ZX - РЕВЮ
N6 (6) 1991

НАУЧНО-ТЕХНИЧЕСКИЙ КООПЕРАТИВ "ИНФОРКОМ". 107241, МОСКВА, Б-241, А/Я 37

Дорогие читатели.

Сейчас Вы держите в руках шестой выпуск ZX-РЕВЮ, а это означает, что половина пути нами пройдена, можно подводить первые итоги.

Судя по тем письмам, которые мы получаем, ZX-РЕВЮ находит добрый отклик среди любителей "Спектрума", и мы с большим интересом продолжаем это начинание.

Но не все так хорошо, как хотелось бы. Нас не устраивает содержание рекламного приложения, надо сказать правду - оно не пользуется популярностью, а в то же время читатели просят давать больше познавательных материалов.

Начиная с этого номера мы ликвидируем этот раздел, увеличив объем материалов, представляющих всеобщий интерес. Конечно, те немногие объявления, которые уже оплачены нашими читателями, мы еще дадим, но больше плату за объявления не принимаем.

Это не значит, что мы лишаем Вас связи друг с другом. Во-первых, кому есть что сказать, может это сделать и через раздел "ФОРУМ". Во-вторых, в порядке исключения мы все-таки будем бесплатно иногда давать некоторые объявления наших читателей, но при условии, что они будут иметь общественный интерес, при наличии свободного места и в порядке очереди. В первую очередь просьбы об экстренной помощи и объявления от инвалидов, поскольку они наиболее нуждаются в общении с единомышленниками.

Разумеется, в скромных пределах мы оставляем за собой возможность представлять свои разработки и разработки НТК "ПЛЮС" (это условие нашего сотрудничества).

Кстати, в этом выпуске нет технического раздела, но это явление временное, связанное с отпускным периодом, далее он будет восстановлен.

Нет в этом выпуске и раздела "Секреты ПЗУ". Большой объем очередной главы, посвященной процедурам, выполняющим печать на экране и принтере вынуждает нас опять сделать очередной выпуск двойным.

Все чаще поступают вопросы о подписке на 1992 год. Мы пока не готовы сообщить ничего конкретного. Ясно только одно - будет подписка не будет, а "ZX-РЕВЮ" все равно будет.

Те, кто поверил нам в этом году и подписался, будут получать его и в будущем, пусть не в форме подписки, а в какой-то другой, но мы знаем, что нам работа нужна. Вы ее ждете и те, кто нами зарегистрирован, всегда будут своевременно проинформированы.

По этой причине мы и не стали прекращать прием подписки в этом году. Напоминаем, что вновь подписавшиеся получат все вышедшие выпуски. Стоимость подписки за весь год - 90 руб.

Пока же мы формируем портфель материалов на 1992-ой год и, если он окажется очень толстый, рассмотрим вопрос о значительном увеличении нашего объема.

И, в заключение, просьба ко всем нашим читателям. Мы хотели бы выписать наиболее популярные местные ежедневные или еженедельные газеты нашей страны, чтобы лучше знать чем живут регионы. Не можете ли Вы дать нам совет, что наиболее охотно читают в Вашей местности?

Может быть Вы даже сможете выслать экземпляр такого издания для ознакомления, из него мы сразу узнаем и координаты редакции.

Надеемся на вашу подсказку.

С уважением,"ИНФОРКОМ".

Раздел для начинающих

Мы продолжаем публикацию заметок для начинающих нашего читателя из г. Дубна, т. Скитева.

Сегодня раздел посвящен графике пользователя, так называемой UDG-графике.

При работе с графикой пользователя обычно сталкиваются с определенными трудностями. Для графики отводится 21 символ от "A" до "U" при условии печати этих знаков в графическом режиме (курсор G).

Каждый знак состоит из 6 строчек по 8 символов в каждой. Рисуем матрицу 8x8, и в ней рисуем то изображение, которое хотели бы видеть. Заменим закрашенные квадраты единицами, а незакрашенные - нулем. Затем выберем символ, который хотим заменить своим, например "A", и наберем следующую программу.

```
10 POKE USR "A",BIN 00111100
20 POKE USR "A"+1,BIN 01011010
30 POKE USR "A"+2,BIN 01111110
40 POKE USR "A"+3,BIN 11100111
50 POKE USR "A"+4,BIN 10100101
60 POKE USS "A"+5,BIN 10100101
70 POKE USR "A"+6,BIN 10100101
80 POKE USR "A"+7,BIN 00100100
```

Если после выполнения программы перейдем в графический режим (а это выполняется одновременным нажатием клавиш CAPS SHIFT и 9) и напечатаем литеру "A", то увидим изображение паука. Это изображение будет находиться в памяти компьютера до тех пор, пока не будет выключено питание или нажата кнопка "RESET".

Программа заняла восемь строчек, а заменила только одну литеру. Можно сделать эту программу по-другому, используя оператор "READ" и список "DATA".

Правда, предварительно необходимо провести следующую работу - перевести изображение из двоичного кода (а мы создавали его в двоичном коде 0,1) в десятеричный код. Для этого можно, например, воспользоваться оператором "BIN". Дав прямую команду PRINT BIN 00111100, получим результат - 60.

Затем следующая строка PRINT BIN 01011010 дает 50 и т.д. Таким образом определяем значения всех строк и заносим их в список DATA. Теперь программа выглядит так:

```
10 FOR X=0 TO 7
20 READ Y
30 POKE USR "A"+X,Y
40 NEXT X
50 DATA 60,90,126,231,165,165,165,36
```

Программа делает то же самое, что и предыдущая.

Продолжение паука (падающий паук).

```
110 FOR X=0 TO 7
120 READ Y
130 POKE USR "T"+X,Y
140 NEXT X
150 DATA 16,16,16,16,16,16,16,16
160 FOR L=0 TO 20
170 PRINT AT L,3: INK 0; "T"
```

Здесь черточка под буквой "T" означает, что это вовсе не буква, "T", а тот знак, который закреплен за клавишей "T" в графическом режиме. Пожалуйста не ошибитесь при наборе. Аналогично будем поступать и далее.

```
180 PRINT AT L+1,3; INK 2;"A"
190 NEXT L
```

Продолжение паука (убегающий паук).

```
210 PAUSE 10
220 FOR C=3 TO 30
230 PRINT AT 21,C; " "
240 PRINT AT 21,C+1; INK 2; "A"
```

```
250 NEXT C
```

В разработке "Большие возможности Вашего Спектрума" есть вариант перевода компьютера на русский шрифт, создав русские символы в графическом режиме.

Работая с графикой пользователя можно создавать и более сложные рисунки, состоящие из нескольких символов.

Попробуйте ввести следующую программу.

```
10 FOR I=0 TO 47
20 READ N
30 POKE USR "A"+I,N
40 NEXT I
50 DATA 0,2,3,7,15,19,83,126
60 DATA 0,64,64,224,240,200,202,126
70 DATA 127,115,120,31,15,4,4,28
80 DATA 254,206,30,243,240,32,56,0
90 DATA 127,187,120,27,15,4,28,0
100 DATA 254,254,30,216,240,32,32,56
```

Выполним эту программу, результатом будет изменение в графическом режиме символов A,B,C,D,E,F. Если будет сообщение - OUT OF DATA, внимательно проверьте Ваш листинг. В каждой строке DATA должно быть 8 чисел. Теперь можно командой NEW очистить память, листинг исчезнет, но результат работы программы останется (до команды RESET).

Введем следующую программу. Символы, стоящие в кавычках, будем вводить в графическом режиме.

Прыгающие фигурки.

```
110 BORDER 5; PAPER 5: CLS
120 PLOT 60,71: DRAW 135,0
130 PAUSE 50
140 PRINT AT 10, 14; "AB"; AT 10,18;"  "
150 PRINT AT 11,14;"CD";AT 11,18; "AB"
160 PRINT AT 12,14;"  ";AT 12,18;"EF"
165 BEEP .05,0
170 PAUSE 50
180 PRINT AT 10,14;"  ";AT 12,18;"  "
190 PRINT AT 12,14;"EF";AT 12,18;"  "
205 BEEP .05,20
210 GO TO 130
```

По такому принципу можно создавать достаточно сложные рисунки состоящие из большего количества символов до 21.

Эта программа рисует шахматное поле. В строках 50, 60 после кавычек делаем два пробела, затем два черных квадрата в графическом режиме и так далее, всего в строке 32 символа. А в строках 70, 80 сначала два квадрата, затем два пробела и т.д.

```
10 CLS
20 BORDER 0
30 PAPER 7
40 FOR N=1 TO 6
50 PRINT "  ■■  ■■  ■■  ■■  ■■  ■■  ■■  ■■"
60 PRINT "  ■■  ■■  ■■  ■■  ■■  ■■  ■■  ■■"
70 PRINT "■■  ■■  ■■  ■■  ■■  ■■  ■■  ■■  "
80 PRINT "■■  ■■  ■■  ■■  ■■  ■■  ■■  ■■  "
90 NEXT N
```

Используя эту программу, а также программы, которые мы приводили ранее, сетка на экране, вертикальные цветные полосы, полное окрашивание экрана в основные цвета (R,G,B) и так далее, можно сделать тестовую программу для монитора. Для этого необходимо сделать программу меню, один из вариантов можно найти во втором выпуске "ZX-РЕВЮ" в разделе маленькие хитрости, а в листингах программ соответственно изменить номера.

MEGA BASIC

Сегодня мы заканчиваем печать инструкции по работе с языком программирования MEGA-BASIC и приводим окончание списка дополнительных команд, а также перечень системных сообщений программы.

Начиная со следующего выпуска мы последовательно начнем рассматривать версии другого, не менее мощного языка - BETA BASIC.

* * *

CURRENT_n - по этой команде текущим становится окно n.

DEFG_A$,n... - эта команда позволяет определить символ UDG. Форма UDG задается 8 числами, записанными после a$, которое в свою очередь определяет адрес ОЗУ, где этот символ будет храниться.

DELETE_a,b - эта команда используется для удаления блока программных строк. Здесь a - номер первой строки, а b - номер последней строки удаляемого блока. Если a>b, то на экране появится сообщение "B Integer out of range".

DOKE_a,b - эта команда эквивалентна следующим:

POKE a,b-256*INT(b/256)

POKE a+1,INT(b/256)

DOWN_y,x,a$ - по этой команде строка символов a$ выводится в нижнюю часть экрана.

EDIT_n - по этой команде n-я строка переписывается в область редактора строк и активизируется редактор,

ENDPROC_n - эта команда определяет конец процедуры, имя процедуры может быть записано после команды.

EXAMINE - по этой команде на экран выводится содержимое заголовка вводимой с кассеты программы, т.е. имя, длина в байтах и начальный адрес в ОЗУ вводимого файла.

FADE_n - по этой команде каждый раз уменьшается на 1 содержимое каждого байта атрибутов экрана. Этот процесс повторяется до тех пор, пока значение каждого байта не станет равным n.

FONT_n - эта команда позволяет выбрать необходимый символьный набор из имеющихся в памяти. Команда не действует в режиме MODE 1, так как в этом режиме используется собственный символьный набор.

FX_n,m - команды выбора окна на экране.

GET_0,a,y,x,d,w - эта команда используется для сохранения в памяти изображения экрана.

INVERT - по этой команде инвертируется весь экран; цвет INK меняется на цвет PAPER, а цвет PAPER на цвет INK.

KEY_n,A$ - эта команда используется для задания функциональных клавиш. Здесь n - номер клавиши, а A$ - строка символов, закрепленных за этой клавишей.

MODE_(n),a - эта команда используется для выбора размеров выводимых на экран символов. Если после команды записаны два числа, то первое определяет номер окна.

MON - по этой команде вызывается фронт-панель. При этом пользователь может не только узнать содержимое ячеек памяти и регистров процессора, но и изменить их. Возврат в Мега-Бейсик осуществляется нажатием SPACE.

MTASK_n - по этой команде программа может выполняться одновременно с двух точек.

PAN_n,m - эта команда позволяет перемещать попиксельно вправо и влево содержимое текущего окна экрана без изменения значений атрибутов.

PCLEAR - по этой команде очищается стек PROCEDURE/REPEAT-UNTIL.

PLAY_n,l,s,d,f - эта команда используется для исполнения звуковых эффектов.

POP_n - по этой команде в стек PROCEDURE/REPEAT-UNTIL записывается число n.

PRINTER_n - эта команда используется для вывода на принтер или другие периферийные устройства.

PUSH_n,m - эта команда используется для записи значения в стек PROCEDURE/REPEAT-UNTIL.

PUT_1,a,y,x,d,w - эта команда используется для восстановления изображения, хранящегося в памяти, на экран. Команда противоположна команде GET.

REPEAT - оператор, определяющий начало цикла REPEAT-UNTIL. Если стек PROCEDURE/REPEAT-UNTIL заполнен, то на экране появляется сообщение "PROC stack overflow"

RESTART_n - эта команда эквивалентна команде ON ERROR GO TO, которая существует в других версиях BASIC.

RESTART_OFF - использование этой команды приводит к отмене режима программной обработки ошибок.

SCROLL_n,m - по этой команде осуществляется перемещение вверх или вниз значений текущего окна.

SOFF - команда выключения ISG.

SON - команда включения ISG.

SPEED_n - команда управления скоростью выполнения программы при включенном режиме трейсирования TRON.

SPRINT_x,y,a,b,c$ - это команда вывода на принтер строки символов, размер которых задается пользователем.

SFROFF_n,m - выключение спрайтов.

SPRON_n,m - включение спрайтов.

SPUT_x,y,a,b,c,w,d - эта команда подобна команде PUT, но в данном случае размеры изображения могут быть увеличены в двух направлениях заданием коэффициентов c,b.

SREP_n - это команда повторения звуков, информация о которых записана в звуковом буфере.

STIPPLE_n - эта команда позволяет создавать теневые эффекты для символов, записанных в режиме MODE 4.

SWAP_n,m - это команда для манипуляций с файлом атрибутов. При этом значение байта файла атрибутов меняется с n на m.

TRON - команда включения режима трассировки. При трассировке номер выполняемой программной строки выводится в нижнем левом углу экрана.

TROFF - команда выключения режима трассировки.

UNTIL_a - эта команда определяет конец цикла REPEAT-UNTIL.

VDU a,(a...) - эта команда эквивалентна PRINT CHR$.

WINDOW_y,x,d,w - по этой команде задается размер и расположение на экране текущего окна.

Новые сообщения об ошибках

FX NOT IMPLEMENTED - ошибка, связанная с командой FX...., Вы попытались использовать несуществующее назначение.

ILLEGAL WINDOW - ошибка команд FX или CURRENT. Вы попытались использовать более чем 10 окон.

LINE NOT FOUND - ошибка команды EDIT. Вы попытались редактировать строку, которой нет в программе.

MISSING PARAMETERS - некорректная запись чисел или параметров в командах.

WINDOW TOO LARGE - эта запись может появиться при использовании команды WINDOW, если размеры окна слишком большие.

WINDOW TOO SMALL - эта запись появляется, если один из размеров окна равен нулю.

X TOO LARGE или Y TOO LARGE - выход за пределы экрана.

В нашей почте лежат письма от читателей, которые либо начали работу с МЕГАБЕЙСИКом, либо собираются ее начать, но хотели бы видеть образец программы, написанной на этом языке.

Самое простое, что можно им порекомендовать - это просмотреть листинг файла DEMO, который прилагается к программе. Но поскольку файл DEMO предназначен прежде

всего для того, чтобы наилучшим образом представить программу в работе, а не в просмотре, он выглядит не очень читабельным.

Идя навстречу высказанным пожеланиям, мы приводим распечатку программы, предназначенной для исследования оператора PLAY. Программа взята из журнала YOUR SPECTRUM (N15, июнь, 1985 г.).

Прелесть программы в том, что во-первых Вы наглядно увидите насколько же текст ее более понятен и компактен по сравнению с обычным Бейсиком, а во-вторых, сможете неплохо разобраться с командой PLAY, прощупать ее возможности.

После этой команды следуют несколько параметров и влияние их на работу команды на первый взгляд совсем неочевидно.

Наберите эту программу и запустите. Вертикальным перемещением курсора Вы сможете выбрать тот параметр, который хотите поменять, а горизонтальным перемещением - изменить этот параметр. После этого нажмите "пробел" и слушайте, что у Вас получилось.

```
1000 REM PLAY EXPLORER
1010 REM BY L A PRIVETT
1020 REM MARCH 1985
1030 :
1040 PCLEAR
1050 SETSCREEN
1060 SETDISPLAY
1070 SETVALUE
1080 BOXIN
1090 SCANKEYS
1100 STOP
1110 :
1120 :
1130 :
```

Вот собственно и вся программа - она занимает 7 строк, начиная со строки 1040 - по строку 1100.

