следующей процедуре. Машинные коды, соответствующие этой строке - 6548H...6561H. Вызов процедуры, соответствующей БЕЙСИК-оператору GO SUB 35, выполняется командой CALL 650FH.

Непосредственно связаны с этой строкой строки 25 и 30 и мы рассмотрим их здесь совместно в едином блоке.

СТРОКА 25. Для перевода этой строки в машинные коды необходимо использовать калькуляторный стек, куда записывается число 0.1 и результат операции b-a. Вычитание проще выполнить, используя последовательность команд:

```
LD  HL,(5CB0H)
LD  A,H
SUB L
```

Затем, используя процедуру 2D28H переписываем значение со стека в регистр A и

вызываем процедуру BEEP - CALL 03F8H. Машинные коды, соответствующие этой строке, 650F...6521.

СТРОКА 30. Для вывода символа на экран соответственно в позицию a,72 используем команды LD A,n и RST 10H. Машинные коды - 6522H...6547H.

Расчет b-a, генерация звука.

```
650FH
25871   RST 28H ;Калькулятор.
DEFB  34 EA 23 D7 0A 3D 38
25679   LD HL,(5CB0H)
25882   LD A,H
25883   SUB L
25884   CALL 2D2BH
25887   CALL 03F8H
```

Печать "героя"

```
6522H
25890   LD A,02
25892   CALL 1601H
25895   LD A,15H
25897   RST 10H
25898   LD A,01
25900   RST 10H
25901   LD A,11H
25903   RST 10H
25904   LD A,08
25906   RST 10H
25907   LD A,10H
25909   RST 10H
25910   LD A,68
25912   RST 10H
25913   LD A,16H
25915   RST 10H
25916   LD A,(5CB0H)
25919   RST 10H
25920   LD A,(5B09H)
25923   RST 10H
25924   LD A,96H
25925   RST 10H
25927   RET
```

Создание переменной "a".

```
6548H
25928   LD A,(5B08H)
25931   LD (5CB0H),A
25934   LD B,19H
25936   LD A,B
25937   PUSH BC
25938   LD (5CB1),A
25941   CALL 650FH
25944   CALL 650FH
25947   POP BC
25048   INC B
25949   LD A,B
25950   CP 23
25952   JR NZ,-11
```

СТРОКА 730. Поскольку переменная цикла этой строки у нас хранится в ячейке 5CB0H, то организовать цикл в машинных кодах не представляется особенно трудным. Для этого необходимо извлекать это значение из этой ячейки, увеличивать его на 2 при каждом проходе и сравнивать с числом 14H (20D). Машинные коды - 6562H... ...6573H

Цикл по "a".

```
6562H
25954   CALL 650FH
25957   CALL 650FH
25960   LD A,(5CB0H)
25963   INC A
25964   INC A
25965   LD (5CB0H),A
25968   CP 14H
25970   JR NZ,-12H
```

СТРОКА 740. В этой строке производится подсчет числа "жизней". Для реализации этой строки в машинных кодах необходимо извлекать значение из ячейки 5B02H, уменьшать его на 1 и результат опять записывать в эту ячейку. Для вывода на экран этого значения необходимо первоначально установить необходимые параметры PRINT AT, а затем к числу, записанному в ячейке 5B02H добавить 30H(48D) для перевода его в код ASCII. Машинные коды 6574H...658EH.

Коррекция числа "жизней".

```
6574H
25972   LD A,(5B02H)
25975   DEC A
25976   LD (5B02H),A
25979   LD A,02
25981   CALL 1601H
25984   LD A,16H
25986   RST 10H
25987   LD A,16H
25989   RST 10H
25990   LD A,10H
25992   RST 10H
25993   LD A,(5B02H)
25996   ADD 30H
25998   RST 10H
```

СТРОКА 750. В этой строке переписывается значение переменной x2 так, чтобы x2 = 14H (20D). Машинные коды 658FH...6593H.

Коррекция переменной x2.

```
658FH
25999   LD A,14H
26001   LD (5B02H),A
```

СТРОКА 760. В этой строке выполняется условный переход. Если число "жизней" не равно 0, то выполняется возврат к началу главного цикла - строке 680. Для программы в машинных кодах это переход по адресу 6446. Машинные коды, соответствующие этой строке - 6594H...659AH.

Проверка на конец "жизней".

```
6594H
26004   LD A,(5B02H)
25007   AND A
26008   JP NZ,5446H
```

Процедура END GAME

СТРОКА 770. В этой строке сравнивается достигнутый счет в этой игре с лучшим результатом прошлых игр. Если новый результат выше, то осуществляется переход к строке 790. Эту операцию легко выполнить для программы в машинных кодах, используя команду

```
SBC HL,DE
```

и проверяя значение флага переноса (CARRY). Если флаг будет включен после

команды SBC, то выполняется переход по адресу 6611H. Машинные коды для этой строки - 659BH...65A6H.

Проверка на новый рекорд.

```
659BH
26011   XOR A
26012   LD HL,(5B03H)
26015   LD DE,(5D00H)
26019   SBC HL,DE
25021   JR C,+6AH
```

СТРОКА 780. В этой строке устанавливается новое значение лучшего результата и это значение выводится на экран. В машинных кодах для выполнения этой операции достаточно переписать значения из ячеек 5B03/4 в ячейки 5B00/1. Для вывода на экран используем процедуры STACK VALUE IN BC и PRINT TOP VALUE ON CALCULATOR STACK, предварительно установив параметры PRINT AT. Машинные коды 65A7H...66C2H.

Коррекция рекорда.

```
65A7H
26023   LD HL,(5B03H)
25026   LD (5B00H),HL
26029   PUSH HL
26030   POP BC
26031   CALL 2D2BH
26034   LD A,02
26036   CALL 1601H
25039   LD A,16H
26041   RST 10H
26042   LD A,15H
26044   RST 10H
26045   LD A,1BH
25047   RST 10H
26046   CALL 2DE3H
26051   JR +4C
```

СТРОКИ 790 и 800. В этих строках печатаются сообщения. Коды символов находятся в 65C5H...6610H, а сама процедура печати - 6611H...6619H.

Печать сообщений.

```
65C5H
DEFB 12 01 11 07 16 0C 00 20
DEFB 20 20 20 20 20 20 20 20
DEFB 20 20 47 41 4D 45 20 20
DEFB 4F 56 45 52 20 20 20 20
DEFB 20 20 20 20 20 20 20 16
DEFB 02 00 12 00 20 20 41 6E
DEFB 6F 74 68 65 72 20 67 61
DEFB 6D 65 20 3F 20 28 79 29
DEFB 65 73 20 20 28 6E 29 6F
DEFB 20 20 20 20

6611H
26129   LD DE,65C5H
26132   LD BC,004CH
26135   CALL 203CH
```

СТРОКИ 840...850. В машинных кодах, соответствующих этим строкам, для проверки нажатой клавиши используется системная переменная LAST KEY, а после проверки осуществляется соответствующий переход. Если ни клавиша "y", ни клавиша "n" не будут нажаты, то осуществляется возврат к циклу ожидания нажатия клавиш.

Проверка нажатой клавиши.

```
661AH
26138   HALT
26139   BIT 5,(IY+01)
26143   JR Z,-07
36145   RES 5,(IY+01)
26149   LD A,(5C0BH)
26152   CP 6EH
26154   JR NZ,+03
26156   CALL 0000
26159   CP 79H
26161   JR NZ,-19
```

СТРОКА 860. Перед тем, как выполнится переход к началу главного цикла по команде JP 60C9H, переписывается значение лучшего результата (переменная hi) с помощью команд LD A,n : RST 10H.

Печать нового рекорда.

```
661AH
26163   LD A,02
26165   CALL 1601H
26168   LD A,11H
26170   RST 10H
26171   LD A,05
26173   RST 10H
26174   LD A,16H
26176   RST 10H
26177   LD A,15H
26179   RST 10H
26180   LD A,07
26182   RST 10H
26183   LD A,20H
26185   RST 10H
26186   LD A,20H
26188   RST 10H
26189   LD A,20H
26191   RST 10H
26192   LD A,20H
26194   RST 10H
26195   JP 60C9H
```

ПРОЦЕДУРА HOME

СТРОКА 890. Для вывода на экран здесь также используется процедура для печати символьной строки, но первоначально в данных этой строки переписываются значения x1,y1,x2 и y2. Коды символов сообщения при этом записаны в 669AH...66A5H, перенос значений переменных x1,y1,x2 и y2 - в 66A6H..66B1H, а процедура печати - В 66B2H... ...66BFH.

Печать "героя".

```
669AH
DEFB  11 08 10 08 08 16 02 04
DEFB  15 00 04 96

66A6H
26278   LD HL,(5B06H)
26281   LD (669F),HL
26284   LD HL,(5B03H)
26287   LD (66A3H),HL
26290   LD A,02
26292   CALL 1601H
26295   LD DE,669AH
26298   LD BC,000C
```

```
26301   CALL 203CH
```

СТРОКА 900. Эта строка в программе в машинных кодах не нужна, т.к. расположенные здесь данные для музыки будут загружены непосредственно в память при загрузке машинного кода.

СТРОКИ 910...920. Здесь через CALL 566AH вызывается процедура для проигрыша торжественной мелодии. Сама мелодия расположена в адресах 6656H...6669H, процедура - в 666AH...6699H, а вызов выполняется из точки 66C0H.

Музыка.

```
66C0H
26304   CALL 666AH

6656H
DEFB 00 00 00 00 00 00 ED 0B
DEFB ED 0B F0 10 EC 0B EC 10
DEFB EC 0B EC 10

666AH
26218   LD B,07
26220   LD HL,665CH
26223   LD A,(HL)
26224   LD(6678H),A
26227   INC HL
26226   PUSH BC
26229   PUSH HL
26230   RST 28H ;Калькулятор.
DEFB  34 EC 4C CC CC CC 38
26238   POP HL
26239   LD A,(HL)
26240   PUSH HL
26241   CALL 2D28H
26244   CALL 03F8H
26247   POP HL
26248   POP BC
26249   INC HL
26250   DJNZ -1DH
26252   LD A,01
26254   CALL 3D28H
26257   LD A,14H
26259   CALL 2D28H
26262   CALL 03F8H
26265   RET
```

СТРОКА 930. Изменение значения переменных home и score в машинных кодах см. 66C3H...66D7H, а вывод значения score на экран выполняется с использованием печати числа, находящегося на вершине стека калькулятора. Машинные коды 66D4H...66E9H.

Коррекция переменных.

```
66C3H
26307   LD A,(5B05H)
26310   INC A
26311   LD (5B05H),A
26314   LD HL,(5B03H)
26317   LD DE,0032H
26320   ADD HL,DE
26321   LD (5B03H),HL
```

Печать новых значений.

```
66D4H
26324   PUSH HL
25325   LD A,02
```

```
25327   CALL 1601H
26330   LD A,16H
26332   RST 10H
26333   LD A,15H
26335   RST 10H
26336   LD A,07
26338   RST 10H
26339   POP BC
26340   CALL 2D2BH
26343   CALL 2DE3H
```

СТРОКА 950. В этой строке выполняется проверка: все ли 4 дома уже заняты. Если нет, то осуществляется переход к главному циклу. Программа в машинных кодах, выполняющая эту операцию, проще, чем строка BASIC-программы. Для этого достаточно записать в регистр A значение из ячейки 5B05H и с помощью команды AND 4 маскировать все биты, кроме второго. Если в результате операции флаг нуля (ZERO) не будет установлен, то выполняется переход к главному циклу с помощью команды JP 641E.

Проверка 4-х домов.

```
66EAH
26346   LD A,(5B05H)
26349   AND 04
26351   JP NZ,641EH
```

СТРОКИ 960...980. Эти строки позволяют увеличить скорость игры, когда количество занятых домов достигает 4,8 или 12. Это достигается путем исключения команд RET из процедуры ROLL.

СТРОКА 985. В этой строке проверяется, выше ли число занятых домов, чем 36. Если это так, то пропускается процедура для добавления еще одного паука и осуществляется переход к главному циклу. Для реализации этого в машинных кодах достаточно вычесть от значения переменной "home" число 37. Если при этом флаг переноса (CARRY) не будет включен, то значит число занятых домов >= 37, и выполняется переход по команде JP NC 6129. Машинный код 6714H...6718H.

Изменение скорости игры.

```
66F2H
26354   XOR A
26355   LD A,(5B05H)
26358   CP 04
26360   JR NZ,+06
26362   XOR A
26363   LD(7EA9H),A
26366   JR +19
26368   CP 08
26370   JR NZ,+06
26372   XOR A
26373   LD (7EC2H),A
26376   JR +0F
26378   CP 0C
26380   JR NZ,+06
26382   XOR A
26383   LD (7ED5H),A
26386   JR +5
26368   SUB 25H
26390   JP NC,6129H
```

СТРОКА 990. Работу со случайными числами мы уже обсуждали в четвертой главе, поэтому объяснять порядок преобразования этой строки БЕЙСИК-программы в машинные коды мы здесь не будем. Необходимо только отметить, что для получения случайного числа в этой программе вначале извлекается значение системной переменной SEED, а затем оно

изменяется по следующей программе:

```
LD A,C
AND 1F
SUB 2
```

Все операции выполняются с использованием кодов калькулятора. Машинные коды - 6719H...6740H.

Генерация случайных чисел.

```
6718H
26393   LD BC,(5C76H)
26397   CALL 2D2BH
26400   RST 28; калькулятор.
DEFB  A1 0F 34 37 16 04 34 80
DEFB  41 00 00 80 33 02 A1 03
DEFB  31 38
26419   CALL 2DA2H
26422   LD (5C76P),BC
26426   LD A,C
26427   AND 1F
26429   SUB 02
26431   JR C,+05
```

СТРОКА 1000. Все, что необходимо сделать, это увеличить на 1 значение переменной "a". Машинный код - 6741H.

СТРОКА 1005. В этой строке проверяется, не превышает ли значение переменной "a" величины 31. Если это так, то значение этой переменной обнуляется.

```
6741H
26433   INC A
26434   CP 20H
26436   JR NZ,+01
26438   XOR A
```

СТРОКА 1010. Для того, чтобы проверить свободна ли позиция экрана с координатами (10,a), используется процедура SCREEN$. Для этого первоначально в регистр B записывается значение переменной "a", а в регистр C - значение 0AH (10D), и вызываются процедуры STACK VALUE IN BC и SCREEN$, а затем параметры со стека передаются в регистры A,B,C,D,E. Теперь выполняется проверка, SCREEN$(10,a) = 20H. Если это так, то выполняется переход для проверки следующей позиции экрана по команде JR 6741. Машинные коды, соответствующие этой строке, 6747H...675EH.

Поиск свободного знакоместа.

```
6747H
26439   PUSH AF
26440   LD B,A
26441   LD C,0A
26443   CALL 2538H
26446   CALL 2BF1H
26449   LD A,(DE)
26450   PUSH AF
26451   CALL 16BFH
26454   POP AF
26455   CP 20
26457   JR Z,+03
26459   POP AF
26460   JR -1D
26462   POP AF
```

СТРОКА 1020. Эта строка повторяет предыдущую, за исключением того, что значение переменной "a" увеличивается на 1 перед вызовом процедуры SCREEN$. Обратите внимание, что значение переменной "a" сохраняется перед вызовом процедуры SCREEN$

(PUSH AF), чтобы его можно было использовать в следующей процедуре. Машинные коды 675FH...6776H.

Поиск свободного знакоместа.

```
675FH
26463   INC A
26464   PUSH AF
26465   LD B,A
26466   LD C,0A
26468   CALL 2538H
26471   CALL 2BF1H
26474   LD A,(DE)
26475   PUSH AF
26476   CALL 16BFH
26479   POP AF
26480   CP 20
26483   JR Z,+03
26484   POP AF
26485   JR -36H
```

СТРОКА 1030. Значение переменной "a" извлекается со стека и записывается в регистр A, а затем размещается в участке памяти с именем DATA для следующего вывода на экран процедурой PRINT DATA. Машинные коды 6777H... 6792H.

Печать добавочного паука.

```
6777H
26487   POP AF
26468   DEC A
26489   LD (6782H),A
26492   JR +07
DEFB 11 04 16 0A 19 9E 9F
26501   LD A,02
26503   CALL 1601H
26506   LD DE,677EH
26509   LD BC,0007
26512   CALL 203CH
```

СТРОКА 1039. Здесь повторно вызывается процедура, воспроизводящая фанфары по команде CALL 666A.

СТРОКА 1040. Это строка безусловного перехода в начало цикла. В машинных кодах это выполняется по команде JP 6129H.

Музыка и возврат.

```
6793H
26515   CALL 666AH
26518   JP 6129H
```

Как видите, выполнять перевод БЕЙСИК-программы в машинные коды не так уж сложно. Лучше использовать при написании программ в машинных кодах шестнадцатиричную систему счисления, т.к. она более наглядна, если выполняется запись чисел сразу же в регистровую пару.

Нам осталось только привести машинный код для скроллинга экрана вправо-влево. Это код процедуры ROLL.

Машинный код процедуры ROLL.

```
ORG 32244
RIGHT 32244     LD C,8
L1    32246     PUSH HL
      32247     LD DE,31
      32250     ADD HL,DE
      32251     LD A,(HL)
      32252     SBC HL,DE
      32254     RRA
      32255     LD B,32
L2    32257     LD A,(HL)
      32258     RRA
      32259     LD (HL),A
      32260     INC HL
      32261     DJNZ L2
      32263     POP HL
      32264     INC H
      32265     DEC C
      32266     JR NZ,L1
      32268     RET
      32281     LD B,32
L4    32283     LD A,(HL)
      32284     RLA
      32285     LD (HL),A
      32286     DEC HL
      32287     DJNZ L4
      32289     POP HL
      32290     INC H
      32291     DEC C
      32292     JR NZ,L3
      32294     RET
      32295     LD HL,16479
      32298     CALL LEFT
      32301     LD HL,16512
      32304     CALL RIGHT
      32307     LD HL,16512
      32310     CALL RIGHT
      32313     LD HL,16607
      32316     CALL LEFT
      32319     LD HL,18432
      32322     CALL RIGHT
      32325     LD HL,18432
      32328     CALL RIGHT
      32331     LD HL,18432
      32334     CALL RIGHT
      32337     LD A,(23673)
      32340...32346 - NOP
      32347     AND 02
      32349     JR Z,L5
      32351     LD HL,18496
      32354     CALL RIGHT
      32357     LD HL,18496
      32360     CALL RIGHT
      32363     LD HL,18496
      32366     CALL RIGHT
      32369     JR L6
L5    32371     LD HL,18527
      32374     CALL LEFT
      32377     LD HL,18527
      32380     CALL LEFT
      32383     LD HL,18527
      32386     CALL LEFT
L6    32389     LD HL,18560
      32392     CALL RIGHT
      32395     LD HL,18560
      32398     CALL RIGHT
      32401     LD HL,18560
      32404     CALL RIGHT
      32407     LD HL,18624
      32410     CALL RIGHT
      32413     LD HL,20511
      32415     CALL LEFT
      32419     LD HL,20575
      32422     CALL LEFT
      32425     RET
      32426     LD HL,18560
      32429     CALL RIGHT
      32432     LD HL,18624
      32435     CALL RIGHT
      32438     LD HL,20511
      32441     CALL LEFT
      32444     LD HL,20575
      32447     CALL LEFT
      32450     RET
      32431     LD HL,16479
      32454     CALL LEFT
      32457     LD HL,16512
      32450     CALL RIGHT
      32463     LD HL,18432
      32466     CALL RIGHT
      32469     RET
      32470     LD HL,16479
      32473     CALL LEFT
      32475     LD HL,16607
      32479     CALL LEFT
      32482     LD HL,18560
      32485     CALL RIGHT
      32486     LD HL,18624
      32491     CALL RIGHT
      32494     RET
```

ЧИТАТЕЛЬ - ЧИТАТЕЛЮ

Вместо вступления.

Дорогие друзья! Эту объемную статью не хочется предварять утомительным вступлением, но если Вас абсолютно не интересует музыка и, тем более, у Вас нет идей по работа с программой WHAM, мы все же просим Вас внимательно просмотреть статью. Дело в том, что здесь представлена такая гирлянда идей, полезных советов и маленьких хитростей, которая украсит многие программы, даже если они и не имеют отношения к музыке.

"ИНФОРКОМ".

WHAM! The Music Box. НОВЫЕ ВОЗМОЖНОСТИ

Этот музыкальный редактор наверняка уже знаком читателям "ZX-РЕВЮ". Он помогает не только освоить азы музыкальной грамотности, но и придать особый блеск Вашим собственным программам, тот завершающий штрих, который придает им неповторимый облик.

"WHAM" достаточно широко распространен на рынке программного обеспечения и читатель наверняка имеет ту или иную версию в своем арсенале. Существуют и подробные описания к программе. Более широко развить эту тему вынуждают некоторые тонкости, связанные с ее работой. В частности, неграмотная адаптация под БЕТА-ДИСК многих версий, имеющих хождение на рынке программного обеспечения. В результате этого они неправильно выполняют запись на диск скомпилированного кодового музыкального фрагмента. Предлагаемый материал вносит ясность в этот вопрос. Остановлюсь и на некоторых других особенностях работы этой программы, и главное - довольно существенном ее усовершенствовании.

Надо сказать, что чаще при изложении материала применяется следующий подход: какое-то явление или процедура и как ее можно использовать. Этот материал находится несколько в иной плоскости: конкретная программа и как к ней применить сведения из разных областей, так или иначе имеющие к ней отношение.

Итак, "WHAM". Вот типовой набор файлов, входящих в программу.

```
"WHAM!"    BASIC
"SCREEN$" CODE 16384,6913
"WHAM!1"   CODE 65300,235
"WHAM!2"   CODE 30000,15000
"WHAM!3"   CODE 50000,7000
"M.BOX"    BASIC
```

Первый Бейсик-файл является загрузчиком. Во-первых, он переустанавливает значение RAMTOP=29999. Таким образом, для блоков кодов отводится место с адреса 30000. Во-вторых, именно в этом Бейсик-загрузчике находится команда, переключающая символьный набор на загружаемый утолщенный, стилизованный. Новое значение

системной переменной CHARS задается при помощи POKE:

```
POKE 23606,43: POKE 23607,217
CHARS=42+256*217=55594
```

Переключение CHARS происходит в первой же строке Бейсик-загрузчика и является, кроме того, методом защиты. Это приводит к тому, что при нажатии BREAK ничего не видно: сам символьный набор еще не загружен. Это отпугивает начинающих. На самом деле все просто: переключитесь на символьный набор ПЗУ, для этого подайте "вслепую" прямую команду POKE 23606,0: POKE 33607,60 и все встанет на свое место.

Символьный набор расположен с адреса, на 256 большего, чем CHARS: 55594+256=55650. Если у Вас появятся желание заменить его или русифицировать всю программу, то Вы сможете это сделать, расположив новый символьный набор с адреса 55650. При русификации надо также учесть, что значительная доля текстовых сообщений содержится в машинно-кодовой части программы.

Большая часть программы выполняется в машинных кодах, но программа обращается и к Бейсику в определенных случаях. Это загрузка-выгрузка мелодии, компиляция и запись скомпилированного блока кодов. Одной из первоначальных проблем является то, как остановить программу и выйти в Бейсик. Для этого можно в титульном меню нажать "1" или "2" (загрузка или запись). Когда в нижней части экрана программа запросит ввод устройства - это уже работает Бейсик. Нажимайте BREAK для остановки программы. Запустить опять ее можно командой RUN.

Забегая вперед, скажу, что, когда появилась идея дополнить программу некоторыми новыми функциями, встал вопрос, как уместить все эти дополнения в рамках отведенной памяти. Чудес ведь не бывает. Для набивки новых строк необходимо отодвигать RAMTOP в сторону более старших адресов. Поэтому помимо обычных средств по экономии памяти (о которых будет рассказано ниже), пришлось пожертвовать частью буфера для хранения мелодий.

В первоначальном варианте "WHAM" может хранить в памяти 6 мелодий. Новый, усовершенствованный вариант - на одну меньше. Но такая жертва, по-моему, все же оправдана теми дополнительными возможностями, о которых будет рассказано ниже. Поэтому в новом загрузчике в строке 30 устанавливается значение RAMTOP, равное 31999 а не 29999, как в исходном варианте.

Изначальный текст Бейсик-файла "M.BOX" примерно на 80% был переработан. Во избежание неоднозначности, ниже приведен полностью его новый вариант: "m.box". Если Вы захотите воспроизвести программу у себя, внося изменения в старий файл "M.BOX", проверьте, пожалуйста, аккуратно все строки. Обратите внимание на их номера. Во многих случаях это существенно. Поэтому на начальном этапе без необходимости старайтесь ничего не изменять. Когда новая программа у Вас заработает, тогда можете вносить любые дальнейшие изменения.

Изменениям подверглись и кодовые блоки программы для поддержки ее новых возможностей. Подробно все изменения в кодовых блоках приведены в конце статьи. Условимся, что новые файлы, входящие в "WHAM" будут называться:

```
"WHAM+"  BASIC
"wham $" CODE
"Wham 1" CODE
"wham 2" CODE
"wham 3" CODE
"m.box"  BASIC
```

Это нужно для того, чтобы различать новые и старые версии файлов.

Теперь некоторые вопросы, связанные с адаптацией под БЕТА-диск.

Универсальный загрузчик.

Адаптация загрузчика под БЕТА-диск тривиальна. Заменяем все LOAD"" на RANDOMIZE USR 15619:REM LOAD "ИМЯ".

Но сейчас я хочу предложить прием, позволяющий реализовать универсальный загрузчик, одинаково работающий и с магнитофоном и с БЕТА-диском. Такой прием можно

использовать, в принципе, для любой программы, но для многих - это ненужная избыточность. Однако для таких программ, как "WHAM", которые после адаптации под диск остаются двухрежимными (загрузка и запись сохраняются для ленты и диска) эффектнее выглядит предлагаемый вариант. Он придает большую гибкость программе.

Универсальность основана на том факте, что при использовании БЕТА-диска системная переменная PROG (начало расположения Бейсик-программы) отличается от магнитофонного варианта компьютера на 112 байтов для дополнительных системных переменных. При проверке, если зафиксировано значение PROG=23755, то значит интерфейс Бета-диска отсутствует или неинициализирован. В этом случае программа выполняет загрузку с ленты, Если значение PROG больше, чем 23755, программа переходит на строки загрузки с диска.

Универсальный двухрежимный загрузчик для "WHAM" выглядит так.

```
10 REM "WHAM!" Universal Tape-Disk loader.
20 BORDER 0: PAPER 0: INK 0: POKE 23624,0: CLEAR 31999
25 IF (PEEK 23635+256*PEEK 23636)=23755 THEN POKE 23739,111:LOAD ""SCREEN$: LOAD ""CODE :
      LOAD ""CODE : LOAD ""CODE : GO SUB 100: LOAD ""
30 RANDOMIZE USR 15619: REM : LOAD "wham $"CODE 16384
40 RANDOMIZE USR 15619: REM : LOAD "wham 1"CODE
50 RANDOMIZE USR 15619: REM : LOAD "wham 22"CODE
60 RANDOMIZE USR 15619: REM : LOAD "Wham 3"CODE
70 GO SUB 100
80 RANDOMIZE USR 15619: REM : LOAD "m.box"
100 POKE 23675,88: POKE 23676,255: POKE 23606,42: POKE 23607,217
110 RETURN 9999 SAVE "WHAM+" LINE 10
```

Подпрограмма GO SUB 100 кроме переключения символьного набора выполняет также переключение символов UDG-графики, восстанавливая исходное значение, которое могло быть изменено дисковым загрузчиком "boot".

Основной файл "m. box".

Как уже говорилось, новый вариант потребовал принятия мер по экономии памяти. Некоторые из них уже реализованы в файле "M.BOX". Это, например, задание вначале программы в 10 строке:

```
LET O=NOT PI: LET I=SGN PI: LET N7=VAL "7" ...
```

Буква O похожа на ноль, а буква I - на единицу, так что такая замена практически не сказывается на "читаемости" программы. Кроме того, при помощи переменной N7 задано число 7. Оно часто встречается в программе (например, BORDER N7: PAPER N7:). Замена часто встречающихся чисел переменными позволяет существенно сократить объем программы. Но для БЕТА-диска и всех нововведений этого все же недостаточно. Пришлось бороться буквально за каждый байт. Потребовалось заменить все числа на выражения VAL "число" во всей программе. Только теперь можно быть уверенным, что не произойдет сбоев программы из-за нехватки памяти при работе.