Все остальное - вспомогательные процедуры.

```
2000 @SETSCREEN
2010 WINDOW_0,0,22,64
2020 PAPER 0
2030 INK 6
2040 BRIGHT 1
2050 OVER 0
2060 INVERSE 0
2070 MODE_2
2080 CLS
2090 DRAW 255,0
2100 DRAW 0,175
2110 DRAW 255,0
2120 DRAW 0,-175
2130 FONT_1
2140 LET a$="PLAY_ EXPLORER"
2150 SPRINT_20,10,1,2,A$
2160 INK 3
2170 ENDPROC_SETSCREEN
2180 :
2190 :
2200 :
3000 @SETDISPLAY
3010 LET A$="A B C D E"
3020 LET B$="= = = = ="
3030 INK 5
3040 DOWN_5,8,A$
3050 DOWN_5,12,B$
3060 FONT_2
```

```
3070 SPRINT_10,150,2,2,"PLAY"
3080 PRINT AT 2,40;"USE CURSOR"
3090 PRINT AT 3,40;" KEYS  TO "
3100 PRINT AT 4,40;"SELECT AND"
3110 PRINT AT 5,401"  CHANGE  "
3120 INK 6
3130 PLOT 0,36
3140 DRAW 255,0
3150 INK 6
3160 PRINT AT 8,40;"SPACE FOR"
3370 PRINT AT 9,40;"   PLAY  "
3180 BEEP .1,0
3200 ENDPROC_SETDISPLAY
3210 :
3220 :
3230 :
4000 @SETVALUE
4010 DIM M(5)
4020 DIM V(5)
4030 DIM P(5)
4040 FOR F=1 TO 5
4050 LET P(F)=3+F*2
4060 LET V(F)=0
4070 NEXT F
4080 RESTORE 4180
4090 FOR F=1 TO 5
4100 READ Q
4110 LET M(F)=Q
4120 PRINT AT 3+F*2,16;V(F)
4130 PRINT AT 3+F*2,24;M(F)
4140 NEXT F
4150 LET PRE=1
4160 LET POS=1
4180 DATA 1,255,255,255,255
4200 ENDPROC_SETVALUE
4210 :
4220 :
4230 :
5000 @BOXIN
5010 PLOT 20,60: DRAW 111,0
5020 DRAW 0,79: DRAW -111,0
5030 DRAW 0,-79
5040 PLOT 95,0: DRAW 0,36
5050 PLOT 95,19: DRAW 160,0
5060 MODE_1
5070 PRINT AT 18,26;"A     B";
5080 PRINT "     C      D";
5090 PRINT "       E"
5100 INK 4
5110 PRINT AT 15,25;"MAX"
5120 PRINT AT 15,14;"ACTUAL"
5130 MODE_2
5140 INK 2
5150 PRINT AT 12,40;"Q TO QUIT"
5160 INK 6
5170 PRINT AT 15,40;"L.PRIVETT"
5200 ENDPROC_BOXIN
5210 :
5220 :

6000 @SCANKEYS
6010 LET Z$ = "              "
6020 OVER 1
6030 PAPER 1
6040 INVERSE 1
```

```
6050 PRINT AT P(1),6;Z$
6060 REPEAT
6070 LET FINISH=0
6080 IF IN 63486=239 THEN LEFT
6090 IF IN 61438 = 251 THEN RIGHT
6100 IF IN 61438=247 THEN UP
6110 IF IN 61438=239 THEN DON
6120 IF IN 64510=254 THEN QUIT
6130 IF IN 32766=254 THEN NOISE
6150 BEEP .005,-16
6160 UNTIL_FINISH
6200 ENDPROC_SCANKEYS
6210 :
6220 :

7000 @DON
7010 LET POS=POS+1
7020 IF POS>5 THEN LET POS=5
7030 POSITION
7040 ENDPROC_DON
7070 :
7080 :
7100 @UP
7110 LET POS=POS-1
7120 IF POS<1 THEN LET POS=1
7130 POSITION
7140 ENDPROC_UP
7170 :
7180 :

7200 @POSITION
7210 PRINT AT P(PRE),6; Z$
7220 PRINT AT P(POS),6; Z$
7230 LET PRE=POS
7240 ENDPROC_POSITION
7250 :
7260 :

7300 @LEFT
7310 LET S=POS
7320 LET V(S)=V(S)-1
7330 IF V(S)<0 THEN LET V(S)=0
7340 OVER 0
7350 PRINT AT P(S),16;V(S); " "
7360 OVER 1
7370 ENDFROC_LEFT
7380 :
7390 :

7400 @RIGHT
7410 LET S=POS
7420 LET K=M(POS)
7430 LET V(S)=V(S)+1
7440 IF V(S)>K THEN LET V(S)=K
7450 OVER 0
7460 PRINT AT P(S),16; V(S); " "
7470 OVER 1
7480 ENDPROC_RIGHT
7490 :
7500 :

8000 @QUIT
8010 LET FINISH=200
8020 PAPER 0
8030 INK 7
```

```
8040 OVER 0
8050 CLS
8060 ENDPROC_QUIT
8070 :
8080 :

8100 @NOISE
8110 LET A=V(1): LET B=V(2)
8120 LET C = V(3): LET D=V(4)
8130 LET E=V(5): MODE_1
8140 OVER 0
8150 INK 7
8160 PAPER 0
8170 PRINT AT 20,26;A
8180 PRINT AT 20,32;B
8190 PRINT AT 20,38;C; "  "
8200 PRINT AT 20,46:D; "  "
8210 PRINT AT 20,54;E; "  ";
8220 MODE_2
8250 PAPER 1
8260 OVER 1
8270 INK 6
8280 PLAY_A,B,C,D,E
8310 ENDPROC_NOISE
```

128 K

Итак, сегодня мы продолжаем разговор о некоторых особенностях компьютеров с памятью 128K. Мы уже рассмотрели вопросы, связанные с особенностями организации оперативной памяти и знаем, как подключаются дополнительные страницы при программировании в машинных кодах.

Здесь мы рассмотрим работу системы организации памяти в виде так называемого "ЭЛЕКТРОННОГО ДИСКА" (RAM-диска).

Итак, в Бейсике Вы имеете возможность выгружать программы, данные или машинный код на электронный диск примерно таким же образом, как это делается при выгрузке на ленту. Разница, конечно, в скорости этой операции, да и команда немного отличается. При выгрузке на квазидиск - это SAVE!, а при выгрузке на ленту, как Вы знаете, просто SAVE.

Аналогичным образом работают и команды LOAD!, VERIFY!, MERGE!.

Очевидным недостатком электронного диска является то, что все, что на нем хранится, стирается из памяти при выключении питания.

В нашей статье мы рассмотрим организацию системы "электронного диска" с точки зрения логики машинного кода.

Необходимо отметить, что термины "электронный диск", "квазидиск", "RAM-диск" - конечно же очень неточные, поскольку никаких дисков в компьютере нет. Есть 128 килобайт оперативной памяти, а называется система ее использования "электронным диском" только потому, что она имитирует работу как бы с внешним дисководом.

Страница ОЗУ.

В предыдущих выпусках мы с Вами уже говорили о страничной организации памяти и говорили о страницах ОЗУ и страницах ПЗУ. Сейчас мы коснемся страниц ОЗУ несколько глубже. Как Вы помните, в компьютере есть 8 страниц ОЗУ, пронумерованных от 0 до 7, объемом по 16K каждая, что и дает в сумме 128K.

Сейчас же мы добавим к этому, что эти страницы ОЗУ разделены на две группы, которые назовем:

ОСНОВНАЯ ПАМЯТЬ и ПАМЯТЬ ЭЛЕКТРОННОГО ДИСКА.

На Рис. 1 показано распределение страниц ОЗУ по этим группам.

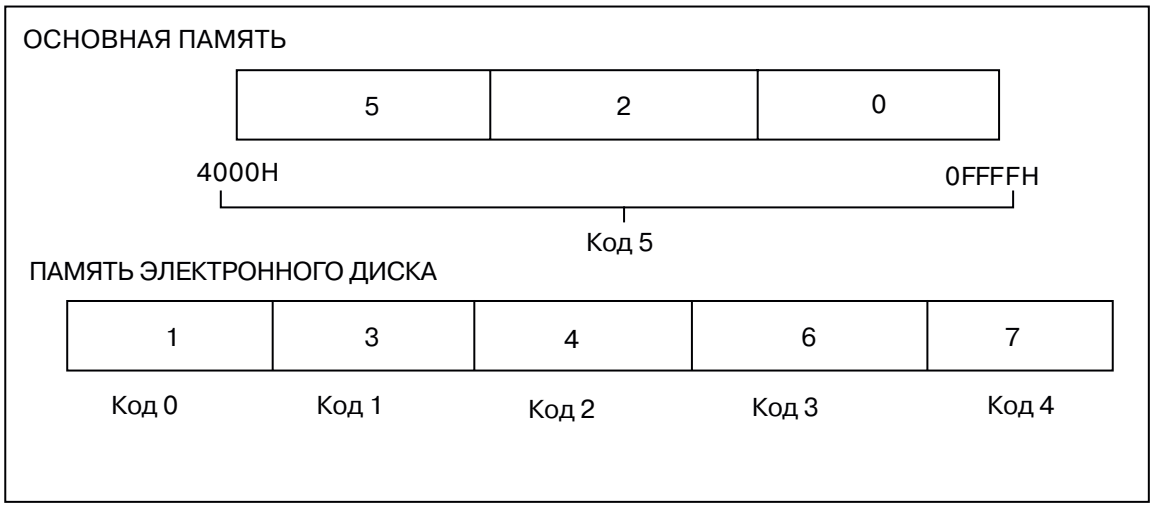


Рис.1

Как видно из рис. 1, основная память содержит три страницы ОЗУ по 16K каждая, что в сумме составляет 48K. Здесь находится память экрана, системные переменные, рабочая БЕЙСИК-программа со своей областью переменных, область информации о каналах, стек калькулятора, графика пользователя и все прочее, чему положено быть. Одним словом, как видите, это раздел 128-го мало отличается от карты памяти стандартной 48-килобайтной

машины. Все нововведения коснулись другого раздела ОЗУ - области памяти электронного диска.

Основная память ОЗУ состоит из страниц 5,2,0 (именно в этом порядке). Страница 5 постоянно впечатана, начиная с адреса 4000H, страница 2 - постоянно, начиная с адреса 8000, а страница 0 - при нормальных обстоятельствах, начиная с адреса 0C000H. Это означает, что для того, чтобы иметь доступ к одной из этих страниц, Вы должны впечатать ее вместо нулевой страницы, начиная с адреса 0C000H (а ее сохранить и впоследствии восстановить). Итак, нельзя иметь одновременный доступ ко всей памяти электронного диска.

Чтобы упростить дальнейшие рассуждения, введем понятие код страницы ОЗУ. Это не то же самое, что номер страницы. На рис. 1 показано, как кодируются страницы основной памяти и памяти электронного квазидиска. Итак, код страницы - число от 0 до 5.

В прошлых выпусках мы говорили о том, что для обозначения памяти выше 64 килобайт четырех шестнадцатеричных разрядов уже недостаточно и ввели пятый разряд - так, что адрес 1C000H обозначает первую страницу ОЗУ и, соответственно, адрес C000H на этой странице, соответственно аналогично можно поступить и с кодами страниц. Рассмотрим начало области памяти электронного диска.

Оно существует на странице ОЗУ с номером 1. Она же - страница с кодом 0. Тогда можно указать на адрес, принадлежащий этой странице, указав на код страницы [0] и на адрес, например C000.

Чтобы избежать путаницы кодов страниц с номерами страниц, мы будем писать код в квадратных скобках.

Итак, начало области электронного диска имеет абсолютный адрес 1C000H, а в терминологии кодов страниц - адрес [0]C000.

Чтобы это было понятнее, вот несколько примеров:

```
1FFFF = [0]FFFF
3C001 = [1]C001
4D800 = [2]D800
6EE00 = [3]EE00
7EBEC = [4]EBEC
4000  = [5]4000
ABCD  = [5]ABCD
F000  = [5]F000
```

Понятно, что одна пара регистров микропроцессора не в состоянии хранить число, большее по размеру, чем четыре шестнадцатеричных разряда, поэтому, чтобы работать с такими адресами в машинном коде, необходимо наличие трех регистров, вместо обычных двух. Причем неважно, в какой системе Вы работаете - с адресацией через номера страниц или через их коды. Обычно, работая в системе электронного диска, удобнее пользоваться кодовой адресацией, а при работе со страничной организацией памяти (как в прошлых выпусках) - удобнее пользоваться абсолютной адресацией по номеру страницы.

В качестве примера рассмотрим такой случай. Нам надо сохранить адрес [4]EBEC в тройном регистре AHL. Тогда регистр A должен содержать 04, а пара HL - число EBEC.

Для чего вообще нужен был переход от номеров страниц к их кодам? Для упрощения операций адресации в машинных кодах. Номера страниц связаны с физической организацией памяти, и потому идут, как видно из рис. 1 не подряд. Коды же страниц - чисто логическая конструкция, и потому идут подряд и манипулировать с ними проще.

Например:

Пусть AHL содержит адрес какого-либо байта в памяти электронного диска. Как отыскать адрес следующего за ним байта (Рис.2):

```
INC HL          - Увеличили адрес на единицу
BIT 7,H         - Проверили старший разряд на 0.
                  Поскольку в нормальных условиях
                  там всегда единица, то появление нуля
                  будет свидетельствовать о том, что
                  страница исчерпана.
RET NZ          - Если не 0, то все в порядке и возврат.
LD HL,C000      - В противном случае переходим на новую
                  страницу и выставляем ее начальный
                  физический адрес C000.
INC A           - Увеличиваем на единицу номер страницы.
RET             - Возврат.
```

Рис.2

А теперь представьте, во что бы превратилась эта нехитрая процедура, если бы мы использовали не коды страниц, а их физические номера. Ведь нельзя же было бы использовать простую и быструю операцию INC A.

На этом мы прервемся до следующего выпуска, в котором подробно рассмотрим карту памяти RAM-диска и разберем несколько примеров по работе с RAM-диском из машинного кода.

FORUM

Мы вновь возвращаемся к вопросу о пределах совместимости отечественных моделей "Спектрума" и будем возвращаться к нему еще не раз, поскольку считаем его одним из наболевших.

На наш взгляд существенный вклад в прояснение ситуации внес т. Симаков из г. Красноярска. Его письмо мы приводим почти полностью и надеемся, что оно заинтересует наших читателей.

Сам он работает с компьютером, собранным по схеме "Балтика" и практически не имеет проблем. Нормально работают и "ELITE" и "TLW2" о которых мы упоминали ранее, но в его компьютере стоит ПЗУ 1982 г., а версии 1987 г. и 1989 г. работают менее надежно.

Знакомые товарища Симакова, владеющие моделью "Дубна" неоднократно обращались к нему с просьбой разобраться с причинами незагружаемости ряда фирменных программ. Было установлено, что от этого страдают программы, имеющие нестандартный загрузчик, т.е. не использующие для загрузки процедуры, размещенные в ПЗУ, а какие - то свои из предварительно загруженного блока (кстати он-то загружается нормально, поскольку его загрузкой управляет ПЗУ).

Сами эти загрузчики похожи на стандартный, но имеют какие-либо особенности, например для вывода счетчика на экран. При этом в них установлены определенные константы, задающие временные интервалы между считыванием отдельных битов с магнитофонного порта (см. РЕВЮ, с. 78-79).

Но все дело в том, что значения этих констант рассчитаны исходя из заданной скорости работы микропроцессора, а в "Дубне" (по крайней мере в попавшихся ему экземплярах) был использован процессор UA880 вместо Z80, имеющий меньшую тактовую частоту.

Разработчики компьютера по-видимому это учли и скорректировали эти константы в ПЗУ компьютера, но загрузка под управлением внешнего загрузчика не идет.

Вскрыв программу и заменив нестандартный загрузчик на обращение к ПЗУ (0556H), ему удалось "реанимировать" несколько программ, например "Exolon", "Freddy Hardest" и ряд других.

По этой же причине по-видимому не работают и TF-COPY и C0FY-86/M, поскольку они тоже применяют свой собственный загрузчик.

* * *

Мы благодарим т. Симакова за информацию и ждем других сообщений по этим вопросам.

МАЛЕНЬКИЕ ХИТРОСТИ

Так сложилось, что экономия памяти стала основной темой этого раздела. По-видимому, это как раз тот вопрос, в котором без хитростей не обойтись. Очень обидно бывает, когда к концу Вашей программы Вы обнаруживаете, что места для чего-то не хватает.

Вторая причина такой направленности нашего раздела в том, что "хитрости" - это нестандартные, а потому нередко не очень удобные приемы и, разумеется, их не применяют без достаточных на то оснований. Острая необходимость сэкономить память - как раз достойное основание.