Листинг файла "m.box".

```
1 REM (C) 1985 MARK TIME LTD (C) 1993 ALANSOFT
2 GO TO VAL "10"
5 GO SUB VAL "90": STOP
6 RANDOMIZE USR VAL "15616"
7 STOP
10 CLS : LET O=NOT PI: LET I=SGN PI: LET N7=VAL "7": BORDER N7: RANDOMIZE USR VAL "50000":
      INPUT '''': LET T$=""
15 LET STR=VAL "50000": LET LEN=VAL "2000"
20 LET KP=PEEK VAL "23559"-VAL"48"
25 IF KP<O OR KP>N7 THEN RUN
30 PRINT AT VAL "5"+KP,VAL "9" ; OVER I; INVERSE I; "            "
35 POKE VAL "23658",VAL "6"
40 IF KP<INT PI THEN GO TO VAL "3000"
45 IF KP=VAL "4" THEN GO TO VAL "8000"
```

```
50 CLS : BORDER VAL "5": PRINT AT VAL "8",0;"  1 -Print HELP-PAGE."'"  2 -Select MUSIC-KEY
      --->"'"  3 -Routine RECOMPILER."'"  4 -Tune TRANSPONER."
55 RANDOMIZE USR (PEEK VAL "51253"+VAL "56576"): PAUSE 0
60 IF INKEY$="2" THEN GO SUB VAL "600": GO TO VAL "55"
65 IF INKEY$="3" THEN GO TO VAL "9000"
70 IF INKEY$="4" THEN GO TO VAL "5000"
75 IF INKEY$="1" THEN CLS : PRINT : RANDOMIZE USR VAL "55420": PAUSE 0
80 RUN
90 IF PEEK VAL "23635"+VAL "256"*PEEK VAL "23636"=VAL "23755" THEN SAVE "m.box" LINE VAL
      "2": RETURN
92 RANDOMIZE USR VAL "15619": REM : ERASE "m.box"
94 RANDOMIZE USR VAL "15619": REM : SAVE "m.bOX" LINE 2
99 RETURN
100 IF T$="M" THEN GO SUB VAL "4000": IF KP=I THEN RANDOMIZE USR VAL "54000": GO TO VAL
      "140"
105 GO SUB VAL "6000": LET ERR=0
110 IF KP=VAL "2" THEN GO TO VAL "200"
115 IF T$="D" THEN LET ERR=USR VAL "15619": REM : LOAD F$CODE STR,LEN
117 IF ERR<>0 THEN GO TO VAL "300"
120 IF T$="T" THEN CLS : PRINT "START TAPE"'"SEARCHING FOR :";F$: IF F$<>"          " THEN
      LOAD F$CODE STR
130 IF T$="T" AND F$="          " THEN PRINT "Loading any tune - no name given": LOAD ""CODE
      STR,LEN
140 IF PEEK STR>VAL "220" THEN POKE VAL "52860",PEEK STR
150 RUN
200 POKE STB,PEEK VAL "52860"
210 IF T$="D" THEN LET ERR=USR VAL "15619": REM : SAVE F$CODE STR,LEN
215 IF ERR<>0 THEN GO TO VAL "300"
220 IF T$="T" THEN SAVE F$CODE STR,LEN
225 IF T$="M" THEN RANDOMIZE USR VAL "54006": FOR A=I TO VAL "10": POKE VAL
      "65289"+A+(TN*VAL"10"),CODE F$(A): NEXT A: RUN
250 PRINT #0;"VERIFY ? (Y/N)": BEEP VAL ",1",VAL "20": GO SUB VAL "500": IF NOT F THEN RUN
260 LET ERR=0: PRINT AT VAL "19",0;"           Verifying...          "
270 IF T$="T" THEN VERIFY F$CODE STR,LEN
280 IF T$="D" THEN LET ERR=USR VAL "15619": REM : VERIFY F$CODE STR,LEN
290 IF ERR=0 THEN PRINT #0;TAB VAL "12"; INVERSE I;"  O.K.  ": BEEP VAL ".1",VAL "20": PAUSE
      VAL "50": RUN
300 PRINT #0; FLASH I;" ERROR ";ERR;" ": BEEP I,0: PAUSE 0: RUN
400 POKE VAL "23658",VAL "8": INPUT I: RETURN
500 POKE VAL "23658",VAL "8": PAUSE 0: INPUT ;: IF INKEY$ ="Y" THEN LET F=I: RETURN
510 LET F=0: RETURN
600 IF PEEK VAL "50908"=VAL "159" THEN POKE VAL "50908",VAL "191": POKE VAL "51253",VAL
      "178": RETURN
610 IF PEEK VAL "50908"=VAL "191" THEN POKE VAL "50908",VAL "159": POKE VAL "51253",VAL
      "170": PRINT AT VAL "13",VAL "27";"   ";AT VAL "14",VAL "27";"   ": RETURN
700 INPUT ;: PRINT #0; FLASH I;"         Please, wait...         ": RETURN
800 GO SUB VAL "400": PRINT #0;"    Select device for ";"LOAD" AND KP=VAL "7";"SAVE" AND
      KP=VAL "4";":"'" Press:   T -TAPE;  D -DISK"
805 PAUSE 0: INPUT ;: LET T$=INKEY$: IF T$<>"T" AND T$<>"D" THEN GO TO VAL "800"
810 RETURN
3000 PRINT #0;TAB VAL "4"; "SELECT PERIPHERAL DEVICE"
3010 PRINT #0''"TAPE MEMORY DISK (SPACE = EXIT) "
3020 PAUSE 0: LET T$=INKEY$: IF INKEY$=" " THEN RUN
3030 IF T$<>"M" AND T$<>"T" AND T$<>"D" THEN GO TO VAL "3020"
3040 GO TO VAL "100"
4000 CLS : PRINT ''''"TUNES STORED IN MEMORY"
4010 LET D=VAL "65300": PRINT '''"NO. TITLE:";
4020 FOR T=I TO VAL "5": PRINT '
4030 FOR D=D TO D+VAL "9": PRINT CHR$ PEEK D;: NEXT D: NEXT T
4040 PRINT #0;"TUNE NUMBER? L"
4050 LET N$=INKEY$: IF N$<"1" OR N$>"5" THEN GO TO VAL "4050"
4060 LET TN=VAL N$: POKE VAL "23681",TN+I: RETURN
5000 CLS : PRINT AT VAL "10", VAL "5"; "Tune Transponierung"
5010 GO SUB VAL "400": PRINT #0; "Press: U -for UP;  D -for DOWN"
5020 PAUSE 0: IF INKEY$<>"U" AND INKEY$<>"D" THEN GO TO VAL "5020"
```

```
5030 LET R=(INKEY$="U")-(INKEY$="D")
5040 INPUT ;: PRINT #0; " 1  - 1 Channel   2  - 2 Channel     3  - 1 and 2 Channel"
5050 PAUSE 0: IF INKEY$<"1" OR INKEY$>"3" THEN GO TO VAL "5050"
5060 LET C=VAL INKEY$
5070 INPUT "Tune change (12=octave): ";S
5100 LET V=0: GO SUB VAL "700"
5110 IF C<>VAL "2" THEN LET A=VAL "60001": GO SUB VAL "5200"
5120 IF C<>I THEN LET A=VAL "61001": GO SUB VAL "5200"
5130 IF V=0 THEN LET V=I: GO TO VAL "5110"
5140 INPUT ;: PRINT AT VAL "10",VAL "5"; "Transponierung - O.K.":BEEP VAL ".1",VAL "20":
      PAUSE 0: RUN
5200 IF PEEK A=VAL "64" OR A=VAL "61000" OR A=VAL "62000" THEN RETURN
5210 IF PEEK A>VAL "40" AND PEEK A<VAL "244" THEN LET A=A+I: GO TO VAL "5200"
5220 LET T=S*R+PEEK A
5230 IF T>VAL "255" THEN LET T=T-VAL "256"
5240 IF T<0 THEN LET T=T+VAL "256"
5250 IF V=0 THEN IF T>VAL "40" AND T<VAL "244" THEN GO TO VAL "5300"
5260 IF V=I THEN POKE A,T
5270 LET A=A+I: GO TO VAL "5200"
5300 INPUT ;: PRINT AT VAL "15",0;"  Tune ";("1" AND A<VAL "61000");("2" AND A>VAL
      "61000");" Channel is wery ";("high." AND R=I);("low." AND R=-I)
5310 PRINT "Transponierung is not available.": BEEP I,0: PAUSE 0: RUN
6000 DIM F$(VAL "10"): INPUT "Press ENTER for CATALOGUE DISK" AND T$="D"''"FILENAME: ";F$
6010 IF F$="          " AND T$="D" THEN GO TO VAL "6100"
6020 RETURN
6100 RANDOMIZE USR VAL "15619": REM : CAT
6110 GO TO VAL "6000"
7000 LET CHN1=VAL "57266": IF WHIND=VAL "3000" THEN LET CHN1=VAL "57464"
7010 LET CHN2=CHN1+LEN1+I
7020 LET HL=VAL "60001": LET DE=CHN1: LET BC=LEN1: GO SUB VAL "7500": LET HL=VAL "61001":
      LET DE=CHN2: LET BC=LEN2: GO SUB VAL "7500"
7030 LET TLENG=CHN2+LEN2-VAL "57000"
7031 LET CHN1=CHN1-VAL "57000"+ASEM
7032 LET CHN2 = CHN2-VAL  "57000"+ASEM
7040 PRINT AT VAL "10",VAL "6";"CODE LENGTH : ";TLENG
7050 POKE VAL  "57030",INT   (CHN1/VAL  "256"): POKE VAL  "57029",CHN1-VAL  "256"*PEEK VAL
      "57030"
7055 POKE VAL "57002",INT ((CHN1-I)/VAL "256"): POKE VAL "57001",CHN1-I-VAL "256"*PEEK VAL
      "57002"
7060 POKE VAL "57034",INT (CHN2/VAL "256"): POKE VAL "57033",CHN2-VAL "256"*PEEK VAL "57034"
7065 POKE VAL "57008", INT ((CHN2-I)/VAL "256"): POKE VAL "57007",CHN2-I-VAL "256"*PEEK VAL
      "57008"
7080 RETURN
7500 LET Z=VAL "256": LET H=INT(HL/Z): LET L=HL-H*Z: LET D=INT(DE/Z): LET E=DE-D*Z: LET
      B=INT(BC/Z): LET C=BC-B*Z: RESTORE VAL "7700": FOR Z=VAL "16416" TO VAL "16428": READ
      S: POKE Z,S: NEXT Z: RANDOMIZE USR VAL "16416":RETURN
7700 DATA VAL "33",L,H,VAL "17",E,D,I,C,B,VAL "237",VAL "176",VAL "201",PI
7701 PRINT #0;"RETURN OPTION 1,2 OR 3"
7702 PRINT AT VAL "12",0;"1,KEYPRESS"'"2,ALWAYS"'"3,TUNE END"
7703 IF INKEY$="1" THEN RETURN
7704 IF INKEY$="2" THEN PRINT AT VAL "4",VAL "20";"ALWAYS  ": POKE VAL "57020",VAL "62":
      RETURN
7706 IF INKEY$="3" THEN POKE VAL "57063",VAL "225": POKE VAL "57064",VAL "225": POKE VAL
      "57065",VAL "251": POKE VAL "57066",VAL "201": PRINT AT VAL "4",VAL "20";"TUNE END":
      LET A$=" N/A ": PRINT AT VAL "7",VAL "20";A$;AT VAL "9",VAL "20";A$: RETURN
7707 GO TO VAL "7702"
8000 PAPER N7: INK I: BORDER VAL "5": CLS : PRINT  INK N7;AT I,0;"             "; INK I:
      RANDOMIZE USR VAL "55640"
8110 POKE VAL "23681",I: LET LEN1=USR VAL "54032"-VAL "60000"
8115 PRINT AT VAL "6",VAL "20";LEN1
8116 IF LEN1=VAL "999" THEN PRINT AT VAL "6",VAL "20";"NOT PRESENT"
8120 POKE VAL "23681",VAL "2": LET LEN2=USR VAL "54032"-VAL "61000"
8125 PRINT AT VAL "8",VAL "20";LEN2
8126 IF LEN2=VAL "999" THEN PRINT AT VAL "8",VAL "20";"NOT PRESENT"
8130 LET WHIND=USR VAL "54056"
```

```
8133 IF VHIND=VAL "3000" THEN PRINT AT VAL "5",VAL "20";"*PRESENT*"
8140 IF LEN1=VAL "999" OR LEN2=VAL "999" THEN GO TO VAL "8800"
8141 LET SM=LEN1-I: IF LEN2<LEN1 THEN LET SM=LEN2-I
8142 IF (LEN1-I)/SM<>INT ((LEN1-I)/SM) OR (LEN2-I)/SM<>INT ((LEN2-I)/SM) THEN GO TO VAL
      "8600"
8150 INPUT "ASSEMBLY ADDRESS: ";ASEM
8160 IF ASEM<VAL "16384" THEN GO TO VAL "8150"
8161 LET TLENG=LEN1+LEN2+VAL "267"+(VAL "198"*(WHIND=VAL "3000")): IF ASEM+TLENG>VAL "65536"
      THEN PRINT #0;"Error :-no room for tune.": BEEP I,0: LET ASEM=VAL "85536"-TLENG:
      PRINT AT VAL "21",0;"    MAX. ADDRESS: ";ASEM: PAUSE 0: GO TO VAL "8150"
8162 PRINT AT VAL "3",VAL "20";ASEM
8164 POKE VAL "55703",INT (ASEM/VAL "256");POKE VAL "55702",ASEM-VAL "256"*PEEK VAL "55703":
      RANDOMIZE USR VAL "55700"
8165 IF LEN1=LEN2 THEN GO SUB VAL "7700": INPUT "":PRINT AT VAL "12",0''': GO TO VAL "8170"
8166 PRINT #0;"CHANGE RETURN OPTION TO ALWAYS ?": GO SUB VAL "500":   IF F THEN POKE VAL
      "57020",VAL "62": PRINT AT VAL "4",VAL "20";"ALWAYS "
8170 GO SUB VAL "700": GO SUB VAL "7000": INPUT :
8176 INK 0: PRINT AT VAL "12",0;"ROUTINE COMPILATION COMPLETED OK",,
8177 PRINT '"ADJUSTMENT POKES :-",
8178 PRINT "REPLAY SPEED :";ASEM+VAL "35";",(230 TO 255)"'"BORDER COLOUR:";JASEM+VAL
      "26";",(0 TO 7)"
8179 PRINT "TO RUN - RANDOMIZE USR ";ASEM
8180 POKE VAL "57035",PEEK VAL "52860"-VAL "4": POKE VAL "57026",PEEK VAL "52851"
8205 POKE VAL "56814",INT (ASEM/VAL "256"): POKE VAL "56813",ASEM-VAL "256"*PEEK VAL "56814"
8207 POKE VAL "56812",INT (TLENG/VAL "256"): POKE VAL "55811",TLENG-VAL "256"*PEEK VAL
      "56812"
8208 POKE VAL "56718",PEEK VAL "56811": POKE VAL "56719",PEEK VAL "56812"
8210 GO SUB VAL "6000": PRINT AT VAL "21",0,,: PRINT AT VAL "2",VAL "8"; INK I;"TUNE NAME :
      ";F$
8220 FOR A=I TO VAL "10": POKE VAL "56800"+A,CODE (F$(A)): NEXT A
8230 GO SUB VAL "800"
8240 IF T$="T" THEN PRINT #0;"Start tape, then press any key.":PAUSE 0: BORDER N7: INPUT ;:
      RANDOMIZE USR VAL "58700": GO TO VAL "8300"
8250 LET ERR=USR VAL "15619": REM : SAVE F$CODE 57000,TLENG
8260 IF ERR<>0 THEN GO TO VAL "300"
8270 RANDOMIZE USR VAL "56728"
8300 LET STR=VAL "57000": LET LEN=TLENG: GO TO VAL "250"
8600 PRINT AT VAL "12",VAL "8";"WARNING"'"End marker mismatch."'''may cause distorted
      tune."'''Continue ? (y/n)"
8610 GO SUB VAL "500": IF F THEN GO TO VAL "8145"
8630 RUN
8810 PRINT AT VAL "12",0;"Tune compilation abandoned."
8811 PRINT '"No end markers defined."
8813 PRINT "Use the w key to place the end marker for the current channel."
8820 PAUSE 0: RUN
9000 CLS : PRINT AT VAL "10", VAL "6"; "Routine Recompiling": GO SUB VAL "800"
9010 IF T$="D" THEN GO SUB VAL "6090": LET ERR=USR VAL "15619":REM : LOAD F$ CODE 57000
9020 IF T$="D" THEN LET STR=PEEK VAL "23782"+VAL "256"*PEEK VAL"23783": IF ERR<> THEN GO TO
      VAL "300"
9030 IF T$="T" THEN LOAD ""CODE VAL "57000": LET STR=PEEK (PEEK 23649+256*PEEK
      23650+30)+256*PEEK (PEEK 23649+256*PEEK 23650+31)
9040 PRINT AT 0,0;   PAPER  VAL  "7";   INK VAL "7";"              ": GO SUB VAL  "700"
9050 RESTORE VAL "9000": FOR A=VAL "16384" TO VAL "16397": READ B: POKE A,B: NEXT A:
      RANDOMIZE USR VAL "16384"
9060 DATA VAL "33",VAL "96",VAL "234",VAL "54",VAL "41",VAL "17",VAL "97",VAL "234",SGN
      PI,VAL "207",VAL "7",VAL "237",VAL "176",VAL "201"
9070 IF PEEK VAL "57035">VAL "250" THEN POKE VAL "60000",VAL "255"
9080 IF PEEK VAL "57035"<=250 THEN POKE VAL "60000",PEEK VAL "57035"+VAL "4"
9090 POKE VAL "52860",PEEK VAL "60000"
9100 LET A1=VAL "57000"-STR+(PEEK VAL "57001"+VAL "256"*PEEK VAL "57002")+I: LET A2=VAL
      "60001"
9110 GO SUB VAL "9200": LET M1=A2
9120 LET A1=A1+VAL "2": LET A2=VAL "61001"
9130 GO SUB VAL "9200": LET M2=A
9140 CLS : PRINT AT VAL "10",VAL "7": "Recompiling - O.K.": BEEP VAL ".1",VAL "20"
```

```
9150 GO SUB VAL "400": PRINT #0;"Delete markers""Tune end""? (Y/N)": GO SUB VAL "500"
9160 IF F THEN POKE M1,VAL "41": POKE M2,VAL "41"
9170 RUN
9200 POKE A2,PEEK A1
9210 IF PEEK A1<>VAL "64" THEN LET M=A1+I: LET A2=A2+I: GO TO VAL "9200"
9220 RETURN
```

После загрузки основного Бейсик файла "m.box", происходит старт программы с начальной строки. Несколько начальных строк выполняют вспомогательные функции. RUN5 5 - самозапись файла на ленту или диск (в зависимости от его наличия проверка в строке 90 по PROG выполняется аналогично тому, как это сделано в универсальном загрузчике). Переключение символьного набора - GO SUB 8. Обратное переключение его GO SUB 9.

Фактическое начало программы - строка 10. Титульная заставка прорисовывается при помощи подпрограммы в машинных кодах, расположенной по адресу 50000. Последняя устроена так, что при первом старте после загрузки сначала воспроизводится демонстрационная мелодия, прекращающаяся после нажатия на любую клавишу. Эта мелодия представляет собой скомпилированный фрагмент, расположенный и стартуемый с адреса 42500. При последующих запусках программы командой RUN, мелодия при прорисовке заставки не воспроизводится.

Когда подпрограмма в кодах RANDOMIZE USR 50000 возвращается в Бейсик, в ячейке 23559 сохраняется код нажатой клавиши. Строка 20 возвращает номер нажатой цифровой клавиши в переменную KP и дальнейшее распределение происходит в зависимости от ее значения. Так, строка 40 переводит на подпрограмму записи-считывания, строка 45 - на компиляцию. В том случае, если были нажаты клавиши "5" - "6", возврата в Бейсик вообще не происходит. А вот если нажата клавиша "7", то вместо "HELP PAGE" вызывается дополнительное системное меню: "SYST. MENU" - (соответствующее изменение текста титульного меню произведено в кодовом блоке). В "SYST.MENU" заложены все дополнительные возможности, но об этом чуть позже.

А сейчас более подробно об адаптации файла "m.box" под БЕТА-ДИСК. Изменения будут касаться процедур загрузки и выгрузки. Это пункты "1" и "2" в титульном меню программы. Эти процедуры имеют много общих строк, так что технология адаптации упрощается. В исходном варианте "WHAM" после нажатия клавишей "1" или "2" строка 40 переводит на блок строк с 3000, выполняющий задание устройства ввода-вывода. Внизу экрана появляется новое меню, предлагающее выбрать устройство:

```
TAPE, MEMORY, DRIVE (SPACE=EXIT)
```

Так как микродрайвы в нашей стране не слишком распространены, удобно было DRIVE заменить на DISK с изменением всех Бейсик-строк, относящихся к микродрайву. Заменена прежде всего строка, выводящая это меню - 3010.

Строка 3040 завершает выбор устройства и переводит на строку 100. Здесь выполняются дальнейшие действия, связанные с загрузкой-сохранением.

Строка 100 - выполнение подпрограммы записи-считывания мелодии при работе с буфером памяти: GO SUB 4000.

Подпрограмма GO SUB 6000 (строка 105) - это ввод имени файла. Удобно будет при выборе диска предложить пользователю вывести его каталог. Поэтому подпрограмма устроена так, что в случае дисковода, если не вводя имени нажать ENTER, будет выдан каталог диска. Такое решение удобнее, чем дополнительное меню, запрашивающее вывод каталога, так как освобождает оператора от лишних действий. Это, казалось бы, несущественная мелочь, но как раз из таких мелочей складывается впечатление о той или иной программе. Приятно иметь дело с программой, где все продумано до мелочей. Я не пытаюсь представить эту программу как эталон, усовершенствованию нет предела, но Вы можете в своих разработках использовать те моменты и приемы, которые уже хорошо зарекомендовали себя.

В строке 105 кроме ввода имени, задается нулевое начальное значение параметра ERR - кода ошибки TR-DOS. Далее, в строке 10 происходит разделение на процедуры

записи и загрузки. Если запись, то переход на строку 200. Если загрузка, то - на строку 115.

Как Вы знаете, в случае ошибки загрузки с магнитофона, программа остановится с сообщением "Tape loadind error". В случае ошибки при загрузке с диска, мы ничего не заметим. Поэтому необходимо предусмотреть специальные меры по перехвату ошибок TR-DOS. Для этого код ошибки возвращается в переменную ERR для контроля выполнения команды.

Строка 300 должна сигнализировать о невыполненной команде TR-DOS. Здесь применяется упрощенный прием, когда не расшифровывается подробно причина неудачи, а просто выводится код ошибки. Как правило, бывает достаточно даже просто одного сигнала об ошибке, чтобы вспомнить, что, например, файл с таким именем уже есть на дискете. Но лучше, все же, вывести код ошибки ERR. Это не отнимет много памяти, как подробные комментарии на все случаи жизни. Для дальнейшей оценки кода ошибки можно воспользоваться руководством к TR-DOS, но на практике, как правило, в этом не бывает необходимости.

После завершения записи программа предлагает выполнить верификацию. Если в ответ на запрос ввести любую клавишу, кроме "?", то от нее можно отказаться. Верификация реализуется в строках с 250.

Строки загрузки с магнитной ленты и записи на ленту остались без изменения.

Изменения, представленные выше, предназначены для загрузки и выгрузки на диск исходных текстов мелодий. Рассмотрим теперь проблемы, возникающие при компиляции мелодии и записи готового блока кодов, но сначала немного о компиляции. Программа устроена так, что можно скомпилировать музыкальный фрагмент в произвольно заданный адрес. Причем, наблюдательный пользователь наверняка заметил, что выгрузка кодового блока на магнитную ленту происходит необычно на слух: нет привычной паузы между заголовком и самим блоком кодов. Это наводит на мысль, что выгрузка выполняется нестандартной процедурой. Для чего это понадобилось? Выгрузка скомпилированного блока кодов происходит всё время из одного и того же места памяти: из адреса 57000. Хитрость заключается в том, что заголовок файла подменяется другим, в который заносится адрес, заданный Вами при компиляции, а сам массив кодов выгружаются всегда из 57000 с уже изменённой адресацией под заданный Вами адрес. Запись скомпилированного музыкального фрагмента выполняется в строке 8210 при помощи процедуры в кодах 56700. Вот эта программа.

```
56700:      LD    IX,56800
  .         LD    DE,00017
  .         SUB   A
  .         SCF
  .         CALL  1222
  .         NOP
  .         LD    IX,57000
  .         LD    DE,Длина
  .         LD    A,255
  .         SCF
  .         CALL  1222
  .         EI
56727:      RET
```

Как видим, запись производится в два этапа. Сначала записывается блок кодов, имитирующий заголовок кодового файла. Этот блок расположен с адреса 56800 и имеет стандартную для заголовка длину 17 байтов. При компиляции туда заносятся заданный стартовый адрес и получившаяся длина кодового блока. Затем выполняется вторая часть программы - запись самого блока кодов, как уже говорилось, из адреса 57000 (длина тоже заносится при компиляции).

Самый простой приём при адаптации под диск может быть следующий. Запись выполнить при помощи Бейсика, обычной командой TR-DOS, из адреса 57000. Это выполняют строки 8230 ... 8260.

Естественно, в этом случае в заголовках блоков, скомпилированных в разные адреса, будет всегда стартовый адрес 57000. Загружать записанный таким образом

скомпилированный блок надо, обязательно указывая адрес загрузки, тот который Вы задали при компиляции. Поэтому, если Вы завтра забудете адрес, для которого сегодня скомпилировали мелодию, то, скорее всего, Вы уже никогда не сумеете воспользоваться этим блоком кодов.

Поэтому, для устранения этого недостатка, процедура записи дополняется небольшой программой в машинных кодах, которая, будучи выполненной сразу же после записи файла, "подменяет" заголовок на диске на новый, с заданным Вами стартовым адресом.

Расположена эта программа следом за процедурой записи, с адреса 56728 (#DD96):

```
#DD98     0E0A       LD    C,#0A
#DD9A     CD133D     CALL  #3D13
#DD9D     79         LD    A,C
#DD9E     2AEDDD     LD    HL,(#DDED)
#DDA1     22E65C     LD    (#5CE6),HL
#DDA4     0E09       LD    C,#09
#DDA6     CD133D     CALL  #3D13
#DDA9     C9         RET
```

Для того, чтобы подробнее разобраться, как она действует, откройте ZX-РЕВЮ № 1-2 за этот год на стр. 25 и вспомните, как выполняются программы TR-DOS из машинного кода: в регистр C заносится код команды и вызывается подпрограмма 15636 (3D13H).

Программа работает так. Вначале в регистр C заносится код 10, что соответствует процедуре поиска файла в каталоге диска. Параметры для поиска - спецификация файла, то есть его имя и тип. Эти данные уже заданы - они остались в системных ячейках после записи файла. После завершения поиска, результат будет в регистре C. Следующей будет выполнена процедура записи 16 байтов заголовка с изменённым параметром START на место старого. В регистре A задаётся номер заголовка (из регистра C), согласно требованиям к последующей команде при C=9 - записи заголовка файла. Теперь надо то значение адреса, которое было задано при компиляции, занести (в двухбайтном виде) в область спецификации файла в таблице системных переменных TR-DOS, а именно, в ячейки 23782, 23783 (5CE6H, 5CE7H). Для этого вспомним, что стартовый адрес уже занесён при компиляции в 17-байтный заголовок магнитофонного файла. Два его байта находятся по адресу 56813 (DDEDH). Поэтому берём его оттуда и при помощи регистра HL переписываем в системные переменные TR-DOS. Далее осталось только выполнять подпрограмму 15635 (3D13H) с параметром C=9. Теперь заголовок переписан с тем значением адреса, которое было задано при компиляции и проблем с адресом запуска больше не будет. Практически, при записи такой процедурой, процесс записи внешне никак не отличается от традиционного.

Рассмотренная процедура выполняется в строке 8270. Строка 8300 задаёт параметры: стартовый адрес и длину для верификации. Последняя осуществляется блоком строк, начиная с 250.

Надо сказать, что показанный приём может применяться и в других Ваших программах. Блок кодов, переписывающий заголовок, может использоваться, например, при желании наоборот, засекретить истинное значение стартового адреса или с иными целями. Сама записывающая процедура может работать в любом месте памяти.

Как "бороться" с COPY.

От некоторых пользователей, в основном, начинающих, можно услышать претензии по поводу того, что после нажатия "7" в титульном меню, нормально выводится на экран страничка - подсказка ("HELP - PAGE"), но после этого "программа зависает". HELP - PAGE выводится при помощи блока в кодах, значит в нём причина "зависания"? Тогда устранить её будет сложно! Но на самом деле всё очень просто.

"Зависание" происходит на команде COPY (расположенной в старом варианте "WHAM" в строке 50 Бейсик - программы) на тех компьютерах, где имеется интерфейс для принтера LPRINT - 3 с собственным теневым ПЗУ. Оно активизируется, когда компьютер пытается выполнить команду COPY. Но он не может её выполнить, так как предварительно должен быть задан режим для получения графической копии экрана командами:

```
LPRINT CHR$ 0: PRINT CHR$ 1
   ...    ...
LPRINT CHR$ 0: PRINT CHR$ 6
```

Выходов из этого положения два: сложный и простой. Сначала - сложный (но он пока не реализован в Бейсик - файле "m. box"). Для этого надо организовать команду, инициализирующую интерфейс принтера в одной из начальных строк, чтобы она выполнялась при старте программы. Причём команда эта по возможности должна сама определять, инициализирован интерфейс принтера или нет. Вот так это можно выполнить для LPRINT-3:

```
IF PEEK VAL "23296"<>VAL "95" THEN LPRINT: LPRINT CHR$ NOT PI; CHR$ VAL "5"
GO TO VAL "10"
```

Контроль (упрощённый) того, что интерфейс инициализирован, выполняется по наличию в буфере принтера (с адреса 23296) драйвера печати, перебрасываемого туда из теневого ПЗУ при инициализации интерфейса LPRINT-3.

Но надо сказать, что начальная инициализация поможет правильно распечатать HELP - PAGE на принтере, но не поможет "бороться с COPY", если у Вас выключен или отсоединён принтер. Ну действительно, зачем всё время выполнять графическую копию экрана, может Вы хотите просто посмотреть страничку - подсказку и ничего больше. Можно, конечно, организовать запрос, например, "НАЖМИТЕ [P] ДЛЯ ПЕЧАТИ НА ПРИНТЕРЕ". Но гораздо изящнее всё это выглядит, когда выполняется автоматически. Помните о мелочах!

Попробуйте так:

```
75 IF INKEY$="1" THEN CLS: PRINT: RANDOMIZE USR VAL "55420"
76 IF kp=N7 THEN BORDER N7: PAPER N7: INK 0: CLS: PRINT: RANDOMIZE USR VAL "55420"
77 IF IN 123<>255 THEN COPY
78 PAUSE 0: RUN
```

Команда RANDOMIZE USR 55420 в строке 75 (в старом файле "M. BOX" это происходит в строке 50) выводит на экране HELP - PAGE, после чего происходит контроль, включён ли принтер и готов ли он к печати. Если готовность принтера есть, тогда IN 123 даст значение 127 (выключится 7 бит) и будет выполнена распечатка графической копии экрана. Таким образом, если Ваш принтер выключен, то строка 77 никак не будет влиять на работу программы, но как только Вам понадобится "твёрдая копия", достаточно будет всего лишь включить принтер.

Но даже такой "мудрый" подход к команде COPY проблему решает только отчасти, так как невозможно предусмотреть заранее все многообразие интерфейсов принтеров, имеющих хождение в стране. Не у всех же LPRINT-3. Да и вряд ли имеет смысл тратить много сил и компьютерной памяти для решения проблемы в рамках этой программы. Ну в самом деле? Это же не текстовый редактор, где печать является определяющим параметром. Поэтому, предложу теперь второй выход - простой, но подходящий на все случаи жизни: вообще исключить команду COPY из программы. Она (программа) не сильно пострадает от этого. Кстати, COPY встречается в программе ещё в одном месте: в старом варианте "WHAM" это строка 8210. Там по замыслу авторов должны выдаваться на печать параметры скомпилированного музыкального фрагмента. COPY надо удалить и в этой строке, ограничившись выводом данных только на экран.

Компиляция.

Теперь поговорим подробнее о самой компиляции. В принципе, при компиляции возможно задание любого адреса. Но определённые ограничения вносит строка 8160. В старом варианте "WHAM" она запрещает задавать адрес меньше 32768. но если Вы захотите скомпилировать мелодию, скажем, для размещения её в нулевой Бейсик-строке или в дисплейном файле, то Вы не сможете это сделать. Для чего авторами программы внесено такое ограничение, наверное, уже никогда не узнать. Но Вы можете устранить этот дефект простой заменой числа в строке 8160.

Кроме того, наверное ограничение накладывает строка 8161 старого "M. BOX". Если мелодия расположена близко к концу памяти (65535), то заданный адрес может оказаться

недоступен, хотя при полученной реальной длине Вы можете рассчитать, что скомпилированный блок кодов поместился бы до конца памяти. Для устранения этого надо также изменить числовые параметры в строке 8161 (см. листинг "m. box").

Теперь будет возможно разместить мелодию в любом месте ОЗУ, и только адреса ПЗУ для компиляции будут недоступны. Причём, если задать заведомо большой адрес, например 65500, то программа после вывода сообщения о недопустимой длине, сама рассчитает и предложит Вам такое предельное значение адреса, которое позволит Вам расположить скомпилированный блок вплотную к концу ОЗУ (это опять, казалось бы, мелочь, но как приятно работать с такой дружелюбной программой!).

Ещё один момент, связанный с компиляцией, на который хочется обратить внимание пользователя. В строке 7500 старого варианта "WHAM" есть такой фрагмент:

```
... FOR Z=VAL "23296" TO VAL "23308": READ S: POKE Z, S: NEXT Z: RANDOMIZE USR VAL "23296" ...
```

Здесь создаётся вспомогательный кодовый блок, необходимый для компиляции.

С одной стороны мы говорили о корректной работе интерфейса принтера, а с другой - в драйвер печати, находящийся в буфере принтера, вносятся изменения, разрушающие его… Если Вы решите убрать команду COPY из программы и отказаться от вывода на печать, то не произойдёт ничего плохого. Но если Вы всё же сохраните печать (как это было показано выше для интерфейса LPRINT-3), то приведённый фрагмент будет систематически нарушать процесс печати. Вспомним, что печать (COPY) используется при компиляции, как раз после того, как содержимое буфера принтера испорчено приведённым выше фрагментом.

Хотя в предлагаемом варианте "WHAM" команда COPY исключена, всё же зарезервирована возможность для дальнейшего усовершенствования, пока не реализованного - распечатки полного текста на принтере. Поэтому для вспомогательного кодового фрагмента найдено другое место. Так как его работа временная, то есть возможность расположить его в экранной памяти, прямо в дисплейном файле, на том месте, где нет никакой информации. А для того, чтобы не "портить" изображение, надо предварительно "замаскировать" его, напечатав на этом месте пробелы с одинаковым белым цветом INK и PAPER, включив эту печать, например, в строку 8000. Здесь печатается 13 пробелов для "маскировки" блока кодов, который будет создан на этом месте строкой 7500. В последней - соответствующим образом изменены значения адресов.

Теперь вспомогательный блок кодов создаётся прямо на экране, но для пользователя это происходит совершенно незаметно.

Другим недостатком процесса компиляции является такой момент. После завершения её, на экран выводятся комментарии по поводу ячеек, в которых можно задать темп мелодии и цвет бордюра. При запуске скомпилированного блока кодов Вы увидите, что и темп и цвет бордюра не совпадает с тем, что были заданы Вами в редакторе перед компиляцией. Поэтому придётся ещё неоднократно вносить изменения в кодовый блок, добиваясь требуемого темпа воспроизведения.

Этот недостаток можно легко устранить, если процесс компиляции дополнить специальной Бейсик - строкой. Параметры, темп и цвет бордюра хранятся в памяти:

темп - в ячейке 52860,

а цвет бордюра - в 52851

и достаточно всего лишь перенести их в скомпилированный блок кодов. Но при этом надо учесть один момент. Перед занесением параметра, определяющего темп, надо внести определённые коррективы из-за разницы во времени работы воспроизводящих процедур. Экспериментальным путём подобрана величина смещения - заносится значение на 4 меньше, чем в ячейке 52850 (строка 8080).

Теперь после завершения компиляции Вам не потребуется вносить изменения - блок сразу же готов к работе.

Кодировка мелодии.

Исходный (нескомпилированный) текст мелодии, которая в данный момент редактируется, располагается с адреса 60000. Это - рабочая область. Общая длина массива - 2000 байтов. При сохранении мелодии в памяти этот массив перебрасывается в буфер, находящийся в старом "WHAM" с адреса 30000 длиной 12 Кб, а в новом - с 32000, длиной 10 Кб. Рабочая область с 60000 поделена на две равные части по 1Кб. С адреса 60000 расположены коды первого канала, с адреса 61000 - второго канала. Изначально область, где расположен текст мелодии, заполнена кодом 41. Этот код для текста мелодии означает отсутствие ноты или "пауза" - то есть то же, что "пробел" для обычного текста.

В ячейке 60000 находится код, определяющий скорость воспроизведения мелодии. Коды вводимых нот будут располагаться начиная с адреса 60001. Для второго канала - с 61001.

Для обозначения нот используются коды (в порядке повышения тона) с 244 по 255 и далее с 0 по 40. Самая низкая нота - "до" 1-ой октавы имеет код 244. Следующая за ней "до - диез" 1-ой октавы - код 245 и т. д. Код 255 соответствует ноте "си" 1-ой октавы. Следующая нота - "до" 2-ой октавы имеет код 0. А самая высокая нота - "ми" 5 октавы (клавиша SYMB. SH. в режиме 4 октавы) имеет код 40.

Коды с 42 по 239 для нот не используется, они соответствуют тем звуковым эффектам, которые можно реализовать в программе: имитируют звуки барабана, ударника, и т.д. Вставляя шумовые эффекты в мелодию, надо учитывать, что при этом в память заносятся два числа: в области памяти для 1 и для 2 каналов. Пара этих чисел определяет звучание эффекта. Этим объясняется их многообразие. Кстати, здесь кроется причина того, что поставив музыкальный эффект в каком-то месте, пользователь с удивлением обнаруживает, что удалить его он не может. Для нот удаление выполняется клавишей ENTER (при этом в память заносится код 41). Для звуковых эффектов это не проходит, так как эффект определяется не одним числом, как нота, а двумя. Поэтому для того, чтобы удалить эффект, надо нажать ENTER в одном канале, затем вернуться на 1 шаг назад (клавиша "0"), переключить канал клавишей "T" и еще раз нажать ENTER, только тогда звуковой эффект будет стерт.

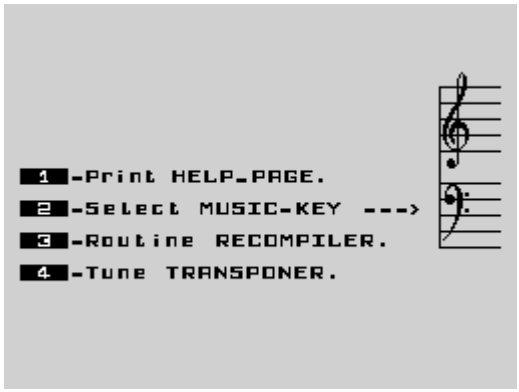
При завершении мелодии, перед тем, как ее скомпилировать, необходимо "зациклить" ее, установив в обоих каналах маркер "конец мелодии". Кстати, он может быть различным для каждого канала, например, 1-ый канал является солирующим и имеет максимальную длину, а 2-ой канал, сопровождая мелодию, все время повторяет одну и ту же короткую последовательность нот, с периодом в несколько тактов.

"Концу мелодии" соответствует код 64. Он ставится клавишей "W". Не все знают о том, что маркер "конец мелодии" легко можно удалить. Это выполняется так. Надо подойти к концу и остановиться на самом последнем такте мелодии, затем нажать ENTER. При этом в память будет занесена "пауза" на то место, где был маркер "конец мелодии". Код 64 просто заменится кодом 41. Это выполняется в каждом канале отдельно.

Теперь мы подошли к наиболее интересной части, о которой говорилось выше.

Дополнительные функции "WHAM".

Они реализованы через дополнительное системное меню "SYST.MENU", вход в которое происходит теперь из титульного меню при нажатии клавиши "7" (в старом "WHAM" - это соответствовало "HELP PAGE"). При нажатии "7" программа попадает на строку 50. Вид системного меню представлен на рис 1.



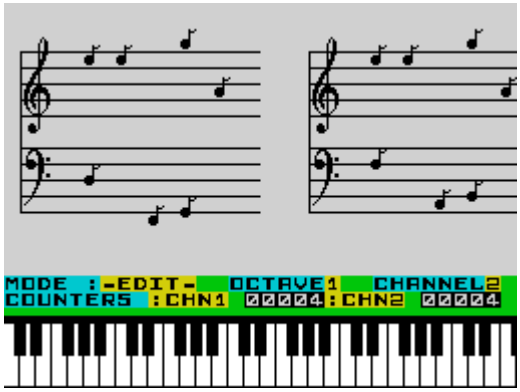
При нажатии "1" выводится страничка-подсказка, аналогично тому, как это выполнялось в старом "WHAM". При нажатии "2" происходит смена нотного ключа на нижнем нотном стане. Этот момент требует комментария.

Нотный стан.

Динамический диапазон "WHAM" охватывает четыре полных октавы с первой по четвертую (и частично захватывает пятую). Эти октавы относятся к скрипичным, басовые октавы в "WHAM" не используются.

Тем не менее, верхний нотный стан - верхние пять линеек - предназначен для обозначения нот в скрипичном ключе, а нижний - в басовом. Поэтому Вы можете заметить, что, например, нота "соль" 3-ей октавы, находящаяся на предпоследней линии верхнего нотного стана, не соответствует таким образом ноте "соль", для 1 октавы, находящейся на последней линейке на низшем нотном стане. Но учитывая то, что нижний нотный стан представлен в басовом ключе, все выглядит верно.

Это может оказаться неудобным если, например, надо набрать с нот мелодию, написанную так, что и верхний и нижний нотные станы находятся в скрипичном ключе. Поэтому добавлена новая функция - переключение нотного ключа на нижнем нотном стане. Теперь при переходе к редактированию мелодии (например, "6" в титульном меню), на экран выводится изображение нотных ключей. Нижний нотный стан формируется теперь тоже в соответствии с нотным ключом: в зависимости от последнего, с разницей в одну линейку. Это поясняет рисунок 2.



Переключение выполняется при помощи подпрограммы GO SUB 600. Формирование изображения нотных ключей производится подпрограммой в машинных кодах, выполняемой в строке 55.

В соответствии с типом нижнего нотного ключа, вход в подпрограмму печати ключей должен происходить с разных точек входа: для басового ключа - 56746 (DDAAH), для скрипичного - 56754 (DDB2H).

Для изображения ключей применен специальный символьный набор, а точнее, 23 дополнительных символа, соответствующие кодам с 33 по 55 включительно (символы с "!" по "7"). Их образы расположены на свободном месте с адреса 56816 по 56999 включительно. Изображение скрипичного ключа, например, складывается из 6 строк по 2 знакоместа. "Текст", изображающий нотные ключи, состоит из трех фрагментов, расположенных по адресам 65184 (FEA0H), 65218 (FEC2H) и 65264 (FEF0H), то есть по адрес 65299 и таким образом в новом варианте включен в файл "Wham 1"CODE.

Подпрограмма печати нотных ключей переключает CHARS на этот новый символьный набор, производит печать нотных ключей, выводя на экран строку текста, "вырисовывающую" нотный ключ, а затем возвращает прежнее значение CHARS.

Эта процедура расположена с адреса 56746 (DDAAH), следом за приведенной выше процедурой записи на диск заголовка файла.

```
DDAA  11A0FE   LD   DE,#FEA0
DDAD  012200   LD   BC,#0022
DDB0  1806     JR   #DDB8
DDB2  11C2FE   LD   DE,$FEC2
DDB5  012E00   LD   BC,#002E
DDB8  2A365C   LD   HL,(#5C36)
DDBB  E5       PUSH HL
DDBC  21E8DC   LD   HL,#DCE8
DDBF  22365C   LD   (#5C36),HL
DDC2  CD3C20   CALL #203C
DDC5  11F0FE   LD   DE,#FEF0
DDC8  012400   LD   BC,#0024
DDCB  CD3C20   CALL #203C
DDCE  E1       POP  HL
DDCF  22365C   LD   (#5C36),HL
DDD2  C9       RET
```

Подпрограмма на Бейсике GO SUB 600 в зависимости от типа ключа вносит изменения в формирователь нижнего нотного стана: для скрипичного ключа нотные линейки должны быть сдвинуты на 1 линию вниз по сравнению с басовым. Кодовый блок программы корректируется при помощи POKE.

Рекомпиляции.

Это очень мощная новая возможность программы "WHAM". Но сначала некоторые дополнительные сведения о компиляции.

При компиляции мелодии происходит следующее. Сначала подготавливается воспроизводящая процедура. Надо сказать, что ее длина непостоянна и зависит от исходного текста мелодии, наличия в ней шумовых эффектов. Звучание нот в обоих каналах определяется "нотной" процедурой, а шумового эффекта - "шумовой" процедурой. Если шумовые эффекты отсутствуют, длина воспроизводящей только "нотной" процедуры будет равна 267 байт. Если в мелодии используется хоть один шумовой эффект, то длина воспроизводящего блока увеличивается на 198 байт - столько занимает "шумовая" процедура.

Затем к воспроизводящей процедуре пристыковывается нотная последовательность - берется фактическая длина исходного текста мелодии, отмеченная в рабочей области маркером "конец мелодии" для каждого голоса, формируется скомпилированный блок, как Вы уже знаете, с адреса 57000. В полученном блоке кодов изменяется адресация машиннокодовых команд переходов с абсолютной адресацией. Так выполняется "привязка" к тому адресу, который Вы задали при компиляции. Получается фрагмент кодов, состоящий из двух частей: "воспроизводящая процедура" и "нотная последовательность". Первая же команда воспроизводящей процедуры имеет вид:

```
LD HL,ADDR
```

где ADDR - адрес, с которого начинается в памяти нотная последовательность.

Исходя из этих данных, сделана подпрограмма, которая выполняет операцию, обратную компиляции, то есть выделяет из скомпилированного блока кодов исходный текст мелодии. Это очень часто нужно на практике. Например, Вы озвучили свою программу мелодией собственного сочинения, А потом захотели внести какие-то изменения в эту мелодию. Если Вы не сохранили исходный текст мелодии, то для выполнения этой задачи Вам пришлось бы набирать всю мелодию заново. Наличие подпрограммы рекомпиляции позволит Вам не сохранять все исходные тексты мелодий, а в любой момент получить из скомпилированного блока кодов полноценный исходный текст, доступный для дальнейшего

редактирования.

Вызов подпрограммы рекомпиляции происходит при нажатии "3" в системном меню. При этом программа переходит на строку 9000. После выбора устройства (магнитофон или дисковод) происходит загрузка скомпилированного блока кодов. Она выполняется в адрес 57000, то есть туда, где как раз готовился к выгрузке скомпилированный блок.

Теперь необходимо определить стартовый адрес STR, для которого блок был скомпилирован. В случае диска, эти данные можно взять из системных переменных TR-DOS, где они остались после загрузки (строка 9020). В случае загрузки с магнитофона, стартовый адрес можно прочитать в заголовке загружаемого файла. Он (заголовок) считывается во временную рабочую область памяти. Сразу же после завершения загрузки, пока временная рабочая область еще не уничтожена, его можно прочитать при помощи PEEK. Это выполняет строка 9030. В этой строке не используется прием по сокращению памяти типа VAL "число", так как рассчет значения числа приводит к изменениям во временной рабочей области и преждевременному разрушению информации о считанном заголовке.

Строки 9040... 9050 при помощи вспомогательного блока кодов, формируемого в дисплейном файле, производят очистку области памяти с 60000 длиной 2000 - где располагается текущая мелодия. Сюда будет выполняться рекомпиляция. Строки 9070-9090 выполняют восстановление темпа воспроизведения. Правда, надо опять, как при компиляции, учитывать разницу во время работы машиннокодовых процедур, воспроизводящих мелодию.

Строка 9100 рассчитывает адрес, с которого располагается нотная последовательность. Коды, расположенные с этого адреса, надо переписать в рабочую область с адреса 60001. Это выполняется при помощи подпрограммы GO SUB 9200. Если встречается код 64, соответствующий маркеру "конец мелодии", подпрограмма 9200 заканчивает работу. Адрес расположения маркера "конец мелодии" запоминается в переменной M1 (строка 9110). Строки 9120-9130 выполняют аналогичные действия для второго канала. Адрес маркера конца - в переменной M2.

После звукового сигнала, свидетельствующего о завершении рекомпиляции (строка 9140), в строке 9150 следует запрос: надо ли уничтожить маркеры "конец мелодии". Это удобно, если Вы собираетесь продолжить мелодию, сделать ее длиннее, чем она была. В этом случае нажмите клавишу "Y", иначе - любую другую (если Вам надо всего лишь скорректировать некоторые ноты, то маркеры удалять незачем).

На этом подпрограмма рекомпиляции заканчивает свою работу и происходит возврат к титульному меню. Теперь восстановленный исходный текст мелодии находится в рабочей области. Его можно обрабатывать дальше, как обычно: редактировать, прослушать, сохранить и т.д.

Транспонирование мелодии.

В эту подпрограмму можно попасть при нажатии "4" в системном меню. Это очень удобный режим, если надо изменить тональность или перезадать мелодию для другой октавы.

Программа транспонирования расположена со строки 5000. Прежде всего необходимо ответить на запрос, в какую сторону требуется изменить тональность: вниз или вверх. Это реализуется в строках 5010...5030.

Следующий запрос (строки 5040-5060) - для какого канала выполнить транспонирование: 1 - только для первого; 2 - только для второго; 3 - для обоих каналов.

В строке 5070 идет последний запрос: на какую величину произвести смещение. Минимальная величина - 1 тон. Если надо изменить тональность на 1 октаву, то величина смещения должна быть равна 12.

Работа начинается в строке 5100 и выполняется в два этапа: вначале - проверка, затем - собственно замена кодов. В строке 5100 обнуляется флаг режима - переменная V. Если V=0, то проверка, если V=1, то замена кодов. При V=0 происходит предварительный просмотр всего текста мелодии от начала до конца - возможно ли вообще осуществить транспонирование - не выйдет ли результирующая мелодия за пределы динамического

диапазона "WHAM". Это выполняется подпрограммой GO SUB 5200 для двух каналов - соответственно в областях памяти с адреса 60001 и с 61001 в зависимости от заданных каналов (строки 5110-5180). Если в результате проверки выясняется, что должен быть получен тон, недоступный для "WHAM", слишком высокий или слишком низкий, то процесс проверки прерывается с выдачей сообщения о невозможности транспонирования. Это выполняется переходом на строку 5300. После чего - возврат к титульному меню. А если проверка доведена до конца, то в строке 5130 устанавливается флаг V=1, что означает "транспонирование возможно". Можно переходить к замене кодов. Это выполняется путем возврата на строку 5110. Здесь процесс вызова подпрограммы GO SUB 5200 повторяется вторым циклом, но уже со значением флага V=1. В этом случае строка 5260 производит замену кодов. Выход из подпрограммы 5200 по достижении маркера "конец мелодии" или, при его отсутствии, по достижении последнего адреса рабочей области.

Строка 5140 подводит итог транспонирования, после чего - возврат в титульное меню. Дальше - обычная работа с полученной мелодией.

Надо сказать, что работа подпрограмм рекомпиляции и транспонирования, организованная в Бейсике, не отличается быстродействием, так что, выполняя их, наберитесь терпения. "Неторопливость" подпрограмм усугубляется внесением элементов экономии памяти: VAL "число" и заменой чисел переменными. Но учитывая, что этими возможностями не каждую минуту придется пользоваться, с этим можно смириться. Лучше было бы, конечно, построить такие процедуры в машинных кодах, тем более, что это не так уж сложно и вполне доступно начинающим. Попробуйте, если хотите, заменить Бейсик машинным кодом. Я же - ограничусь идеей.

Сокращение программы.

Как известно, после загрузки программы мы имеем в памяти шесть демонстрационных мелодий. Кроме того, непосредственно в памяти может находиться седьмая. Это удобно, когда Вы в первый раз загружаете программу - можно послушать их, посмотреть, как задаются ноты и эффекты. Но когда Вы освоитесь и будете заниматься самостоятельным набором мелодий для оформления своих собственных программ, демонстрационные мелодии не понадобятся. Вы все равно будете удалять их, записывая в память на их место свои заготовки. Каждая мелодия занимает в памяти по 2 Кб. Исключив их из загрузки, можно сократить программу на 12 Кб. А если еще "выбросить" цветную заставку (около 7 Кб) то получится ощутимая экономия.

Владельцы магнитофонных версий, представьте, во что выльется сокращение времени при каждой загрузке! Здесь речь идет не об урезании программы вообще, а о "персональной копии" - специальной версии, предназначенной для быстрой загрузки. И не об удалении демонстрационных мелодий вообще, их надо сохранить, но отдельно, на ленте (или дискете), чтобы всегда можно было при желании загрузить опять. Для этого надо каждую мелодию сначала загрузить из буферной памяти в рабочую область редактора, а затем сохранить на ленте (или диске). После того, как демонстрационные мелодии надежно сохранены, можно приступить к модификации кодового блока "WHAM!2" CODE. Начиная с адреса 30000, длиной по 2 Кб, здесь расположено 6 мелодий. Итого - по адрес 41999 включительно. Можно сделать так:

```
LOAD "WHAM!2"CODE 30000,15000
SAVE "wham 2"CODE 42000,3000
```

Осталось только дополнить загрузчик небольшой подпрограммой, которая будет инициализировать область с 32000 (в старом варианте это надо было делать с 30000), заполняя ее кодом 41 -"пауза". Вот эта подпрограммка:

```
LD    HL,32000
LD    (HL),041
LD    DE,32001
LD    BC,09999
LDIR
RET
```

Для ее формирования и запуска универсальный Бейсик-загрузчик "WHAM", приведенный в начале статьи, надо дополнить следующими строками:

```
102 FOR a=20480 TO 20493: READ b: POKE a,b: NEXT a
104 RANDOMIZE USR 20480
106 DATA 33,0,125,54,41,17,1,125,1,15,39,237,176,201
```

Кроме того, надо очистить (а точнее, заполнить пробелами или, например, словом: "EMPTY") ту область, где хранятся названия демонстрационных мелодий. Это область с адреса 65300, по 10 байтов на каждое название. Эти исправления надо сделать в файле "WHAM!3"CODE.

Для устранения загрузки картинки, из загрузчика можно удалять строку 30 и из строки 25 исключить:

```
... LOAD ""SCREEN $: ...
```

и перекомпоновать заново файлы для магнитофонного варианта.

Теперь, когда описание программы и всех ее новых функций завершено, переходим к тем изменениям, которые надо выполнить в кодовых блоках.

Изменение кодовых блоков.

Ниже приводится программа на Бейсике, выполняющая все необходимые изменения в кодовых блоках "WHAM!1"CODE и "WHAM!2"CODE. По завершении она выдает новые файлы "wham 1"CODE и "wham 3"CODE.

Программа предназначена для работы с магнитофоном. Владельцы "Спектрумов" с БЕТА-диском, наверняка сами сумеют внести необходимые коррективы.

```
10 CLEAR 31999: POKE 23675,88: POKE 23676,255
20 LOAD "WHAM!1"CODE
30 LOAD "WHAM!3"CODE
40 POKE 50908,159: POKE 50909,79: POKE 50921,223: POKE 50922,71
50 POKE 51248,0: POKE 51253,170: POKE 51254,221
60 FOR A=51374 TO 51406: READ B: POKE A,B: NEXT A
70 DATA 022,017,001,127,049,057,056,053,032,077,065,082,075,032,084,073,077,069,043,127,049,
      057,057,051,032,065,076,065,078,083,079,070,064
80 FOR A=51495 TO 51527: READ B: POKE A,B: NEXT A
90 DATA 083,089,083,084,046,077,069,078,085,019,000,022,015,007,032,080,082,069,083,083,032,
      067,079,078,084,082,079,076,032,075,069,089,032
100 FOR A=56728 TO 56786: READ B: POKE A,B: NEXT A
110 DATA 014,010,205,019,061,121,042,237,221,034,230,092,014,009,205,019,061,201,017,160,
      254,001,034,000,024,006,017,194,254,001,046,000
120 DATA 042,054,092,229,033,232,220,034,054,092,205,060,032,017,240,254,001,036,000,205,060
      ,032,225,034,054,092,201
130 FOR A=56816 TO 56999: READ B: POKE A,B: NEXT A
140 DATA 000,000,000,000,000,000,000,255,016,040,044,076,076,076,076,255,128,128,128,128,
      128,128,128,255,092,088,088,120,112,112,096,255
150 DATA 128,129,131,131,135,134,142,255,224,192,192,064,064,064,064,255,156,153,187,187,
      183,183,182,255,224,248,252,252,078,070,070,255
160 DATA 182,179,147,153,136,132,131,255,070,070,068,068,072,080,224,255,128,128,128,134,
      143,143,142,135,064,054,064,064,064,064,128,000
170 DATA 000,000,000,006,015,015,014,007,128,128,128,128,128,128,128,128,135,140,136,144,
      144,158,159,255,128,192,224,096,118,118,112,255
180 DATA 159,142,128,128,128,128,128,255,112,112,112,112,102,102,096,255,128,128,128,129,
      129,131,130,255,192,192,128,128,000,000,000,255
190 DATA 132,136,144,160,128,128,128,255,000,060,064,124,006,056,060,000,000,126,002,004,
      008,016,016,000
200 FOR A=USR "A" TO USR "B"+7: READ B: POKE A,B: NEXT A
210 DATA 008,120,248,248,248,112,000,000,000,000,000,009,014,008,008,008
220 FOR A=USR "E" TO USR "F"+7: READ B: POKE A,B: NEXT A
230 DATA 248,112,000,000,000,000,000,000,009,014,008,008,008,120,248,248
240 FOR A=65184 TO 65299: READ B: POKE A,B: NEXT A
250 DATA 016,000,017,007,022,008,027,035,033,033,022,009,027,047,048,033,022,010,027,049,
      050,033,022,011,027,051,052,033,022,012,027,053
260 DATA 033,033,016,000,017,007,022,008,027,046,128,128,022,009,027,035,034,033,022,010,
      027,035,036,033,022,011,027,037,038,033,022,012
```

```
270 DATA 027,039,040,033,022,013,027,041,042,033,022,014,027,045,044,128,022,002,027,033,
      034,033,022,003,027,035,036,033,022,004,027,037
280 DATA 038,033,022,005,027,039,040,033,022,006,027,041,042,033,022,007,027,043,044,128
290 FOR N=1 TO 6: RESTORE 300: FOR M=1 TO 10: READ B: POKE (65300+10*(N-1)+M-1),B: NEXT M:
      NEXT N
300 DATA 045,045,069,077,080,084,089,045,045,032
400 SAVE"wham 1"CODE 65184,352
410 SAVE"wham 3"CODE 50000,7000
```

Цифры в строках DATA напечатаны одно под другим только для упрощения "читаемости" программы. Вы можете набирать их так, как Вам удобно (например, вместо "002" разумеется, можно набрать "2").

На этом мы заканчиваем изложение некоторых вопросов, связанных с музыкальным редактором "WHAM", но ни в коем случае не заканчиваем музыкальную тематику. Наоборот, планируем всячески ее развивать в дальнейшем. В первом номере "ZX-РЕВЮ" будущего года мы продолжим рассказ о музыкальных редакторах. Будет рассмотрен другой вариант редактора "WHAM" - рассчитанный на работу с трехканальным музыкальным сопроцессором AY-8910 (8912). А затем, в дальнейшем, программы "SOUND-TRACER" и "SONGCOMPILER", также рассчитанные на музыкальный сопроцессор. С их помощью можно создавать звуковое сопровождение такого высокого качества и широкой палитры звучания, что оно становится практически соизмеримо с музыкой, которую мы слышим по радио.

ВОЗВРАЩАЯСЬ К НАПЕЧАТАННОМУ

НЕСТАНДАРТНАЯ ЗАГРУЗКА

В "ZX-РЕВЮ" №1-2 за этот год на стр. 31 мы опубликовали процедуру загрузки с изменением цвета бордюрных полос, взятую из игры "BARBARIAN-3", присланную Комисаровым Павлом из пос. Видово Мурманской обл. В этот раз мы продолжаем рассказ об этой процедуре, о иной нестандартной загрузке, а также связанных с этим некоторых приёмах защиты, пользуясь дополнительным материалом, любезно предоставленным нам нашими корреспондентом.

Павел прислал дополнения к загрузчику, которые позволяют изменить его свойства.

Правда, следует заметить, что, тестируя присланные им листинги, мы получали иногда несколько иные результаты, чем те, о которых пишет Павел. Поэтому публикуемый материал является коллективным творчеством: в основе - то, что прислал наш корреспондент, но с нашими изменениями и дополнениями. Итак, модернизация загрузчика блоков кодов без заголовка.

1. Загрузка кодов, начиная со старших адресов к младшим. Для этого надо сделать такую замену: FE55H - 2B. Это приведёт к тому, что по адресу FE54H вместо команды INC IX получится команда DEC IX. Здесь надо отметить два момента. Во-первых, в вызывающей программе в регистре IX указывается не начальный, а конечный адрес загрузки (начальный адрес плюс длина DE), так как загрузка идёт "наоборот". Другая особенность заключается в том, что, задав, скажем, для загрузки экрана, длину 6912 байтов. Вы обнаружите, что последний байт блока кодов (он будет первым байтом дисплейного файла) не грузиться. Поэтому, при подготовке блока к загрузке, следует задать длину на 1 байт большую, то есть 6913 байтов. Для загрузки экранного файла, например, вызывающая программа, может выглядеть так.

```
LD   IX,#5B00
LD   DE,#1B01
LD   A,#FF
SCF
CALL #FDE8
RET
```

При этом загружаемую "картинку" необходимо сначала "перевернуть" и увеличить её длину на 1 байт, например такой программой на Бейсике:

```
CLS: LOAD "" SCREEN$
FOR a=0 TO 6912: POKE (46912-a), PEEK(16384+a): NEXT a
SAVE "name" CODE 40000, 6913
```

При выполнении записи на ленту в строке 3 пропустите заголовок, а только затем включайте магнитофон, то есть запишите только сам кодовый блок.

При загрузке со старших адресов к младшим, будет слышно, как вначале грузятся атрибуты, а затем - графика.

К сказанному можно добавить такой факт. В своё время нам попалась игра "ROLLER COASTER" (ELITE), в которой загрузка велась тоже "наоборот". Причём запись была отвратительного качества и едва читалась. Эта программа была переписана с магнитофона на магнитофон, так как ни один копировщик не мог её скопировать из-за большой длины. Даже копировщики с компрессированием памяти (типа "TF-COPY"). При более детальном

рассмотрении выяснилось, что загрузка идёт там тоже со старших адресов в младшие, причём начинается с экрана (с атрибутов), затем пробегает область всего ПЗУ, затем, после достижения нулевых значений в счётчике адреса и его переполнения, продолжается с конца памяти и заканчивается буфером принтера. То есть, загружается (точнее говоря, делают вид, что загружаются) все 64K памяти! При этом переписывается и машинный стек, обеспечивая после окончания загрузки и возврата по RET автостарт программы. Сам факт существования ещё "не взломанной" программы уже говорит о высоком качестве защиты.

Однако, такой метод защиты имеет смысл, если Ваша программа большая - использована практически вся память. Иначе сколько же лишнего времени будет занимать загрузка! Это очень "действует на нервы" и резко снижает симпатию к программе. Это тоже надо учитывать.

Однако, возвращаемся к загрузчику от BARBARIANa.

2. При загрузке пилоттоны хаотично изменяют свой цвет, переливаясь всеми цветами радуги. Для получения этого эффекта в загрузчике от BARBARIANa нужно изменить всего одну ячейку: FE90H - 00. Можете также попробовать другой вариант: FE8FH - ED, FE90H - 5F.

3. При загрузке по бордюру проскакивают тонкие короткие разноцветные штрихи. Этот эффект получается, если изменить загрузчик следующим образом:

```
FE8F   FD5F        LD     A,R
FE93   D3FE        OUT    (#FE),A
FE95   E600        AND    #00
       FE97 F608   OR     #06
FE99   D3FE        OUT    (#FE),A
FE9B   37          SCF
FE9C   C9          RET
```

4. Бордюр чёрный, индикация считывания (бордюрные полосы) производится в заданных позициях (знакоместах) экрана. Надо внести следующие изменения:

```
FE32   EE00        XOR    #00
FE79   3E0B        LD     A,#0B
FE8F   3E00        LD     A,#00
FE91   2F          CPL
FE92   3290FE      LD     (#FE90),A
FE95   E606        AND    #06
FE97   EE04        XOR    #04
FE99   F642        OR     #42
FE9B   ...
```

Дальнейшая часть программы зависит от того, в каком месте экрана Вы хотите осуществить вывод бордюрных полос. Например, пусть это будут два знакоместа экрана: левое верхнее и соседнее с ним справа. Адреса ячеек в файле атрибутов, соответствующие этим знакоместам будут: 5800H и 5801H и продолжение программы будет таким:

```
FE9B   320058      LD     (#5800),A
FE9E   320158      LD     (#5801),A
FEA1   37          SCF
FEA2   C9          RET
```

При этом в выбранных знакоместах должен находиться какой-нибудь графический образ. Тогда те пикселы, которые выключены (имеют цвет PAPER) будут иметь чёрный цвет и меняться не будут, а включенные пикселы (имеют цвет INK) будут переливаться красно-жёлтыми полосами, воспроизводя такие же полосы, какие мы видим по бордюру экрана при загрузке. Довольно-таки оригинальный эффект получается: одно и то же знакоместо имеет чёрный фон и, в то же время, на включенных пикселах наблюдаем красные и жёлтые полосы. Вы спросите, как же такое вообще может быть? Ведь "Спектрум" устроен так, что в принципе одно знакоместо не может иметь больше двух цветов: INK и PAPER? Дело здесь в том, что на получающуюся картину накладывается ещё одно явление - строчная развёртка экрана.

Поэтому быстрая смена цвета INK приводит к тому, что во время развёртки кадра одна строка имеет, красный цвет, а другая строка развёртки уже имеет жёлтый цвет, что и создаёт необычный эффект.

Ещё на один оригинальный способ загрузки, применённый в программе "KRAKOUT-II", обратил внимание Павел. Смысл её состоит в том, что блок кодов с одним пилот-тоном может вести загрузку в разные адреса. Например, картинка там загружается так: верхняя треть экрана, затем атрибуты верхней трети, нижняя треть, атрибуты нижней трети, средняя треть, атрибуты средней трети. Программа при этом выглядит так:

```
LD   IX,#4000
LD   DE,#0800
LD   A,#FF
SCF
DI
INC  D
EX   AF,A'F'
DEC  D
CALL #05EA
LD   IX,#5800
LD   DE,#0100
CALL #05A9
RET
```

Это тоже можно несложно использовать, нужно только предварительно переделать загружаемый блок под этот загрузчик (как при загрузке экрана "наоборот"). Например, для того, чтобы загружать экран так: верхняя треть, затем её атрибуты, средняя треть и её атрибуты, нижняя треть и её атрибуты, надо вначале подготовить блок кодов к загрузке при помощи такой программы на Бейсике.

```
1 CLS: LOAD "" CODE 16384
2 FOR a=0 TO 2048: POKE (40000+a),PEEK (16384+a): NEXT a
3 FOR a=0 TO 256:  POKE (42048+a),PEEK (22528+a): NEXT a
4 FOR a=0 TO 2048: POKE (42304+a),PEEK (18432+a): NEXT a
5 FOR a=0 TO 256:  POKE (44352+a),PEEK (22784+a): NEXT a
6 FOR a=0 TO 2048: POKE (44608+a),PEEK (20480+a): NEXT a
7 FOR a=0 TO 256:  POKE (46656+a),PEEK (23040+a): NEXT a
8 SAVE "name" CODE 40000,6912
```

Для загрузки этого блока кодов (без заголовка) может быть использована следующая программа в кодах.

```
AB00  DD210040   LD   IX,#4000
AB04  110008     LD   DE,#0800
AB07  3EFF       LD   A,#FF
AB09  37         SCF
AB0A  F3         DI
AB0B  14         INC  D
AB0C  08         EX   AF,A'F'
AB0D  15         DEC  D
AB0E  CD6A05     CALL #056A
AB11  DD210058   LD   IX,#5800
AB15  110001     LD   DE,#0100
AB18  CDA905     CALL #05A9
AB1B  DD210048   LD   IX,#4800
AB1F  110008     LD   DE,#0800
AB22  CDA905     CALL #05A9
AB25  DD210059   LD   IX,#5900
AB29  110001     LD   DE,#0100
AB2C  CDA905     CALL #05A9
AB2F  DD210050   LD   IX,#5000
AB33  110008     LD   DE,#0800
```

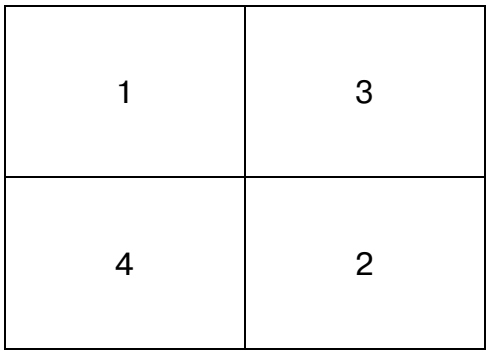
```
AB36   CDAA905     CALL   #05A9
AB39   DD21005A    LD     IX,#5A00
AB3D   110001      LD     DE,#0100
AB40   CDA905      CALL   #05A9
AB43   FB          EI
AB44   D8          RET    C
AB45   CF          RST    8
AB46   1A          DEFB   #1A
```

Для примера, этот блок кодов расположен с адреса AB00H, но он может находиться в любом другом месте.