Но как бы там ни было, будете Вы применять эти приемы или нет - дело Ваше, но узнать побольше о своем компьютере Вы наверное сможете, а сейчас для нас это самое главное.

Какие же программы в первую очередь нуждаются в "хитростях"? В первую очередь текстовые, а именно:

- адвентюрные игры;
- обучающие программы;
- деловые игры.

То есть, это программы с большим количеством текстовых блоков.

Рассмотрим пример:

```
10 PRINT "Нажмите любую клавишу".
```

Занимает такая строка 29 байтов в программной области Бейсика (PROG) и ничего не занимает в области Бейсик-переменных (VARS), но это только пока. После запуска программы всякий раз, когда Вы будете выполнять эту строку, в области VARS под нее будет расходоваться память.

Если Вы предполагаете использовать ее не один раз, то можете поступить так:

Сбросьте программу - NEW, а затем наберите

```
LET a$="нажмите любую клавишу".
```

Если теперь Вы дадите команду LIST, то не получите ничего и это не удивительно, ведь у Вас нет никакой программы.

Теперь наберите:

```
10 PRINT a$
```

а затем дайте прямую команду GO TO 10 и Вы увидите, как Ваше сообщение появится на экране. Волшебство? Нисколько! Просто Ваше сообщение находится в области переменных и хранится там в переменной a$. В области VARS оно занимает 24 байта, а в области PROG всего 8, т.е. в сумме 32 - Вы сэкономили 21 байт.

Если бы вместо GO TO 10 Вы использовали RUN, то получили бы сообщение "VARIABLE NOT FOUND", потому как RUN очищает область программных переменных. Работая таким образом, Вы обязаны использовать GO TO и более того, если в программе есть переменные, содержащиеся только в области VARS, то выгружать ее можно только с номером строки автостарта, чтобы она сама стартовала без необходимости давать RUN.

Кстати, такой прием позволяет немного сбить с толку малоопытного, но любопытного "раскольщика". Ведь по команде LIST, которую он освоил в первый день после приобретения компьютера, он увидит далеко не всю информацию.

"Раскольщика" поопытнее можно перехитрить например другим приемом.

Скажите, пожалуйста, вооружившись всеми своими знаниями, что бы Вы подумали, увидев в первых строках программы запись типа:

```
10 LET start=CODE "A"
20 GO TO start
```

Наверное, Вы открыли бы справочник (если Вы его у нас приобрели) или хотя бы страницу 81 прошлого выпуска "ZX-РЕВЮ" и узнали бы, что код буквы A равен 65. То есть эта запись как бы эквивалентна GO TO 65. Далее Вы просмотрели бы строку 65, где настраивается вся работа программы и пошли бы по ложному пути.

Хитрость может быть например в том, что программист в 10-й строке написал вовсе

не букву A, не верьте своим глазам. Это вполне может быть графический символ из графики пользователя UDG, закрепленный за клавишей A, который пока ничей от нее по образу не отличается, но имеет совсем другой код - 144, и тогда запись эта эквивалентна GO TO 144. А может быть это какой-то другой код, если он предварительно "перевесил" образ буквы "A" на клавишу "F", тогда это будет код 149.

При всей примитивности этого примера, он дает Вам представление о мотиве, которым нередко пользуются профессионалы и, когда он подмешан в густую кашу других элементарных в общем-то приемов, то сильно портит кровь любопытным, особенно если они не знают с чего начать.

А теперь рассмотрим некоторые другие приемы экономии памяти.

В своих текстовых строках Вы можете использовать управляющие символы, это такие:

CHR$ 5 - "запятая PRINT"

CHR$ 16...CHR$ 21 - цветовые атрибуты

CHR$ 8 - BACKSPACE (смещение курсора влево на одну позицию).

Пожалуй о "запятой PRINT" надо поговорить особо. Дело в том, что в наборе символов "Спектрума" существуют две запятые. Они выглядят (на экране) совершенно одинаково, но имеют разное назначение.

Во-первых, это обычная запятая, символ номер 44 (CHR$44), которая не является управляющим кодом, а просто обычным символом.

Во-вторых - это запятая оператора PRINT. Это уже управляющий код. Номер этого символа - 6 (CHR$6). Вот пример ее применения:

```
10 LET a = 5
20 LET b = 10
30 PRINT a,b
```

Вы, конечно, знаете, что делает этот управляющий код, а если не знаете, то попробуйте и увидите. Попробуйте также

```
30 PRINT a;b
```

И еще попробуйте

```
30 PRINT a'b
```

Итак, этот код дает компьютеру команду - выполнить печать на правом поле экрана, а если Вы и так на нем, то перейти на новую строку.

Кстати, напомним, что для того, чтобы просто перейти на следующую строку можно использовать символ " ' " (апостроф).

Итак, предположим, что Вам надо выдать на экран какое-то сообщение длиной более 32 символов. Вам надо строку делить, а делить как попало Вы не хотите - Вам надо это сделать в строго определенном месте.

Вариант первый - строку подбивают дополнительными пробелами (расход памяти).

Вариант второй - использовать апостроф (" ' ") - расход 3-х байтов.

Это можно сделать, если вставить CHR$6 внутрь строки, что в общем-то сделать непросто и дело вот в чем.

Предположим, Вам надо набрать:

```
PRINT "Это первая печатная строка a это вторая".
```

Если Вы поставите запятую перед "a", то это будет обычная запятая (CHR$44).

Если Вы хотите вставить туда CHR$6, то пришлось бы закрывать кавычки:

```
PRINT "Это первая печатная строка", "a это вторая".
```

А здесь есть расход трех байтов.

Можно, однако, этого избежать следующим путем:

- наберите

```
PRINT "это первая печатная строка
```

- перейдите в режим E (CAPS SHIFT + SYMB SHIFT)

- держа клавишу CAPS SHIFT, нажмите "6".

Курсор приобретет желтый цвет. Это произошло потому, что Вы своими действиями ввели два управляющих кода (CHR$16 + CHR$6): CHR$16, как известно управляет цветом INK, а CHR$6 после него означает "желтый".

Нам с Вами символ CHR$6 нужен, а вот CHR$16 - нет, его надо удалить. Удалим его с помощью DELETE (CAPS SHIFT + "0") и курсор сразу "прыгнет" на другое поле. Это значит,

что код CHR$6 воспринят. А теперь закончим наше сообщение:

```
а это вторая"
```

Пока не нажимайте ENTER.

Давайте убедимся, что зазор между "строка" и "а" реально существует. Прогоните курсор влево к началу строки (CAPS SHIFT + "5") и смотрите на него. Видите, как он перепрыгнул через зазор? Вы видите этот зазор? Конечно видите, а вот для компьютера его как бы и не существует, поэтому и нет расхода памяти на заполнение пробелами этого зазора.

Теперь нажмите ENTER и давайте расстанемся до следующего выпуска, который будет большим и даст возможность рассмотреть ещё несколько приемов, связанных с внедрением в текстовую строку управляющих символов.

ПРОФЕССИОНАЛЬНЫЙ ПОДХОД

(Продолжение. Начало на стр. 49, 70)

Разработка программы.

Мой подход к разработке программ базируется на той профессиональной подготовке, которую я получил на фирмах ICL и IBM.

Сам процесс разработки не зависит от того, какой язык программирования я применяю. Конкретное же содержание каждого этапа может изменяться в зависимости от того, какими инструментальными средствами я располагаю на своей машине и, в определенной степени, от типа разрабатываемой программы.

Этап 1. Постановка задачи.

Для чего предназначена программа, что она будет делать? Для игровых программ есть несколько важных моментов, которые должны быть определены уже на ранней стадии:

- каким будет экранный интерфейс (какой метод будет применяться для связи между программой и пользователем);
- тема игры;
- каков предполагаемый размер игры и сколько памяти можно отвести под хранение экранов и таблиц данных.

Выходная информация:

Это прежде всего экраны, но сюда можно отнести и распечатки на принтере, если они предполагаются, здесь же должны быть намечены те выгрузки на ленту или диск, которые могут потребоваться по ходу игры. И, конечно же, должен использоваться звук. Любая программа, разумно его использующая, только выигрывает от этого.

Я говорю о выходной информации, не сказав о входной вовсе не потому, что считаю ее более важной. Просто Вы не определитесь с тем, что надо дать программе, если не знаете, что Вы будете от нее брать.

Для игровых программ стоит рассмотреть в качестве выходных характеристик такие очевидные параметры, как:

- степень использования цвета;
- предполагаемая скорость;
- притягательность и т.п. Составьте список того, что Вы хотели бы иметь в своей программе и, по мере ее разработки, периодически к нему возвращайтесь для сверки.

Входная информация:

Сюда относятся те данные, которые поступают в программу во время игры от клавиатуры или от джойстика. Если необходимо во время игры делать подзагрузку, то сюда же можно отнести данные, считываемые с ленты или диска.

От того, как происходит управление игрой в большинстве случаев зависит судьба этой игры - успех или неудача.

Применение джойстика всегда вносит в игру ограничение в связи с тем, что у него только одна кнопка. Ах, если бы у него было две функциональных кнопки! Тогда первая исполняла бы какое-то действие, а вторая - переключала бы вид этого действия. Например, первая означала бы "огонь", а вторая переключала бы оружие: нож, пистолет, автомат, гранатомет и т.п.

Однако кнопка у нас только одна и все функции, которые Вы хотели бы иметь, надо все-таки как-то к джойстику привязать. Здесь очень легко можно зайти так далеко, что игра окажется трудной в освоении.

Тщательно продумывайте технологию управления игрой. Не хватайтесь за первый попавшийся вариант. Взвесьте несколько альтернативных, выберите лучший и будьте готовы в любой момент от него отказаться, если окажется, что для других людей он слишком сложен.

Файлы и таблицы данных:

На этой стадии достаточно только сделать грубую прикидку того, какие данные могут потребоваться программе во время работы. Оцените их размер, предварительно наметьте метод, которым данные будут выбираться из таблиц.

Например, если в игре есть много комнат, надо наметить, как эти комнаты будут нумероваться и как будут кодироваться двери, связывающие разные комнаты между собой.

Первый этап должен закончиться твердым ответом на вопрос: возможно ли создание той программы, которую я задумал или все надо начать сначала. Обычно к этому моменту мой стол уже завален грудой набросков, эскизов, таблиц, которые смогут пригодиться на следующем этапе.

Этап 2. Проектирование программы.

Теперь настало время решать как Вы будете программировать все то, что задумали и наметили на первом этапе.

Работа начинается с подчистки и увязки результатов первого этапа. Я предпочитаю сначала строго расписать все таблицы данных, которые были намечены и заканчиваю эту работу присвоением имен переменным, которые будут хранить эти данные. Обратите внимание на то, что значительная часть пути уже пройдена, а компьютер Вам до сих пор еще не был нужен. Именно в этом и состоит суть моего подхода. Тщательная проработка структур данных на бумаге до начала программирования позволяет разбить предполагаемую программу на несколько независимых частей, с каждой из которых работать проще, чем со всей программой в целом. Каждая часть, в свою очередь делится на процедуры. Каждая процедура выполняет отдельную конкретную задачу, достаточно простую, чтобы можно было запрограммировать ее за один прием, без необходимости делить на части.

Правильное проектирование программы - самый критический фактор для успеха или провала всей Вашей работы. Плохо спроектированные или вообще неспроектированные программы занимают столько времени на программирование и отладку (если вообще отлаживаются до конца), что когда они готовы, то обычно быстродействие их работы бывает неудовлетворительным. Вам очень дорого будет стоить пренебрежение этим этапом. Начинающему не терпится начать программирование, он полагает, что методом проб и ошибок, внося коррективы в программу по ходу разработки, решит все проблемы. Опытный программист, наоборот, отдает половину своего времени именно этой работе без компьютера, потому что знает, что когда коррекций в программе становится слишком много, то они начинают влиять друг на друга и быстро губят всю работу. Более того, еще на докомпьютерном этапе он уже наметит, что будет подлежать последующим уточнениям и коррекциям с целью наилучшей играбельности программы и так организует программу, чтобы эти уточняемые параметры не влияли друг на друга.

Итак, одним словом, для всех языков программирования желательно, а при программировании в машинном коде (или на Ассемблере) абсолютно необходимо расчленить задачу на множество микрозадач, каждую из которых можно запрограммировать так, чтобы текст этой микропрограммы мог сформироваться и уместиться в Вашей голове.

Этап 3. Кодирование.

На этом этапе записывается процедура в машинных кодах (на ассемблере), но сначала надо завести рукописный справочник по всем переменным, используемым в Вашей программе.

Если Вы полагаете, что и так все запомните, то лучше не экспериментируйте, для этого надо иметь необычную память. По каждой переменной я записываю имя, размер, содержание, примечание.

Например:

Имя	Размер	Содержание	Примечание
BOMB	1 байт	0 – нет 1 - есть	Наличие бомбы устанавливается в процедуре GETBOMB. Изменяется в процедуре USEBOMB.

Кроме справочника по переменным, надо вести еще и справочник по процедурам:
- имя процедуры;
- что делает;
- содержание регистров на входе;
- содержание регистров на выходе;
- какие переменные использует.

Например:

Имя	Что делает	Вход	Выход
FIRE	Инициализирует пакет процедур, управляющих стрельбой.	A - вид оружия 0…3 0 - нож 1 - пистолет 2 - автомат 3 - гранатомет BC,DE,HL – не определены F - флаг C=0 – выполнить стрельбу - флаг C=1 – поменять оружие - прочие флаги не определены	A – сила выстрела от 0 до 255
Используемые переменные:	WEAPON 1 байт Вид оружия CHARGE 3 байта Количество зарядов на каждый вид оружия		

Конечно, если Вы пишете программу с помощью Ассемблера, то эту информацию можно включать в исходный текст в виде комментария, но во-первых, лучше иметь эти листки на каждую процедуру под рукой, во-вторых, Вы быстрее приобретете профессиональный стиль и, в-третьих, это серьезный шаг вперед на пути создания собственной библиотеки процедур, которая позволит Вам в несколько раз сократить время на разработку будущих проектов.

Не надо пытаться написать сразу всю программу. Начните с программирования самых малых, не взаимодействующих с другими процедур. Их Вы быстро напишете и испытаете.

Этап 4. Отладка.

Я предпочитаю всякую процедуру проверять сразу же после того, как написал. Как правило, для этого необходимо приписывать несколько строк, чтобы обеспечить те данные, которые эта процедура должна получать из главной программы. Иногда даже необходимо сразу писать несколько процедур, испытывать и отлаживать совместно, если они друг без друга не отлаживаются, но чем их меньше, тем лучше. Есть закон, согласно которому если размер текста возрастает вдвое, то количество ошибок в нем возрастет в четыре раза. Есть и другой закон - если количество ошибок возрастает вдвое, то время на их отыскание и устранение тоже возрастает в 4 раза. Так что сами думайте, какой размер должны иметь Ваши элементарные процедуры. Заканчивается проверка и отладка процедуры тогда, когда Вы уверены, что в проведенном независимом испытании она работает и делает все, что положено. Правда, к сожалению, это еще не дает 100%-ной уверенности в том, что она также надежно будет работать в комплексе с другими при запуске всей программы в целом. Но если этап проектирования был хорошо проработан, то Вы сможете подкорректировать ее работу в составе всей программы без нарушения работы других блоков.

В основном эти проблемы совместной работы отлаженных порознь процедур порождаются взаимообменом данными между процедурами, который происходит через переменные, содержащиеся в заданных ячейках памяти, через регистры процессора или через флаги процессора. Эти ошибки будут минимальными, если Вы в свое время четко расписали для каждой процедуры, что она имеет на входе, что выдает на выходе и с какими регистрами работает.

Чтобы минимизировать вред от ошибок, связанных с взаимодействием между процедурами, я обычно "прикрепляю" новую к уже отлаженной. Так, постепенно, отлаженные процедуры создают среду для проверки и отладки новых.

Что такое хорошо и что такое плохо?

Хорошее программирование означает понятное программирование и минимум ошибок. Помните, что Вам многократно предстоит возвращаться назад и что-то переделывать, поэтому пожалуйста не оставляйте сами себе ловушек. Вот примерный список приемов плохого стиля:

1. Использование "хитрых", головоломных приемов.

Я считаю, что в принципе этого делать не надо. Исключение - тот случай, когда это абсолютно необходимо для достижения максимальной скорости работы, но и здесь это может быть в минимальном количестве особо важных (по быстродействию) процедур.

2. Злоупотребление применением стека.

Лучше оставить стек в покое. Особенно плохо применять его для хранения программных переменных. Вызов переменной по имени (в Ассемблере) - намного понятнее, быстрее и создает меньше условий для появления ошибок.

3. Неправильное использование процедур.

Каждая процедура должна иметь один вход и один выход. Если Вам их нужно больше, то вернитесь на этап 2 и повторите проектирование.

Никогда не имитируйте вызов процедуры (CALL) путем перехода (JP) с последующей манипуляцией стеком. Прием слишком головоломный, чтобы пользоваться им часто. Всегда завершайте процедуру естественным путем (команда RET в конце процедуры).

4. Злоупотребление переходами (JP).

Команда JP, как и GO TO в Бейсике очень затрудняет читаемость программы. При широком применении переходов JP Вы затрудняете себе отладку. Используйте JP только для организации циклов и для исполнения условного ветвления (IF… THEN…).

5. Работа с адресами.

Программируя на Ассемблере, не задавайте адреса, а определяйте только имена переменных, процедур и т.п. Если Вы будете вызывать процедуру по адресу, то все будет хорошо, пока Вам не придется ее переместить в другое место и тогда вся программа рассыплется как карточный домик.

К чертам хорошего стиля я отношу:

1. Определение имен со смыслом (название отражает содержание).
2. Комментарий к каждой процедуре.
3. Комментария к каждому "головоломному" (нестандартному) приему.
4. Применение регистров процессора по назначению:

A - аккумулятор;

B - восьмиразрядный обратный счетчик;

BC - шестнадцатиразрядный обратный счетчик;

HL - для хранения адреса или как двухбайтный аккумулятор.

DE - для хранения адреса в операциях, в которых необходимо использование двух адресов.

IX,IY - для выборки элемента из таблицы с использованием индексной адресации.

ПРИМЕЧАНИЕ ИНФОРКОМА.

Если Ваша программа предполагает обращение к системному ПЗУ "Спектрума", то лучше в регистр IY ничего не засылать, а использовать его для выбора данных из таблицы системных переменных "Спектрума", - прием, ставший стандартным.

Регистр же IX в этом случае наиболее часто используют в операциях загрузки/выгрузки.

В качестве примера распечатки программы в машинных кодах, обещанной в прошлом выпуске "ZX-РЕВЮ", мы приводим процедуру Стива Тернера для печати текстовых сообщений в любой точке экрана с точностью до пиксела (а не только в координатах знакомест).

Она может применяться в играх, например для выдачи на экран текущего счета.

Данный пример распечатает на экране две строки: EXAMPLE OF TEXT TABLE. Первая

строка содержится в относительном адресе 00AB, а вторая - в 00B3. Обратите внимание на то, что перед строкой, выводимой на печать, должен стоять байт, в котором указана длина этой строки:

00AA - 07 (7 символов в слове "EXAMPLE")
00B2 - 0D (13 символов в строке "OF TEXT TABLE").

(Продолжение в следующем выпуске)

```
0000             00010              ; Пример программы
0000             00020              ; печати текста
0000  3E01       00030              LD A,1       ; Номер сообщения
0002  160A       00040              LD D,10      ; Координата “y” места печати
0004  1E0A       00050              LD E,10      ; Координата “x”
0006  CD0A00     00060              CALL TEXT
0009  C9         00070              RET
000A             00080              ; Подпрограмма TEXT
000A             00090              ; A - номер сообщения
000A             00095              ; DE - координаты печати.
000A             00100  TEXT
000A  21AA00     00110              LD HL,TXADD;
000D  0600       00120  LOOK         LD B,0
000F  1803       00130              JR GOTRY
0011  4E         00140  LONG         LD C,(HL)    ; поиск текста
0012  23         00150              INC HL
0013  09         00160              ADD HL,BC
0014  3D         00170  GOTRY        DEC A
0015  20FA       00180              JR NZ,LONG
0017  CD1B00     00190              CALL PRINT
001A  C9         00200              RET
001B             00210
001B             00220
001B             00230              ; Подпрограмма PRINT
001B  22A800     00240  PRINT        LD (INPUT),HL
001E  7A         00250              LD A,D
001F  E638       00260              AND 38H      ; определяем ряд
0021  87         00270              ADD A,A
0022  87         00280              ADD A,A
0023  83         00290              ADD A,E      ; прибавили “x”
0024  5F         00300              LD E,A       ; младший байт
0025  70         00310              LD A,D
0026  1F         00320              RRA
0027  1F         00330              RRA
0028  1F         00340              RRA
0029  E618       00350              AND 018H     ; определяем треть экрана
002B  4F         00360              LD C,A
002C  7A         00370              LD A,D
002D  E607       00380              AND 07H      ; строка в ряду
002F  81         00390              ADD C
0030  C640       00400              ADD 40H      ; старший байт
0032  57         00405              LD D,A
0033  2AA800     00410              LD HL,(INPUT)
0036  ED53A400   00420              LD (OUTLIN),DE
003A  7E         00430              LD A,(HL)
003B  A7         00440              AND A
003C  C8         00450              RET Z        ; неверное значение
003D  4F         00460              LD C,A
003E  ED53A600   00470  PROLET       LD (OUTPUT),DE
0042  23         00480              INC HL
0043  22A800     00490              LD (INPUT),HL
0046  7E         00500              LD A,(HL)
0047  FEFF       00510              CP 0FFH
0049  2016       00520              JR NZ,CHAR
004B             00530              ; переход к новой строке
```

```
004B             00540             ; под первой позицией
004B             00550             ; печати
004B  ED5BA400   00560             LD DE,(OUTLIN)
004F  7B         00570             LD A,E
0050  C620       00580             ADD 20H
0052  5F         00590             LD E,A
0053  D0         00600             RET NC
0054  7A         00610             LD A,D       ; другая треть экрана
0055  C608       00620             ADD 8
0057  FE58       00630             CP 58H
0059  D0         00640             RET NC
005A  57         00650             LD D,A
005B  ED53A400   00660             LD (OUTLIN),DE
005F  183C       00670             JR REJOIN
0061             00680
0061  6F         00690  CHAR        LD L,A       ; поиск символа
0062  2600       00700             LD H,0
0064  29         00710             ADD HL,HL
0065  29         00720             ADD HL,HL
0066  29         00730             ADD HL,HL
0067  ED5B365C   00740             LD DE,(CHASET)
006B  19         00760             ADD HL,DE
006C  ED5BA600   00770             LD DE,(OUTPUT)
0070  7A         00780             LD A,D
0071  FE58       00790             CP 58H
0073  D0         00800             RET NC
0074  0608       00810             LD B,8
0076  7E         00820  EIGHT       LD A,(HL)
0077  12         00830             LD (DE),A    ; выдает
0078  23         00840             INC HL       ; восемь
0079  7A         00850             LD A,D       ; строк
007A  E607       00860             AND 07       ; символа
007C  3C         00870             INC A
007D  FE08       00880             CP 8
007F  200E       00890             JR NZ,NONEW
0081  7A         00900             LD A,D
0082  E6F8       00910             AND 0F8H
0084  57         00920             LD D,A
0085  7B         00930             LD A,E
0086  C620       00940             ADD 20H
0088  5F         00950             LD E,A
0089  3005       00960             JR NC,NEXTSL
008B  7A         00970             LD A,D
008C  C608       00980             ADD 8
008E  57         00990             LD D,A
008F  14         01000  NONEW       INC D
0090  10E4       01010  NEXTSL      DJNZ EIGHT
0092  ED5BA600   01020             LD DE,(OUTPUT)
0096  1C         01030  NEXTLT      INC E
0097  2004       01040             JR NZ,REJOIN
0099  7A         01050             LD A,D
009A  C608       01060             ADD A,8
009C  57         01070             LD A,D
009D  2AA800     01080  REJOIN      LD HL,(INPUT)
00A0  0D         01090             DEC C
00A1  209B       01100             JR NZ,PROLET
00A3  C9         01110             RET
00A4  01120
00A4  0000       01130  OUTLIN      DW 0         ; начальная строка
00A6  0000       01140  OUTPUT      DW 0         ; адрес печати
00A8  0000       01150  INPUT       DW 0         ; адрес текста
5C36             01160  CHASET      EQU 23606    ; указывает на
00AA             01170                           ; адрес шрифта в ПЗУ ми-
00AA             01180                           ; нус 256 байтов
00AA             01190             ; Таблица текста
```

```
00AA  07          01200  TXADD       DEFB 7
00AB  4558414D    01210               DEFM 'EXAMPLE'
00AF  504C45
00B2  0D          01220               DEFB 13
00B3  4F462054    01230               DEFM 'OF TEXT TABLE'
00B7  45585420
00BB  5441424C
00BF  45
00000 TOTAL ERRORS
```

В ВАШУ ЗАПИСНУЮ КНИЖКУ

Поскольку в этом номере нет технического приложения, мы начинаем новый цикл, который может заинтересовать всех тех, кто умеет держать в руках паяльник.

Проведя анализ многочисленных зарубежных источников, мы выбрали несколько десятков оригинальных периферийных устройств и в ближайших выпусках дадим о них краткую информацию.

Пусть это будет ответом на довольно многочисленные просьбы наших читателей порекомендовать им, какими разработками стоит заняться. Посмотрев, что бывает для "Спектрума", Вы может быть определите и область приложения собственных сил.

Сразу оговоримся, что наши источники информации имели рекламный характер и потому более подробной технической информацией по этим устройствам мы не владеем, тем более что как Вы конечно знаете, "ИНФОРКОМ" не занимается аппаратными вопросами.

Для справки в обзоре приводим также названия фирм-производителей и цену в английских фунтах стерлингов.

Интерфейсы дисководов.

1. Disciple.

Фирма: Miles Gordon.

Цена: 69.95 ф. ст.

Поддерживает до двух дисководов с объемом памяти до 800K на каждом. Загрузка программы 48K занимает 3,5 секунды. Совместим с любыми стандартными дисководами (одно- и двухсторонними, одинарной и двойной плотности; с 40 и 80 дорожками: диаметром 3.0; 3.5; и 5.25 дюйма).

Имеет кнопку snapshot для остановки программы в любом месте и мгновенной выгрузки ее на диск.

Имеет встроенный интерфейс "Центроникс" для подключения принтера, поддерживает при работе с принтером команды LLIST, LPRINT, COPY.

Имеет встроенные порты как под Кемпстон-джойстик, так и под Синклер-джойстик.

Операционная система позволяет "Спектруму" служить головной станцией для обслуживания принтера, двух дисководов и еще 64-х "Спектрумов".

2. Beta 128 Disk Interface.

Фирма: TECHNOLOGY RESEARCH

Цена: 109.25 ф. ст.

Может поддерживать до 4-х дисководов практически любого типа. Основное достоинство - в наличии кнопки "magic button" для остановки работающей программы и сброса ее на диск.

В нашей стране наибольшее распространение нашли версии именно этого интерфейса.

Основной недостаток - операционная система не обходит испорченные сектора, поэтому для серьезной работы может применяться только с большой осторожностью.

3. Delta Disk Interface.

Фирма: TECHNOLOGY RESEARCH

Цена: 149.00 ф. ст.

Дополнительно к предыдущей модели имеет встроенный интерфейс для принтера "Центроникс" и расширение памяти ОЗУ до 128K.

4. Discovery 1

Фирма: OPUS

Цена: 99 ф. ст. (с дисководом 3,5 дюйма).

Самая дешевая и популярная модель. Отличается высокой надежностью и нашла наибольшее применение в зарубежных странах.

Может поддерживать до двух дисководов. Хоть и поставляется с дисководами 3,5 дюйма, но может дооснащаться дисководами 5,25 дюйма.

Интерфейс, блок питания и до двух дисководов размещаются в едином блоке, от него же питается и сам компьютер.

Дополнительно интерфейс имеет порт джойстика (Кемпстон) и принтера (Центроникс), а также выходной порт для подключения прочей аппаратуры.

Интерфейсы джойстика.

1. Single Port.
Фирма: DK'Tronics
Цена: 9.95 ф. ст.

Имеет один порт для стандартных джойстиков, имитирующий команду IN 31 (Кемпстон-интерфейс).

2. Dual Port.
Фирма: DK'Tronics
Цена: 13.00 ф. ст.

Имеет два порта. Первый - для кемпстон-джойстика, второй - имитирует клавиши 6,7,8,9,0 - Синклер (правый).

3. Programmable Joystick Interface.
Фирма: DK'Tronics
Цена: 14.95 ф. ст.

Позволяет использовать джойстик с любой игрой. Поставляется с программным обеспечением, позволяющим запрограммировать его от клавиатуры.

4. Comcon Programmable Joystick Interface.
Фирма: Comcon
Цена: 19.95 ф. ст.

Программируемый интерфейс для джойстика, очень легко и просто перепрограммируется путем установки шести перемычек в позиции, необходимые для имитации джойстиком действия определенных клавиш. Может поддерживать джойстики с двумя независимыми кнопками, например одна кнопка "огонь", а вторая - "бомбометание". Выбор настройки возможен в любое время, в том числе и когда игра уже полностью загружена.

5. Fox Programmable Joystick interface.
Фирма: Fox Electronics Ltd.
Цена: 24.95 ф.ст.

Программируемый интерфейс. Совместим с любыми программами. Одновременно может хранить в памяти расположение клавиш для 16 программ. Имеет встроенный источник питания на батареях, поэтому не теряет настройку и после отключения. Совместим со всеми джойстиками "Атари"-типа, поддерживает функцию "автострельба".

Звуковые устройства.

1. Digital Sound Sampler.
Фирма: Datel Electronics
Цена: 49.99 ф.ст,

Анализатор звука (оцифровщик, дигитайзер). Позволяет Вам переводить в "двоичную" цифровую форму любые звуки и сигналы и записывать их в память компьютера. Воспроизводиться звук может в любой тональности, как вперед, так и назад, с восходящим

тоном или с нисходящим, с эхоэффектом, в замкнутых циклах и т.п.

Поставляется с пакетом программного обеспечения. Кроме перечисленных, имеет еще ряд возможностей.

2. Sound Sampler.
Фирма: CHEETAH
Цена: 45.95 ф.ст.
Анализатор звука.

3. Sweet Talker.
Фирма: CHEETAH
Цена: 24.95 ф. ст.
Синтезатор речи.

4. Mega Sound.
Фирма: CHEETAH
Цена: 10.95 ф. ст.
Усилитель звука.

5. Specdrum.
Фирма: CHEETAH
Цена: 29.95 ф. ст.
Синтезатор звука ударных инструментов.

6. MIDI Interface.
Фирма: CHEETAH
Цена: 49.95 ф. ст.
Интерфейс MIDI для подключения внешних электромузыкальных инструментов.

7. Micon
Фирма: XR1 SYSTEMS
Цена: 109.00 ф. ст.

Это интерфейс типа MIDI высокой сложности. Позволяет подключать внешнее музыкальное оборудование, регулировать звук не только по высоте, но и по длительности и окрасу, причем все это одновременно по восьми каналам. К интерфейсу прилагается математическое обеспечение.

Прочие устройства

1. Snapshot
Фирма: DATEL ELECTRONICS
Цена: 24.99 ф. ст.

Программа "замораживается" нажатием кнопки. Ее можно выгрузить, можно ввести в нее необходимые изменения через POKE. После загрузки программа будет стартовать точно с того места, в котором она была выгружена.

2. Robotics and Model Control (ROBOTEK)
Фирма: DATEL ELECTROHICS
Цена: 29.99 ф. ст.

Имеет 4 независимых выхода для подключения роботов, двигателей, световых источников, бытовой техники и т.п. Имеет 8 независимых входов от чувствительных устройств и т.п. Легкое и удобное в работе устройство поставляется со всеми необходимыми шнурами.

3. Lightwriter

Фирма: DATEL ELECTRONICS
Цена: 14.99 ф. ст.

Это световое перо поставляется с необходимым комплектом математического обеспечения и с соответствующим интерфейсом. С его помощью можно рисовать на экране, в том числе кривые и замкнутые линии, стирать изображения, заполнять контуры, менять цвета INK и PAPER. Все необходимые функции пера выбираются из экрана с помощью самого пера.

4. Kempston Mouse
Фирма: KEMPSTON
Цена: 69.95 ф.ст.

Манипулятор мышь предназначенный для выполнения графических работ, в частности, для изображения гладких кривых.

Мышь совместима с Кемпстон-интерфейсами. К сожалению, математического обеспечения под нее кроме адаптированной версии Артстудии нет, хотя возможно, что Артист-2 тоже с ней совместим. Мышь использует команду IN для определения своего положения относительно курсора на экране.

x - координата: IN 64479
y - координата: IN 65503
Мышь имеет две клавиши.

5. Star Mouse
Фирма: SAGA SYSTEM
Цена: 49.95 ф.ст.

Эта мышь немного меньше, чем ее кемпстоновская сестра. Она имеет только одну клавишу. Ее можно разбирать и чистить.