Если контроль качества считывания не нужен, то замените команду по адресу AB44H на RET (C9H).

Изложенное явление можно использовать не только для загрузки картинки, но и для любого другого массива кодов, причём количество таких дописываемых по отдельности блоков может быть достаточно большим, создавая много хлопот взломщикам.

А теперь предлагаем ещё один способ нестандартной загрузки, присланной Комиссаровым Павлом. Он "вытащил" его из программы "FREDDY-2". Этот загрузчик загружает экранный файл длиной 6912 байт по четвертям экрана в шахматном порядке. Приблизительно это выглядит так:



Причём загрузка изображения в каждую четверть идёт так: Сначала - графика, но не как обычно - линиями с шагом по 8 пикселов, да ещё с делением по третям экрана, а пиксельные линии идут подряд, одна за другой, сверху донизу, картинка разворачивается плавно и непрерывно, пока не заполнится вся четверть экрана. Затем - загрузка атрибутов, но опять, не как обычно - по строкам сверху вниз - а по столбцам, то есть изображение окрашивается слева направо. И так повторяется для каждой четверти.

Для того, чтобы загрузить таким образом картинку, надо её сначала подготовить. Павел предлагает программу на Бейсике, которая это выполняет аналогично тому, как это делалось при загрузке "наоборот".

```
1 CLEAR 39999: LOAD "" CODE 40000, 6912
10 LET na=50000
20 FOR q=1 TO 4: FOR s=0 TO 11: READ adr: PRINT s; CHR$ 143;: FOR k=0 TO 7: FOR a=0 TO 15
30 POKE na, PEEK adr: LET adr=adr+1: LET na=na+1: NEXT a: LET adr=adr+240: NEXT k: NEXT s:
      PRINT: PRINT
40 READ adr: LET ka=adr: FOR a=0 TO 15: FOR b=0 TO 11: POKE na, PEEK adr: LET adr=adr+32:
      LET na=na+1: NEXT b
50 LET adr=ka+1: LET ka=ka+1: NEXT a: NEXT q
60 BEEP .1, 32: SAVE "name" CODE 50000, 6912: STOP
70 DATA 40000,40032,40064,40096,40128,40160,40192,40224,42048,42080,42112,42144,46144
80 DATA 42192,42224,42256,42288,44112,44144,44176,44208,44240,44272,44304,44336,46544
90 DATA 40016,40048,40080,40112,40144,40176,40208,40240,42064,42096,42128,42160,46160
100 DATA 42176,42208,42240,42272,44096,44128,44160,44192,44224,44256,44288,44320,46528
```

После запуска программа ждёт загрузки нормальной картинки, а после загрузки начинается перекодировка. Пока она выполняется, на экран выводятся цифры. По ним можно определить, много ли осталось ждать до завершения работы. Программа должна

вывести четыре строки чисел от 0 до 11.

После этого программа предложит записать полученный блок кодов. При записи пропустите заголовок, и только после него включайте магнитофон.

Загрузка полученного блока кодов без заголовка выполняется при помощи следующего загрузчика:

```
EA60  ED7360EB  LD    (#EB60),SP
EA64  37        SCF
EA65  3E00      LD    A,#00
EA67  FE01      CP    #01
EA69  08        EX    AF,AF'
EA6A  F3        DI
EA6B  3E0F      LD    A,#0F
EA6D  D3FE      OUT   (#FE),A
EA6F  DBFE      IN    A,(#FE)
EA71  1F        RRA
EA72  E620      AND   #20
EA74  F602      OR    #02
EA76  4F        LD    C,A
EA77  BF        CP    A
EA78  C248EB    JP    NZ,#EB48
EA7B  CDE705    CALL  #05E7
EA7E  30F8      JR    NC,#EA78
EA80  211504    LD    HL,#0415
EA83  10FE      DJNZ  #EA83
EA85  2B        DEC   HL
EA86  7C        LD    A,H
EA87  H5        OR    L
EA88  20F9      JR    NZ,#EA83
EA8A  CDE305    CALL  #05E3
EA8D  30E9      JR    NC,#EA78
EA8F  069C      LD    B,#9C
EA91  CDE305    CALL  #05E3
EA94  30E2      JR    NC,#EA78
EA96  3EC6      LD    A,#C6
EA98  B8        CP    B
EA99  30E0      JR    NC,#EA7B
EA9B  24        INC   H
EA9C  20F1      JR    NZ,#EA8F
EA9E  06C9      LD    B,#C9
EAA0  CDE705    CALL  #05E7
EAA3  30D3      JR    NC,#EA78
EAA5  78        LD    A,B
EAA6  FED4      CP    #D4
EAA8  30F4      JR    NC,#EA9E
EAAA  CDE705    CALL  #05E7
EAAD  D248EB    JP    NC,#EB48
EAB0  79        LD    A,C
EAB1  EE03      XOR   #03
EAB3  4F        LD    C,A
EAB4  2600      LD    H,#00
EAB6  06B0      LD    B,#B0
EAB8  CD4EEB    CALL  #EB4E
EABB  118048    LD    DE,#4880
EABE  D5        PUSH  DE
EABF  111040    LD    DE,#4010
EAC2  D5        PUSH  DE
EAC3  119048    LD    DE,#4890
EAC6  D5        PUSH  DE
EAC7  DD210040  LD    IX,#4000
EACB  0604      LD    B,#04
EACD  C5        PUSH  BC
EACE  DDE5      PUSH  IX
EAD0  160C      LD    D,#0C
EAD2  DDE5      PUSH  IX
EAD4  1208      LD    E,#08
EAD6  DDE5      PUSH  IX
EAD8  0610      LD    B,#10
EADA  C5        PUSH  BC
EADB  CD4EEB    CALL  #EB4E
EADE  3068      JR    NC,#EB48
EAE0  C1        POP   BC
EAE1  DD7500    LD    (IX+0),L
EAE4  DD23      INC   IX
EAE6  10F2      DJNZ  #EADA
EAE8  DDE1      POP   IX
EAEA  DD24      INC   I
EAEC  1D        DEC   E
EAED  20E7      JR    NZ,#EAD6
EAEF  DDE1      POP   IX
EAF1  DD7C      LD    A,I
EAF3  E618      AND   #18
EAF5  47        LD    B,A
EAF6  DD7D      LD    A,X
EAF8  E6E0      AND   #E0
EAFA  07        RLCA
EAFB  07        RLCA
EAFC  07        RLCA
EAFD  BC        OR    B
EAFE  3C        INC   A
EAFF  47        LD    B,A
EB00  E618      AND   #18
EB02  F640      OR    #40
EB04  DD67      LD    I,A
EB05  78        LD    A,B
EB07  E607      AND   #07
EB09  0F        RRCA
EB0A  0F        RRCA
EB0B  0F        RRCA
EB0C  47        LD    B,A
EB0D  DD7D      LD    A,X
EB0F  E61F      AND   #1F
EB11  B0        OR    B
EB12  DD6F      LD    X,A
EB14  15        DEC   D
EB15  20BB      JR    NZ,#EAD2
EB17  DDE1      POP   IX
EB19  DD7C      LD    A,I
EB1B  E618      AND   #18
EB1D  0F        RRCA
EB1E  0F        RRCA
EB1F  0F        RRCA
EB20  F658      OR    #58
EB22  DD67      LD    I,A
EB24  0610      LD    B,#10
EB26  C5        PUSH  BC
EB27  060C      LD    B,#0C
EB29  DDE5      PUSH  IX
EB2B  C5        PUSH  BC
EB2C  CD4EEB    CALL  #EB4E
EB2F  3017      JR    NC,#EB48
EB31  DD7500    LD    (IX+0),L
```

```
EB34  012000     LD    BC,#0020
EB37  DD09       ADD   IX,BC
EB39  C1         POP   BC
EB3A  10EF       DJNZ  #EB2B
EB3C  DDE1       POP   IX
EB3E  DD23       INC   IX
EB40  C1         POP   BC
EB41  10E3       DJNZ  #EB26
EB43  C1         POP   BC
EB44  DDE1       POP   IX
EB46  1085       DJNZ  #EACD
EB48  ED7B60EB   LD    SP,(#EB60)
EB4C  FB         EI
EB4D  C9         RET
EB4E  06B4       LD    B,#B4
EB50  2E01       LD    L,#01
EB52  CDE305     CALL  #05E3
EB55  D0         RET   NC
EB56  3ECB       LD    A,#CB
EB58  B8         CP    B
EB59  CB15       RL    L
EB5B  06B0       LD    B,#B0
EB5D  30F3       JR    NC,#EB52
EB5F  C9         RET
```

(C) Р. Купцов, Д. Мальтов, 1993г.

ИСПРАВЛЕННЫЕ ОШИБКИ ПЗУ.

В последних двух номерах прошлого года мы опубликовали обзор зарубежной прессы, обобщив те сведения, которые имеются по ошибкам в стандартном системном ПЗУ "Спектрума".

Работа получила благожелательные отзывы от наших читателей. По-видимому, им важно всё, что связано с ПЗУ, а тем более с его ошибками.

Вместе с тем встал интересный вопрос: "А как обстоит дело в ПЗУ ТУРБО-90?". Результатами своих исследований по этому поводу делятся наши корреспонденты Роман Купцов и Мальтов Денис из г. Новгорода. Тема не закрыта, мы опубликуем и другие взгляды, освещающие этот вопрос, тем более, что интересно бы рассмотреть и свои "собственные" ошибки Турбо-ПЗУ.

КОРР: Мы являемся постоянными читателями "ZX-РЕВЮ" с 1992 года. Ваш журнал выше всяких похвал! Сами мы занимаемся программированием три года и Вы нам в этом очень помогли, особенно в том, что связано с программированием в машинных кодах. У Вас постоянно свежий и хорошо поданный материал, а Ваши статьи по секретам ПЗУ в 1991 г. - просто класс! Неоценимая помощь любому программисту.

ИФК: Спасибо, конечно, друзья на добром слове, но позвольте нам остаться при своём мнении. Статьи эти о ПЗУ в 1991 году были, возможно, полезны, но мы их рассматриваем как СОВЕРШЕННО НЕУДАЧНЫЙ опыт и РЕШИТЕЛЬНО ими НЕДОВОЛЬНЫ. Знаете ведь: "Первый блин... и т.д.". В общем, случился с нами приступ внутренней аллергии. Так бывает, когда хочешь сделать что-то хорошее, а чувствуешь, что получается не то.

Потому мы и прекратили их давать. Мы тогда только начинали и опыта общения с читателями было маловато. Сейчас мы всё сделали бы по-другому. И будьте уверены, сделаем! В наших планах на 2-ой квартал 1994 года стоит книга о системном ПЗУ, которая действительно класс! Мы готовим удобный и понятный способ подачи материала и уверены, что книга будет отличной. Так что позвольте Ваши комплименты принять авансом, а мы постараемся этот аванс отработать по-честному.

. . . . . .

. . . . . .

КОРР: Мы хотим поделиться с Вами и со всеми читателями некоторыми находками из ПЗУ ТУРБО-90. Немного покопавшись в нём, мы обнаружили (не без помощи "ZX-РЕВЮ"), что некоторые из ошибок системного ПЗУ там исправлены.

Исправлено:

1. Процедура обработки немаскированного прерывания INT, начиная с адреса 0066H, т. е.

```
PUSH AF
PUSH HL
LD   HL,(5CB0)
LD   A,H
OR   L
JR   Z,0070H; ошибка исправлена, в фирменном ПЗУ здесь стоит JR NZ,0070H
```

2. Ошибка деления, т. е.:

```
IF 1/2  <> 0.5 THEN PRINT "Ku-Ku".
```

В Турбо-ПЗУ здесь "Ku-Ku" не печатается.

3. Ошибка "-65536"; т.е. PRINT IN "-65536" даёт - 65536.

4. CHR$ 9 работает.

5. CHR$ 8 работает правильно.

6. STR$ работает правильно, т. е.

```
PRINT "Ku-Ku" + STR$ 0.5
```

даёт:

```
Ku-Ku0.5
```

7. Ошибка SCREEN$ исправлена, т.е.

```
IF "X"=SCREEN$(0,0) THEN PRINT "Ku-Ku"
```

Ku-Ku не печатается, когда в текущем знакоместе (0, 0) нет "X"

8. Исправлена ошибка курсора текущей строки: т.е.

```
9000 PRINT\9001\EDIT,
```

где значок "\" означает ENTER, выдаёт на редактирование строку 9000, а значок ">" в нижней части экрана не появляется.

9. Исправлена ошибка ведущего пробела.

Неисправленные ошибки:

1. Особенность регистровой пары H'L' (альтернативной).
2. Ошибка оператора PLOT.
3. Ошибка CLOSE$.
4. Ошибка Scroll?
5. Ошибка K-режима.
6. Ошибка проверки синтаксиса.

Непроверенные ошибки.

1. Ограничение по использованию регистровой пары IY.
2. Особенности пользовательской функции FN.
3. Ошибки кодов управления цветом.
4. Ошибка MOD_DIV (кальк-р).
5. Ошибка E_TO_FP (кальк-р).
6. Ошибка INKEY$ #0.

Adventure Games

"Adventure Building system"

Окончание. Начало см. в предыдущем выпуске.

Чтобы завершить демонстрационную программу, начатую нами в предыдущем номере "ZX-РЕВЮ", загрузите программу с ленты_B (Листинг_4) и добавьте строки: с 1100 по 3315 из Листинга_5. Сохраните готовую демонстрационную игру командой RUN 9990. Скомпонуйте файлы готовой демонстрационной программы в следующем порядке:

Файл "INTRO" (Лист._3)
Файл "adventure" (Лист._4+5)
Файл "system"CODE
Файл "objects"DATA
Последние два файла сгенерированы при помощи головной программы ABS.

Листинг_5.
Текст 2-ой части демонстрационной игры. Продолжение файла "adventure".

```
1100 IF NO>MOB THEN LET R$(1)="ЭТО НЕВОЗМОЖНО!": GO TO 50
1105 IF PEEK (O+NO)=99 THEN LET R$(1)="ПРОВЕРЬТЕ СВОЙ ИНВЕНТАРЬ!": GO TO 50
1110 IF PEEK (O+NO)<>PEEK F THEN LET R$(1)="НАВЕРНОЕ, ЭТО ВАМ ПРИСНИЛОСЬ!": GO TO 50
1115 IF PEEK (F+99)>=MAX THEN LET R$(1)="У ВАС ВЕДЬ ТОЛЬКО ДВЕ РУКИ,": LET R$(2)="ВАМ ВСЕ
     ЭТО НЕ УНЕСТИ.": GO TO 50
1120 IF NO=7 THEN CLS : GO SUB 8800: PRINT AT 2,10;"ПОЗДРАВЛЯЕМ": GO SUB 8800: PRINT AT
     5,3;"ВЫ ЗАКОНЧИЛИ СВОЮ МИССИЮ.": GO SUB 8800: GO TO 1602
1125 POKE (O+NO),99: POKE (F+99),PEEK (F+99)+1: LET R$(1)="O.K." : GO TO 50
1200 IF NO>MOB THEN LET R$(1)="НЕ БУДЕМ ДЕЛАТЬ ГЛУПОСТЕЙ!": GO TO 50
1205 IF PEEK (O+NO)<>99 THEN LET R$(1)="ПРОВЕРЬТЕ СВОЙ ИНВЕНТАРЬ!": GO TO 50
1210 IF NO=9 AND PEEK F<>11 THEN POKE (O+9),O: POKE (O+2),PEEK F: POKE (F+2),O: POKE
     (F+99),PEEK (F+99)-1: LET R$(1)="O.K. ОН СРАЗУ ЖЕ ПОГАС.": GO TO 50
1215 IF NO=9 THEN POKE (O+9),O:POKE (O+2),PEEK F: POKE (F+2),O: POKE (F+99),PEEK (F+99)-1:
     RANDOMIZE USR SS: PRINT INK 5;"O.K. НО ОН ТУТ ЖЕ ГАСНЕТ!!!": GO SUB 8800: GO TO 100
1220 IF NO=8 THEN LET R$(1)="НО ВЕДЬ ОН НАДЕТ НА ВАС.": GO TO 50
1230 POKE (O+NO),PEEK F: POKE (F+99),PEEK (F+99)-1: LET R$(1)="O.K.": GO TO 50
1300 IF NO<10 THEN GO TO 1350
1302 IF NO<>16 THEN GO TO 1315
1305 IF (PEEK F=8 OR PEEK F=9) THEN LET R$(1)="ДВЕРЬ СДЕЛАНА ИЗ ДОБРОТНОГО": LET
     R$(2)="ДЕРЕВА И В ДАННЫЙ МОМЕНТ": LET R$(3)="ЗАКРЫТА, НО НЕ ЗАПЕРТА."
1310 IF PEEK (F+4)=1 THEN LET R$(3)="ОСТАЕТСЯ ШИРОКО ОТКРЫТОЙ."
1313 GO TO 50
1315 IF NO<>21 THEN GO TO 1335
1317 IF (PEEK F=14 OR PEEK F=15) THEN LET R$(1)="ВОРОТА СДЕЛАНЫ ИЗ НЕВАЖНОГО": LET
     R$(2)="ДЕРЕВА И В ДАННЫЙ МОМЕНТ": LET R$(3)="ЗАКРЫТЫ ИЗНУТРИ НА ЗАСОВ."
1320 IF PEEK (F+7)=1 THEN LET R$(3)="ОСТАЮТСЯ ОТКРЫТЫМИ."
1322 GO TO 50
1325 IF (PEEK F=15 AND (NO=23 OR NO=18) AND PEEK (F+6)=1) OR (PEEK F=7 AND PEEK (F+5)=1 AND
     NO=24) THEN LET R$(1)="ЗДЕСЬ ЯМА, БУДЬТЕ ОСТОРОЖНЫ!": GO TO 50
1330 IF ((PEEK F=15 AND (NO=23 OR NO=18)) OR (PEEK F=7 AND NO=24)) THEN LET R$(1) =
     "ЧУВСТВУЕТСЯ ЗАБОТЛИВЫЙ УХОД,": LET R$(2)="НО НЕДАВНИЙ ДОЖДЬ ОСТАВИЛ": LET
     R$(3)="РАСКИСШУЮ ПОЧВУ!": GO TO 50
1335 IF NO=22 AND ((PEEK F=7 AND PEEK (F+5)=1) OR (PEEK F=15 AND PEEK (F+6)=1)) THEN LET
     R$(1)="ЭТО ДОВОЛЬНО ГЛУБОКАЯ ЯМА,": LET R$(2)="И ЕСЛИ ТУДА СВАЛИТЬСЯ, ТО...!": GO TO
     50
1340 IF PEEK F=15 AND NO=26 AND PEEK (F+6)=0 THEN LET R$(1)="ВЫГЛЯДИТ ТАК, КАК БУДТО ЗДЕСЬ":
     LET R$(2)="КТО-ТО КОПАЛ.": GO TO 50
1345 GO TO 1375
1350 IF (PEEK (O+NO)<>PEEK F AND PEEK (O+NO)<>99) THEN LET R$(1) ="НО ВЕДЬ ЕГО РЯДОЙ С ВАМИ
     НЕТ!": GO TO 50
```

```
1355 IF NO=1 OR NO=8 THEN LET R$(1)"КРАСИВЫЙ ПЛЕД, ВЫГЛЯДИТ КАК": LET R$(2)"ИЗ ФРАНДУЗСКОЙ
      ШЕРСТИ.": GO TO 50
1360 IF NO=2 OR NO=9 THEN LET R$(1)"НЕБОЛЬШОЙ ФАКЕЛ, СОВЕРШЕННО": LET R$(2)"ОБЫКНОВЕННЫЙ
      НА ВИД.": GO TO 50
1365 IF NO=3 THEN LET R$(1)"ОНА ДОСТАТОЧНО ПРОЧНА.": GO TO 50
1370 IF NO=7 THEN LET R$(1)"ЧТО ЗА ПОДОЗРЕНИЯ!": LET R$(2)"(ОН НАСТОЯЩИЙ. СКОРЕЙ БЕРИ
      ЕГО": LET R$(3)"И ПОКОНЧИМ С ЭТИМ ДЕЛОМ!": GO TO 50
1372 IF NO=6 THEN LET R$(1)"НУ ЧТО ВЫ ЕРУНДОЙ ЗАНИМАЕТЕСЬ!": GO TO 50
1375 LET R$(1)"ВЫГЛЯДИТ СОВЕРШЕННО ОБЫКНОВЕННО,": LET R$(2)"НЕТ НИЧЕГО СТРАННОГО.": GO TO
      50
1400 GO TO 100
1500 RANDOMIZE USR SS: PRINT TAB 3; PAPER 2;"DHHHHHHHHHHHHHHHHHHHHHHHHE"
1501 RANDOMIZE USR SS: PRINT TAB 3; PAPER 2; INK 6;"A"; INK 7;" В ВАШЕМ РАСПОРЯЖЕНИИ: "; INK
      5;"G"
1502 RANDOMIZE USR 64968
1503 LET HE=0
1504 FOR X=0 TO (MOB-1)
1506 IF PEEK (PBS+X)=0 THEN LET X=MOB: GO TO 1512
1508 RANDOMIZE USR SS: PRINT TAB 3; PAPER 2;   INK 6;"A "; INT 7;O$(PEEK (PBS+X)); INK;"G":
      LET HE=HE+1
1510 NEXT X
1512 IF HE=0 THEN RANDOMIZE USR SS: PRINT TAB 3; PAPER 2; INK 6 ;"A"; INK  7;"НЕТ СОВСЕМ
      НИЧЕГО"; INK 6;"G"
1513 RANDOMIZE USR SS: PRINT TAB 3; PAPER 2; "CBBBBBBBBBBBBBBBBBBBBRBF"
1514 GO TO 195
1600 CLS
1602 PRINT AT 8,2;"ХОТИТЕ ПОПРОБОВАТЬ ЕЩЕ РАЗ?"; AT 11,8;"ВВЕДИТЕ Y ИЛИ N"
1604 POKE 23658,8: INPUT A$
1606 IF A$="Y" THEN RANDOMIZE USR 65058: GO TO 100
1608 IF A$="N" THEN GO SUB 8800: GO SUB 8800: GO TO 9999
1610 BEEP .2,20: GO TO 1600
1700 REM ЗАПИСЬ
1701 RANDOMIZE USR 65034
1702 CLS
1704 PRINT AT 7,0; " ДЛЯ ВРЕМЕННОГО СОХРАНЕНИЯ ПОЗИЦИИ В ПАМЯТИ КОМПЬЮТЕРА ВВЕДИТЕ ""B""
      (ВРЕМЕННО)"
1706 PRINT '' " ДЛЯ ПОСТОЯННОГО СОХРАНЕНИЯ ПОЗИЦИИ НА МАГНИТНОМ НОСИТЕЛЕ ВВЕДИТЕ ""П""
      (ПОСТОЯННО)"
1708 POKE 23658,0: INPUT A$: IF A$<>"B" AND A$<>"П" THEN GO TO 1708
1710 IF A$ = "B" THEN GO TO 100
1712 SAVE  "DATA"CODE 64130,150
1714 GO TO 100
1800 REM ЗАГРУЗКА
1801 CLS
1802 PRINT AT 7,0; "ДЛЯ ВОССТАНОВЛЕНИЯ ВРЕМЕННО СОХРАНЕННОЙ ПОЗИЦИИ ИЗ ПАМЯТИ ВВЕДИТЕ ""B""
      (ВРЕМЕННО)"
1804 PRINT '' "ДЛЯ ЗАГРУЗКИ ПОСТОЯННО СОХРАНЕН-НОЙ ИГРЫ С МАГНИТНОГО НОСИТЕЛЯВВЕДИТЕ  ""П""
      (ПОСТОЯННО)"
1805 POKE 23658,0: INPUT A$: IF A$<>"B" AND A$<>"П" THEN GO TO 1806
1808 CLS : IF A$="B" THEN GO TO 1814
1811 INK 0
1812 LOAD "DATA"CODE 64130,150
1815 INK 6
1814 IF PEEK 64190=0 THEN PRINT AT 12,0; "ВЫ НИ РАЗУ НЕ СОХРАНЯЛИ ИГРУ."'' "НЕТ ДАННЫХ ДЛЯ
      ВОССТАНОВЛЕНИЯ! ": BEEP .5,15: PAUSE 150: GO TO 100
1815 RANDOMIZE USR 65046
1816 GO TO 100
1915 IF PEEK F=9 AND NO=17 AND PEEK (F+3)=0 AND PEEK (F+11)=0 THEN POKE (F+3),1: POKE
      (F+11),1: LET R$(1)"ОТКРЫВ ЕГО, ВЫ СЛЫШИТЕ ТИХИЙ": LET R$(2)"ГОЛОС, ДОНОСЯЩИЙСЯ ИЗ
      ГЛУБИНЫ": LET R$(3)"БУФЕТА: ""ШВЫРНИ ШКАТУЛКУ":   LET R$(4)"С ЧЕРДАКА И УВИДИШЬ,
      ЧТО БУДЕТ.""": GO TO 50
1917 IF PEEK F=9 AND NO=17 AND PEEK (F+3)=1 THEN LET R$(1)"БУДЬ ВНИМАТЕЛЬНЕЕ!": LET
      R$(2)"ОН УЖЕ ОТКРЫТ": GO TO 50
1920 IF PEEK F=9 AND NO=17 AND PEEK (F+3)=0 THEN POKE (F+3),1:LET R$(1)"O.K. НО ЧТО ТАКОЕ?
      ВСЕ ТИХО!": GO TO 50
```

```
1925 IF NO=3 AND (PEEK (O+3)=99 OR PEEK (O+3)=PEEK F) THEN LET R$(1)= "НЕСМОТРЯ НИ НА КАКИЕ
     УСИЛИЯ": LET R$(2)="ОТКРЫТЬ ЕЕ НЕ УДАЕТСЯ!": GO TO 50
1930 IF PEEK F=11 AND NO=25 AND PEEK (F+8)=0 THEN LET R$(1)="НЕ ОТКРЫВАЕТСЯ! ЗАПЕРТ НА
     КЛЮЧ.": GO TO 50
1935 IF PEEK F=11 AND NO=25 AND PEEK (F+8)=1 THEN LET R$(1)="ОН УЖЕ ОТКРЫТ!": GO TO 50
1940 GO TO 45
2000 IF ((PEEK F=8 OR PEEK F=9) AND NO=16) OR ((PEEK F=14 OR PEEK F=15) AND NO=21) THEN LET
     R$(1)="ОКАЗЫВАЕТСЯ, ЗДЕСЬ НЕ ЗАПЕРТО!": GO TO 50
2005 IF ((PEEK F=8 OR PEES F=9) AND NO=16 AND PEEK (F+4)=1) OR ((PEEK F=14 OR PEEK F=15) AND
     NO=21 AND PEEK (F+7)=1) THEN LET R$(1)="НО ВЕДЬ УЖЕ ОТКРЫТО НАСТЕЖЬ!": GO TO 50
2010 IF PEEK F=11 AND NO=25 AND PEEK (F+8)=0 AND PEEK (O+5)=99 AND PEEK (F+9)=0 THEN POKE
     (F+8), 1: POKE (F+9),1: POKE (O+3),11:LET R$(1)="СЕЙФ ОТКРЫВАЕТСЯ, И ВНУТРИ ВЫ": LET
     R$(2)="ВИДИТЕ ДЕРЕВЯННУЮ ШКАТУЛКУ.": GO TO 50
2015 IF PEEK F=11 AND NO=25 AND PEEK (F+8)=1 THEN LET R$(1)="А ОН ТЕПЕРЬ НЕ ЗАПЕРТ!": GO TO
     50
2020 IF PEEK F=11 AND NO=25 AND PEEK (F+8)=0 AND PEEK (O+5)<>99 THEN LET R$(1)="ДЛЯ ЭТОГО
     НУЖЕН КЛЮЧ!": GO TO 50
2030 IF NO=3 AND (PEEK (O+3)-PEEK F OR PEEK (O+3)=99) AND PEEK (O+5)=99 THEN LET R$(1)="КЛЮЧ
     К ШКАТУЛКЕ НЕ ПОДХОДИТ.": GO TO 50
2035 GO TO 45
2100 IF (PEEK F<>7 AND PEEK F<>15) THEN LET R$(1)="ЗДЕСЬ НЕЛЬЗЯ КОПАТЬ": GO TO 50
2105 IF (((PEEK F=15 AND (NO=23 OR NO=18 OR NO=22 OR NO=26)) OR (PEEK F=7 AND (NO=24 OR
     NO=22))) AND PEEK (O+4)<>99) THEN LET R$(1)="И КАК ЖЕ ЭТО СДЕЛАТЬ?": LET R$(2)="ДЛЯ
     ЭТОГО НУЖЕН ПОДХОДЯЩИЙ": LET R$(3)="ИНСТРУМЕНТ.": GO TO 50
2110 IF PEEK F=15 AND PEEK (F+10)=0 AND (NO=18 OR NO=22 OR NO=23 OR NO=26) THEN POKE
     (F+10),1: POKE (O+5),15: POKE (F+6),1: LET R$(1)="ВЫ ИСПОРТИЛИ КРАСИВУЮ ЛУЖАЙКУ": LET
     R$(2)="ГЛУБОКОЙ ЯМОЙ. НО ВОТ ЧТО-ТО": LET R$(3)="БЛЕСТНУЛО... ЭТО КЛЮЧ!": GO TO 50
2115 IF ((PEEK F=15 AND PEEK (F+5)=1) OR (PEEK F=7 AND PEEK (F+5)=1)) THEN LET R$(1)="НЕТ
     СПАСИБО, БОЛЬШЕ НЕ НАДО!": GO TO 50
2120 IF PEEK F=15 AND PEEK (F+6)=0 AND PEEK (O+4)=99 AND (NO=23 OR NO=22 OR NO=18 OR NO=26)
     THEN POKE (F+6),1: LET R$(1)="O.K. ВЫ ВЫКОПАЛИ ЯМУ В САДУ.": GO TO 50
2125 IF PEEK F=7 AND PEEK (F+5)=0 AND PEEK (O+4)=99 AND (NO=24 OR NO=22) THEN POKE (F+5),1:
     LET R$(1)="O.K. ВЫ В ПОЛЕ ВЫКОПАЛИ ЯМУ.": GO TO 50
2130 GO TO 45
2200 IF PEEK (O+9)=99 AND NO=9 THEN LET R$(1)="НО ОН И ТАК ГОРИТ!": GO TO 50
2205 IF NO=2 AND PEEK (O+2)=99 AND PEEK F<>11 THEN POKE (O+2),0: POKE (O+9),99: POKE
     (F+2),1: LET R$(1)="O.K. ТЕПЕРЬ ФАКЕЛ ГОРИТ.": GO TO 50
2210 IF NO=2 AND PEEK (O+2)=99 THEN POKE (O+2),0: POKE (O+9),99 : POKE (F+2),1: RANDOMIZE
     USR SS: PRINT INK 5;"O.K. ТЕПЕРЬ ФАКЕЛ ЯРКО СВЕТИТ!": GO SUB 8800: GO TO 100
2215 GO TO 45
2300 IF PEEK (O+2)=99 AND NO=9 THEN LET R$(1)="НО ВЕДЬ ОН НЕ ГОРИТ!": GO TO 50
2305 IF NO=9 AND PEEK (O+9)=99 AND PEEK F<>1 THEN POKE (O+9),0: POKE (O+2),99: POKE (F+2),0:
     LET R$(1) = "O.K. ФАКЕЛ ПОГАС.": GO TO 50
2310 IF NO=9 AND PEEK (O+9)=99 THEN POKE (O+9),0: POKE (O+2),99: POKE (F+2),0: RANDOMIZE USR
     SS: PRINT INK 5;"O.K. ФАКЕЛ БОЛЬШЕ НЕ СВЕТИТ.": GO SUB 8800: GO TO 100
2315 GO TO 45
2400 IF PEEK F=11 AND (NO=19 OR NO=14) THEN POKE F,6: GO TO 100
2405 IF PEEK F=6 AND (NO=20 OR NO=14) THEN POKE F,2: GO TO 100
2410 IF PEEK F=2 AND NO=15 THEN POKE F,6: GO TO 100
2412 IF PEEK F=6 AND NO=15 THEN POKE F,11: GO TO 100
2415 IF (PEEK F=14 OR PEEK F=15) AND PEEK (F+7)=0 THEN LET R$(1)="ЭТО НЕСЕРЬЕЗНО, ЛУЧШЕ
     ОТКРОЙТЕ": LET R$(2)="ВОРОТА ИЗ САДА!": GO TO 50
2420 IF (PEEK F=14 OR PEEK F=15) AND PEEK (F+7)=1 THEN LET R$(1)="И ЗАЧЕМ ВАМ ЭТО ДЕЛАТЬ?":
     LET R$(2)="ВЕДЬ ВОРОТА ОТКРЫТЫ.": GO TO 50
2425 GO TO 45
2500 IF PEEK F=2 AND (NO=20 OR NO=15) THEN POKE F,6: GO TO 100
2505 IF PEEK F=6 AND (NO=19 OR NO=15) THEN POKE 64280,11: GO TO 100
2510 GO TO 45
2600 IF (PEEK F=8 OR PEEK F=9) AND NO=16 AND PEEK (F+4)=1 THEN POKE (F+4),0: LET R$(1)="O.K.
     ТЕПЕРЬ ДВЕРЬ ЗАКРЫТА.": GO TO 50
2605 IF (PEEK F=14 OR PEEK F=15) AND NO=21 AND PEEK (F+7)=1 THEN POKE (F+7),0: LET
     R$(1)="O.K. ТЕПЕРЬ ВОРОТА ЗАКРЫТЫ.": GO TO 50
2610 IF PEEK F=9 AND NO=17 AND PEEK (F+3)=1 THEN POKE (F+3),0: LET R$(1)="O.K. ТЕПЕРЬ БУФЕТ
     ЗАКРЫТ.": GO TO 50
```

```
2615 IF PEEK F=11 AND NO=25 AND PEEK (F+8)-1 THEN LET R$(1)="КАЖЕТСЯ, ДВЕРЦУ ЗАКЛИНИЛО!": GO
     TO 50
2620 GO TO 45
2700 IF PEEK F=2 AND NO=3 AND PEEK  (O+3)=99 AND PEEK (F+11)=1  THEN POKE (O+7),6: POKE
     (O+6), 6: POKE (O+3),O: POKE (F+99),PEEK (F+99)-1: LET R$ (1) ="ВОТ ЭТО БРОСОК!!!":
     LET R$(2) = "ОТ УДАРА ШКАТУЛКА РАЗБИЛАСЬ.":   LET R$(3)="ЧТО-ТО ИЗ НЕЕ ВЫПАЛО И
     БЛЕСНУВ,": LET R$(4) = "ПОКАТИЛОСЬ ПО ПОЛУ КОНЮШНИ.": GO TO 50
2705 IF PEEK F=2 AND NO=3 AND PEEK (O+3)=99 THEN LET R$(1)="Я БЫ ЭТОГО НЕ ДЕЛАЛ.":   LET
     R$(2)="НАДО ЕЩЕ КОЕ-ЧЕГО": LET R$(3)="ДОБИТЬСЯ ОТ БУФЕТА!":   GO TO 50
2710 GO TO 1200
2800 IF ((PEEK F=15 AND PEEK (F+6)=1) OR (PEEK F=7 AND PEEK (F+5)=1)) AND PEEK (O+4)<>99 AND
     NO=22 THEN LET R$(1)="И КАК ЖЕ ВЫ СОБИРАЕТЕСЬ ЭТО": LET R$(2)="СДЕЛАТЬ БЕЗ НУЖНОГО
     ИНСТРУМЕНТА!": GO TO 50
2802 IF (PEEK F=15 AND PEEK (F+6)=0 AND (NO=22 OR NO=26)) OR (PEEK F=7 AND PEEK (F+5)=0 AND
     NO=22) THEN LET R$ (1)="ШИКАРНО ПОЛУЧИЛОСЬ!": GO TO 50
2805 IF PEEK F=7 AND NO=22 AND PEEK (O+4)=99 AND PEEK (F+5)=1 THEN POKE (F+5),O: LET
     R$(1)="O.K. ВЫ ЗАКОПАЛИ ЯМУ В ПОЛЕ.": GO TO 50
2810 IF PEEK F=15 AND NO=22 AND PEEK (O+5)=15 THEN LET R$(1)="ОСТОРОЖНО! ЗАКОПАЕТЕ КЛЮЧ!":
     GO TO 50
2812 IF PEEK F=15 AND NO=22 AND PEEK (O+4)=99 AND PEEK (F+6)=1 THEN POKE (F+6),O: LET
     R$(1)="O.K. ВЫ ЗАКОПАЛИ ЯМУ": LET R$(2)="И ОСТАВИЛИ НА ЛУЖАЙКЕ": LET
     R$(3)="НЕКРАСИВОЕ ПЯТНО!": GO TO 50
2815 GO TO 45
2900 IF PEEK F=11 AND NO=25 AND
2902 (F+8)=1 THEN LET R$(1)="НЕ ПОЛУЧАЕТСЯ, ДВЕРЬ ЗАЕЛО!": GO TO 50
2905 GO TO 45
3000 IF NO=1 AND PEEK(O+1)=99 THEN POKE (O+1),O: POKE (O+8),99: POKE (F+1),1: POKE
     (F+99),PEEK (F+99)-1: LET R$(1)="ОТЛИЧНО! ОСВОБОДИЛАСЬ РУКА.": LET R$(2)="ДА И ПЛЕД
     ВАМ ОЧЕНЬ ИДЕТ.": GO TO 50
3005 IF NO=1 AND PEEK (O+8)=99 THEN LET R$(1)="ОН УЖЕ ОДЕТ НА ВАС!": GO TO 50
3008 IF NO=1 THEN LET R$(1)="НЕ ПОЛУЧАЕТСЯ, СНАЧАЛА НАДО": LET R$(2)="ВЗЯТЬ ЕГО В РУКИ.": GO
     TO 50
3010 GO TO 45
3100 IF NO=8 AND PEEK (F+99)=MAX THEN LET R$(1)="НЕ ВЫХОДИТ! ОБЕ РУКИ ЗАНЯТЫ.": GO TO 50
3102 IF NO=8 THEN POKE (O+1),99 : POKE (O+8),O: POKE (F+1),O: POKE (F+99),PEEK (F+99)+1: LET
     R$(1)="O.K. НО ПОЧЕМУ?": LET R$(2)="РАЗВЕ ВЫ К НЕМУ ЕЩЕ": LET R$(3)="НЕ ПРИШЛИ?": GO
     TO 50
3105 IF NO=1 THEN LET R$(1)="НО ЕГО НЕТ НА ВАС!": GO TO 50
3110 GO TO 45
3200 IF PEEK F=2 AND (NO=15 OR NO=27) THEN POKE F,6: FOR X=-5 TO 15: BEEP .04,K: NEXT X: FOR
     X=14 TO -5 STEP -1: BEEP .08,X: NEXT X: BEEP 1,-6: GO TO 100
3205 IF (NO=27 OR NO=5 OR NO=5) THEN LET R$(1)="O.K. НО НИЧЕГО НЕ СЛУЧИЛОСЬ!": GO TO 50
3210 GO TO 45
3300 IF NO<10 AND PEEK (O+NO)<>99 THEN LET R$(2)="ПРОВЕРЬТЕ СВОЙ ИНВЕНТАРЬ!": GO TO 50
3305 IF NO=5 AND PEEK (F+8)=O THEN LET R$(1)="И КАК ЖЕ НАДО ИСПОЛЬЗОВАТЬ": LET R$(2)="ЭТОТ
     КЛЮЧ?": GO TO 50
3310 IF NO=4 THEN LET R$(1)="ПОДУМАЙТЕ... - НАВЕРНОЕ,": LET R$(2)="ВЫ ЗНАЕТЕ, ЧТО МОЖНО
     ДЕЛАТЬ": LET R$(3)="С ПОМОЩЬЮ ЛОПАТЫ!":GO TO 50
3315 LET R$(1)="ЧТО ЗНАЧИТ **ИСПОЛЬЗОВАТЬ**!?": LET R$(2)="НЕЛЬЗЯ ЛИ УТОЧНИТЬ?": GO TO 50
```

Теперь пришло время расшифровки некоторых ключевых моментов, которые помогут Вам отслеживать логику работы программы.

Управление адвентюрной игрой.

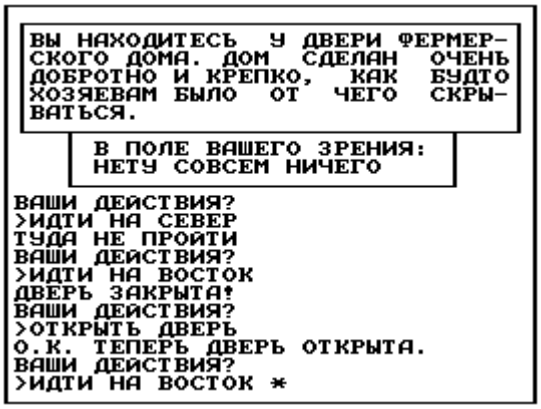
БЕЙСИК-программа управляет игрой с помощью следующих данных:

- номер глагола (или команды) (VB);
- номер существительного (или направления) (NO);
- таблицы текущего расположения объектов;
- таблицы состояния флагов.

В прошлом номере "ZX-РЕВЮ" мы рассмотрели параметры, задаваемые при помощи строк DATA (Лист. 2). Теперь, подводя итог сказанному, мы приводим полный перечень слов (Листинги 1 и 2), распознаваемых программой, в виде двух таблиц, отдельно для глаголов и существительных (и прочих слов). Для каждого слова указан его номер.

Анализатор текста.

Система, анализирующая текст, позволяет вводить до 30 символов, среди них могут быть русские и латинские заглавные буквы, цифры и символы. Русские буквы набираются непосредственно буквенными клавишами, за исключением Ч, Э, Ш, Щ, Ю. Последние набираются одновременно с нажатием SYMB. SH. клавишами A, S, F, G, X. Для "забоя" последнего введённого символа используется клавиша "DELETE" или "КУРСОР ВЛЕВО". Приглашение процедуры ввода изображается символом ">", а курсор - в виде символа "*".



Как уже говорилось, процедура построена таким образом, что анализирует пробелы между словами, выделяя первое и последнее слово. Первое слово считается глаголом, а последнее - существительным или направлением перемещения. Поэтому команды "БРОСИТЬ ЭТУ ЛОПАТУ", "БРОСИТЬ КОРОТКУЮ ЛОПАТУ" дают тот же эффект, что и просто "БРОСИТЬ ЛОПАТУ".

Анализатор просматривает введённый текст команды и возвращается в БЕЙСИК с числом VB, соответствующим номеру использованного глагола и с числом NO, соответствующим номеру использованного существительного. Любое нераспознанное слово выдаёт в БЕЙСИК число 200 для глагола и число 201 для существительного.

Таблица 1.

ГЛАГОЛЫ	
ОСНОВНЫЕ	ДОПОЛНИТЕЛЬНЫЕ
ИДТИ.....0	ОТКРЫТЬ.....9
СЕВЕР.....0	ОТПЕРЕТЬ.....10
С.....0	КОПАТЬ.....11
ЮГ.....0	ВЫКОПАТЬ.....11
Ю.....0	ЗАЖЕЧЬ.....12
ВОСТОК.....0	ГАСИТЬ.....13
В.....0	ПОГАСИТЬ.....13
НАПРАВО....0	ПОДНЯТЬСЯ.....14
ЗАПАД.....0	ЗАЛЕЗТЬ.....14
З.....0	ЛЕЗТЬ.....14
НАЛЕВО.....0	СПУСТИТЬСЯ....15
ВВЕРХ.....0	СЛЕЗТЬ.....15
Х.....0	ЗАКРЫТЬ.....16
ВНИЗ.....0	ШВЫРНУТЬ.....17
Н.....0	ЗАКОПАТЬ.....18
ВЗЯТЬ.....1	ЗАПЕРЕТЬ.....19
БРАТЬ.....1	ОДЕТЬ.....20
ПОЛОЖИТЬ...2	НАДЕТЬ.....20
БРОСИТЬ....2	СНЯТЬ.....21
ПРОВЕРИТЬ..3	ПРЫГАТЬ.....22
ОСМОТРЕТЬ..3	СПРЫГНУТЬ.....22
СМОТРЕТЬ...4	ЗАПРЫГНУТЬ....22
ИНВЕНТАРЬ..5	ИСПОЛЬЗОВАТЬ..23
И.....5	
СТОП.....6	
КОНЕЦ.....6	
СОХРАНИТЬ..7	
ЗАГРУЗИТЬ..8	

Эта информация может обыгрываться в БЕЙСИКЕ как угодно, но в нашей демонстрационной программе применяется самое простое решение, - в этом случае пользователь всегда получает ответ "НЕ ПОЛУЧАЕТСЯ".

Если же Вы даёте команду, состоящую из одного слова, например, "СМОТРЕТЬ", то за глагол Вы получите VB=4, а за отсутствие существительного - должны были бы получить NO=201, что вызовет реакцию "НЕ ПОЛУЧАЕТСЯ", хотя нам такая реакция не нужна. Для того, чтобы избежать подобных ситуаций, мы включаем глагол "СМОТРЕТЬ" и в список существительных тоже, при этом присваиваем ему порядковый номер 99 (см. Листинг_1, строка 2190 и Таблицу_2).

Со всеми другими глаголами (командами), которые могут использоваться без существительного, надо поступить тем же образом.

Возможен и другой вариант, когда такому глаголу Вы присваиваете номер не 99, а соответствующий тому существительному, которое с этим глаголом обычно связано. В демонстрационной программе мы так поступили со словом КОПАТЬ - ему присвоен номер 22 (см. Листинг_2, строка 2192 и Таблицу_2), потому что КОПАТЬ и КОПАТЬ ЯМУ - понятия равнозначные. Поэтому пусть читатель, бросивший беглый взгляд на таблицы глаголов и существительных без чтения текста, не упрекает нас в незнании основ русской грамматики.

Таблица_2

СУЩЕСТВИТЕЛЬНЫЕ И ДРУГИЕ СЛОВА
ИНВЕНТАРЬ
ПЛЕД.....1
ШЕРСТЯНОЙ ПЛЕД.....1
ФАКЕЛ.....2
МАЛЕНЬКИЙ ФАКЕЛ.....2
ШКАТУЛКА.....3
ДЕРЕВЯННАЯ ШКАТУЛКА.....3
ЛОПАТА.....4
КОРОТКАЯ ЛОПАТА.....4
КЛЮЧ.....5
БРОНЗОВЫЙ КЛЮЧ.....5
ЩЕПКИ.....6
ДЕРЕВЯННЫЕ ЩЕПКИ.....6
АЛМАЗ.....7
КРУПНЫЙ АЛМАЗ.....7
ПЛЕД, НАДЕТЫЙ НА ПЛЕЧИ....8
ГОРЯЩИЙ ФАКЕЛ.....9
НАПРАВЛЕНИЕ
СЕВЕР.....С.....10
ЮГ.....Ю.....11
ВОСТОК.....В...НАПРАВО....12
ЗАПАД.....З...НАЛЕВО.....13
ВВЕРХ.....Х.....14
ВНИЗ.....Н.....15

ПРОЧИЕ СЛОВА
ДВЕРЬ.....16
БУФЕТ.....17
ЛУЖАЙКА.....18
СТУПЕНИ.....19
ЛЕСТНИЦА.....20
ВОРОТА.....21
КОПАТЬ.....22
ЯМУ.....22
САД.....23
ПОЛЕ.....24
СЕЙФ.....25
ПЯТНО.....26
ПРЫГАТЬ.....27
КОМАНДЫ
СМОТРЕТЬ.....99
ИНВЕНТАРЬ.....99
И.....99
СТОП.....99
КОНЕЦ.....99
СОХРАНИТЬ.....99
ЗАГРУЗИТЬ.....99

Обратите внимание, что для тех существительных, которые имеют длину, меньше заданной для идентификации (4 символа) - это, например, существительные ЯМА, САД - задавать их надо в родительном падеже, то есть: ЯМУ (а САД, ПОЛЕ, СЕЙФ и т.д. и в родительном и в именительном падеже совпадают). Ведь подаваемая команда может быть: ВЫКОПАТЬ ЯМУ или ОСМОТРЕТЬ ЯМУ или ПРЫГНУТЬ В ЯМУ и т. д. Если же допускать такие команды, которые потребуют так же и слова ЯМА, например, ПРОВЕРИТЬ, КАК ВЫКОПАНА ЯМА, то существительные надо задавать дважды: ЯМА и ЯМУ, но присваивать им одинаковые номера.

Обратите также внимание на разницу в использовании некоторых глаголов. Это, например, СМОТРЕТЬ и ОСМОТРЕТЬ. Первый не относится к какому-либо предмету

конкретно и подразумевает, как бы оглядеться по сторонам, при этом перерисовывается экран и заново сканируется и отображается табло "В ПОЛЕ ВАШЕГО ЗРЕНИЯ". Глагол СМОТРЕТЬ может использоваться самостоятельно, без существительного. Другой глагол - ОСМОТРЕТЬ - самостоятельно использоваться не может и служит для получения каких-то дополнительных сведений о конкретном предмете, например, о ДВЕРИ, ПОЛЕ, ШКАТУЛКЕ, ЩЕПКАХ от неё, когда она будет разбита и т. д.

Различаются такие глаголы ОТКРЫТЬ и ОТПЕРЕТЬ. Учитывайте, что, например, ДВЕРЬ может быть закрыта, но не заперта на замок, а СЕЙФ закрыт, да ещё и заперт.

Таблица расположения объектов.

Переменная O определяет начало этой таблицы. В ней находятся данные о текущем расположении 50 объектов. Для каждого объекта в любой момент времени здесь содержится число, определяющее номер локации, в которой в данный момент находиться объект.

Например, для объекта с номером 1 (ПЛЕД) Вы можете узнать его расположение командой PEEK (O+1), а изменить его состояние - командой POKE (O+1), N. Здесь "N" - новое место размещения объекта.

Если PEEK (O+X)=99, то это означает, что объект находится у Вас в руках (или на Вас).

Если PEEK (O+X)=0,это значит, что объект в данный момент не виден.

По ходу игры Бейсик-программа вносит изменения в таблицу расположения объектов.

Пример расшифровки логической конструкции:

```
IF PEEK (O+9)<>99 THEN ...
```

означает следующее:

ЕСЛИ У ВАС ПРИ СЕБЕ НЕТ ГОРЯЩЕГО ФАКЕЛА, ТО ...

Таблица состояния флагов.

Некоторые объекты имеют одно и то же имя, например ФАКЕЛ. В то время, когда это просто ФАКЕЛ или МАЛЕНЬКИЙ ФАКЕЛ, то это объект номер 2. В то же время, если это ГОРЯЩИЙ ФАКЕЛ, то это уже объект номер 9. Кроме того, по ходу игры могут происходить разные события с предметами, не входящими в инвентарь. Это, например, отпирание и открывание дверей, фиксация того, что интересующее событие произошло и т.д. Все эти моменты отражаются в таблице флагов.

Переменная F определяет адрес ячейки, с которой начинается эта таблица. В ней возможно расположение данных о 100 объектах, причём некоторые флаги зарезервированы для особых случаев.

Определяется состояние флага предмета X командой PEEK (F+X), а изменяется командой POKE (F+X), Z, где Z - новое состояние предмета.

По ходу игры Бейсик-программа вносит изменения в таблицу состояния флагов.

Флаг (F+0) содержит номер текущей локации - то есть место, где Вы сейчас находитесь. Расшифровка логических конструкций в программе будет выглядеть так. Например:

```
IF PEEK F=7 THEN ...
```

означает следующее:

ЕСЛИ ДЕЙСТВИЕ ПРОИСХОДИТ В ПОЛЕ ТО ...

Другой пример:

```
... POKE F, 6 ...
```

означает:

... МЫ ОКАЗЫВАЕМСЯ В КОНЮШНЕ ...

Флаг (F+99) служит для отражения количества предметов (ИНВЕНТАРЯ), которые герой имеет при себе в данный момент.

Пример логической конструкции:

```
POKE (F+99), PEEK (F+99)+1
```

означает ни что иное, как то, что Вы берёте объект в руки. Другой пример:

```
IF PEEK (F+99)=MAX THEN . . .
```

означает:

ЕСЛИ У ВАС В РУКАХ МАКСИМАЛЬНОЕ КОЛИЧЕСТВО ПРЕДМЕТОВ, ТО ...

Когда игра начинается, все флаги, за исключением (F+0) имеют нулевое значение. В процессе игры, когда герой надевает ПЛЕД, флагу (F+1) присваивается единица, а когда он зажигает ФАКЕЛ, единица присваивается флагу (F+2) и т. д.

Ниже приводится таблица флагов, использующихся в игре. Значение 1 присваивается соответствующему флагу, если выполняется действие, показанное в таблице.

ТАБЛИЦА СОСТОЯНИЯ ФЛАГОВ
F+0=NO - НОМЕР ТЕКУЩЕЙ ЛОКАЦИИ
F+1=1, если ПЛЕД НАДЕТ НА ПЛЕЧИ
F+2=1, если ФАКЕЛ ЗАЖЖЁН
F+3=1, если БУФЕТ ОТКРЫТ
F+4=1, если ДВЕРЬ ОТКРЫТА
F+5=1, если ВЫКОПАНА ЯМА В ПОЛЕ
F+6=1, если ВЫКОПАНА ЯМА В САДУ
F+7=1, если ВОРОТА ОТКРЫТЫ
F+8=1, если СЕЙФ НЕ ЗАПЕРТ
F+9=1, если ШКАТУЛКА НАЙДЕНА
F+10=1, если КЛЮЧ НАЙДЕН
F+11=1, если БУФЕТ ОТКРЫВАЛСЯ ХОТЯ БЫ 1 РАЗ
F+99=NO - ЧИСЛО ВЗЯТЫХ ОБЪЕКТОВ

Если с этим объектом происходит противоположное действие (дверь опять закрывается), то флаг должен обнуляться. Однако могут быть ситуации, когда состояние одного объекта оценивается несколькими флагами. Например, если ОТКРЫТЬ БУФЕТ в первый раз, то установится в единицу два флага: F+3 и F+11. Если теперь ЗАКРЫТЬ БУФЕТ, то сбросится флаг F+3, а флаг F+11 так и останется в состоянии 1.

Теперь можно проследить, как в игре формируются более сложные действия. Ситуация, когда зажигается факел, например, моделируется следующим образом:

```
POKE (F+2), 1: POKE (0+2)=0: POKE (0+9), 99
```

То есть, во-первых, устанавливается флаг зажжённого факела, во-вторых, исчезает объект под названием МАЛЕНЬКИЙ ФАКЕЛ, в третьих - у Вас в руках появляется новый объект: ГОРЯЩИЙ ФАКЕЛ.

Ситуация, когда Вы кладёте горящий факел, например, в саду (при этом он гаснет), будет выглядеть так:

```
POKE (0+9), 0: POKE (F+2), 0: POKE (0+2), 15:
```

Сначала исчезает объект ГОРЯЩИЙ ФАКЕЛ и обнуляется его флаг, а затем в САДУ появляется объект МАЛЕНЬКИЙ ФАКЕЛ.

Теперь, когда Вы имеете представление о том, как моделируются те или иные логические конструкции, можно проследить действие программы сначала.

Демонстрационная программа.

В строке 100 начинается прорисовка игрового экрана. На нём изображён только текст, так задумано в концепции отладочной системы ABS. Однако ничего не мешает Вам изменить 7000-ные строки так, чтобы, например, запускался декомпрессор, выводя в верхней трети (или двух третях) экрана рисунок локации. Эти экраны можно подготовить, например, при помощи компрессора, опубликованного в "ZX-РЕВЮ" №3-4, стр. 61-63.

Строка 102 выполняет подпрограмму из большого блока строк, который начинается со строки 7000. Описание локации вводится обычным оператором PRINT в строках 7000 + номер локации, умноженный на 10. Это происходит в зависимости от номера локации PEEK F. Для АМБАРА (1 ЛОКАЦИЯ) будет GO SUB 7000, для ЧЕРДАКА (2 ЛОКАЦИЯ) - GO SUB 7010 и т. д. Длина описания каждой локации - не более 10 БЕЙСИК-строк. Каждое описание должно заканчиваться командой RETURN.

Строка 104 (вместе со строкой 7110) обеспечивают реакцию программы на событие, когда Вы попадаете в погреб без ГОРЯЩЕГО ФАКЕЛА.

Строки 105-129 обеспечивают проверку того, что можно увидеть в этой локации. Если

что-то есть, то эта информация также выводится на экран в виде табло: "В ПОЛЕ ВАШЕГО ЗРЕНИЯ". Блок в машинных кодах сам выполняет проверку того, какие объекты следует объявить в этих ситуациях, а начать выполнять процедуры Бейсика. Вместо слов можно вводить изображение соответствующих предметов при помощи символов UDG-графики, печатая их оператором PRINT прямо поверх картинки.

Процедура RANDOMIZE USR SS (строка 195) проверяет необходимость и осуществляет скроллинг экрана и должна выполняться всякий раз, после того, как происходит перерисовка экрана.

Далее, в строке 200, запускается большая процедура в кодах, обеспечивающая сканирование клавиатуры и печать на экране вводимых команд, а также их первичный анализ - "Анализатор текста", о котором уже говорилось выше.

Результатом работы этой процедуры являются значения двух ячеек памяти: 64114 и 64115. В первой из них содержится номер использованного глагола, во второй - существительного (или направления перемещения). Они передаются в Бейсик-переменные в строках 202 и 204. Номер глагола будет теперь VB, а номер существительного - NO.

Строка 208 разрешает при отсутствии ГОРЯЩЕГО ФАКЕЛА в ПОГРЕБЕ только следующие действия: ИДТИ, ЗАЖЕЧЬ, ПОГАСИТЬ, ПОДНЯТЬСЯ, СТОП, ИНВЕНТАРЬ, СОХРАНИТЬ и ЗАГРУЗИТЬ.

Если в результате работы процедуры ввода команды не зафиксирован ни один из допустимых глаголов, то значение ячейки 64114 будет равно 200. Аналогично, если в команде не зафиксировано ни одно из заданных существительных (или направления перемещения), то значение ячейки 64115 будет 201. Как поступить в этих случаях - решать Вам. В демонстрационной программе в этом случае строка 228 адресует на строку 45, где выводиться сообщение "НЕ ПОЛУЧАЕТСЯ".

Если глагол и существительное идентифицированы, то дальнейшее распределение производиться в строке 235, в зависимости от значения глагола: происходит переход на подпрограммы обработки глаголов, расположенные в большом блоке строк, который начинается со строки 1000.

Поскольку наиболее вероятно, что Вы будете оформлять программу с разделённым экраном, нужно было бы вводить проверку на необходимость скроллирования перед тем, как печатать сообщение от программы на экране. Чтобы избежать такой проверки, мы в демонстрационной программе оформляем реакцию программы в виде символьных переменных R$(1), R$(2), R$(3), R$(4), которые печатаются с помощью главной печатающей процедуры (строка 50).

Обработка глаголов.

Каждая подпрограмма обработки глаголов должна иметь не более 100 Бейсик-строк. Номер первой строки определяется по номеру глагола: 1000 + номер глагола, умноженный на 100. Так обеспечивается переход, например в случае ИДТИ - на строку 1000, в случае ВЗЯТЬ - на строку 1100 и т. д.

Вот полный перечень строк для соответствующих глаголов.

```
ИДТИ..........1000
ВЗЯТЬ.........1100
ПОЛОЖИТЬ......1200
ОСМОТРЕТЬ.....1300
СМОТРЕТЬ......1400
ИНВЕНТАРЬ.....1500
СТОП..........1600
СОХРАНИТЬ.....1700
ЗАГРУЗИТЬ.....1800
ОТКРЫТЬ.......1900
ОТПЕРЕТЬ......2000
КОПАТЬ........2100
ЗАЖЕЧЬ........2200
ПОГАСИТЬ......2300
ЛЕЗТЬ.........2400
СПУСТИТЬСЯ....2500
```

```
ЗАКРЫТЬ........2600
ШВЫРНУТЬ.......2700
ЗАКОПАТЬ.......2800
ЗАПЕРЕТЬ.......2900
НАДЕТЬ.........3000
СНЯТЬ..........3100
ПРЫГАТЬ........3200
ИСПОЛЬЗОВАТЬ...3300
```

Специфичной является подпрограмма перемещения, расположенная со строки 1000. Мы попадаем на неё независимо от направления, во всех случаях предусмотренного перемещения. В строке 1002 происходит изменение номера слова, определяющего переменные. Так, для СЕВЕР вместо 10 получается значение 1, для ЮГ - вместо 11 - значение 2 и т. д. Это необходимо для корректного дальнейшего выполнения программы в кодах.

В строке 1005 выполняется следующая проверка. Если Вы находитесь перед дверью и идёте на восток или находитесь в прихожей и идёте на запад и в обоих этих случаях дверь закрыта, то вывод текста "ДВЕРЬ ЗАКРЫТА".

Строка 1010. Аналогичная проверка для ворот.

Строка 1020. Если Вы пытаетесь идти в поле, в котором выкопана яма или в саду, в котором выкопана яма, то Вы в неё падаете. В этом случае игра для Вас заканчивается.

В строке 1090 выполняется подпрограмма в кодах, которая выполняет перемещение в другую локацию согласно таблице допустимых перемещений. Входным параметром для этой процедуры является значение содержимого ячейки 64115 (оно было занесено туда в строке 1002). Другим параметром для этой процедуры является значение флага (F+0), то есть текущая локация. Результат работы этой процедуры возвращается также в ячейку 64115, теперь там значение, соответствующее новой локации. Если же там ноль, то это значит, что в направлении, которое Вы задали, перехода нет. Этот контроль производится в строке 1092. В строке 1093 происходит изменение флага текущей локации - Вашего положения. Строка 1094 возвращает на начало программы с новыми условиями - в новой локации.

Обратите внимание, что зацикливание происходит тремя способами. Первый - если команда не воспринимается - на строку 45 с выводом сообщения "НЕ ПОЛУЧАЕТСЯ". Второй - на строку 50 - без изменения верхней части экрана - то есть без изменения локации и отображения предметов, находящихся в поле Вашего зрения. Третий способ - переход на строку 100 (как при старте игры) - это перерисовка всего экрана (переход в новую локацию или команда СМОТРЕТЬ). При этом также изменяется и табло "В ПОЛЕ ВАШЕГО ЗРЕНИЯ:".

Примеры расшифровки подпрограмм.

Теперь, для примера, мы дадим расшифровку реакции программы на некоторые Ваши действия. Потом Вы должны будете продолжать эти рассуждения для всех строк программы, если хотите в деталях разобраться в том, что происходит в разных ситуациях.

Итак, например, строка 1100 - сюда мы попадаем в случае обработки глагола ВЗЯТЬ.

Строка 1100 накладывает ограничение на попытку взять объект, не являющийся инвентарем (то есть имеющий больший номер).

Строка 1105 недоумевает по поводу попытки взять объект, который уже находится у Вас в руках.

Строка 1110 удивляется, если Вы хотите взять инвентарь, который в принципе существует, но его сейчас нет в этой локации.

Строка 1115 возмущается, если Вы хотите поднять больше объектов, чем это разрешено программой.

Строка 1120 радуется, если Вы берёте в руки АЛМАЗ - это конечная цель миссии.

Строка 1125 соглашается с Вами, если Вы берёте остальные объекты инвентаря, которые не были зафиксированы в предыдущих строках.

Крупным блоком является блок, обрабатывающий глагол ОСМОТРЕТЬ (ПРОВЕРИТЬ). Здесь проверка разветвляется на несколько ветвей для ускорения анализа.

Строка 1300 разделяет проверку на относящуюся к инвентарю и прочим объектам.

Строка 1305 отражает закрытое, а строка 1310 - открытое (в случае выключенного флага двери) состояние двери. Строка 1312 зацикливает программу.

Строка 1315 аналогично двери перехватывает проверку ворот.

Строки 1317 и 1320 аналогично двери отражают открытое или закрытое состояние ворот (в зависимости от флага ворот). После чего строка 1322 зацикливает программу.

Строка 1325 предупреждает о выкопанной яме, если Вы находитесь в саду и осматриваете САД или ЛУЖАЙКУ и при этом яма выкопана (флаг ямы в саду - установлен) или Вы находитесь в поле с выкопанной ямой и осматриваете ПОЛЕ.