Единственный пакет графического обеспечения, совместимый с ней - это управляемый через пиктограммы CAD пакет. Панель пиктограмм на экране дает возможность выполнять операции черчения и рисования.

Фирма сделала возможность работать при помощи этой мыши со своим редактором "THE LAST WORD". Эта мышь имеет меньшие возможности, чем предыдущая, но с учетом значительно более низкой цены выглядит привлекательнее.

6. AMX Mouse
Фирма: AMX
Цена: 69.95 ф.ст.

Эта мышь решительно выигрывает по сравнению с двумя предыдущими. Пакет, ее сопровождающий, выглядит внушительно. Многие фирмы, выпускающие математическое обеспечение, неоднократно заявляли о готовности работать под нее, по крайней мере в базах данных, электронных таблицах и графических пакетах. "Артстудия" имеет опцию для этой мыши. Мышь подключается через прилагаемый интерфейс, имеющий одновременно порт "Центроникс" для подключения принтера.

Это единственная мышь из всех, имеющая три клавиши. Они могут независимо программироваться, их потенциал очень велик.

Солидный пакет прилагаемых программ включает в себя "AMX ART" - программу для управления аппаратным обеспечением; "MOUSE CONTROL" - программу для добавления в Бейсик новых 23-х команд, связанных с управлением мышью; "ICON DESIGNER", "CALCULATOR" И "PUZZLE".

7. Datapen
Фирма: DATAPEN MICROTECHNOLOGY
Цена: 20.95 ф.ст.

Это светоперо имеет характеристики, которых нет у других. Операционная кнопка находится на пере, что дает возможность не использовать клавиатуру. Оно может работать в

любых условиях внешнего освещения. Работает с точностью до пикселя.

Имеет 20 заранее определенных команд, позволяющих наносить геометрические объекты (треугольники, линии, окружности и т.п., а также текст и символы графики пользователя). Работает с цветом, имеется накладываемая сетка, легкость стирания, возможность выгрузки рисунков на ленту.

8. Videoface
Фирма: DATA-SCIP
Цена: 69.00 ф.ст.

Устройство предназначено для оцифровывания изображений. Совместимо с микродрайвом, а также дисковыми системами "Опус" и "Бета".

Время оцифровки экрана - 0. 27 сек. Все математическое обеспечение управляется от меню. Имеет возможность хранить в памяти 6 полных экранов. Может применяться в развлекательных и профессиональных целях.

Необходимые шнуры прилагаются.

9. Scorpion
Фирма: MICRO-ROBOTICS
Цена: 249.00 ф.ст.

Этот контроллер содержит 24K ОЗУ и 32K ПЗУ. Он открывает необъятные возможности по части подключения аналоговых, цифровых источников сигналов, также для управления сервомеханизмами, светодиодными экранами, источниками инфракрасного излучения и многое др. Если у Вас в голове есть какой-нибудь проект по использованию "Спектрума", Вы без сомнения найдете возможность применить "Скорпион".

Существенный недостаток - очень высокая цена.

10. Spectrum Doctor Screenex
Фирма: SCREEN MICROCOMPUTERS DISTRIBUTION
Цена: 57.43 ф.ст.

Устройство предназначено в первую очередь для профессионалов.

"Доктор "подключается к внешнему разъему "Спектрума" и прогоняет через компьютер серию тестов. Телевизор Вам уже не нужен, т.к. на панели прибора имеются 15 светодиодов, показывающих состояние систем компьютера. Выполняются проверки питания, адресной шины и шины данных, блока ULA и процессора.

Проверяет наличие неправильных байтов в ПЗУ и исправность ОЗУ. Для проверки ОЗУ низших 16K и верхних 32K выполняются отдельно.

Скорость и простота диагностирования таковы, что блок может быть полезен тем, кто выполняет ремонтные работы.

ADVENTURE PROJECT

Игра - дело серьезное, а игра адвентюрная - дело серьезное вдвойне. Если Вы полагали, что дав серию статей по методике работы с адвентюрными игровыми программами, мы с ними закончили, то Вы чуть-чуть ошибаетесь.

Сегодня мы начнем новый цикл статей и покажем на примере адвентюрных программ как могут применяться принципы искусственного интеллекта. А это уже согласитесь дело совсем нешуточное и, возможно, знание этих элементарных базовых принципов поможет когда-нибудь в будущем в Вашей практической работе.

Первые адвентюрные игры в основной представляют собой просто путешествие по игровым полям (локациям), информация о которых содержится в памяти ЭВМ. Но с течением времени программы стали усложняться, и в первую очередь усложнение пошло как по пути введения в игру элементарной графики, так и по пути введения некоторых базовых концепций, связанных с искусственным интеллектом. Первыми образцами такого подхода стали The Hobbit и Sherlock фирмы MELBOURNE HOUSE и в какой-то степени Valhalla фирмы LEGEND.

Такой подход позволил дать игровым персонажам возможность вести в общем программном окружении как бы собственную жизнь. В программах появился эффект "реального времени". И Ваш главный герой, которым Вы управляете, получает возможность общаться с другими персонажами и "жить" в игровом пространстве.

И, как Вы увидите, делается это совсем несложно. Все, что для этого надо - это определить правила поведения героев и правила их реакции на игровые ситуации, возможные в ходе игры.

Вы, должно быть, знаете, что когда дело доходит до обработки правил, то очень хорошо подходит язык ПРОЛОГ, который кстати для "Спектрума" тоже реализован в версии "ZX-Микропролог", но мы в рамках данной статьи будем обсуждать базовые концепции на примерах, написанных на БЕЙСИКЕ.

Конечно, БЕЙСИК далеко не идеален, но зато понятен всем, а уж применять основные принципы Вы можете где хотите и на том языке, какой Вам будет удобнее - ПАСКАЛЬ, СИ и др.

Мы не будем останавливаться на методике разработки сценария игры - это важный момент, но не тема данной статьи. Для этого Вам необходимы фантазия, вдохновение, художественный вкус. Скажем только, что как правило ситуации реальной жизни имеют более мощный эффект, чем надуманные конструкции.

Когда сценарий игры продуман, на бумаге расписывается карта локаций, по которым могут путешествовать персонажи.

Программа может иметь несколько игровых пространств, перемещение между которыми выполняется подъемом по лестницам, спуском в ущелья и т.п. В таком случае локации помечаются сначала меткой, указывающей на уровень, а затем на координаты позиции. Эти метки локаций могут не иметь никакого отношения к тому, с какими описаниями они появятся в игре, но служат как бы путеводителем программисту и дизайнеру программы, особенно если это разные люди.

Предположим, что в одном академическом НИИ, расположенном на морском берегу, группа налетчиков похитила уникальный опытный экземпляр часов, способных управлять развитием Вселенной. Преступники скрылись на моторной яхте, а часы перепрограммировали на обратный ход. Время пошло вспять.

Вас зовут Инф. Вам, с Вашими помощниками Орой и Комом, надо найти налетчиков и вернуть часы до того, как обратный ход времени превратит Вас в ребенка или вообще уничтожит Вас как еще не появившегося на свет.

Игра начинается на берегу моря. Двигаясь вправо, Вы попадаете в моторную лодку. Остальные направления показаны на рис.1.

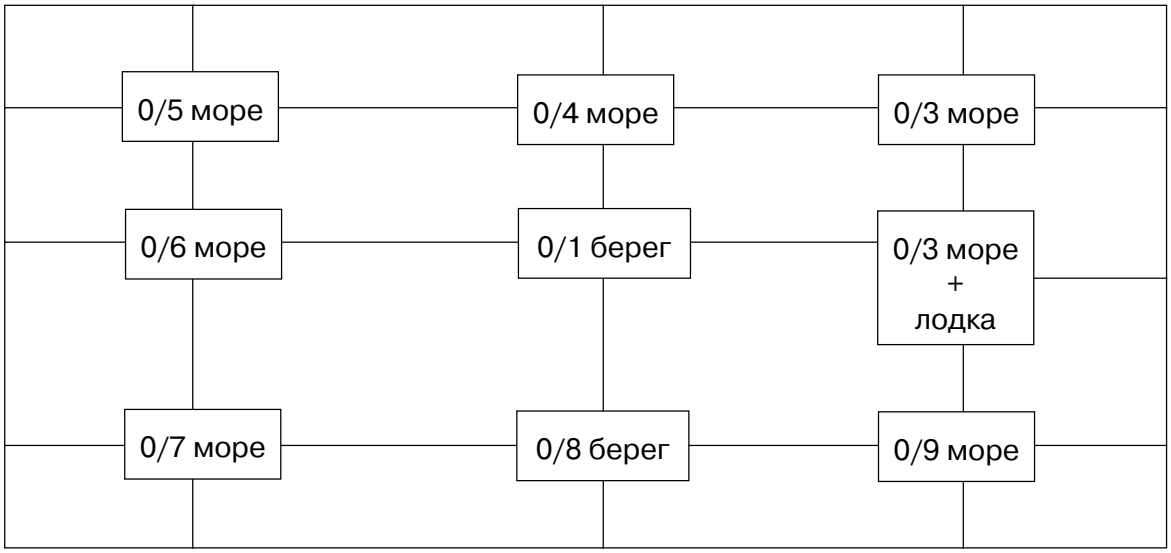


Рис.1

На этом этапе Вам для описания локации достаточно одного-двух слов, но Вы-то конечно помните, что в игре каждая локация будет появляться со своим каким-то достаточно подробным описанием.

Лучше всего завести отдельную картотеку. Пометьте каждую карточку шифром, определяющий номер локации. Затем дайте краткое описание, полное описание и список объектов (предметов), которые могут быть здесь обнаружены.

Карточка может выглядеть примерно так, как показано на рис.2.

0/2 МОРЕ

Вы упали в ледяную воду. От холода у Вас начинают неметь конечности. Еще немного, и вы утонете.

+ ЛОДКА

Вы взобрались на борт моторной лодки, слегка покачивающейся на волнах.

ПРЕДМЕТЫ:

- ключ зажигания;
-

Рис. 2

В качестве уровня игрового пространства здесь взят 0. Всякий уровень, находящийся выше, будет обозначаться положительным числом, а уровень ниже - отрицательным. Например, когда Вы настигнете налетчиков и подниметесь на борт яхты, уровень будет +1, а когда спуститесь в трюмное помещение, то -1.

Когда Вы перейдете от разработки проекта непосредственно к программированию, то обнаружите, что локации и объекты могут конкурировать друг с другом и сделать программирование более сложным, чем ему положено быть. Например, лодка может поставить Вас перед проблемой, которая хоть и не очевидна, но оказывает влияние на структуру программы и ход игры. Проблема в том, чем считать лодку - локацией или объектом, поскольку она перемещается.

Если ее считать объектом, то можно ею не заниматься до тех пор, пока не дойдете до размещения объектов. Описание локации в этом случае такое, как на рис.2.

Учет лодки как объекта, имеет определенные преимущества, поскольку тогда ее можно перемещать по ходу игры. Более того, с ней могут перемещаться и персонажи. С точки зрения гибкости игры гораздо удобнее иметь лодку и перемещаться с нею по

локациям, чем описывать путь по фиксированным локациям пространства игры. При таком подходе единственная структура данных, необходимая нам пока - это обычный массив.

На рис.3 показан текст программы, описывающей приведенную выше карту в виде двумерного массива размером 3x3. Каждый элемент массива имеет место для 10 символов для краткого описания локации.

```
10 DIM g$(3,3,10)
20 FOR i=1 TO 3
30 FOR j=1 TO 3
40 READ g$(i,j)
50 NEXT j: NEXT i
60 DATA "Море","Море","Море"
70 DATA "Море","Берег","Море"
80 DATA "Море","Берег","Море"
90 REM PRINT ROUTINE
100 FOR i=1 TO 3
110 FOR j=1 TO 3
120 PRINT g$(i,j);" ";
130 NEXT j
140 PRINT
150 NEXT i
```

Рис.3

Если Вы пишете программу в машинном коде, Вы сведете все данные в единую таблицу, а на конкретный элемент будете указывать с помощью индекса (смещения).

Конечно, Вы понимаете, что приведенный выше пример расточительно использует ограниченную память компьютера. Так, семь раз повторяется локация с описанием "море" и два раза "берег". Вы, конечно, найдете способы сэкономить здесь немалый объем. Надо также иметь в виду и ограничения по скорости, если работаете на БЕЙСИКе. "Спектрум" может затратить пол-минуты на то, чтобы обработать массив всего лишь 100x100.

Теперь перейдем к проектированию персонажей. Этот процесс состоит из трех основных этапов:

- характеристика;
- изменчивость;
- коммуникабельность.

Последние два взаимосвязаны. Действительно, взаимодействие между персонажами по ходу игры неминуемо приводит к изменению их действий. Любые взаимодействия отражаются на атрибутах, присвоенных персонажам, а те в свою очередь приводят к изменениям во взглядах на внешний мир и, соответственно к изменению поведения по отношению к другим персонажам.

Действия и перемещения одного персонажа вызывают перемещения других персонажей (и объектов!) в игровом пространстве. Например, в зависимости от смелости одного персонажа он может взять или не взять какой-либо объект на борту вражеской яхты (пищу или, скажем оружие), а это действие может заставить страдать другой персонаж в определенное время (от голода или от безоружности). По этой причине фактор изменчивости является важнейшим при проектировании игры с элементами ИИ (искусственного интеллекта).

Первый этап - задание характеристик персонажам понятен. Его сложность зависит от того, насколько сложной Вы планируете сделать программу. Уже давно в играх сложилась определенная система атрибутов, которая включает в себя:

- силу;
- интеллект;
- ловкость;
- пси-энергию;
- стойкость.

Если с первыми тремя все понятно, то пси-энергия характеризует магические способности героя, а стойкость - способность выдержать определенное количество ударов, не будучи травмированным.

Вы видите, что такая система атрибутов нацелена на игры, в которых сражение - это естественное состояние героев, для чего собственно и идет игра.

Эта система не годится для более интеллектуальной программы с более сложным поведением персонажей, поскольку никаких качеств характера героя она не описывает, а потому не сможет определять его самопроизвольное поведение в игровой среде, что необходимо для программы с искусственным интеллектом.

Желательно, чтобы в программе герои могли бы испытывать чувства, например, такие, как боль, страх, ненависть и другие, неотъемлемо присущие человеку.

Вернемся к нашему проекту программы. Предположим, пока Ком остается в лодке, Вы с Орой ныряете в пучину моря, где во мраке видите очертания затонувшей яхты и силуэты двух чудовищ, охраняющих ее - Крама и Брога. Они сильны, но малоподвижны. В то же время, у них очень неплохо с интеллектом. Настолько неплохо, что они не будут ввязываться в драку или продолжать ее, если у них нет стопроцентной уверенности закончить ее без особого ущерба для себя. Атрибуты, которыми может быть в дальнейшем описано поведение всех персонажей, могут быть такими, как показано на рис.4.

Атрибут	Изменчивость	Диапазон изменения +10/-10
Эмоции	x	стабильные / нестабильные
Энергия	x	энергичный / вялый
Сила	x	большая / малая
Интеллект		гений / тупица
Совесть	x	большая / отсутствует
Образованность		высокая / низкая

Рис. 4

Отмеченные знаком "x" атрибуты могут меняться по ходу игры. Диапазон изменения от "-10" до "+10". Так, например, если Ваша энергичность имеет атрибут "-5", то в очередной ситуации, в которой она будет учитываться, она зачтется не в Вашу пользу. Если по ходу программы Вы будете делать ошибки, то атрибуты эмоций, силы и совести также будут изменяться.

Ключ к использованию этих атрибутов в системе искусственного интеллекта состоит в том, как они взаимосвязаны, как влияют один на другой и как влияют на поведение персонажей. Нет больше однозначно хороших или однозначно плохих героев. Теперь Вы можете покорять врагов, а можете даже делать из них друзей.

Конечно, по ходу игры эти атрибуты могут изменяться сотни раз, но в начале игры они должны быть заданы, например, так, как показано на рис.5.

Атрибут	Инф	Ора	Крам	Брог
Эмоции	5	-2	-3	2
Энергия	4	-1	6	-3
Сила	3	2	4	4
Интеллект	7	2	4	4
Совесть	6	5	-4	-3
Образованность	6	6	1	2

Рис. 5

Определив систему атрибутов, Вы теперь можете выработать критерии поведения своих персонажей, образовав комплексные атрибуты, см. рис.6.

Характеристика персонажа	От чего зависит
Гнев	Энергия, Интеллект, Сила
Мужество	Энергия, Сила, Совесть
Трусость	Энергия, Сила, Совесть
Счастье	Интеллект, Эмоции, Энергия
Настроение	Эмоции, Энергия, Совесть

Рис. 6

Мы позже займемся выведением формул для расчета характеристик персонажей через их атрибуты, но сейчас имейте в виду, что все здесь зависит от Вас. Вы сами можете задать эти формулы. Например:

мужество = энергия + сила + совесть

или скажем:

мужество = 2*энергия + 5*сила + 3*совесть.

Вы же можете вычислить и другие какие-либо характеристики, не указанные выше, например, способность к самопожертвованию.

Скоро Вы увидите, что зная атрибуты персонажей, Вы сможете на глаз предсказать их поведение в той или иной ситуации, а также что произойдет, если например Инф совершит какое-то действие.

Благодаря такому подходу, персонажи могут приобрести некоторую автономию. Они смогут сами "решать", что им делать в той или иной ситуации, а не быть марионетками в руках играющего. Такая игра становится намного интереснее, неожиданнее, да и вообще меняется сама ее природа.

После того, как данные по персонажам определены в исходной таблице (рис.5), можно приступить к программированию. Предельно простой сценарий показан на рис.7.

```
100 LET c$="":LET p$="":LET n$=""
110 REM заполняем банк атрибутов
120 DIM a(4,6):REM - 4 персонажа, 6 атрибутов
130 RESTORE 190
140 FOR i=1 TO 4
150 FOR j=1 TO 6
160 READ a(i,j)
170 NEXT j
180 NEXT i
190 DATA 5,4,3,7,6,6: REM - Инф
200 DATA -2,-1,2,2,5,-6: REM -Ора
210 DATA -3,6,4,4,-4,1: REM - Крам
220 DATA 2,-3,4,4,-3,2: REM - Брог
230 REM - начало экспериментального сценария
240 CLS
250 PRINT "Мрачные очертания проступают"
260 PRINT "перед Инфом и Орой."
270 PRINT "Они видят корпус яхты, на"
260 PRINT "которую указал локатор."
300 REM - ввод противников
310 IF a(3,2) OR a(3,3) OR a(3,5)<-1 THEN LET c$="Брог":
      LET f$="одно чудовище охраняет":
      GO TO 340
320 IF a(4,2) OR a(4,3) OR a(4,5)<-1 THEN LET c$="Крам":
      LET f$="одно чудовище охраняет":
      GO TO 340
330 IF c$="" THEN PRINT "яхта появляется перед Вами":
```

```
      PRINT "дверь в рубку открыта":
      GO TO 380
340 REM
350 PRINT "Корпус яхты прорисовывается":
360 PRINT "в темноте"
370 PRINT f$;"дверь в рубку.":
      GO TO 400
380 PRINT "Вы проходите в помещения"
390 STOP
400 REM - решение стоит ли сражаться
410 IF a(1,2) AND a(1,3) AND a(1,5)<-8 THEN LET p$="Ора":
      LET n$="Инф":
      GO TO 440
420 IF a(2,2) AND a(2,3) AND a(2,5)<-8 THEN let p$="Инф":
      LET n$="Ора":
      GO TO 440
430 LET p$="Инф и Ора": STOP
440 REM - один из них уплывает
450 PRINT n$; "Холодеет при мысли о схватке и"
460 PRINT "уплывает к лодке."
470 PRINT p$; "остается с чудовищем"
480 PRINT "один на один": STOP
```

Рис. 7

Вам должно быть понятно из этого примера как учитывается влияние атрибутов на поведение персонажей. Возьмем чудовищ.

Строка 310. Если у Крама не все в порядке с силой или с энергией или с совестью, он не выползает на поле битвы.

То же самое делает и Брог в строке 320.

Если ни тот, ни другой не охраняют вход, то через строку 330 Вы с Орой проникаете на яхту.

Аналогично и с положительными героями. Их поведение рассматривается в строках 400.

Есть только одна особенность. Чудовища анализируются оператором OR, а ваши герои оператором AND. Это сделано специально, чтобы у первых было больше оснований не вступать с Вами в схватку, ведь если ИЛИ энергии, ИЛИ силы, ИЛИ совести у них мало, они уйдут.

Для вас же наоборот, надо, чтобы ОДНОВРЕМЕННО И того, И другого И третьего не хватало, чтобы бросить товарища в беде. Конечно, это менее вероятно и возможно лишь в совсем отчаянном положении, когда действительно сражаться бесполезно, а надо думать о конечной задаче.

Вам должно быть ясно, что приведенный пример страшно расточителен по расходу памяти и предельно примитивен. Конечно, на БЕЙСИКе реально такие вещи не пишут. Но как иллюстрация концепции он годится и должен быть понятен. Степень сложности, которую Вы определите для себя - дело Ваше. Вы можете анализировать сотни факторов, составлять любые уравнения для расчета критериев - это ваше право.

И напоследок обратим здесь еще внимание на один аспект взаимосвязи. Предположим, что у Брога был кинжал. Что тогда, если он не вышел на битву? Вам стало легче от того, что Вы не пострадали, но зато Вы лишились возможности завладеть кинжалом и впоследствии его использовать. Вы даже никогда и не узнаете, чего лишились - вот что самое интересное. Это тот случай, когда программа начинает жить собственной жизнью, независимо от Вас.

(Продолжение в следующем выпуске)

Развивая начатый разговор об адвентюрных играх, мы сегодня даем две полных инструкции по работе с программами этого жанра. Первая программа - Sherlock - классический представитель этого направления, давно завоевавшая устойчивую популярность. Вторая программа - Miami Vice. Ее особенность в том, что это одна из программ, объединивших в себе адвентюрный жанр с жанром деловой игры (MANAGEMENT) на базе криминального сюжета. В последние годы таких программ стало немало - это The Sidney Affair (INFOGRAMES), Vera Cruz Affair (INFOGRAMES), The Fourth Protocol (CENTURY), и многие многие другие.

SHERLOCK

Фирма MELBOURNE HOUSE.
Год выпуска: 1985.

Игра относится к жанру адвентюрных (текстовых). Действие происходит в Лондоне конца XIX века и его ближайших окрестностях. Вы играете роль Шерлока Холмса и должны раскрыть ряд преступлений. Если Вы погибнете, игра прекратится.

В своих путешествиях Вы встретите многих персонажей, окажетесь в странных ситуациях. Люди и предметы откроются Вам совсем не такими, какими казались на первый взгляд. Ваша работа похожа на работу кукловода - Вы будете контролировать все, что Холмс будет делать или говорить. Свои команды Вы будете отдавать компьютеру на простом "Инглийском" языке (это разновидность английского). Словарь у компьютера большой. Он содержит свыше 800 слов, может выполнять 53 разных действия.

Вы вводите свои указания в виде простых предложений. Верхняя часть экрана служит для изображения места действия и описания ситуации. Графическое изображение появляется, когда Вы входите в новое место, но при повторном входе его уже не будет. Чтобы вызвать рисунок, надо дать команду LOOK (смотреть).

Здесь же описываются все действия, совершаемые Вами или другими персонажами. You take the note (Вы берете записку). Watson examines the pistol (Уотсон изучает пистолет). Нижняя часть экрана служит для связи между Вами и компьютером. Все, что Вы наберете, будет показано здесь заглавными буквами. Здесь же изображаются запросы и подсказки от компьютера, например: I don't understand the word "DOR" (Я не понимаю слово "DOR").

Стирание неправильного символа - CAPS SHIFT + 0.

Стирание всей командной строки - SYMBOL SHIFT + 0.

Правила "инглийского" языка.

1. Каждое предложение должно иметь глагол. Оно может состоять только из глагола.

SIT - сидеть.

CLIMB - взбираться.

Значение глагола может быть изменено добавлением наречия:

FOLLOW QUICKLY - быстро преследовать.

QUICKLY COUNT THE MONEY - быстро пересчитать деньги.

2. Глагол должен стоять первым.

Правильно:

OPEN THE WINDOW CAREFULLY WITH THE STICK - открыть окно с помощью палки.
Неправильно:
WITH THE STICK CAREFULLY OPEN THE WINDOW - палкой осторожно открыть окно.

3. Прилагательные должны стоять перед существительными.
OPEN THE GREEN DOOR - открыть зеленую дверь.

4. Предлоги, такие как WITH (с); UNDER (под); ON (на); OFF (из) и др. обычно должны стоять перед существительными.
OPEN THE DOOR WITH THE KEY - открыть дверь с помощью ключа.
PICK UP THE NOTE - поднять записку.
Иногда они могут стоять и после существительного. Допускаются, например:
TURN THE LIGHT ON - включить свет.
PICK THE NOTE UP - поднять записку.

5. Можно увязывать несколько команд в одну с помощью союза AND (и), например:
PICK UP THE NOTE AND TAKE THE LAMP OUT OF THE HOUSE - поднять записку и вынести лампу из дома.
TAKE THE MONEY AND RUN - взять деньги и убежать.

6. Можно применять запятые и точки, но фраза не должна быть длиннее 128 символов.
TAKE MONEY, RUN - взять деньги, убежать.
PICK UP THE NOTE. TAKE THE LAMP OUT OF THE HOUSE - поднять записку. Вынести лампу из дома.
Всегда имейте в виду, что в ответ на Ваши действия другие персонажи действуют независимо. Если Вы дадите слишком много инструкций в одной команде, то результат может оказаться неожиданным.

Шерлок умеет "говорить". Это позволяет Вам инструктировать другие персонажи. Вы можете, по крайней мере дать ему команду сказать кому-либо, что Вы хотите, чтобы тот сделал. Разумеется, поскольку эти персонажи независимы, то они сами решают делать ли им то, что Вам нужно.
SAY TO WATSON "READ NOTE" - сказать Уотсону: "прочитай записку".
Заметьте, что то, что Шерлок должен сказать, заключено в кавычки. К тому, что он говорит применимы также все правила "инглийского" языка. Старайтесь не говорить очень много одному персонажу за один раз, т.к. ему это может надоесть, и он не захочет Вам помогать.
Другие персонажи, в отличие от компьютера, не сообщат Вам о том, что они не понимают того, что им говорят, а просто сочтут Ваше поведение довольно странным. И если им Ваши действия покажутся бессмысленными, вряд ли они станут помогать Вам.
Если Вы, например, начали свой диалог с Уотсоном так: SAY TO WATSON "HELLO" - сказать Уотсону "привет", - то далее уже не надо повторять команду SAY TO WATSON. Дальнейшая беседа идет просто путем применения кавычек. "READ THE DAILY CHRONICLE" - прочитай "Дэйли Хроникл". Если же после этого Вы захотите обратиться еще к кому-либо, то опять надо будет начать с SAY TO.

Обсуждение.

Холмс не только может давать указания другим персонажам, но может обсуждать обстоятельства дела вместе с Уотсоном и Лестрейдом. При этом он может отвечать на вопросы, которые они ему ставят. То есть могут быть составлены предложения, которые не вызывают немедленных действий со стороны персонажей. Лестрейду и Уотсону можно, например, предложить о чем-то подумать.
Такие заявления в адрес других персонажей не могут начинаться с глагола (чтобы отличаться от команд) и должны быть построены по одной из двух форм. Первая форма

описывает взаимоотношения между субъектом и объектом: BROWN KILLED SMITH - Браун убил Смита. MAJOR FOULKES HAS THE PISTOL - Майор Фоулкс имеет пистолет.

Вторая форма содержит глагол TO BE (быть) для описания какого-либо аспекта. THE GUN IS GREEN - Пистолет зеленый. WATSON IS INNOCENT - Уотсон невиновен.

Очень часто Холмсу удается извлекать необходимую ему информацию из свидетелей. Это можно сделать командой TELL ME ABOUT ... - Расскажите мне о ...

Естественно, в начале перечня должна стоять инструкция SAY TO. Например, SAY TO WATSON "TELL ME ABOUT THE PISTOL" - сказать Уотсону "Расскажите мне о пистолете".

Вы можете расспрашивать кого угодно и о чем угодно - об информации, о людях, о предметах, об их адресах и об алиби.

Обобщение команд.

Вам может быть неудобным давать описание всем предметам, которые Вы хотите, например, поднять. Для этого Вы можете обобщать свою команду, используя слова ALL (все), EVERYTHING (все), EXCEPT (за исключением). Например:

ALL BOTTLES - все бутылки;

EVERYTHING EXCEPT GREEN BOTTLES - все, кроме зеленых бутылок;

EXAMINE EVERYTHING - обследовать все;

OPEN ALL EXCEPT THE GREEN DOOR - открыть все, кроме зеленой двери;

CLOSE ALL DOORS EXCEPT THE GREEN ONE - закрыть все двери, кроме зеленой.

Здесь также можно использовать союз AND (и) для получения более сложных структур: TAKE THE NOTE AND THE LAMP OUT OF THE HOUSE - вынести записку и лампу из дома.

DROP THE SHORT AND THE LONG ROPES - бросить короткую и длинную веревки.

Сокращения.

Вполне возможно быть максимально кратким в диалоге с компьютером без нарушения взаимопонимания. Конечно, чем короче Ваше предложение, тем больше вероятность непонимания.

Рассмотрим игровую ситуацию. На экране горит описание You are in your sitting room. To the west there is your door to the north is a green door. - (Вы находитесь в гостиной. К западу от Вас расположена Ваша дверь, а к северу - зеленая дверь). Предположим, что Вы решили дать команду:

OPEN THE GREEN DOOR - открыть зеленую дверь, но для краткости набрали только слово:

OPEN - открыть. Для Вас ситуация ясна, но компьютер ответит:

OPEN WHAT? - что открыть?

Если теперь Вы дадите команду OPEN ALL - открыть все, то увидите результат своих действий:

You open your door. - Вы открываете Вашу дверь.

You open the plain chest. - Вы открываете шкаф.

Если теперь Вы опять дадите команду OPEN, то получите ответ:

I SEE NOTHING TO OPEN - я не вижу, что можно открыть.

Если бы Вы сразу дали команду

OPEN YOUR DOOR - открыть Вашу дверь, то сразу бы и получили результат: YOUR DOOR IS OPEN - Ваша дверь открыта.

Как видите, за краткость иногда приходится платить.

Описывая объект, Вы можете давать его название и при нем определения (если они есть). Предположим, что Вы видите вкусное пенящееся пиво в бутылке. Тогда можно дать одну из следующих команд:

DRINK BEER - пить пиво;

DRINK DELICIOUS BEER - пить вкусное пиво;

DRINK FOAMING BEER - пить пенящееся пиво;

DRINK DELICIOUS FOAMING BEER - пить вкусное пенящееся пиво.

Во всех случаях Вы утолите жажду. Нельзя использовать местоположение предмета в

качестве его определения:

DRINK BEER IN BOTTLE - пить пиво в бутылке. Это неправильно, т.к. BOTTLE - не прилагательное.

Нельзя также делать что-либо с какой-либо вещью более чем одним образом. Например, можно сказать:

PUT THE ROPE ON THE CHAIR - положите веревку на стул, или:

PUT THE ROPE ON THE TABLE - положите веревку на стол.

Но нельзя сказать: PUT THE ROPE ON THE TABLE AND CHAIR - положите веревку на стол и на стул. И тем более нельзя: PUT THE ROPE ON EVERYTHING - положите веревку на все.

Движение и перемещение.

После того, как Вы приходите на новое место, на экране появляется изображение и игра останавливается на время, чтобы Вы могли рассмотреть экран. Нажатие любой клавиши продолжит игру.

Существует несколько способов для перемещения из одного места в другое.

1. С помощью курсорных клавиш 5,6,7,8 - соответственно: запад, юг, север, восток. При этом эта клавиша должна быть первой в команде. Этим методом нельзя двигаться вверх (UP) или вниз (DOWN), а также например на северо-восток (NORTHEAST).

2. В восьми направлениях, а также вверх и вниз можно двигаться, набирая команду по буквам:

E

EAST - восток

GO EAST - идти на восток

QUICKLY GO EAST - идти на восток быстро.

Перемещение в заданное место.

Если Вы знаете, куда Вам надо пройти, то можете давать команду, используя место назначения:

GO TO LODGINGS - идти к домику.

Если Вам надо переместиться в удаленное место, то Вы можете пользоваться кэбом или поездом, но для этого Вам нужны деньги. Чтобы успеть на поезд, Вы должны знать время его отправления.

Кэбы и поезда.

Кэб можно нанять на большинстве лондонских улиц. Но извозчик ничего не знает, кроме названия улиц. Так, чтобы поехать на вокзал Виктория, надо дать команду: SAY TO CABBIE "GO TO BUCKINGHAM PALACE ROAD" - сказать извозчику "ехать на Бэкингем Палас Роуд".

Чтобы попасть на поезд, Вы должны приехать на нужный вокзал и пройти на нужную платформу. Во время движения в кэбе или на поезде время идет в реальном масштабе. В это время Вы можете обсуждать с Уотсоном обстоятельства дела или отдыхать.

Вход и выход.

Если Вы знаете свое местоположение, например перед входом в строение, можете задавать команду на вход например так:

ENTER LODGINGS - войти в домик

GO INTO LODGINGS - пройти в домик.