Строка 1330 отражает результаты проверки, если Вы находитесь в саду и осматриваете сад или лужайку, но яма не выкопана (этот случай был перехвачен предыдущей строкой) или в поле, в котором тоже нет ямы.

Строка 1335 показывает результат проверки ЯМЫ, если Вы находитесь в поле, в котором выкопана яма или в саду, в котором выкопана яма.

Строка 1340 - результат осмотра ПЯТНА, если Вы находитесь в саду и яма закопана.

Строка 1345 - это результат всех остальных проверок.

Строка 1350 перехватывает проверки тех предметов, которых нет в этой локации и нет у Вас.

Строка 1355 показывает результат осмотра ПЛЕДА или ПЛЕДА НАДЕТОГО НА ПЛЕЧИ.

Строка 1360 - результат осмотра ФАКЕЛА или ГОРЯЩЕГО ФАКЕЛА.

Строка 1365 - осмотр ШКАТУЛКИ.

Строка 1370 - осмотр АЛМАЗА.

Строка 1372 - осмотр ЩЕПОК.

Строка 1375 - все прочие случаи, не предусмотренные программой.

Подробную расшифровку некоторых строк мы дали только для примера. С остальными Вы теперь сможете разобраться самостоятельно.

Кодовый блок программы.

Эта информация представляет определённый интерес для "хакеров". Речь идёт о блоке кодов "system" CODE. В том случае, если Вы не изменяли данные, заданные в Листингах 1 и 2, адрес загрузки блока кодов - переменная SAVE - будет равен 63400. Этот блок занимает пространство до 65368 (начало символов UDG - графики, стандартно расположенных здесь). Если Вы увеличили число слов, распознаваемых программой, то адрес загрузки Вашего блока отодвинется в сторону младших адресов. Здесь надо проконтролировать, чтобы он не перекрывался с загружаемым символьным набором, который сейчас расположен с адреса 50000 и имеет длину 768 байт (см. Листинг_3) - может этот адрес и изменить.

С адреса загрузки SAVE блока "system" CODE расположены те данные, которые были заданы в строках DATA программы ABS (вместе с Листингом_2). Здесь расположена таблица переходов между локациями, как она была задана в 3000-х строках, группы четвёрок символов, заданных для идентификации слов и т.д.

Начиная с адреса 64000, блоки, расположенные здесь, имеют постоянные адреса и не изменяют своего положения.

Ячейки 64114 и 64115 являются рабочими ячейками, в которых хранятся выходные параметры анализатора текста и подпрограмм перемещения.

С адреса 64116 расположена таблица данных, передаваемых через эту область из программы ABS в демонстрационную программу (см. строки 9957-9960 Листинга_4).

С адреса 64130 зарезервирована область размером 150 байт. Она служит для временного сохранения текущего состояния игры, которое определяется таблицей состояния флагов (100 байтов) и таблицей текущего расположения объектов (50 байтов). Сами эти таблицы расположены с адреса 64230 (этому числу равна переменная F, а переменная O на 99 байтов больше, то есть 64379). Ячейка 64130 - первая ячейка сохраняемой области - соответствует значению флага (F+0), то есть текущей локации. Это число не может быть равно нулю (не бывает нулевой локации). При старте демонстрационной программы, в эту ячейку заносится ноль (строка 9961 Листинга_4),

предохраняя, таким образом, от сбоя при попытке загрузить состояние, если оно не было прежде сохранено. Эту проверку выполняет строка 1814 Листинга_5.

С адреса 64430 расположены процедуры в кодах.

Процедура по адресу 64440 проверяет необходимость и выполняет скроллинг экрана, обеспечивая в заданных пределах сохранение на экране Ваших команд и ответов программы на них.

Процедура по адресу 64495 (строка 200) - анализатор текста.

Процедура по адресу 64930 сканирует таблицу расположения объектов, выявляя те предметы инвентаря, которые находятся в текущей локации и видны (строка 105 Листинг_4).

Процедура по адресу 64968 (строка 1502) выполняет аналогичную операцию для того инвентаря, который находится у Вас в руках (или на Вас).

Процедура по адресу 65005 (строка 1090) - процедура перемещения.

Процедура по адресу 65058 (строка 9980 Листинг_4) выполняет инициализирующие действия при старте программы, в частности, обнуляет 150-байтовую область с адреса 64280 - таблицу флагов, заносит значение стартовой локации в таблицу флагов (F+0) а также исходное расположение объектов в таблице текущего расположения.

Создание собственной игры.

Когда Вы отладите программу и демонстрация заработает, перед Вами встанет следующий вопрос: "А как мне написать свою игру?"

Во-первых, прежде чем приступать к созданию собственной игры, прогоните демонстрацию и внимательно изучите все детали. Во-вторых, даже и не подходите к компьютеру, пока тщательно не продумаете сценарий своей игры на бумаге. Вы должны, по крайней мере, разработать карту и разместить на ней все участвующие в игре объекты с указанием, к какому типу они относятся.

Сначала спланируйте весь сценарий просто на бумаге, после чего можете приступать к работе с головной программой (Листинги 1 и 2, объединённые при помощи MERGE, иными словами, то, что теперь записано на ленте _A). Ещё раз отметим, что самый простой способ понять, что и как надо делать - это максимально подробно проследить логику работы демонстрационной программы.

С чего практически начать?

Посмотрите на Листинг_4 - кроме блока строк с 7000 - это почти готовый дебют для новой игры. Надо лишь добавить к нему блоки строк с 1700 и с 1800 - СОХРАНЕНИЕ и ЗАГРУЗКА, да ещё со строки 1600 - КОНЕЦ и получится прекрасная заготовка для Вашей собственной игры. Осталось только добавить (или изменить имеющиеся) строки, отвечающие за закрытые или запертые двери, освещение, монстров и разные ловушки, которые будут в Вашей игре. Эти изменения надо ввести в блок перемещения - со строки 1000.

Скорость работы и объём памяти.

При создании собственной игры размер занимаемой памяти может стать критичным. Место, отведённое для Бейсик - программы можно расширить, отодвинув символьный набор в сторону старших адресов, но так, чтобы он не наложился на блок кодов "system" CODE, об этом уже говорилось несколько выше. Практически для этого надо изменить параметры оператора CLEAR и изменить адрес загрузки символьного набора "chr" CODE. Надо также в соответствии с новым адресом загрузки символьного набора изменить значение CHARS, задаваемое в строке 20 Листинг_3.

Другим решением бывает замена часто встречающихся чисел переменными. Например, 0 и 1 постоянно встречаются в листинге. Поэтому, например, если присвоить: LET M=0: LET J=1 и по всему тексту программы заменить 0 и 1 на M и J Вы получите существенную экономию памяти. Числа 0 и 1 могут заменяться иначе. 0 - NOT PI, а 1 - SGN PI. В то же время, если этот подход довести до неразумных пределов, он может сказаться на скорости работы программы.

Структура подпрограмм.

Подпрограмма "ПРОВЕРИТЬ" или "ОСМОТРЕТЬ" по-видимому, у Вас будет одной из крупнейших подпрограмм, но можно ускорить время её работы, если разбить её на несколько блоков. Так, например, это делает строка 1300 в демонстрационной программе, разветвляя исполнение в зависимости от номера объекта по разным ветвям. Но и дальнейшая разбивка этих ветвей тоже возможна.

FORUM

Проблемы ELITE

Внимание, друзья! Мы представляем Вам нашего корреспондента из Альметьевска - Абдрахманова Руслана. Впервые мы приводим свидетельство пилота, который лично видел РАККСЛУ.

А начиналось все с "ZX-РЕВЮ". Руслан начал выписывать его только в 93-м году и журнал стал как бы "ключом" к тайнам "Speccy", позволившим заглянуть дальше, чем LOAD и Press any key.

КОРР: ... До недавнего времени программа ELITE меня не интересовала: паутинная графика, никакого звука, непонятный сюжет. Но после публикации "The Dark Wheel" все переменилось. Я стал отчаянно торговать, воевать, искать приключения. После рейтинга Competent дело застопорилсь. Но тут вмешался случай. По воле провидения я нашел планету RAXXLA!!! И вот, как это произошло.

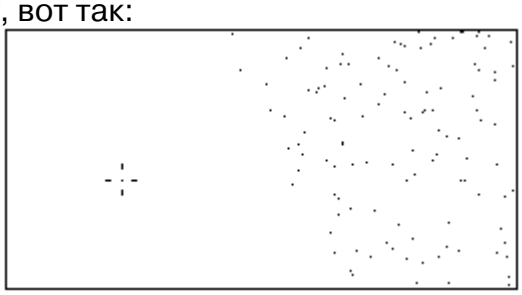
... На последние деньги я купил галактический гиперпривод и стартовал с планеты LAVE. На экране пошли концентрические круги и вдруг в блоке питания что-то щелкнуло, но программа не сбросилась, переход продолжался. Очутился я у планеты URBA. Правление - диктатура. Желая посмотреть карту, я нажал "O" и тут увидел, что нахожусь в галактике без номера, а около меня нет ни одной планеты, кроме RAXXLA! Представьте себе мои чувства?!

МФК: Это представить невозможно, это надо пережить. Кстати, уважаемый Руслан, в пылу переживаний Вы могли не заметить, что кодов для планеты URBA тоже нет в знакогенераторе игры, как и для Раккслы. Это является косвенным подтверждением того, что могут существовать планеты с именами, состоящими не только из стандартных слогов. Очень хорошо, что Вы запомнили название этой планеты,

КОРР: ... Я сразу понял, что надо записать результат, но для этого пришлось пробиваться на станцию планеты URBA. У нее был высокий технологический уровень -10, но, тем не менее, космос кишел пиратами. Меня бесконечно атаковали "Фер-де-Лэнсы", хотя статус у меня был чистый. Кроме них там были "Кобры" и астероиды.

С трудом пробившись на станцию, я обнаружил, что к продаже там имелись только золото и радиоактивные материалы, причем все это по очень высоким ценам. Я конечно записал свое состояние на ленту, но, как потом выяснилось, безрезультатно. Этот блок оказался длиной всего 99 байтов и никуда не пошел.

А на Раккслу меня не пустили, выдав сообщение "Hyperspace, Range?", хотя топлива было на 7 световых лет. Других планет рядом не было. На карте галактики я обнаружил, что она заполнена всего наполовину, вот так:



Крестиком отмечено место, где был я.

Остается добавить, что я пользуюсь версией M128, а гипердрайв был у меня установлен перед отлетом на планету DIZO.

Я не утверждаю, что всем надо взламывать свои блоки питания, но может быть мой случай кому-нибудь поможет.

ИФК: Спасибо, Руслан за интересный отчет. Нам кажется, что Вы грамотно и четко держали себя в этой экстремальной ситуации и сделали все, что могли. Наконец-то у нас появилось хоть косвенное, но свидетельство. Единственное, о чем стоит пожалеть, так это о том, что Вы не прислали распечатку отгруженного блока. Хоть в нем всего лишь 99 байтов, но он, тем не менее, мог бы быть весьма интересен. По-крайней мере, по содержащимся в

нем кодам можно было бы попробовать восстановить эту неизвестную галактику и попробовать по ней полетать.

Будет очень обидно, если у Вас этот блок не сохранился.

КОРР: У меня есть просьба к экспертам. Я был бы очень рад прочитать что-то о программах "DRILLER", "TAI-PAN", "READ HEAD", "NORTH & SOUTH". Эти игры многоуровневые и разобраться с ними начинающему пока не под силу.

ИФК: Очень многие, желающие попробовать свои силы, спрашивают нас, какие игры мы могли бы порекомендовать для исследования. Просьба Руслана - достойный ответ на такие письма.

* * *

Поклонник незабвенной ELITE, пилот Дм. Егоров из Уфы (DANGEROUS) подсказывает тем, кто имеет статус FUGITIVE и не может законным образом совершить посадку на станцию. Став невидимым с помощью CLOAKING DEVICE, Вы сможете обмануть офицеров службы безопасности.

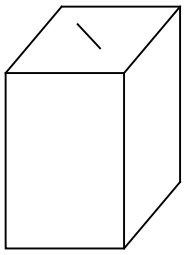
* * *

Нам пишет Владимир Бакаев из г. Челябинска.

Здравствуйте!

Вчера впервые прочитал выпуски "ZX-РЕВЮ" 1991 и 1992 г. и открыл для себя, что "Элитой" увлекся не я один. Мой рейтинг "Competent", стаж - 5 месяцев. Прочитав о том, что Вы просите писать обо всем, что встречается в этой игре очень редко, спешу сообщить об одном происшествии.

Решив как-то начать с нуля, я загрузил свой корабль на 100 Cr и с LAVE отправился на ZAONCE (извините, эту планету пишу по памяти). После J-прыжка поймал сигнал станции и стал к ней приближаться. Вскоре она появилась на экране. Но вид был необычным. Обычно это сначала точка, которая, по мере приближения становится вращающимся додекаэдром. А это была вертикально стоящая черточка. Я начал думать, что это корабль поколений, т.к. размеры его огромны. Но сигнал "S" шел от него. Подлетев поближе, я увидел, что "это" похоже на брусок.



На верхней грани по диагонали - линия. Подлетев очень близко, я решил облететь вокруг. Но не тут-то было. Корабль мой был разрушен (никаких сообщений).

Опять вылетев с LAVE и подлетев к "ZAONCE", увидел ту же картину. Подлетев близко со стороны верхней грани и не рискуя опять взорваться, в последний момент решил влететь в черточку посреди нее. (Была надежда, что это вход в станцию). И это мне (я надеюсь) удалось. Но так как маневр не был подготовлен, то, задевая стенки тоннеля, корабль, истощив энергоблоки, был разрушен.

Больше эта станция мне не встречалась. Буду рад, если кто-нибудь встречался с ней и знает, что это такое.

* * *

В "ZX-РЕВЮ" N5-6 за этот год наш читатель из Москвы Т.М. Канразе попросил осветить недавно появившуюся программу ELITE-2.

Ему не удалось установить в ней существенных отличий и она очень напоминает

"версию-3" по классификации (N3-4,1993). Для прояснения ситуации мы адресовали этот вопрос к нашим читателям. Ответ последовал незамедлительно, тоже из Москвы, причем кажется, что из невинного вопроса нашего читателя рождается новое и интересное направление исследований. Вот, что пишет Алексей Лапшин (AZURE soft.)

КОРР: Недавно, копаясь в новых демонстрационных программах, я обнаружил несколько выпусков новосибирского электронного журнала TELESHOW. В первом же номере я нашел нечто сенсационное. Мне удалось вытащить эту статью с диска в текстовый редактор, не трогая содержания. Я лишь убрал опечатки и характерные ошибки. На авторские права я, конечно же не претендую, но не послать ее Вам было невозможно. Привожу ее целиком.

ИФК: Мы свидетельствуем свое глубокое уважение редакции и создателям TELESHOW и с радостью бы вошли в контакт с целью установления взаимовыгодного сотрудничества, но, не зная адреса редакции, обречены на ожидание возможного контакта. Вдруг до них дойдет это наше обращение.

Приведенная ниже информация является фактически рекламой программы ELITE-2, причем не просто рекламой, а лучшей ее формой - пропагандой. Фирма, подготовившая ее, отлично знает свое дело, жаль только нет координат, к кому следует обратиться за приобретением новой игры.

Мы окажем всемерную помощь в рекламе и распространении игры тем авторам, которые подготовили этот интересный материал. Обращайтесь, Вы не будете разочарованы. У нас созданы дистрибуторские сети и они могут Вам помочь.

*** ELITE 2 (48 K) ***

Вы можете продолжить свою боевую и деловую карьеру, загрузив запись состояния, полученную в ELITE, но можете начать все сначала, с выдачи пилотской лицензии на планете LAVE в галактике 1. В любом случае первоначально у Вас имеется тот же корабль Кобра-МК3. Если управление Вам незнакомо, воспользуйтесь электронным справочником. В этом рекламном листе приводятся вкратце лишь отличия новой версии.

Действие игры ELITE-2 происходит в 3192 году. Галактическая Федерация по-прежнему использует высоконадежные орбитальные станции типа "Coriolis", но теперь они обязательно оборудованы усовершенствованными средствами наблюдения и опознавания. это означает, что в виду базы можно беспрепятственно использовать любые виды оружия для отражения атаки врагов Федерации: Таргонов, пиратов, нарушителей. На базах размещаются любые корабли, кроме Таргонов и пиратов и, зависнув недалеко от входа, можно наблюдать, как кто-нибудь покидает станцию и уходит в гиперпространство. Кроме того, "Coriolis" обязательно несет флот полицейских кораблей типа Viper, которые в любой момент готовы обрушиться на голову агрессоров, атаковавших станцию или мирный корабль.

В ELITE-2 получено значительно большее приближение к фантастической реальности этой игры. В отличие от ELITE, в новой версии возможно воочию наблюдать, как другие корабли взаимо- и противо- действуют, выполняя сложные маневры в пространстве, используя лазеры, ракеты и прочее оборудование, подобно тому, как это делаете Вы сами. На первых порах, особенно при полетах на неблагополучные планеты, анархические и Феодальные особенно, имеет смысл искать в пространстве союзников, и ни в коем случае не отказываться от полицейского эскорта. Следует также с умом использовать разногласия, возникающие между различными пиратскими кланами, и жадность охотников, легко отвлекающихся на собирательство обломков и контейнеров - для того, чтобы оторваться от погони.

Применение устройства невидимости теперь не является панацеей - так же, как это делаете Вы сами, другие пилоты пытаются поразить невидимого противника, используя локатор и электронную систему наведения ракет, правда, вероятность попадания по

невидимке значительно ниже.

В пространстве Вы можете встретить все старые корабли: Asp, Fer de Lance, Cobra, Pithon, Viper, Sidewinder, Adder, Krait, Thargoid. Кроме того, начали выпускаться новые корабли классов Gecko и Mamba, а Таргоны приняли на вооружение новую разновидность корабля вторжения - с шестью гранями, получивший среди охотников условное название Targoid-6.

Станции во время технического обслуживания в обязательном порядке снабжает все корабли новым оборудованием. Все корабли класса Cobra, в том числе и Ваш, получает новый локатор (включается по CAPS+L), и модифицированный компас (CAPS+C). Новый локатор наиболее полезен в ближнем бою, и прекрасно дополняет старый. В межзвездном пространстве новый компас автоматически переключается в режим гирокомпаса и показывает стороны света N-S, W-E, соответствующие северу-югу, западу-востоку в карте галактики. Гирокомпас позволяет ориентироваться в межзвездном пространстве и совершать перелеты от одной планетной системы к другой без гиперпереходов, на J-двигателе.

В пределах планетной системы новый компас в режиме 1 работает, как и прежде, показывая направление на планету, а в режиме 2 начинает попеременно показывать направление на Солнце и на планету. В частности, это очень удобно и полезно для быстрого поиска пояса астероидов, который обычно начинается на достаточно большом удалении от Солнца.

Технологический прогресс неостановим и на планетах уровня-15 Вы уже можете купить Super-Laser (33000 Cr). Кроме того, на экспериментальных двойных станциях, расположенных вблизи солнц в планетарных системах, лежащих на главной диагонали, возможно приобрести новый корабль "Кобра MK4", но об этом будет сказано ниже, в разделе, посвященном новым миссиям.

Произошли кое-какие изменения во внутренней и в экономической политике. Ввиду готовящегося крупномасштабного вторжения Таргонов, Федерация решила легализовать торговлю оружием - Firearms перестали быть запрещенным товаром.

Полицейские корабли - неприкосновенны. Уничтожение полицейскогого равно как и уничтожение орбитальной станции, принадлежащей Федерации, превращает Вас в нарушителя номер 1. В галактиках 2-8 таких нарушителей встречает вблизи планеты полицейский кордон.

Как и прежде Вам могут быть выданы специальные миссии: по спасению беженцев со станций, солнце которых превращается в новую, по уничтожению корабля-невидимки и по уничтожению базы, захваченной Таргонами. Но, кроме этих трех миссий (которые в новой игре могут повторяться неоднократно), добавлено несколько новых.

Миссия-4 - "Mineral Life". В планетных системах на главной диагонали галактики в поясе астероидов Вы можете, разрабатывая астероиды, подобрать необычные объекты - "минеральную жизнь". Каждая тонна породы, содержащей "живые кристаллы", может быть продана за 1000 Cr. Однако, существуют особые условия для сохранения жизнеспособности этих привередливых космических овечек. Их должно быть не менее 3-х тонн и не более 63-х тонн, потому их нельзя продать в количестве 1 или 2 тонны одному покупателю.

В момент выполнения гиперперехода они питаются драгоценными камешками Вашего сейфа, съедая по 1 грамму за каждую тонну живого груза. И если скольких-то каратов драгоценного сена им не хватит, в Ваших отсеках окажется обычная дешевая порода. А в случае гибели этой минеральной жизни (по любой причине) отсеки вашего корабля окажутся зараженными кристаллическим лишаем, который уже не даст Вам возможность вновь превратить свой корабль в летающую ферму, пока отсеки не будут очищены при выполнении миссии по спасению беженцев или при доставке Desent'a.

Миссия 5 - "Alien Computer".

Уничтожив корабль Таргонов, можно подобрать контейнер, содержащий это устройство, позволяющее запускать имеющиеся на борту Alien Items вместо ракет. Для запуска используются клавиши "D+F", но этот вид оружия эффективен только против самих же Таргонов.

Следует добавить, что Alien Computer обеспечивает еще и автоматический возврат запущенных Alien Items на борт корабля после уничтожения цели.

Миссия 6 - "Кобра MK4". В планетных системах на главной диагонали галактики вблизи Солнц расположены спаренные станции "Кориолис" - это экспериментальные лаборатории Федерации. При подлете к ним, Вам предлагают купить корабль класса "Кобра MK-4". Однако, добраться сюда не просто. Многочисленные корабли, а в основном это Asp'ы и Gecko ведут здесь ожесточенные сражения и без эскорта Alien Items делать там нечего.

От сопровождения следует вовремя избавиться, чтобы "двойные базы" не приняли Вас за враждебную флотилию. Для покупки нового корабля требуется 250000 Cr, но он стоит этого! Имея Кобру MK4, Вы можете купить до 4-х устройств сразу. Причем покупка четырех Large Cargo Bays увеличивает грузоподъемность до 80 тонн, а покупка нескольких Energy Units дает соответствующий рост скорости подзарядки энергетических отсеков. На Кобре MK4 возможен перелет в любую галактику - достаточно нажать номер галактики от 1 до 8, пока идет предстартовый отсчет. Лазеры на новой Кобре установлены на поворотном лафете, на "горбу" корабля. В любой момент Вы можете повернуть лафет по часовой стрелке (CAPS + A) и поставить на нужный экран нужный Вам лазер.

При нажатии клавиши "J" даже и при наличии поблизости больших масс достигается значительное увеличение скорости и маневренности Кобры MK4.

Миссия-7 - "Space War". В течение более, чем столетия Таргоны готовились к войне с цивилизованными галактиками. Их эскадры бороздили межзвездное пространство, и лишь изредка их корабли появлялись на границах планетных систем. Совсем недавно армада кораблей вторжения вновь пыталась захватить орбитальную станцию федерации, но благодаря отважному коммандеру база была взорвана и угроза нашествия приостановлена.

Но вот наступил час Икс и Таргоны начинают крупномасштабное вторжение в одну из галактик. Каждый раз, когда Вы получаете сообщение "MAYDAY! Galactic N Invaded!", область, захваченная Таргонами, увеличивается. Планетарные системы, находящиеся в состоянии активных боевых действий, обозначаются как RESISTING. Системы, в которых сопротивление подавлено, а орбитальные станции захвачены Таргонами, отмечены, как OCCUPIED. На оккупированных планетах флот Таргонов организует кордон при подлёте к планетам и гиперперелёты между охваченными огнём планетами становятся опасными. Межзвёздное пространство заполнено эскадрами Таргонов, перехватывающими корабли федерации.

Вы можете остаться посторонним наблюдателем, но если Вы настоящий патриот Федерации, то непременно пожелаете вступить в ряды её добровольных защитников.

Сделать это можно на любой воюющей, но ещё не оккупированной планете. Вам присваивается звание - лейтенант, и выдаётся первая миссия - эвакуация гражданских лиц, раненых и беженцев из воюющих (RESISTING) систем. После того, как командование решит, что Вы справляетесь с этим заданием нормально, оно присваивает Вам звание - капитан и даёт задание по доставке оружия и продовольствия на воюющие планеты. Исполняя эти первые две миссии, Вы совершаете полёты между воюющими и невоюющими системами. Вам достаточно часто будут встречаться как одиночные Таргоны, так и целые флотилии перехватчиков. Это обеспечит достаточную практику для выполнения последующих миссий.

Когда, наконец, Вы получите звание - майор и задание по доставке десанта на оккупированные территории, Вы должны быть уже достаточно опытным бойцом, чтобы преодолеть кордон из кораблей Таргонов и прорваться к захваченной ими базе. Взяв на борт десантный корабль типа Sidewinder, Вы должны будете добраться до базы Таргонов и высадить десант. При этом Вы не сможете использовать Alien Computer, т.к. грузовой отсек занят десантным кораблём. Не сможете использовать и ракеты, вместо них запускается десантный корабль.

Запуск десанта следует выполнять с как можно более близкого расстояния к базе, чтобы до телепортации через силовое поле станции его не успели расстрелять Таргоны, обороняющие базу. Сразу после успешной высадки десанта Вы можете сами телепортироваться на станцию, включив Docking Computer - база уже контролируется

силами федерации. Чем глубже в тыл Таргонов Вам удастся высадить десант, тем большее количество оккупированных планет выйдут из-под контроля Таргонов.

Последняя, предлагаемая Вам вместе со званием полковник, миссия состоит в поиске командной базы Таргонов, расположенной где-то в глубинах космоса. Штаб сможет Вам сообщить лишь приблизительное название планетной системы, на расстоянии в несколько световых лет от которой находиться эта база. Такое же задание даётся всем пилотам, поступившим на службу добровольно. В принципе, исход войны уже предрешён, цивилизованные галактики снова в безопасности.

Следует отметить, что добровольцы воюют на своих собственных кораблях, которые вооружаются на собственные сбережения. Но при этом они в значительной степени остаются самостоятельными в выборе маршрутов движения, действиях и т.д.

* * *

KOPP: И, в завершение, несколько комментариев от новосибирцев:

Во-первых, программе требуется дисковод. Это связано с тем, что программа сильно защищена ..... (строки письма корреспондента с соображениями о практическом подходе к копированию программы опущены по известным причинам..).

Во-вторых, в программе существует ещё множество тонкостей, которые Вы сами сразу же найдёте. Есть и ещё некоторые “вещи”, о которых Вы узнаете, поиграв приличное время.

В-третьих, программе требуется очень узкий список аппаратных и программных средств. Так, авторами сообщается о необходимости работы в TR-DOS 5.03. Испытания показали, что и на TR-DOS 5.04s программа тоже почему-то работает.

В-четвёртых, в программе корабли больше соответствуют фирменному описанию. Те счастливцы, которые имели доступ к фирменному описанию, знают, что увидев "Кобру MK3" или "Питон" следует выключить лазер и прикинуться "бедной овечкой" ...

ИФК: Здесь есть какой-то элемент непонимания. Дело в том, что к "счастливцам" следует отнести всех читателей "ZX-РЕВЮ", т.к. фирменное описание у нас есть и то описание, которое мы давали в "ZX-РЕВЮ" №2, 1991г. - это и был наш перевод фирменного описания, правда чуть-чуть сокращённый ради экономии места. При этом, к сожалению, выпали чертежи кораблей, но и их мы потом тоже дали вдогонку в "ZX-РЕВЮ" №10, 1991г.

KOPP: ...

В-пятых... Но пожалуй уже хватит. Приобретите себе программу и узнаете обо всём сами".

В конце от себя добавлю, что по состоянию на середину октября 1993 года на московских радиорынках настоящей ELITE 2 замечено не было (а у новосибирцев, между прочим, она есть!). То, что уже длительное время продаётся в Москве под этим названием - Версия 3 по Вашей классификации.

ИФК: Ну, вот, конец с концами и сошлись. Ясно теперь, почему Камразе Т.М. не нашёл в ней ничего особенно нового.

KOPP: И ещё любопытная информация по этому поводу. На компьютере AMIGA (500, 500+, 600, 1000, 1200 . . и т.д.) к августу 93-го года тоже, как ни странно, существовало три версии игры. Это ELITE (классика), ELITE Plus (соответствующая "версии 3"). И новая (!) ELITE 2.0, которая наверное и похожа на вышеописанную "спектрумовскую" ELITE 2.

Кстати, каталоги игр для АМИГИ подозрительно напоминают спектрумовские. Один только сериал DIZZY (1-8!) чего стоит, но об этом можно было бы писать ещё долго.

ИФК: Спасибо, Алексей. Наши почитатели DIZZY теперь кстати будут знать, что их ждёт в ближайшие годы.

В заключение Алексей очень тепло отзывается о своей машине ATM-turbo 512++ v4.50 (память как у АМИГИ, возможности как у СПЕКТРУМА). Второй знак "плюс" означает доработку до почти полной совместимости - классная машина! Алексей очень рекомендует

также текстовый редактор ZX-WORD v1.01 из Харькова.

Ещё раз спасибо, Алексей, мнение эксперта очень важно нашим читателям.

Вопросы совместимости.

Алекс Вебер из посёлка Краснозёрское Новосибирской области готов помочь тем, у кого есть проблемы с запуском программы Silent Service. Суть проблемы в том, что после загрузки БЕЙСИК - блока, программа стартует, появляется надпись S. Service, выдаётся короткий звуковой сигнал и ... программа сбрасывается.

Вскрытие показало, что загрузчик в машинных кодах встроен в БЕЙСИК после REM и там сразу за выдачей звукового сигнала стоит опрос порта 31 (Кемпстон-джойстик). Если на джойстике - 0, то программа сбрасывается.

Итак, нужно позаботиться, чтобы во-первых Кемпстон-джойстик был подключён, а во-вторых, чтобы на нем был не ноль, например, нажать кнопку джойстика во время опроса. После нормального старта загрузки кнопку можно и отпустить.

Другой подход - убрать этот опрос из загрузчика.

Непонятно, какие цели преследовал тот, кто таким образом переделал загрузчик.

И ещё: в программах, взломанных INX'S SOFT и MICROPOL, а также Б. Гильбертом после 90-го г. уже встроены POKES, но стоят за оператором REM. Поэтому если Вы хотите ими воспользоваться, то удалите REM в БЕЙСИК - загрузчике и запустите его снова, а затем грузите программу дальше.

Спасибо за "Программирование в машинных кодах". По Вашей книге я научился разбираться с кодом фирменных программ.

ИФК: Спасибо и Вам, Алекс. Ваши слова - для нас отличная реклама. Мы знаем, что эта книга пользуется особой популярностью и надеемся на то, что наши новые книги, стоящие в плане на 1994 год тоже получат Ваше признание.

* * *

Масленников В.Г. из г. Чебоксары пишет о том, что почти на всех отечественных моделях не идут программы SHORT CIRQUIT и TOP GUN. Причина - в отсутствии порта FFH.

Опрашивание порта в программе SHORT CIRQUIT выглядит примерно так:

```
LOOP  LD A,28H
      IN A,(FFH)
      INC A
      JR Z,LOOP
      RET
```

Подпрограмма зациклена до тех пор, пока с порта FFH поступает число FFH. Сделано это для того, чтобы спрайты на экране не мерцали.

Вместо этого фрагмента можно вставить в программу другой, имеющий ту же длину и не портящий никаких регистров, кроме аккумулятора:

```
      PUSH HL
LOOP  LD A,(HL)
      INC HL
      INC A
      JR NZ,LOOP
      POP HL
      RET
```

Для тех же, кому важнее конечный результат, можно порекомендовать загрузить блок длинной 41986 в копировщик COPY - COPY под адрес 23296 и дать:

```
POKE 33621,229; 126; 35; 60; 32; 251; 225; 201
```

Аналогично ремонтируется и TOP GUN. Для этого блок 40100 загружается в COPY - COPY под адрес 24625 и:

```
POKE 26642,251; 118; 118; 0; 0; 0; 0; 0
POKE 56160,229; 126; 35; 60; 32; 251; 225
```

Значение такого совета трудно переоценить и мы сердечно благодарим Владислава Геннадьевича от имени всех читателей, которым помогли его советы. Вполне возможно, что с такой технологией удастся "отремонтировать" ещё не одну программу. Ждём сообщений.

Советы и секреты.

Нам пишет THE CAT из г. Харькова.

Привет, ИНФОРКОМ! Посылаю выполненное мной в жанре "компьютерная новелла" описание игры SAMURAI WARRIOR (к сожалению, фирму - изготовителя указать не могу - известно только, что взломана она была командой S. S. CAPTAIN & BAMSEE) в 1989 году.