Можете проходить через открытые двери:

GO THROUGH THE DOOR - пройти в дверь.

Можно также проходить через окна (WINDOWS) и т.п. Сквозь них можно также смотреть (LOOK). Это может быть очень полезным, если Вы хотите знать, куда попадете, если пройдете в этом направлении. Бывает можно увидеть, кто Вас там ждет!

Преследование других людей.

Если Вы знаете, куда отправился тот, с кем Вы хотите поговорить, то можете отправиться за ним следом, например:

FOLLOW WATSON - идти за Уотсоном.

Хождение по улицам Лондона.

В этой игре Ваши возможности хождения по Лондону ограничены. Это согласуется с характером Холмса и вызвано ограничением памяти компьютера. Он никогда не ходил пешком туда, куда можно доехать в кэбе.

Ход времени.

Течение времени оказывает большое влияние на игру. Даже когда Холмс ничего не делает, а только рассуждает, время идет и другие персонажи действуют.

Игра идет в реальном времени. Оно изображается на экране телевизора и все время персонажи независимо действуют. Взаимосвязь событий во времени - критический фактор в данной игре.

Единственно, когда компьютер не работает - это во время набора очередной команды.

Течение времени может быть ускорено командой WAIT (ждать).

День и ночь.

Поскольку время идет, день сменяется ночью. Если Вы попадете без фонаря в темное место, то ничего не увидите. К счастью, в некоторых местах есть искусственное освещение. В темноте Вы можете потерять ориентиры, ждать в темноте опасно.

Сон.

Холмс, когда глубоко погружается в дело, может подолгу обходиться без сна, но такой энергии нет у окружающих его людей. Им приходится спать. Однако помните, что большинство наиболее темных дел вершится под покровом ночи.

Ускорение времени.

Если Вы дадите команду WAIT (ждать), то некоторое время ничего не будете делать. Будьте осторожны, последствия могут быть непредсказуемы. Команда WAIT имеет два формата:

WAIT - по ней Вы ничего не делаете в течение 5 минут.

WAIT UNTIL ... - ждать до ... - по этой команде Холмс ничего не делает до заданного времени.

Например.

WAIT UNTIL 10 - ждать до 10 часов;

WAIT UNTIL 11 PM - ждать до 11 часов вечера;

WAIT UNTIL 9:30 AM - ждать до 9:30 утра.

Поведение персонажей.

Все персонажи игры живут собственной жизнью и ведут себя относительно независимо. Их поведение зависит как от Ваших действий, так и от действии других персонажей и поэтому в казалось бы равных ситуациях оно может быть различным. Каждая ситуация в игре является уникальной и однозначного решения игры не может быть. Повторяя игру снова и снова, Вы будете сталкиваться с новыми ситуациями. Играть в нее можно не один раз. Всякий раз это будет новая игра.

Сотрудничество с другими персонажами.

Многие проблемы игры могут быть разрешены только если Вы проявите готовность к сотрудничеству с другими героями. Столько много дел надо сделать Холмсу, что без чужой помощи не обойтись. Хотя он и работает независимо от полиции, тем не менее ему нужна ее помощь для отыскания дополнительных улик и получения информации.

Деньги.

Без них Вы далеко не уйдете. Вам надо покупать билеты на поезд, оплачивать информацию и другие услуги. Фунты вводятся в следующем формате:

.. 99 - 99 фунтов

Для фунтов и шиллингов:

.. 99 19S. или .. 99 19/-

Только для шиллингов:

10/- или 10S.

Для шиллингов и пенсов:

13/11

Только для пенсов:9d.

Команда на уплату денег:

PAY или PAY TO:

PAY TO WATSON .. 1 - уплатить Уотсону один фунт.

PAY 6D TO THE CABBIE - уплатить 6 пенсов извозчику.

Стратегия игры.

Мы Вам советуем во время игры создать собственную карту, чтобы Вы могли легко возвращаться назад.

Когда Вы впервые приходите на какое-то место, компьютер дает Вам его подробное описание и список возможных выходов. При повторном посещении описание очень краткое. Если Вам нужны подробности, надо дать команду LOOK.

В игре имеются и определенные физические законы.

1. Вы не можете поднять предмет, который превышает Ваши силы, а также если Вы уже несете тяжелый груз. Это же относится и к остальным персонажам игры, но если они крепче Вас, то и поднять смогут больше.

2. Вам не обязательно владеть предметом (нести его), которым Вы собираетесь как-то воспользоваться. Если камень лежит на земле, Вы можете давать команду:

THROW STONE AT WINDOW - бросить камень в окно.

Исключение составляют предметы, которыми уже владеют кто-нибудь из других персонажей игры.

3. Некоторые предметы могут играть роль емкостей или контейнеров. Это чемоданы, бочки и т.п.

Нельзя положить в контейнер предмет, если он слишком велик для этого. Очевидно нельзя этого сделать и если емкость закрыта.

4. Некоторые емкости могут быть прозрачными, а некоторые нет. Вы можете увидеть, что находится внутри прозрачного контейнера, а для непрозрачного надо его сначала открыть. Для этого может потребоваться ключ.

5. Некоторые предметы могут ломаться. Вы должны быть осторожны в обращении с ними. Например, если Вы дадите команду сломать дверь с помощью бутылки, результат может быть совсем противоположным - разобьется бутылка.

Повторение команд.

Нажатие @ приводит к повторению ранее поданной команды.

Изучение текущей ситуации.

1. LOOK. Можно давать сокращенно "L".

2. LOOK THROUGH объект. По этой команде дается изображение и описание того, что можно увидеть через открытую дверь или, скажем, окно.

3. INVENTORY. По этой команде выдается описание всего того, что Вы имеете в своем распоряжении. Сокращение - "I".

4. EXAMINE объект. Эта команда позволяет более детально рассмотреть любой предмет, находящийся в пределах вашей досягаемости.

Прерывание игры и выгрузка позиции.

1. PAUSE - игра приостанавливается до тех пор, пока не будет нажата какая-либо клавиша.

2. SAVE - по этой команде производится выгрузка незаконченной игры на ленту. После выгрузки игра продолжается как обычно.

3. LOAD - загрузка ранее отложенной позиции.

4. QUIT - прекращение текущей игры.

5. PRINT - по этой команде включается принтер, если он подсоединен.

6. NOPRINT - эта команда отбивает команду PRINT.

MIAMI VICE

Фирма: "OCEAN"
Год выпуска: 1987

Цель игры - сорвать крупную сделку по переправке контрабанды.

Кроккет и Таббс получили сведения о том, что утром в четверг готовится доставка в город крупной партии контрабанды стоимостью миллион долларов. Получатель - некто мистер "J" - гангстер со стажем, ныне занимающийся респектабельной деятельностью и имеющий влиятельных покровителей.

Известно только, что сделка состоится в одном из портовых заведений и что произойдет она между полуночью воскресенья и утром четверга.

Ваша задача - выехать на место и внедриться в сеть, начав с мелких агентов, обычных завсегдатаев портовых баров. Если вести себя с ними правильно, то через них можно выйти на поставщиков. Еще одно звено в цепи - содержатели различных казино. Если их сразу не пристрелить, то через них можно получить много различной информации.

Управление

1. Интерфейс-2 (Синклер-джойстик)
2. Кемпстон-джойстик
3. Курсор-джойстик
4. Клавиатура

Управление клавиатурой

X – влево C – вправо

M – вниз L - вверх

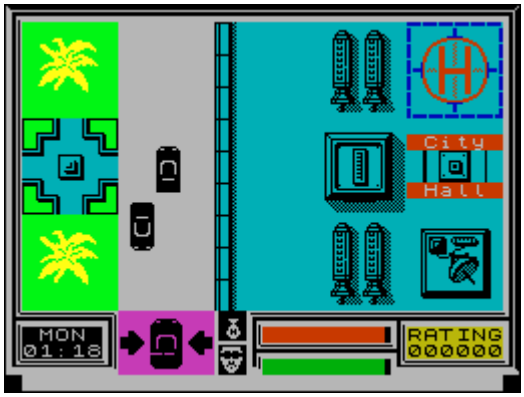
от Y до P – огонь

CAPS LOCK - остановка игры, запуск - повторным нажатием.

E - выход из игры (работает только после паузы).

Стрельба из окна машины.

Сначала нужно перейти в режим стрельбы. Это выполняется кнопкой "Огонь". При этом высвечивается автомобиль на дисплее. Чтобы выстрелить в указанном направлении нажмите "Огонь" еще раз.



Чтобы изменить направление стрельбы, пользуйтесь соответствующими клавишами.

Примечание: Пассажир не может стрелять из машины влево.

Вход в здание.

Остановите машину на дороге. Нажмите и отпустите "Огонь", теперь нажмите "Влево". Дисплей автомобиля изменится. На нем изображается символ автомобиля с двумя мигающими стрелками (справа и слева). Если в машине есть вещественные доказательства, то справа будет мигать стилизованное изображение мешка, а если в машине задержанный, то изображение человека. Чтобы выйти из машины и войти в помещение, нажмите "Огонь" в тот момент, когда стрелка на дисплее есть.

Чтобы не входить, нажмите "Огонь", когда стрелки на экране нет.

Примечание: Если в машине есть улики или задержанный и Вы выйдете из машины и войдете в здание, то они будут утрачены.

Управление внутри зданий.

Вверх - проход через дверь.

Вниз - режим "Выбор".

Влево, вправо - соответственно.

"Огонь" - стрельба в текущем направлении.

Улики (сумки с контрабандой) собираются простым перемещением на них.

При встрече с агентом, его имя загорается на экране сообщений. Чтобы задержать, его надо догнать. Иногда это удается сделать только после перестрелки (если он останется жив). Если же он сдается добровольно, то автоматически включается режим "Выбор" для заключения с ним сделки. Выбирайте тот или иной вариант из предлагаемых нажатием клавиши "Огонь".

Если допрос прошел успешно, Вы должны узнать имя или адрес явки, или время, или сумму сделки.

Чтобы получить очки, надо доставить арестованного гангстера в городское управление полиции (CITY HALL).

Специальные службы.

Они не имеют экранов, показывающих их внутреннее устройство, но компьютер издает характерный звук при посещении этих зданий.

Полицейское управление (CITY HALL).

Сюда доставляются арестованные и улики. Если машина с ними попадает в аварию, то они теряются.

Вы не получите очков за преступника, который откажется говорить, но получите за имевшееся при нем вещественное доказательство. Кроме того, после допроса в полицейском управлении Вам передадут его показания (примерно через 3 часа после доставки).

Госпиталь (HOSPITAL) - его можно посещать один раз в день для восстановления сил.

Расписание.

Ваш шеф снабдил Вас расписанием возможных появлений членов банды в местах явок, но оно действует только на понедельник.

Имеются 2 возможных варианта на первую половину дня и четыре варианта на вторую. Все встречи в течение 12 часов идут по одному расписанию, хотя заранее неизвестно, по какому именно.

Первая половина суток (A.M.)

Кто	Где	Когда	
		или	или
BLADES	SAMS BAR	2:00	9:20
PADDY	ISLAND BAR	3:20	8:00
MAC	FATS BAR	4:40	6:40
RONNIE	DIXIE BAR	6:00	5:20
BONZO	LAMP BAR	7:20	4:00
CHICO	SURFER BAR	8:40	2:40

Вторая половина суток (P.M.)

Кто	Где	Когда			
		или	или	или	или
DINO	JOES BAR	0:20	9:40	6:00	5:20
SHARKY	VINES BAR	1:20	8:40	7:00	4:20
TOOTS	SAMS BAR	2:20	7:40	8:00	3:20
HUGGY	ISLAND BAR	3:20	6:40	9:00	2:20
DUKE	FATS BAR	4:20	5:40	10:00	1:20
FRANKIE	DIXIE BAR	5:20	4:40	1:00	10:20
EDDIE	LAMP BAR	6:20	3:40	2:00	9:20
HAWKER	SURFER BAR	7:20	2:40	3:00	8:20
REEMO	JOES BAR	8:20	1:40	4:00	7:20
SNOWY	VINES BAR	9:20	0:40	5:00	6:20

Время

1 секунда реального времени = 1 минута игры.

Если Вы приедете на явку до того, как там появились члены банды, то они засекут Вашу машину у входа и успеют скрыться. Встреча будет сорвана.

Если Вы прибудете в течение 4-х минут после начала встречи, то они сумеют разбежаться, но оставят вещественные доказательства.

Если Вы прибудете от 4-х до 6-ми минут после назначенного времени, то сумеете захватить гангстера вместе с уликами.

Если Вы приедете от 8-ми до 12-ти минут после начала сбора, они успеют разойтись, но если Вы поспешите, то успеете "сесть на хвост" отъезжавшей с преступниками машины (она красного цвета).

После 12-ти минут будет слишком поздно.

Если Вы не сумеете схватить м-ра "J", то игра начнется сначала, с полуночи следующего воскресенья и до четверга. Все встречи членов банды будут идти по тому же расписанию, кроме тех, кто был арестован или убит.

Очки.

Игра заканчивается автоматически при задержании м-ра "J" или гибели Кроккета или Таббса.

Очки начисляются за:

- доставку улик в полицейское управление;
- доставку гангстеров, если они заговорили (но если это была не ложь);
- уничтожение автомобилей (красного цвета) с пытающимися скрыться преступниками. Очки снимаются за:
- аварию автомобиля;
- уничтожение автомобилей обычных граждан;
- за каждую рану Кроккета и Таббса;
- за неудачную попытку подкупа;
- за несостоявшийся арест, если Вы объявили преступнику, что он арестован.

Если Вы не задержите м-ра "J" к полудню четверга, то потеряете очки.

* * *

Запишу или обменяю программы к "ZX-Spectrum". В коллекции более 500 программ. Цены - низкие. Каталоги выгружу на ленту.

341048, УССР, Мариуполь - 48, а/я 48010, Александру.

Обмен и запись программ к Spectrum. Вышлю тематический каталог. К запросу приложить конверт с обратным адресом. 630102, Новосибирск, ул. Богаткова, д.50, кв.29. Ескевич Алекс Александрович.

Предлагаю программы и литературу по "Спектруму". Возможен обмен. В коллекции 500 программ. Каталог высылается бесплатно. Пишу программы на заказ для организаций и частных лиц.

Обращаться по адресу: 301650, Тульская обл., г. Узловая, ул. Завенягина, д.26, кв.43, Панарьину Артуру Васильевичу

* * *

Владельцам ПК-01 "ЛЬВІВ"

Если Вы хотите знать все секреты своего компьютера и быстро научиться программировать, приобретайте кассету с записью системных и игровых программ.

Вы найдете практические советы по программированию и защите программ.

Содержание:

- все о ПК " ЛЬВІВ";
- универсальный отладчик;
- АССЕМБЛЕР;
- ДИСАССЕМБЛЕР;
- редактор текстов;
- графический редактор;
- монитор;
- копировщик программ;
- BS монитор;
- ОЗУ-V2;
- RENUM (расширение БЕЙСИКа);
- шахматы;
- шашки;
- и другие системные и игровые программы.

Всего 170 рублей, и Вы будете обладать большими возможностями.

Оплату за кассету выполнять почтовым переводом по адресу:

41651С, г. Ахтубинск-1,ул. Волгоградская, д. 14, кв. 12.

Мызину Вячеславу Николаевичу.

STRATEGIC GAMES

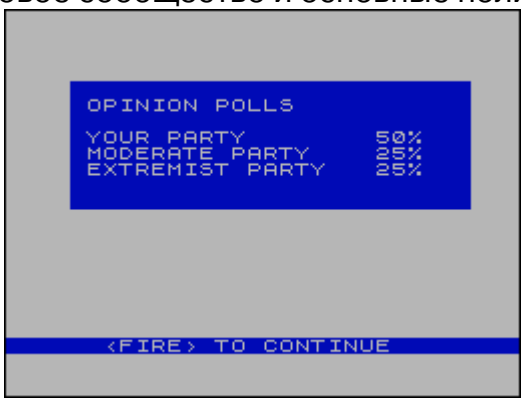
PRESIDENT

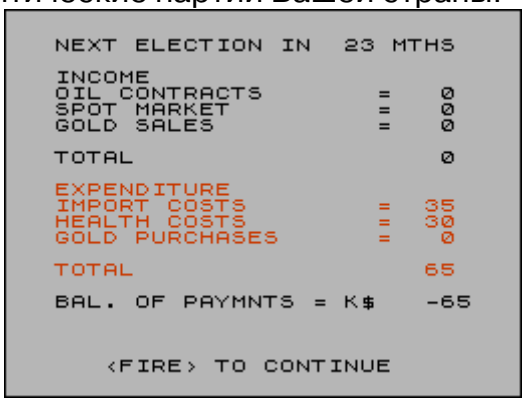
ADDICTIVE 1986

Автор игры - Кевин Томс. Ему же принадлежит и ставшая очень популярной менеджерская игра FOOTBALL MANAGER.