Жанр игры определить довольно трудно - я бы назвал это ADVENTURE/ACTION. Игру отличает динамичность, прекрасно выполненная графика и интересный сюжет. Напечатайте, если можно...

ИФК: Печатаем, причем с огромным удовольствием. Пробным камнем для "новеллы", как Вы знаете, является вопрос ЗАХОЧЕТСЯ ИЛИ НЕТ ЧИТАТЕЛЮ СЫГРАТЬ В ЭТУ ИГРУ?! Попробовали Вашу новеллу на себе и так захотелось сыграть... В общем, ещё десяток подобных новелл и у нас некому будет выпускать "ZX - РЕВЮ", все будут играть, играть и ещё сто раз играть.

Единственное, что успокаивает, так это то, что написать так, как это сделали Вы, очень непросто и вряд ли на нас в ближайшее время обрушится ЦЕЛЫЙ ДЕСЯТОК.

Мы надеемся, что Вы простите небольшие комментарии от нас, вставленные после Вашей новеллы. Сами ведь понимаете, нам тоже хочется прислониться к хорошему делу, а то ещё уволят по сокращению штатов...(шутка).

КОРР: И ещё в раздел "ФОРУМ". В игре "IMPACT" можно получить вечную жизнь, если в строке 30 загрузчика убрать команду REM перед стоящим там POKE (У меня версия игры, в которой после загрузки первого блока, остановки и нажатия "BREAK" на экране появляется надпись "Disked by Billy".

Вот и всё. С большим уважением. THE CAT.

P.S. Кто бы помог (если можно) достать программы "LORD OF THE RINGS" и "SHADOWS OF MORDOR"?

310142 Харьков а/я 1968, THE CAT.

* * *

Нам пишет Шалыминов С.П. из Нижнего Новгорода.

КОРР: Хочется дополнить план игры к программе Aliens, опубликованный в № 5-6 за этот год. Можно видеть, что комнаты пронумерованы от 1 до 255, но нигде на плане нет комнат с номерами 249...252. Попасть в них можно только из комнаты 248, предварительно расстреляв там все коконы. План принимает вид:

```
          XXX
          255
          254
    .... 205 ....
          253
250 249 248 251 252
          247
          ...
```

ИФК: Спасибо за ценное дополнение. Сейчас уже не установить, то ли в письмах авторов эти комнаты были не разысканы (черновики не сохранились), то ли это мы при наборе плана их пропустили. Скорее второе, чем первое. Тем не менее, мы рады Вашему письму и если вина лежит на нас, то это к счастью, ведь благодаря этой ошибке мы имели честь познакомиться с Вами.

КОРР: ...Я это узнал только методом проб и ошибок. Долгое время я играл, держа всю карту в памяти, пока не устал. Тогда составил свой план и обнаружил этот парадокс с

"пропавшими" комнатами. Начал их искать, и вот … результат перед Вами.

Я тоже очень люблю составлять карты к играм и составил их к Robin Good, Dan Dare, Equinox, Gunfright, Death Wish 3, Resque, на подходе карта к Strike Force Cobra. Если кто-то нуждается в них, могу прислать.

603163, Н. Новгород, Ул. Бринского, д.5, к.1, кв.80 Шалыминову С.П.

Ещё я люблю разыскивать POKES или проверять взятые из других источников. Вот несколько, найденных или лично проверенных POKES:

Stop the Express - 34467,0; 34929,0; 35260,0

Curse	63033,0; 64613,0
Renegade	342204,182; 42789,182
Nebulus	32913,0
Zybex	45277,0; 45368,0
L. Ninja	236578,0
Robot Escape	54697,0; 54698,50; 54699,3
Bionic	34247,0; 34274,0

Мне очень нравятся Ваши публикации по ELITE. Сам я пользуюсь версией M128, переделав её под дисковод. Вот только никак не могу сделать отгрузку/подгрузку для диска. Не могу понять, как там это организовано.

ИФК: Может быть кто-то даст четкие советы по этому поводу нашему уважаемому корреспонденту и другим читателям?

КОРР: ...Я тоже раскрутил словарь в программе и тоже знаю, что кода "XX" в нем нет, но я точно знаю, что есть названия планет, которые состоят не только из тех слогов, что есть в словаре, так что рано предполагать, что "Раккcла" не существует.

ИФК: Это интересно! Особенно если учесть информацию, полученную от Абдрахманова Руслана (см. выше).

КОРР: В этой версии почему-то не проходит безтопливный перелёт со станции на станцию, а вот в версии Be-Be Soft у меня это получалось.

В целом я отношу версию M128 К сложным версиям. Пираты здесь, как правило, ASP MK II, KRAIT, SIDEWINDER, реже - PYTON, совсем редко - COBRA MK III. Ни разу не встретил ADDER. Fer-De-Lance можно встретить в двух случаях: если летишь к звезде и статус - Clean, он не стреляет первым. Если же у Вас статус Fugitive - он нападает сразу, даже если появится в переднем экране.

ИФК: Типичное поведение "Охотника за призами", занимающегося боевыми действиями ради спортивного интереса. Можете быть уверены, его рейтинг - ELITE или близкий к нему.

КОРР: ... Полиция (VIPER) весьма маневренна. На неё лучше не нападать. А станция - вообще святыня! Стоит один раз по ней выстрелить и по одному начнут вылетать полицейские корабли, до 10 штук. Как правило, это смертельно, если не уйти в гиперпространство.

Очень часто встречаются астероиды с отшельниками, причём по внешнему виду их не отличить от обычных. Если по нему выстрелить, начинает маневрировать и выпускать корабли - KRAIT и SIDEWINDER.

Вообще, кажется, что это самая интересная версия игры из тех, что я видел.

ИФК: Мы надеемся, что наши пилоты в поисках острых ощущений учтут это мнение. Спасибо за Ваше письмо.

* * *

Гаврилов Сергей Александрович из г. Костромы приводит проверенный им POKE для программы REX-1

23624, 0; 39402, 0; 40057, 0

* * *

Свои POKES, найденные лично, Вам предлагает читатель из Красноярска Татаренко А.А.

1. SPELLBOUND - 27871, 0; 36133, 0; - энергия или
   35101, 195; 35102, 53; 35103, 206
2. DIZZY 2 - 29289, 201 - неуязвимость
3. DIZZY 3 - 42481, X - кол-во жизней
   63001, 0 - беск. жизни
4. DIZZY 4 - 29623, 0; 29624, 195 - беск. жизни
5. DIZZY 5 - 51291, 0 - беск. жизни
6. STAR BOWLS - 47806, X - кол-во жизней
   46278, 0 - беск. жизни
7. SCUMBALL - 49098, X - жизни
8. COCORAMA - 43855, 0; 48658, 0 - беск. жизни
9. TOMCAT - 37174, 0 - беск. жизни
   37171, 201 - неуязвимость
10. TRACER - 50273, 0 - время
    50613, 0 - энергия
    50649, 0 - беск. жизни
11. TWISTER - 42412, 0; 42490, 0 - энергия
12. DARK FUSION - 50407, 0; 50408, 0 - беск. жизни
13. XENON - 25148, 201 - жизни
14. CONQUEST (ERBE SOFT) - 62370, 0 - останавливает противников

* * *

А эти POKES пришли к нам из Белоруссии, их прислал наш читатель из г. Гомеля Владимир Хронов.

Gomel Technosoft'92

Все POKES даны для игр в развёрнутом виде.

1. FROST BYTE 33989, 0; 36560, 0
2. JOE BLADE-3 43059, 195
3. KRION 45004, 0
4. PANAMA JOE 38633, 60
5. GOODY 46187, 0
6. CAPTAIN TRUENO 38310, 24; 38363, 201
7. DRAKONUS 64215, 0
8. SAVAGE 1 39446, 0
9. ELVEN WARRIOR 36767, 0
10. CROM 56190, 0 - Energy
    56220, 0 - Lives
11. YETI 47894, 0 - Lives

* * *

Семёнов Алексей из г. Балаково лично проверил следующие POKES:

ELEPHANT ANTIC 35354, 255; 35363, 255
BALL BREAKER 35840, 0
BALL BREAKER 2 38874, 0; 35938, 0; 35729, 99
KING KONG 30033, 0 - бессмертие
30038, 0 грязь к ногам не липнет;
30053, 24 огонь не жжёт;

30063, 24 бочки не сбивают.
30043, 24
30048, 24

ALPIN GAMES — остановить программу кнопкой BREAK, удалить строку 5620 и дать CONTINUE.

PHOTO ALBUM — 56203, 0
56364, 0 время стоит
56249, 0
51765, 0 неогр. ошибки

IMPACT — пароли:
EGGS, CHIP, LEAD, TICK, CASE, FACE, J.D.

BREAKNECK — 46090, 0 - жизни
46013, X - (0<X<200) - номер комнаты.

* * *

Наш читатель Шабалин М.М. из г. Ухты вскрыл "жучок" в программе "RENEGADE" и готов им поделиться.

Этот "жучок" относиться к тем, кто не может пройти четвёртый уровень игры (босса с пистолетом) и заключается в том, что когда в Вас стрельнет "босс", сразу нажимайте кнопку "PAUSE" (если Вы выбирали клавиатуру, то должны были её назначить). После этого летящая пуля исчезает и это облегчает прохождение уровня.

* * *

У Алекса Кучкина из г. Петропавловска - Камчатского для наших читателей заготовлено несколько POKES и есть один вопрос.

TOTAL ECLIPSE — 47793, 0 - вода,
47690, 0 - здоровье.

ROBOCOP — 39577, 24 - энергия.

N. O. M. A. D. — 40703, 0 - бессмертие.

NINJA COMMANDO — 51763, 0 - бессм.

TURTLES — 53744, 0 - энерг.
47503, 0 - жизни.

MASTER BLASTER — 49988, 0 - жизни.

К сожалению, прочие POKES мы дать не смогли по причине неразборчивости письма.

Теперь вопрос. Алекс очень любит авиаимитаторы, особенно F-16 фирмы Digital Interceptor. К сожалению, после поражения программа высвечивает надпись PLAY ... и чего-то ждёт. Есть подозрение, что в ней не все блоки. Может кто-то дать раскладку блоков этой программы?

* * *

Изящный букет секретов прислал наш юный читатель из г. Находка Приморского Края Максим Кобяков. Сначала несколько POKES:

1. AFTER BURNER - 37934, 0; 37935, 9; 37936,0
2. FRIGTHMARE - 44051, 0; 48455,0
3. LAST DUEL - 37605,0; 40063,0 - 1 игрок
4. LED STORM - 37337, 201
5. RING WARS - 39417, 0; 39534, 0
6. STREET FIGTHER - 42147, 195

7. ZYBEX - 45277, 0; 45368, 0
8. DIZZY 5 - 51291, 0; 41815, 0

Все POKES лично проверены Максимом.

Кроме того, вниманию читателей предлагаются пароли для программы MEGANOVA:

26719 - 2-ая миссия;

16640 - 3-я миссия.

Полезный "жучок" из программы NEW YORK WARRIORS:

Если начать игру вдвоём и в первом блоке убить первого игрока, чтобы он больше не появлялся, а вторым игроком пройти этот блок до конца, то во втором блоке первый игрок появиться снова, но уже с 15-ю жизнями.

Этот способ применим в любом блоке.

* * *

Просьба о помощи.

Наш постоянный читатель из г. Талнах Красноярского края Фёдор Юхнович обращается к экспертам - профессионалам. Он считает, что проработка программ - имитаторов им удаётся лучше всего, пример тому работы Хоминича и Жарова по программе ACADEMY.

ИФК: Дорогие друзья! Мы ежемесячно получаем тысячи писем, в которых нам выражается благодарность за то, что мы делаем. Мы ценим Ваше участие и рады этому, но наконец-то к нам пришло письмо с оценкой труда наших корреспондентов. Наверное, не очень справедливо оставлять за кадром тех, кто бескорыстно делится с Вами всем, что знает и умеет.

ПРИМЕЧАНИЕ: статьи наших корреспондентов мы помечаем символом (C) ("копирайт"), свидетельствуя тем самым своё уважение к их авторскому праву. Статьи, не помеченные этим символом и без указания автора, принадлежат "ИНФОРКОМу" и выполнены его сотрудниками в порядке служебного задания.

КОРР: ... Помогите разобраться с программой-имитатором космического полёта - программой ENTERPRI (аналог ELITE).

ИФК: Мы с этой программой не знакомы и тоже с интересом выслушали бы отзывы о ней.

СОВЕТЫ ЭКСПЕРТОВ

THE DOUBLE

J.S. Ltd.

Эксперт Матвеев Ю.А. г. Москва.

Вы когда-нибудь представляли себя в роли менеджера футбольного клуба? Если нет, то попробуйте сыграть в неплохую игру "THE DOUBLE" ("ДУБЛЬ") фирмы J.S.Ltd.

Программа "THE DOUBLE" является достойным представителем игр жанра BUSINESS/MANAGEMENT. Любители футбольных баталий, имеющие в своем распоряжении "Спектрум" и эту игру, получат возможность испытать свои способности в роли руководителя английского футбольного клуба.

Ваша задача - привести свой клуб к золотым медалям чемпионата и выиграть Кубок, то есть сделать "дубль", добившись двойного успеха.

Поначалу Вам достается команда среднего уровня, имеющая в своем составе несколько неплохих игроков, на которых, собственно говоря, держится все. Но по мере того как Вы, используя выделенные Вам деньги, покупаете новых, более талантливых футболистов и проводите мудрую политику в подборе состава на очередной матч, Ваша команда начинает расти на глазах. В начале чемпионата Вы присматриваетесь к футболистам, экспериментируете с составом. В конце концов Вы приходите к выводу, что с некоторыми игроками Вам не по пути и с ними придется расстаться. Первые попытки продать слабых футболистов в другие клубы будут неудачными, и это понятно. Кому нужен лишний балласт? Ведь каждому игроку приходится платить немалую зарплату из бюджета клуба. Многие менеджеры придерживаются принципа: "Мы не настолько богаты, чтобы покупать дешевых игроков". Возьмите и Вы этот принцип на вооружение. Не отчаивайтесь, если не удается быстро продать футболиста. Вполне возможно, что его купят позже, когда из-за травм ведущих игроков некоторые менеджеры будут вынуждены снизить свою требовательность.

После загрузки программы Вы сразу попадете в главное меню:
MAIN MENU
CLUB REPORT (отчет по клубу) LEAGUE DETAILS (информация о ходе чемпионата)
FA FIXTURES (матчи на Кубок) CONTROL MENU (меню управления игрой)
QUIT GAME (прекратить игру)
CONTINUE (продолжить игру)

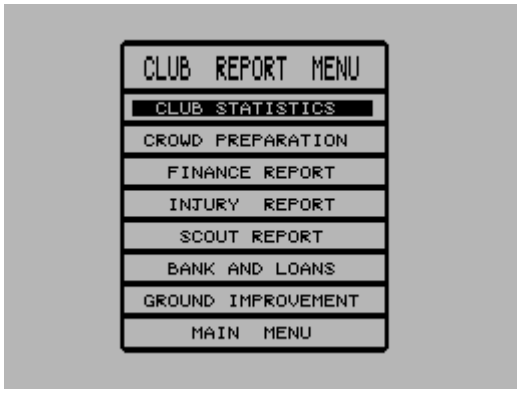
В начале игры установлены следующие управляющие клавиши:
5 - влево 7 - вверх
6 - вниз 9 – вправо
ENTER - выбор

Вы можете использовать джойстик, если в опции CONTROL главного меню, произведете соответствующую застройку. Там же находятся опции сохранения (SAVE GAME) и загрузки (LOAD GAME) отложенной игры.

После выбора управления Вы переходите непосредственно к игре.

Итак, Вам предлагается возглавить футбольный клуб LEICESTER CITY, который выступает в Первом Дивизионе. Отношение к Вам со стороны дирекции клуба достаточно хорошее (об этом можно судить по строке MANAGEMENT CREDIBILITY <кредит доверия> над главным меню). Руководство клуба надеется, что Вы оправдаете высокое доверие, которое Вам оказано, и приведете команду к золотым медалям.

CLUB REPORT MENU



Здесь есть следующие разделы:
CLUB STATISTICS (статистика клуба).
CROWD PREPARATION (планирование посещаемости матчей).
FINANCE REPORT (финансовый отчет).
INJURE REPORT (отчет по травмам игроков).
SCOUT REPORT (отчет наблюдателя).
BANK AND LOANS (расчеты с банком).
GROUND IMPROVEMENT (развитие стадиона).
MAIN MENU (возврат в главное меню).

Раздел CLUB STATISTICS:

Выбрав этот раздел, Вы сможете узнать о средней посещаемости Вашего стадиона (AVERAGE GATE), О количестве мест на стадионе (GROUND CAPACITY), а также такие данные, как номер Дивизиона, в котором играет ваша команда (LEAGUE DIVISION), место, занимаемое командой в турнирной таблице (LEAGUE POSITION) и количество зрителей, побывавших на последней домашней игре (LAST HOME GATE). Тут же Вам предлагается еще одно подменю:

SQUAD DETAILS (Информация по команде).
STAFF DETAILS (Информация по персоналу).
MENU (Возврат в предыдущее меню).

После выбора опции SQUAD DETAILS, Вам представится возможность познакомиться с составом команды, посмотреть статистику на каждого игрока. Здесь приведены данные и о еженедельной зарплате игроков. В низу экрана еще одно меню:
SELL PLAYER (продать игрока).
PRINTOUT (вывести данные на принтер).

MENU (возврат в предыдущее меню). При желании Вы можете продать любого футболиста. Установите курсорную строку на фамилию игрока и выберите опцию SELL PLAYER.

При наличии заявок на покупку футболиста на экране появится таблица с названиями клубов, заинтересовавшихся Вашим предложением, и суммами, которые они предлагают за игрока. Чтобы отказаться от сделки надо выйти из этого подменю. Для оформления сделки, нужно выбрать курсорной строкой устраивающую Вас сумму и нажать FIRE. С этого момента футболист уже не играет в Вашей команде, а деньги за него поступили на Ваш счет в банке.

Вы можете оценить силу каждого игрока по зарплате (WAGE), которая ему выплачивается. Но будьте осторожны! Сумма еженедельных выплат не всегда отражает уровень игры футболиста. Это, возможно, связано с привилегированным положением некоторых игроков. Звездный час у них уже давно в прошлом. Однако, за свои прежние заслуги они еще имеют хорошую зарплату, которая может быть гораздо выше, чем у молодых. Впрочем, бывают и другие ситуации, молодым, подающим надежду футболистам, руководство клуба иногда также завышает зарплату, как бы авансируя их будущие успехи. Вы, кстати, не занимаетесь этими вопросами и все решения о зарплате лежат на совести директора клуба.

Чтобы верно оценить уровень игры каждого отдельного игрока, Вы нанимаете независимый экспертов - наблюдателей (SCOUT), которые могут описать достоинства и отличительные характеристики любого интересующего Вас футболиста.

Для начала войдите в подменю STAFF DETAILS. У Вас имеется один эксперт, который трудится под вашим чутким руководством с окладом в 450 фунтов стерлингов в неделю. Вы можете нанять второго эксперта. Это будет стоить еще 450' в неделю. В этом подменю предлагается за те же деньги нанять и физиотерапевта (PHYSIOTHERAPIST), который будет необходим, как воздух, когда игроки начнут получать травмы в матчах. Впрочем, Вам не обязательно нанимать терапевта и второго эксперта. В ваших силах даже уволить и первого эксперта, дабы сэкономить средства и побыстрее набрать круглую сумму в банке, но делать этого не рекомендуется из-за проблем, которые могут впоследствии возникнуть.

Раздел CROWD PREPARATION:

Здесь Вам сообщают о месте проведения следующей встречи. В том случае, если Ваша команда играет дома, Вам предлагают ввести Ваш прогноз о количестве зрителей, которые придут на матч. Эти цифры будут использованы при подготовке полиции к матчу, которая будет обеспечивать безопасность на стадионе. Если Вы ошибетесь в своем прогнозе и назовете заниженные цифры, то полиция, не готовая к большому количеству зрителей, не сможет поддержать порядок на стадионе и болельщики могут устроить беспорядки. На центральных матчах тура всегда присутствуют эксперты из Федерации футбола и в случае возникших беспорядков Ваш клуб может быть оштрафован на круглую сумму, если в Федерации решат, что это произошло по вашей вине. Крайне нежелательно так ошибаться несколько раз подряд, Вы можете даже потерять свое кресло, поэтому будьте осмотрительны и перед каждой домашней встречей уделяйте внимание безопасности стадиона. Завышенный прогноз хоть и не влечет за собой таких мрачных последствий, но все же влияет на финансовое положение клуба, а значит и на кредит доверия к Вам со стороны руководства.

Раздел FINANCE REPORT:

Здесь Вы получите полную раскладку всех доходов и расходов клуба.
GATE RECIEPTS (плата за входные билеты).
TRANSFERS (плата за продажу/покупку игроков).
PLAYERS WAGES (Зарплата игроков).
STAFF WAGES (зарплата персонала).
TRAVEL AND HOTEL (расходы на переезды и гостиницы).
FIRES AND DAMAGES (штрафы и повреждения).
POLICE EXPENCES (оплата полиции).

GROUND RENT (арендная плата за землю).
GROUND IMPRONEMENT (расходы на расширение стадиона).

Раздел INJURY REPORT:

В этом разделе сообщается о травмах, полученных футболистами. Если в Вашей команде все благополучно, то выводится сообщение: NO INJURIES (травм нет). Если есть травмированные, то сообщается фамилия футболиста, вид травмы и количество недель, на которое игрок выбывает из основного состава.

Вам предлагается решить: освободить травмированного футболиста от тренировок полностью или нет.

В разделе есть подменю, в котором определяется вид тренировки для травмированного футболиста:

NO WORKOUT (без тренировки).
LIGHT WORKOUT (легкая нагрузка).
MEDIUM WORKOUT (средняя нагрузка).
HARD WORKOUT (тяжелая нагрузка).
Игроки могут получить следующие виды травм:
TORN LIGAMENTS (порваны связки).
BROKEN ARM (перелом руки).
ANCLE INJURE (повреждена лодыжка).
BROKEN LEG (перелом ноги).
HARMSTRING - растяжение сухожилий.
BROKEN NOSE - перелом носа.

Существуют и другие виды травм, если Вы достаточно глубоко погрузитесь в эту игру, то вам придется иметь с ними дело.

При легких видах травм игроки выбывают на небольшой срок, а при тяжелых травмах этот срок может достигать полугода.

Раздел SCOUT REPORT:

Этот раздел позволяет ознакомиться с отчетами экспертов-наблюдателей, которых Вы посылаете на очередные матчи с целью изучения уровня игры футболистов, выставленных на продажу другими клубами. Вы, конечно, можете рассчитывать только на свои силы и обойтись без услуг экспертов, но в этом случае высока вероятность купить "кота в мешке". Для того, чтобы отправить эксперта на матч, выберите с помощью курсорной строки опцию SEND ON MISSION, затем задайте дивизион, в который направляется эксперт, а потом выберите команду и футболиста, которого следует посмотреть. Эксперт прощается с Вами и со словами: "Увидимся после матча" отправляется выполнять поручение.

Прочитать последнее сообщение эксперта можно, если выбрать опцию READ LAST REPORT. Аналогичные действия проводятся и со вторым экспертом (если он есть).

Рекомендуется в начале сезона, пока не начались переходы футболистов из клуба в клуб, оценить с помощью экспертов игроков своей команды.

После очередного тура чемпионата эксперт возвращается с отчетом на футболиста, которым Вы заинтересовались. Отчет состоит из пяти характеристик плюс оценка приблизительной цены за игрока. Эксперт описывает футболиста, используя спортивный сленг. Словарь эксперта приведен в конце настоящего описания.

Раздел BANK AND LOANS:

В этом разделе приводятся данные по Вашему счету в банке.
WEEKLY INTEREST (процент по вкладу в неделю).
WEEKLY REPAYMENTS (процент по долгам в неделю).
CURRENT BALANCE (сумма на счете).
OUTSTANDING LOANS (Внешний долг).
Здесь же Вы можете провести с банком финансовые операции:
APPLY FOR LOAN (попросить кредит)

MAKE PART REPAYMENTS (выплатить часть долга)

Раздел GROUND IMPROVEMENT:

Вы можете расширить стадион своего клуба, увеличив количество зрительских мест, что в принципе может повысить доход от проданных билетов, если команда будет играть успешно и посещаемость матчей будет расти. Таблица данных:

PRESENT SPECTATOR CAPACITY - (вмещаемость стадиона).

MAXIMUM SPECTATOR CAPACITY - (максимально возможное количество мест).

COST/1000 EXTRA SPECTATORS - (плата за каждую тысячу дополнительных мест) и подменю:

INCREASE SPECTATOR CAPACITY -(увеличить число зрительских мест).

CLUB REPORT MENU (выход).

Расширять стадион Вам придется не скоро, ибо это влечет за собой слишком больше расходы. Да и посещаемость матчей с участием Вашей команды в первое время невелика.

LEAGUE DETAILS

Здесь Вам предлагается вся справочная информация по чемпионату. При выборе этой опции Вы оказываетесь в подменю: LEAGUE FIXTURES (предстоящие матчи).

LEAGUE RESALTS (результата матчей).

LEAGUE TABLES (турнирные таблицы)

CLUB STATISTICS (статистика по клубам).

MAIN MENU (возврат в вышестоящее меню).

Раздел FA FIXTURES:

Здесь указывается пары команд, которые встречаются в играх на Кубок. Первые встречи начинаются на двадцать второй неделе чемпионата. Жеребьевка клубов проводится за несколько недель до старта игр на Кубок.

CONTINUE.

Эта опция включается, когда все необходимые приготовления закончены и Вы переходите к расстановке игроков из поле на предстоящий матч. После того, как Вы расставите игроков и нажмете FIRE, Вас спросят нужно ли выводить результаты матчей на принтер. После Вашего ответа на экране появится таблица с результатами всех матчей тура. Помимо счета в таблице указываются составы команд, а напротив фамилий игроков, забивавших в этой игре, пишется число забитых мячей.

Если Вы неправильно спрогнозировали число зрителей, и силы полиции не смогли обеспечить порядок на стадионе, то в этом случае после статистики игры с участием Вашей команды сообщается о имевших место беспорядках. Здесь же приводится заявление полиции о том, что они были неверно информированы о количестве зрителей. Далее следует сообщение, что на матче присутствовал инспектор от Федерации футбола.

По окончанию тура вопрос о беспорядках на стадионе будет рассмотрен в Федерации футбола и Вал клуб могут оштрафовать. Иногда, правда, Федерация приходит к выводу, что в этом нет Вашей вины и оставляет клуб в покое.

К десятой неделе чемпионата начинаются торги на бирже игроков. До этого приходят заявки на продажу того или иного игрока. Вот здесь Вам и потребуется помощь Ваших экспертов. Просмотрев такую заявку и выбрав подходящего игрока, Вы передаете все данные о нем своему эксперту. Эксперт, посмотрев этого футболиста, сообщает Вам его отличительные особенности и цену, за которую его можно будет приобрести. Помните, однако, что эксперт тоже не Бог и он никогда не сможет назвать Вам точные цифры. Цена игрока определяется на аукционе. Футболист переходит в тот клуб, который объявит за него максимально цену. Иногда бывает и так, что игрок остается в своей команде под давлением клуба или из-за травмы. Вы, кстати, тоже не можете продать травмированного игрока.

Не хотелось бы раскрывать всех секретов этой замечательной игры, но об одном сказать можно. В Вашей карьере менеджера будут взлеты и падения. Пока Вы не наберете

необходимый опыт Вас ждут неприятные разборки в кабинете директора клуба, проклятия болельщиков и недовольство команды. Возможно, Вы будете с позором уволены с работы и Вас приютит какая-нибудь другая, более бедная и слабая команда. Зато потом, когда раскроется во всей красе Ваш талант, ждите приглашений от прославленных богатых клубов, известных всему миру. Тогда Вы, попивая кофе и раскуривая толстую сигару, будете сами выбирать клуб, который достоин того, чтобы в него пришел работать такой известный и талантливый менеджер, как Вы.

Приложение. СЛОВАРЬ ЭКСПЕРТОВ

Характеристика каждого игрока складывается из пяти отдельных, независимых друг от друга высказываний. Для удобства пользования словарей, фразы разбита на пять групп. Первую фразу эксперта надо искать в первой группе, вторую во второй и т.д. Для характеристики вратарей все фразы, относящиеся к ним, выделены в отдельный раздел, состоящий из пяти групп.

Внимательный читатель обратит внимание на то, что абсолютное большинство заявлений экспертов об игроках имеют положительную тональность. Никто никого не ругает и не порочит. В этом, наверное что-то есть, но все же Вам надо как-то суметь выделить отдельные нюансы этих высказываний. Дерзайте сами, для этого нужен только опыт.

1. ВРАТАРИ

A

GOOD HANDLING (хорошо обращается с мячом).
NOT MACH GETS AWAY FROM HIM (мимо него и муха не пролетит).
COMFORTABLE ON THE BALL (спокойно действует с мячом).
THE SAFEST PAIR OF HANDS IN THE GAME (это надежная пара рук).

B

GETS RID OF THE BALL WELL (хорошо выбивает мяч в поле).
GOOD AT CLEARING HIS LINES (хорошо действует на выходах).
OK ON CLEARANCES (неплохо забирает мяч).
ACCURATE WITH HIS KICKING (точно выбивает мяч).

C

QUICK TO REACT (быстро реагирует)
SUPERB REFLEXES (превосходная реакция).
VERY ALERT (всегда начеку).
MOVES WELL (хорошо перемещается).

D

AWAKE OF SITUATIONS (хорошо чувствует ситуацию).
UNCANNY POSITION SENSE (невероятное чувство позиции).
OCCASIONALY GETS CAUGHT IN TWO MINDS (иногда берет мяч интуитивно).
READS THE GAME SUPERBLY (превосходно чувствует игру).

E

HE COULD BE A SHIP (возможно, это "верняк").
HE WON'T COME CHEAP - HE'S GOOD (этого парня дешево не возьмешь).
WORTH LOOKING AT BOSS (босс, к нему стоит присмотреться).

2. ПОЛЕВЫЕ ИГРОКИ

A

DOESN'T DWELL ON THE BALL (не мешкает с мячом).
LOOKS COMFORTABLE ON THE BALL (на первый взгляд, спокойно действует с мячом).
NICE CONTROL (хороший контроль мяча).
VERY GOOD CONTROL AROUND THE BOX (отлично контролирует мяч на любом участке поля).
DOESN'T FLAP UNDER PRESSURE (отлично действует в сложной ситуации).
IMPRESSIVELY COOL ON THE BALL (хладнокровен в обработке мяча).
SUPERB BALANCE AND CONTROL (превосходно удерживает и контролирует мяч).
NOT AFRAID TO TAKE ON OPPONENTS (смело идет на плотную игру).

B

NOT FAINT HEARTED (смелый игрок).

COMPETITEVE (упорный).

HUSTERS DEFENDERS WELL (хорошо обводит защитников).

HE'S DETERMINED IN THE TACKLE (решителен в схватках).

DISHES OUT A LOT OF STICK (Пресекает проходы соперников).

OPPONENTS HAVE A BATTLE WITH THIS LAD (этот парень доставляет соперникам много хлопот).

GETS WELL AND TRULY STUCK IN (удачно и быстро подключается к атаке).

CHALLENGES EVERYTHING IN THE BOX (умеет все на поде).

C

UNSELFISH (не передерживает мяч).

DISTRIBUTES THE BALL WELL (хорошо распределяет мячи).

SPRAYS HIS PASSES WELL (делает неплохие передачи).

USES THE BALL WELL (ХОРОШО владеет мячом).

GREAT VISION - HIS PASSES ARE CLASS (отлично видит поле - его пасы великолепны).

HIS PASSING IS PURE MAGIC (его передача - просто волшебство).

HIS PASSING IS REAL QUALITY (надежно играет в пас).

SETS UP HIS FELLOW FORWARDS WELL (хорошо питает мячами линию атаки).

D

THIS LAD'S HUNGARY FOR GOALS (остро нацелен на ворота).

THIS LAD WILL SCORE A FEW (этот парень еще себя покажет).

LOVES TO SHOOT GIVEN HALF A CHANCE (бьет при любой возможности).

HIS FINISHING IS LETHAL (его удар смертелен).

HAS THE OCCASIONAL POT AT GOAL (забивает время от времени).

NIGHT SCORE A FEW (из него может быть толк).

ALWAYS LOOKING TO GET ON SCORE SHEET (он рвется забивать).

NOT AFRAID TO SHOOT AT GOAL (бьет из любых положений).

E

NICE TEMPERAMENT WORTH WATCHING(прекрасный темперамент, за ним стоит понаблюдать).

KEEP AN EYE ON THIS LAD BOSS (босс, присмотритесь к этому парню).

COULD BE THE BARGAIN YOUR LOOKING FOR (возможно, это то, что Вы ищете).

HIS GOALS WILL REPAY HIS FEE I'M SURE (я уверен, он стоит своих денег).

A REAL ASSET - NO DOUBT ABOUT IT (отличное приобретение, можно не сомневаться).

A NATURAL, QUICK, SHARP. - (одаренный, быстрый, точный игрок).

URBAN UPSTART

Эксперт Дурченков Алексей г. Челябинск

Программа "URBAN UPSTART" не является такой же логический головоломкой, как, скажем, "SHERLOCK", однако если Вы решите заняться ее прохождением, она доставит Вам немало приятных часов.

Если у Вас нет пока опыта в общении с адвентюрными программами, но Вы ищете, с чего бы начать - начните с программы "URBAN UPSTART" - и Вы не пожалеете о своем выборе. Течение игры очень спокойное. Ваш герой погибнуть не может, самое страшное, что может Вас ожидать - временный арест в КПЗ или же принудительное лечение в муниципальном госпитале. Итак, если Вы решились - начнем.

Городок Скартхопрс - забытое Богом местечко, где даже дети держат за пазухой кнопочные ножи. Здесь все изменения сводятся к тому, что с годами на домах появляется новый слой краски. Люди годами ждут свободного рабочего места и зачастую умирают, так и не дождавшись его. Немногие люди приезжают сюда. Сможете ли Вы спастись из этой глуши, сбежать к цивилизации?

Церковные колокола пробили 3 раза. Попробуйте спастись и, может, Вам повезет.

Внимание!
ИНФОРКОМ ПРЕДУПРЕЖДАЕТ:

Чтение этой статьи может быть опасным для тех, кто предпочитает проходить игры без подсказок.

Приведенные сведения в значительной степени раскрывают технику прохождения игры.

Для тех, кто захочет использовать подсказки только при острой необходимости, мы разбили текст статьи на разделы, привязанное к конкретным эпизодам. Надеемся, что автор простит нам такое вмешательство, это сделано в интересах читателей.

Вот эти разделы (ознакомьтесь с ними до чтения статьи и пользуйтесь именно тем, который Вам нужен в данный момент):

1. Дебют.
2. Книжный магазин.
3. Автобусная остановка.
4. Телефонная будка.
5. Банк.
6. Городской парк.
7. Заброшенный дом.
8. Городская ратуша.
9. Дорога в аэропорт.
10. В аэропорту.

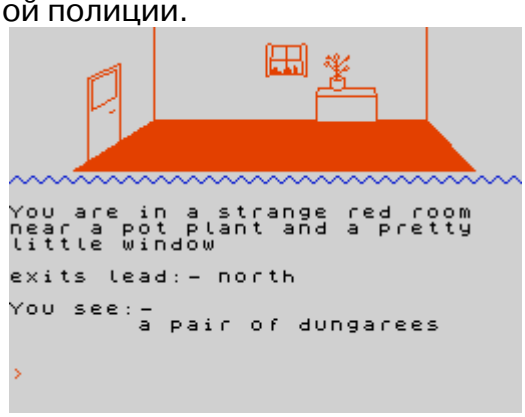
* 11. Полезные советы.

* 12 Нерешенные проблемы.

*/ - чтение раздела безопасно.

1.

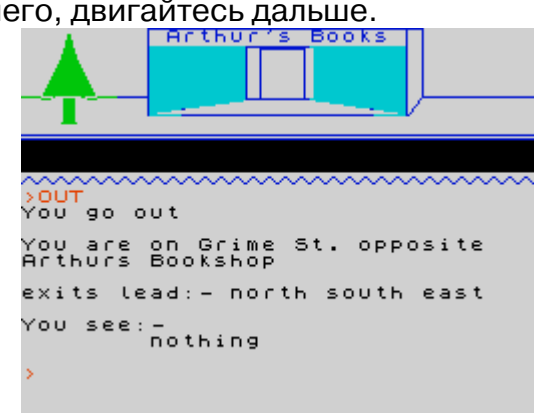
Игру Вы начинаете в маленькой уютной комнате с цветком на окне. Рядом с Вами лежит пара брюк (DUNGAREES), которые не мешало бы взять (TAKE), а потом и одеть (WEAR), учитывая в особенности то, что если Вы будете ходить раздетый по улицам городка, то это вряд ли понравится местной полиции.

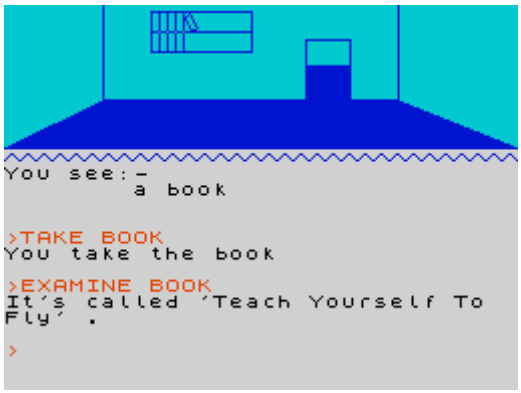


Спустившись на первый этаж, Вы окажетесь перед огромной дверью. Можете попытаться открыть ее и выйти наружу, но из этого вряд ли что получится. Выход один - искать ключ. Найдя ключ (A LARGE KEY), не поленитесь осмотреть и остальные комнаты дома - в них Вы можете обнаружить еще пару полезных предметов - ножницы (A PAIR OF SCISSORS) и банку пива (A CAN OF LAGER), стоящую в холодильнике (FRIDGE). Прихватите все с собой - и в путь. Откройте дверь (UNLOCK DOOR, OPEN DOOR) и смело выходите на улицу (OUT).

2.

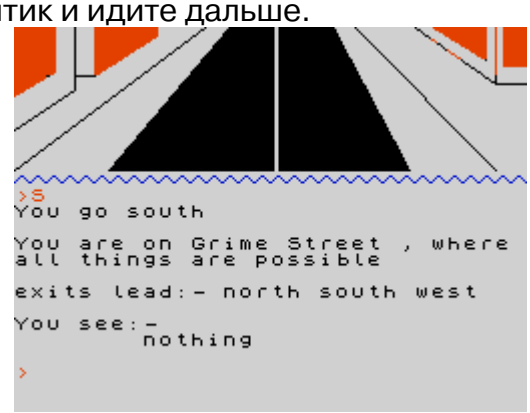
Вы окажетесь на мрачной улице перед уныло стоящим здесь книжным магазином. Зайдя в него (IN), Вы не встретите там покупателей, да и ассортимент имеющейся литературы не балует - на полках стоит всего одна книга (BOOK). Взяв ее и обследовав (EXAMINE), Вы узнаете, что это книга по искусству пилотирования. Решив, что все-таки это лучше, чем ничего, двигайтесь дальше.

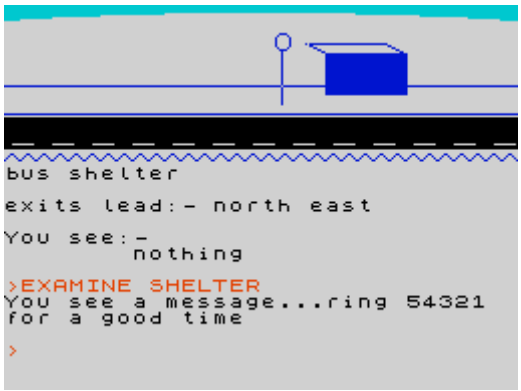




3.

К автобусной остановке, находящейся на мрачной улице, вряд ли когда-либо подходит автобус. Зато там лежит забытый одним из потенциальных пассажиров зонтик (AN UMBRELLA). Надпись на стенке гласит - "Желаете приятно провести время - звоните 54321". Прихватите зонтик и идите дальше.





На углу Вашего дома стоят два мусорных бачка, покопавшись в которых Вы можете найти интересное письмо (A LETTER), изучив которое, Вы узнаете, что это уведомление из банка, присланное вместе с кредитной карточкой (A CHEQUE CARD)) и напоминанием, что для того, чтобы узнать Ваш личный номер банковского счета, Вам необходимо позвонить по номеру 77722. Возьмите с собой карточку и хорошо запомните номер телефона банка. Ненужные более предметы - письмо и большой ключ от входной двери можно бросить здесь (DROP) не опасаясь полиции - в конце концов на то она и свалка, чтоб на ней мусорить.

4.

Дойдя до телефонной будки (PHONE BOX), смело заходите внутрь и звоните в банк (DIAL). Приятный голос поблагодарит Вас за звонок и вам сообщат номер банковского счета. Позвоните 54321, если желаете развлечься (хотя напомню - Ваша цель сбежать из города, тратить время на развлечения просто неразумно). Вам тут же сообщат, что время 3 часа 45 минут и 30 секунд. Удивлены? А каких еще развлечений Вы ждали в этом Богом забытом местечке?

5.

Ну, а теперь в банк. На вид банк кажется закрытым, но не пугайтесь - смело заходите в него (WEST). Вставьте кредитную карточку в машину (INSERT CARD). На дисплее высветилась надпись: "Пожалуйста, наберите номер Вашего банковского счета". После набора номера машина выплюнет Вам назад кредитную карточку и Ваши деньги, хотя их всего лишь - пятерка. Карточку можете оставить здесь - она Вам больше не понадобится и, прихватив пятерку (TAKE FIVER), двигайтесь дальше.

6.

Пройдя по проселочной дороге мимо мертвых деревьев, Вы окажетесь около Скартхопрского парка. Обойдя окрестности парка, Вы можете полюбоваться на его достопримечательности - могильный камень на могиле Джона Смита, разрушенную церквушку.

Мрачновато... Но, еще раз укрепившись в своем решении бежать из этого города, прихватите с собой еду (SOME FOOD) и кусочек сыра (SOME CHEESE) - и прочь из парка выполнять свой план побега. Пройдя по аллее Крикунов (названной так, вероятно потому, что ее облюбовали для своих сборищ фанаты местного футбольного клуба) до стадиона, Вы встретите там фаната местных футболистов. Успехами игроки Скартхопрской команды своих болельщиков не балуют и из-за вчерашнего разгромного поражения от "Ливерпуля" и теперь уже твердого последнего места в турнирной таблице настроение у встретившегося Вам фаната преотвратное. Желая на ком-нибудь сорвать зло, он готов убить Вас и вместо побега из города Вам придется валяться в муниципальном госпитале. Поэтому, не теряя времени, отдайте ему прихваченное из дома пиво (GIVE FAN LAGER), прихватите валяющуюся у его ног мышеловку (A RAT TRAP) и уходите прочь из этого квартала, дав себе слово никогда больше здесь не появляться.

Отойдя немного подальше, остановитесь и осмотрите все, что есть у Вас с собой (INVENTORY). В результате Ваших похождений у Вас есть книга по искусству пилотирования самолетов, мышеловка, пятерка денег, ножницы, одетые на Вас брюки, зонтик, кусочек сыра и немного еды. Не густо... Особенно удручает безденежье. Начинать новую жизнь с пятеркой в кармане не очень-то веселая перспектива. Волей-неволей вспоминаются рассказы старожилов о несметных сокровищах, спрятанных в подвале заброшенного дома, которые охраняются полчищами голодных крыс.

7.

Немногие смельчаки пробовали пробраться туда, но никому это пока не удалось. Оценив свою мышеловку и отметив, заодно, что у Вас есть отличная приманка - сыр, Вы решаете рискнуть. Зайдя по дороге в магазин, реклама которого призывно обещает рыбу и чипсы. Вы ничего из обещанного там не увидите, разве что красную селедку (A RED

HERRING), стоит которую только взять, как на Вас набросятся полчища голодных котов. Кстати, второй перевод слова RED HERRING с английского языка - "ложная наводка". Так что решайте сами - брать или же нет. Да к тому же хозяин магазина - скучающий меломан - выпустит Вас из магазина не раньше, чем Вы послушаете по радио старый номер Френка Синатры (LISTEN).

Идя по улице Дождя без зонтика можно простудиться и попасть в больницу, поэтому как только выйдете на эту улицу, поторопитесь открыть зонтик (OPEN UMBRELLA).

Пробежав по мостику через канал (CROSS BRIDGE), Вы окажетесь у цели. Заржавевшая дверь преграждает Вам дорогу. Открыть ее удастся только после того, как Вы подкрепите свои силы едой (EAT FOOD). Спустившись внутрь по шаткой приставной лестнице, Вы моментально попадаете под атаку дюжины облезлых голодных крыс. Не раздумывая снаряжайте мышеловку! (PUT CHEESE IN TRAP, DROP TRAP). Крысы, завидя, как мышеловка сломала хребет самой здоровой из них, решают убраться в свою нору.

В подвале им сможете найти картонный ящик (A CARDBOARD BOX). Вот оно сокровище! Такой, наверное, будет и Ваша мысль. Что же, берите его и откройте ножницами (OPEN BOX WITH SCISSORS). Внутри Вы увидите только пару ботинок, которые, несмотря на прошедшие века, отлично сохранились и выглядят как новые. Что ж, возьмите их и выбирайтесь из дома.

Остановившись подышать свежим воздухом у канала, Вы найдете маленький ключ (A SMALL KEY). Точно такой же Вы видели у своего однокурсника - летчика. Ну конечно же! Из Скартхопрса надо не убегать, отсюда надо улетать! Но вот проблема - на аэродром пустят только при наличии официального пропуска, а он хранится в ратуше, находящейся на Гражданской улице.

8.

Чиновник, служащий в ратуше, не захочет Вас пустить внутрь. Придется отправиться на поиски дальше. Дойдя до западного конца Больничной улицы, Вы обнаружите там Красную папку (SOME RED TAPE). Взяв ее, Вы сможете беспрепятственно проникнуть в ратушу - видимо, Красная капка не только у нас в стране является атрибутом власти. Зайдя в ратушу, возьмите там пропуск на аэродром (SOME OFFICIAL PAPERS).

9.

Теперь Ваш путь лежит в аэропорт. Свернув с Навозной аллеи на улицу Дождя и попав оттуда на стройку, оденьте ботинки, иначе Ваши ноги увязнут в грязи и санитары тут же уложат Вас в больницу.

Внимательно осмотрите пустые трубы, валяющиеся на стройке - в них Вы найдете летный костюм

(EXAMINE PIPES, FLYING SUIT).

Пройдя по аварийному переулку до входа в аэропорт, смело входите внутрь.

10.

Отдайте дежурному офицеру пропуск (GIVE OFFICER PAPERS). Он потребует с Вас пятерку за вход. Ничего не поделаешь, придется отдать ему последние деньги, только тогда он пропустит Вас к самолету (GIVE OFFICER FIVER) и вот наконец Вы у самолета. Смело забирайтесь внутрь (GO IN PLANE), вставляйте ключ в замок зажигания (INSERT SMALL KEY IS PANEL). Моторы самолета ожили, можно было бы и лететь, но не спешите - управлять самолетом Вы не умеете и Ваш полет закончится весьма плачевно, если Вы не прочитаете книгу по искусству пилотирования (READ BOOK). Прочитали? Ну а теперь в путь. Взлетайте - и Вы победили (TAKE OFF).

11.

И в заключение несколько советов:

Не ходите по улицам раздетым, иначе полиция арестует Вас за оскорбление общественной нравственности.

Не изучайте брошенные на улицах машины - Вас арестует за бесцельное времяпрепровождение.

Не бросайте ненужные предметы на улице, иначе будете арестованы за загрязнение окружающей среды.

Не берите красный шарф - футбольные фанаты сильно изобьют Вас за это и санитары отправят в больницу (по-видимому, это атрибут болельщиков команды соперников).

Для того, чтобы выбраться из больницы, найдите белый халат и оденьте его, иначе лечащий врач задержит Вас прямо у входа и вернет на больничную койку.

Для того, чтобы выбраться из полицейского участка, подойдите к сержанту и подавайте команду WAIT до тех пор, пока не зазвонит телефон и Вы не будете автоматически выброшены из участка.

12.

Теперь о проблемах.

В программе имеются предметы, которые не удалось нигде использовать:

Около полицейского участка лежит старая шляпа (AN OLD HAT).

Западнее улицы Дождя, на стоянке грузовиков можно взять молоко (SOME MILK). Его можно выпить (DRINK MILK), но что это дает, установить не удалось.

Севернее выхода из госпиталя находится здание, называемое "большой оффис" (A LARGE OFFICE BLOCK). Попасть в него так и не удалось.

Программа реагирует на команду SCORE. Наивысший результат, которого удавалось достичь - 18 из 22 возможных. Где взять еще 4 недостающих до полной победы балла - не ясно.

Ну вот, наверное, и все.

Компьютерная новелла

Битва Усаги.

(По игре "SAMURAI WARRIOR")

Усаги было всего 15, когда его отец Акира Тойимбо покончил с собой, сделав "сэппуку"[1]. День назад, когда ничто не предвещало печального конца, молодой Усаги стоял вместе с ним во внутреннем дворике отцовского дома, почтительно склонившись перед гостем, только что приехавшим из Киото [2]. На куртке гостя была эмблема дома Минамото [3], сам же Усаги имел честь носить герб дома Ходзе. Гость низко поклонился отцу, Усаги же удостоился легкого кивка головой. В один голос с отцом он произнес: "Добро пожаловать!", - и отец жестом пригласил гостя во внутренние покои.

Говорили они недолго и через час только пыль на Западной дороге запоминала о том, что сегодня у них кто-нибудь был. С востока надвигалась черная туча...

Была уже почти ночь, когда старый Акира Тойимбо позвал сына к себе. Когда Усаги вошел, отец сидел на циновке в белом кимоно. Его седая голова была низко опущена. Первые слова, которые он произнес, были: "На все воля неба, сын мой... " Выдержав паузу, он сказал: "Вчера самураи Тайра напали на храм, где твоя мать укрылась от бренности этого мира. Она умерла вместе с другими монахинями с улыбкой под мечом этих собак...". Усаги вскочил. Но Акира повелительно вскинул руку, приказывая ему сесть. Его глаза на миг блеснули. Усаги медленно опустился на циновки. Отец тихим голосом продолжал: "Я стар, сын мой... - К тому же я не могу нарушить клятву, данную мною Будде, больше не убивать ни одного человека. Но я был и остаюсь самураем и поэтому сегодня покину этот мир. Ты - мой единственный сын, единственная надежда сохранить честь семьи. Отомсти за свою мать. Мое завещание ты найдешь в этой шкатулке", - отец указал в левые угол комнаты. "Мужайся, сын мой, и пусть не дрогнет твоя рука ни сейчас, ни после", - и Акира обнажил свой короткий меч для сэппуку. Усаги, встав по левую сторону от отца, медленно вынул из ножен и занес свою катану[4], готовясь облегчить страдания отца. Перед тем, как вонзить короткий меч себе в живот, акира в последний раз взглянул на сына. Глаза Усаги затуманились, его собственный меч сверкнул в этом тумане, как молния, обрывая жизнь самого дорогого для него человека...

Всю ночь Усаги не сомкнул глаз, обливаясь слезами. "Прощальная песня"[5] его отца гласила:

В ясном небе
Растворяется
Ясной молнии блеск...

Ниже Рукой Акиры было написано:

"Сын мой!

Сейчас, когда мы с твоей матерью снова вместе, мое сердце печется лишь о тебе. Выполни свой долг. Свято чти кодекс чести "Бусидо". Не обижай и защищай слабых. Придет время и тирания Тайра падет. Наш гость сообщил мне, что светлейший князь Еритомо Минамото вскоре выступит против Тайра. Я хочу, чтобы ты, после того, как отомстишь за мать, встал под его знамена. И последнее: убийцу твоей матери зовут Кавасаки Тайра.

Да пребудет Будда с тобой!"

Усаги вышел из дома рано утром. В его кошельке было всего три ре[6]. Первым, кого он встретил, был седой крестьянин, идущий с поля. Усаги был далек от тех столичных щеголей, которые при виде крестьян в лучшем случае брезгливо морщились, а в худшем проверяли на них остроту фамильного меча. Отец учил Усаги уважать труд. "У каждого своя карма[7]", -

говорил он. Усаги слегка склонил голову. Крестьянин низко поклонился. Усаги протянул ему один ре. "Спасибо, добрый самурай", - сказал крестьянин и снова низко поклонился. Усаги пошел дальше по пыльной дороге. Где-то там, за лесом, за двумя городами, в горах была усадьба Кавасаки Тайра...

Вдруг что-то заставило Усаги напрячься. Его руки мгновенно извлекли из ножен катану. Меч и человек слились в одно целое. В следущую секунду с дерева с протяжным криком, взмахнув мечом, спрыгнул человек в черной маске. Но у самой земли его встретил молниеносный росчерк меча Усаги.

Усаги медленно стер с меча кровь и осторожно вложил его в ножны. Он и раньше кое-что слышал о людях, закрывающих свои лица. Их звали "ниндзя". Суеверные крестьяне верили, что они знаются с нечистой силой, могут внезапно появиться, исчезать и сражаться как звери, неистово махая мечом и метая звездочки - "сюрикены". Крестьянин, которого Усаги вскоре встретил, в ответ на вопрос о ниндзя посмотрел на него с ужасом: "...Да, да, господин, мы слышали с них и раньше. Люди поговаривают, что если свернуть с дороги налево по тропинке перед ручьем, то можно забрести в пещеру, где у них логово... люди оттуда не возвращаются. Не ходите туда, благородный господин. Если уж вам нужно попасть в город, идите прямо, через лес..."

Встреченный Усаги самурай в шлеме, похожем на волчью голову, после того, как Усаги приветствовал его поклоном, сообщил: "Ах, ниндзя? Вон там в кустах лежат трупы двоих. Эти бестии прикинулись крестьянами и напали на меня сзади... "

В городе Усаги решил пополнить свои силы и свернул на постоялый двор. У входа его встретил плутоватый крестьянин и предложил азартную игру. Усаги из любопытства протянул ему монету. Сначала он выиграл, а потом проиграл. Молодой самурай решил не искушать судьбу, так как нужно было еще заплатить хозяйке за еду.

У выхода его поджидал какой-то самурай, который явно хватил лишнего: "Поединок до первой крови, самурай!" - заявил он. Усаги пытался отстранить незнакомца, но тот назвал его трусом и Усаги пришлось все-таки извлечь меч из ножен. Но он не хотел убивать соперника и лишь слегка ранил его. Чужой самурай сразу же протрезвел, и, пробормотав извинения, спешно удалился...

Наконец, после долгого пути, он был у ворот усадьбы Кавасаки Тайра. Стражники узнали герб на куртке Усаги и кинулись на него. Усаги вспомнил об отце и матери и это удесятерило его ярость...

Что было дальше, он помнит плохо... кажется, сражаясь против самого Кавасаки Тайра, он видел призрак своего отца из-за спин нападающих...

* * *

Примечание автора:

(1) - Сэппуку (в народе называемое харакири) - обряд ритуального самоубийства самураев.

(2) Тайра, Минамото, Ходзе - самурайские дома в феодальной Японии. В XII веке вспыхнула затяжная кровопролитная война за власть между домами Тайра и Минамото. Дом Ходзе выступил на стороне Минаното.

(3) Киото - резиденция японских императоров.

(4) Катана - длинный меч, используемый самураями в бою, в отличие от короткого, используемого, в основном, для сэппуку.

(5) Прощальная песня - стихотворение (хокку) на тему природы, слагаемое самураями перед смертью. В прощальной песне отца Усаги обыгрывается его имя - Акира ("ясный").

(6) Еримото Минамото - глава дома Минамото, ставший впоследствии сегуном - военным правителем Японии.

(7) Ре - золотая монета.

(8) Карма - закон причинно-следственных связей, по которому человек обязательно получает воздаяние за добрые или злые дела. Ключевое понятие в буддийской философии.

* * *

Примечание "ИНФОРКОМа":

Так получилось, что эта новелла пришла от уважаемого автора в то время, когда мы готовили первый выпуск нового электронного журнала "PC-РЕВЮ" и работали над статьей "Средневековое оружие в программе MIGHT & MAGIC III", исследуя сопутствующую литературу. Обзор этой литературы позволяет сделать пару небольших добавлений.

1. Автор не вполне справедливо называет катану длинным мечом. Конечно же он длиннее, чем упомянутый ритуальный меч для сэппуку, который немного больше кинжала, но все же это не длинный меч, его принято считать коротким.

Дело в том, что в стандартный боевой комплект богатого самурая входило два меча - вакидзаши и катана. Вакидзаши - длинный меч, которым можно действовать как одной, так и двумя руками, а катана - более короткий, вспомогательный меч для левой руки. Конечно, при отсутствии вакидзаши катана становится основным мечом и может считаться относительно длинным. Кстати, именно не очень большие размеры катаны способствовали тому, что это оружие позаимствовали у самураев бойцы ниндзя, т.к. его можно было переносить тайно, укрывая в складках одежды, а при ношении за спиной оно не стесняло быстрых и ловких движений ночных диверсантов.

2. События, описанные в новелле, имеют реальные исторические корни. Они относятся к периоду феодальных войн 12 века, когда власть императоров была заменена военным сегунатом. К сожалению, князя Еритомо Минамото (1147 -1199) не принято считать светлой личностью. Дело в том, что власть дома Тайра сверг Есицунэ Минамото (1159 - 1189), а Еритомо пришел к власти, коварно загубив своего младшего брата.

Впрочем, согласно легенде, он получил по заслугам. Еритомо погиб, упав с лошади, когда ему привиделся призрак его убитого брата. Предполагают, что роль "призрака" сыграл один из хорошо подготовленных ниндзя, отомстивший правителю за подлость.

* * *

Для справки: программа MIGHT & MAGIC III, известная также под названием ISLES OF TERRA выпущена компанией New World Computing и относится к жанру RPG - Role Playing Games - ролевые игры. К сожалению, для "Спектрума" этот жанр малодоступен. Пожалуй, именно в этом жанре, а также в имитаторах IBM имеет преимущество над "Спектрумом". Во всех прочих играх наши читатели могут чувствовать себя на равных с владельцами IBM-совместимых машин, а во многих случаях даже имеют ощутимое преимущество за счет большей интеллектуальности "синклеровских" игр. Впрочем, те, кто заинтересуются этим вопросом подробнее, смогут в ближайшее время стать читателями "PC-РЕВЮ", если хотя бы на время найдут подход к IBM-совместимой машине, а сейчас такая возможность есть уже и в самых отдаленных пунктах.

Мы рассчитываем, что наша стратегия распространения "PC-РЕВЮ" сможет доводить выпуски этого журнала до каждой IBM-совместимой машины в СНГ в течение 5-6 месяцев после выхода очередного номера.

ИНФОРКОМ-ПРЕСС ПРЕДСТАВЛЯЕТ

Дорогие друзья!

Мы продолжаем рекламную компанию по продвижению нового электронного журнала "PC-РЕВЮ". К сожалению, у нас не получается, как было объявлено, осветить в этой выпуске итоги ТЕНДЕРА, объявленного в NN 7-8. Дело в том, что этот выпуск ZX-РЕВЮ передается в типографию 10 ноября 1993 Г., а итоги ТЕНДЕРа будут подводиться только 15 декабря. Такая необычная для нас поспешность с выпуском данного номера связана с тем, что мы кардинально улучшаем качество и формат издания ZX-РЕВЮ-94 (каждый номер - 96 страниц формата A5, цветная обложка, выполненная одним из известнейших российских художников и пр.) и потому должны иметь солидный резерв времени на подготовку и печать первого номера будущего года. Кстати, многие наши читатели сейчас не выписывают РЕВЮ-94, полагая, что в конце года будет выпущен годовой сборник. Это ошибка. При новом оформлении и объеме ZX-РЕВЮ-94 мы не сможем издать годовой сборник при всем своем желании.

Тем не менее, все участники ТЕНДЕРа получат от нас персональное уведомление об его итогах, публично же его результаты мы обнародуем в первой выпуске будущего года.

Пока же мы заканчиваем подготовку первого выпуска и уже можем оценить его себестоимость. Исходя из нее сейчас можно предположить, что по всей видимости цена отсечки будет не ниже 25 тысяч рублей. Таким образом, стоимость лицензии на право распространения одного номера PC-РЕВЮ составит не менее 25 тысяч рублей как для юридических, так и для физических лиц при приобретении лицензии сразу на несколько номеров, например на все 12 выпусков будущего года, конечно будут действовать скидки.

У читателей ZX-РЕВЮ есть несколько возможностей приобретения PC-РЕВЮ (для личного пользования или для участия в его распространении) от очень дорогих до практически бесплатных.

1. Вы хотите читать PC-РЕВЮ, но не хотите участвовать в его распространении.

1.1. Вы сможете обратиться в своем городе к нашему дистрибутору и скопировать у него очередной номер за назначенную им плату. Мы не можем определять ту цену, которую наши дистрибуторы назначат за копирование журнала, но можем предположить, что рынок выведет их на уровень от 2-х до 7 процентов (в зависимости от количества клиентов) от той суммы, которую они сами заплатили за лицензию. Плюс, конечно, стоимость дискеты (дискет).

Это доступный и оперативный метод получения свежих выпусков журнала.

1.2. Если в Вашем городе нет нашего дистрибутора, Вы можете обратиться в любой компьютерный салон или в любую фирму, занимающуюся торговлей компьютерами или программными продуктами в своем городе и предложить им связаться с нами. Может быть, им придется предварительно изучить потенциальный спрос. Если заявок будет хотя бы несколько десятков, они наверное согласятся включиться в нашу дистрибуторскую сеть, им это будет интересно, необременительно, а главное - перспективно. Это дело, которое будет развиваться и кто начнет его раньше, тот пойдет дальше.

1.3. Вы можете вообще ничего не делать, а терпеливо ждать, когда номер PC-РЕВЮ сам придет к Вам, в руки через друзей или по компьютерным сетям поскольку выпуски будут распространяться без защиты от копирования, этот момент наступит через несколько месяцев (ориентировочно 5-6). Это практически бесплатный, но и самый длинный путь.

1.4. Ограниченное количество дискет каждого номера мы будем продавать в

подарочном (коллекционном) исполнении в красочных альбомах через дорогие компьютерные салоны, книжные магазины и музыкальнее магазины г. Москвы по престижно высоким денам. Поступать туда эти номера будут не ранее, чем через полтора месяца после рассылки мастеркопий дистрибуторам.

Очевидно, для Вас это самый неудобный, дорогой и неоперативный способ получения журнала, но он необходим для поддержки маркетинговых усилий наших дистрибуторов в их регионах.

2. Вы готовы подключиться к распространению PC-РЕВЮ.

2.1. В этом случае для обладания свежим выпуском Вам придется предварительно перечислить нам установленную цену за мастердиск и за лицензию на копирование.

2.2. Далее Вы можете делать все, что хотите, Вы можете копировать журнал для всех желающих на дискету заказчика за назначенную Вами плату. Ваши координаты войдут в список официально зарегистрированных дистрибуторов и к Вам будут обращаться читатели.

Вы можете распространять этот журнал и на своих дискетах, используя для этого торговые возможности других фирм.

Вы можете заниматься этим в порядке предпринимательской деятельности без образования юридического лица, что предусмотрено действующий законодательством.

Вы можете специально для этого зарегистрировать фирму или использовать уже имеющуюся и заключить с нами договор.

2.3. Не исключено, что Вы захотите стать Генеральный дистрибутором в своем городе (области, республике) и заключить с нами эксклюзивный договор, согласно которому Вы будете единственным дистрибутором в своем регионе.

Это будет возможно, но не всегда. Мы пойдем на такой договор только в том случае, если Вы в своем регионе будете первым. Разорвать уже сложившиеся связи с партнерами мы конечно же не сможем. И, разумеется, такая генеральная лицензия будет стоить дороже. Это будет зависеть от размера региона. Генеральная лицензия на Эстонию будет стоить дороже, чем на один город Нарву.

Для Москвы и Московской области Генеральная лицензия на эксклюзивное распространение не продается, но обычные лицензии продаются обычным порядком.

Мы с Вами находимся у истоков огромного дела. В настоящее время в России и СНГ установлено свыше трех миллионов IBM-совместиных компьютеров, но специалисты считают, что пока это капля в море. И до каждого компьютера Вы должны добраться. Не исключено, что Вы находитесь у истоков дела, в которое в ближайшие годы будут вовлечены сотни миллионов рублей и девяносто процентов из них - Ваши, так что решайтесь! Да, первое время может быть трудно, может быть придется побегать, покрутиться, кого-то убедить, кого-то организовать, но успех приходит только к подготовленным людям. Если у Вас есть вера в свои силы, Вы пойдете дальше вместе с нами.

Со всеми вопросами, предложениями, пожеланиями, обращайтесь по адресу:

121019, Москва, Г-19, а/Я 15, А/О "ИНФОРКОМ-ПРЕСС"

Содержание