Такая игра, как PRESIDENT - единственное средство проверить Ваши способности по управлению современным государством в одиночку. Вам предстоит принимать все ответственные решения: как распределить продукты питания, где производить поиск нефти, как и когда наносить военные удары по соседним государствам.

Игра как бы разделена на множество разделов, в каждом из которых Вы принимаете решение посредством выбора из альтернативных вариантов. О том, какой эффект имеет Ваше решение, Вы узнаете в конце месяца, когда Вам сообщат как относятся к Вашему правлению мировое сообщество и основные политические партии Вашей страны.



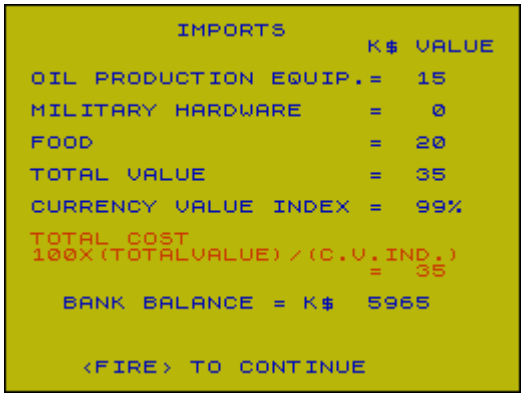


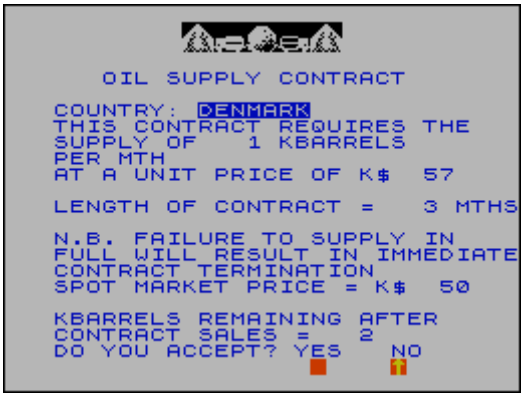
Каждые два года (через двадцать четыре месяца) проводятся новые президентские выборы. Конечно, ловкий игрок будет принимать самые крутые и непопулярные решения сразу после очередных выборов, а в оставшийся срок правления станет завоевывать голоса и доверие избирателей специально для этого рассчитанными взвешенными решениями.

В управление Вам досталась довольно типичная средиземноморская страна, в процветании которой решающую роль играет нефть. На экране изображена карта Вашей страны с нанесенными на нее символами танковых дивизий, нефтеперерабатывающих заводов, нефтехранилищ, посевов зерновых и т.п.

Там же показаны транспортные пути. Вы можете искать нефть и ставить нефтяные вышки, прокладывать дороги и регулировать транспортировку продуктов и товаров.

Время от времени беспокойные соседи будут наносить по Вам воздушные удары, имея в виду в первую очередь поражение нефтяных полей. Интенсивность этих налетов можно снизить тщательно продуманной стратегией размещения зенитных батарей. Могут вражеские силы проводить и наземные атаки, в этом случае Вам предстоит управление своими танковыми частями.





У Вас есть масса возможностей проиграть. При слабой армии вражеские самолеты разнесут заводы и погубят урожай. При чрезмерных затратах на армию население начнет погибать от малярии и туберкулеза, а укреплять армию можно, например, импортом товаров военного назначения.

Это весьма сложная игра. Графика, правда, не потрясает, но вопрос, насколько графика нужна стратегическим играм - достаточно спорный, а в отношении стратегического менеджмента эта игра не может не вызывать восхищения.

FALKLANDS 82

PSS 1986

Сюжет игры относится к Англо-Аргентинскому кризису 1982 года. Эти события еще совсем свежи в памяти большинства наших читателей, и мы останавливаться на них не будем. Скажем только, что операция, начиная с высадки в Сан-Карлосе и до захвата Порта Стенли заняла 25 дней. Аналогично и Вам предстоит повторить это достижение за 25-30 игровых ходов (в зависимости от избранного уровня сложности).

Экран воспроизводит карту северо-восточной части Фолклендских островов. Ваши боевые части показаны в виде элементарных блоков. Достаточно простыми графическими средствами показаны горы и поселки. Аргентинские части на карте не показаны - они скрыты, и Вы их увидите только когда столкнетесь.

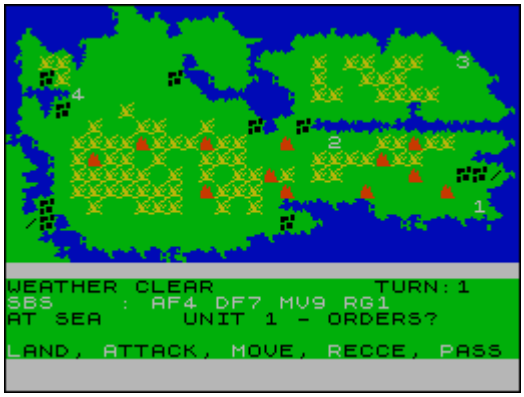
Правила игры очень несложны. Каждая часть имеет 4 показателя:

- фактор атаки (AF);
- фактор защиты (DF);
- скорость перемещения (MV),
- дальность действия (RG).

Фактор атаки характеризует силу боевой части.

Фактор защиты - примерно то же, но на него оказывает влияние рельеф местности, впрочем он же влияет и на скорость перемещения.

Дальность действия для артиллерии - 6 полей, для танков и моточастей – 2, для пехоты - 1.



Вы можете использовать поддержку своей авиации морского базирования, а также огневую поддержку корабельных орудий. В самом начале игры Вы должны назначать сколько-то кораблей для этой задачи, но если назначите слишком много, а аргентинцы атакуют головной флот, то самолеты не поддержат наземные операции, т.к. будут отражать морские атаки.

Поддержка с моря и с воздуха зависит от погоды, а погода с каждый днем становится все хуже и хуже. Поэтому постарайтесь разгромить все, что можно, как можно раньше. Вызов поддержки выполняется вместо хода Вашего полка.

Поскольку на каждом квадрате игрового поля не может быть более одного полка, возможны пробки. Надо аккуратно планировать атаку.

Цель игры не состоит в разгроме аргентинских частей. Вам надо захватить как можно больше населенных пунктов. Если потеряете время в бесконечных стычках, то можете не успеть выполнить главную задачу.

Игра идеальна для начинающих своей простотой. К недостаткам можно отнести не очень хорошую сбалансированность. На высоком уровне сложности аргентинцы сильны настолько, что это решительно не соответствует реальности. На самом же низком уровне играть так легко, что неинтересно.

AUSTERLITZ

LOTHLORIEN1986.

Игра разработана для версии компьютера с оперативной памятью 128K и относится к одной из лучших стратегических игр этой фирмы.

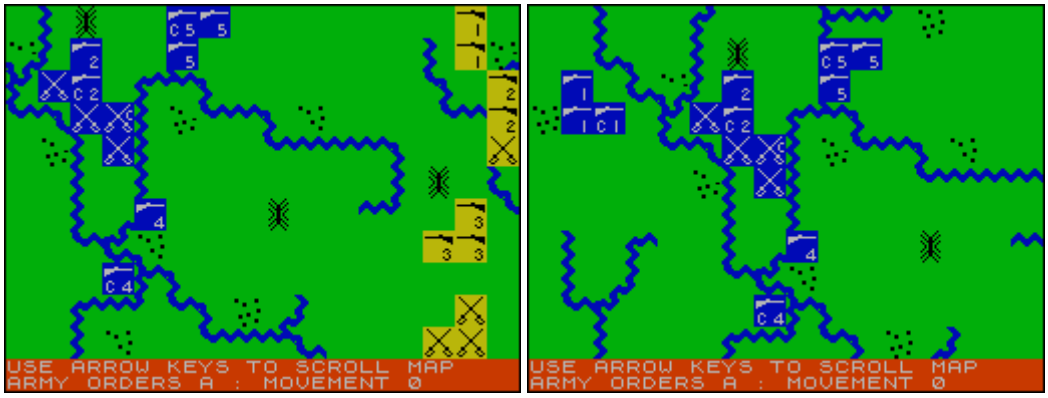
В основу игры положена битва при Аустерлице, в которой Наполеон разгромил значительно превосходящие его объединенные войска Австрии и союзной с ней России. Сражение произошло в 1805 году.

Успех был достигнут тремя факторами. Во-первых, прекрасная обученность войск и, соответственно, высокий моральный фактор.

Во-вторых, стратегический гений Наполеона, талантливое манипулирование войсками.

И, наконец, в третьих, более опытные генералы, по сравнению с войсками союзников. Тактические решения по ходу сражения принимались на месте, оперативно и безошибочно.

Ваша задача - повторить успех Наполеона как в стратегическом, так и в тактическом плане. На тот случай, что вам это удастся, в игре предусмотрены повышенные уровни сложности.



В игре есть две характерные черты, отличающие ее от прочих игр этого жанра.

Во-первых, предполагается, что командиры Ваших частей хорошо подготовлены и во многих случаях решения по отдельным деталям боевых ситуаций принимается и без Вас. Можете даже полностью передать им управление войсками.

Во-вторых, для максимального приближения к реальности существуют скрытые перемещения войск противника. Так, по мере перемещения частей Вы можете терять их из виду и вновь обнаруживать только в результате прямого столкновения.

Механика игры несложна и соответствует большинству игр этого жанра. На результативность боевых ударов оказывают большое влияние фактор местности и фактор морального состояния войска. Местность, во-первых, определяет на какое расстояние то или иное подразделение может быть передвинуто за один ход и, во-вторых, влияет на силу подразделения, находящегося в обороне. Понятно, что горный хребет оборонять проще, чем заболоченную долину.

Моральный фактор учитывается так:

Боевая мощь частей рассчитывается из условия 500 человек - одна единица силы. Тогда часть в составе 4500 человек имеет 9 единиц. При максимальном моральном факторе к ее силе прибавляется еще 5 единиц, что эквивалентно увеличению численности как бы на 3000 человек.

"Аустерлиц" - великолепная игра. Возможно, Наполеон не одержал бы той блистательной победы, если бы его противники промоделировали сражение на "Спектруме".

Техника управления игрой чрезвычайно похожа за технику игры "OVERLORD" той же фирмы, в которой Вы решаете задачу высадки союзных войск в Франции в 1944 году и которую мы очень подробно разберем в следующем выпуске "ZX-РЕВЮ".

* * *

Тем, кто хочет попробовать свои силы в адвентюрных программах!

По просьбам желающих приобщиться к работе с программами этого жанра мы подготовили два сборника программ. Не можем правда сказать, чтобы эти просьбы были многочисленными, но зато они соответствуют нашему основному направлению на расширение интереса к "умным" играм.

Среди программ, представленных в этих сборниках, есть и классические образцы, есть и шедевры, есть и несложные программы, удобные для начинающих.

Считайте, что это как бы маленькая ретроспектива жанра, конечно, здесь не охвачена и десятая часть того, что было разработано, но здесь охвачены основные направления, представлены ведущие фирмы, а если окажется, что это кому-то интересно, то мы ведь можем и продолжить такие сборники.

Если вам удастся пройти какую-нибудь игру от начала и до конца, черкните нам об этом записку, мы будем использовать Ваш опыт для помощи тем, кто "застрянет" по дороге, эти игры проходятся только коллективно.

Цена сборников и порядок оплаты Вам известны, поскольку каждый зарегистрированный получатель "ZX-РЕВЮ" получает вместе с ним ежемесячно и информационный листок "Инфорком предлагает..."

ADVENTURE-1

1. ZZZZ	MASTERTRONIC	1987
2. Robin of Sherlock-1	SILVERSOFT	1985
3. Robin of Sherlock-2	SILVERSOFT	1985
4. Robin of Sherlock-3	SILVERSOFT	1985
5. Red Moon	LEVEL 9	1985
6. Heavy on the Magic	GARGOYLE	1986
7. Bored of the Rings	DELTA 4	1985
8. Sherlock	MELBOURNE HOUSE	1984
9. Gremlins	ADVENTURE INT.	1983
10. Slane King	MARTECH	1987
11. Rebel Planet	U.S.GOLD	1986
12. He Man	ADVENTURE SOFT	1986
13. Witch's Cauldron	MIKROGEN	1987
14. Planet of Death	ARTIC	1982
15. Inca Curse	ARTIC	1982
16. Ship of Doom	ARTIC	1982
17. Espionage Island	ARTIC	1982
18. Neverending Story 1	OCEAN	1985
19. Neverending Story 2	OCEAN	1985
20. Neverending Story 3	OCEAN	1985
21. Aftershock	INTERCEPTOR	1986
22. Wulfan	MASTERTRONIC	1988

ADVENTURE-2

1. Hunchback 1	OCEAN	1986
2. Hunchback 2	OCEAN	1986
3. Hunchback 3	OCEAN	1986
4. Red Hawk	MELBOURNE HOUSE	1986
5. Sorderon's Shadow	BEYOND	1985
6. Stormbringer	MASTERTRONIC	1987
7. Vera Cruz 1	INFOGRAMES	1985
6. Vera Cruz 2	INFOGRAMES	1985
9. Sidney Affair	INFOGRAMES	1987
10. Master of Magic	MASTERTRONIC	1987
11. Hampstead	MELBOURNE HOUSE	1986
12. Spellbound	BEYOND	1984
13. Doomdark's Revenge	BEYOND	1985
14. Worm in Paradise - 1 (с графикой)	LEVEL 9	1985
15. Worm in Paradise - 2 (то же, но без графики, с расширенным текстом)	LEVEL 9	1985
16. Heartland	ODIN	1987
17. Sam Cruise	MICROSPHERE	1987

ЦЕНА ИДЕИ

Многие из читателей ZX-РЕВЮ в свободное от основной работы время работают над какими-то разработками, связанными со своим компьютером. Рано или поздно наступит момент, когда они предложат свою работу потребительскому рынку, а может быть коммерческая организация обратится к ним с заказом на исполнение работы, проведение исследования и т. п.

Как себя повести в этой ситуации, как установить цену на свою разработку, как получить оплату - сразу или по частям?

Этим вопросам мы посвящаем статью "Цена идеи". Страна входит в рынок. Мало кто знает, что это такое, и что будет с обычными товарами в рыночной экономике, но совсем никто не знает что будет с ценообразованием на идеи, не имеющие аналогов, на научные разработки, которые нельзя взвесить и пощупать.

Сразу оговоримся, что материал, который мы здесь даем, уникальный. Еще не пришло время его проверить практикой, и потому мы не претендуем ни на какую методичность, а просто постараемся простыми словами изложить те проблемы, которые надо решать и наметим основные принципы их решения.

Как бы мы ни старались упростить изложение, все равно без некоторых экономических терминов нам не обойтись.

Итак, предположим, Вы стали автором идеи (научной, или программной, или методической разработки, а может быть и технической, например Вы разработали принципиальную схему и описание периферийного устройства).

Товар Ваша идея или не товар?

Еще недавно она не была товаром, а если и товаром, то не Вашим. Она принадлежала той организации, в которой Вы родили эту идею, а еще точнее - никому, поскольку все прекрасно знают о муках изобретателей и рационализаторов. Они зачем-то что-то изобретают, чтобы отрывать от дела важных людей и портить им кровь своими просьбами. А уж о программных или о методических разработках мы и не говорим - раз нет изделия в металле, значит и нет предмета изобретения, а значит и вообще ничего нет.

Никто не пробовал запатентовать методику (мы имеем в виду у нас, а не за рубежом)?

Сейчас можно определенно сказать, что Ваши идеи тот же товар, и он может быть продан и оплачен, но как?

Если бы Вы производили обувь, все было бы просто. Взяли стоимость материалов, прибавили трудозатраты, налоги, накладные расходы, ожидаемую прибыль - вот Вам и цена. Это расчет от себестоимости, здесь он совершенно неприменим.

Вопросы продажи идеи надо решать с других позиций. Продажа идеи - это своего рода продажа лицензии и если Вы сделали программу и предлагаете ее фирме, которая берется ее распространять, то Вы становитесь ЛИЦЕНЗИАРом, а фирма, приобретающая у Вас право на распространение - ЛИЦЕНЗИАТом и вот на особенностях их взаимоотношений, мы и остановимся. Суть не меняется, если Вы сами распространяете свою программу. Все равно - всякий купивший ее у Вас покупает тем самым лицензию на ее применение. А цена на лицензию определяется никак не от себестоимости, возьмем для примера "Кока-Колу". Изобретена почти сто лет назад, затраты на ее изобретение окупились уже миллион раз - а попробуйте купить лицензию, что это будет стоить?

Основной вопрос, который надо решить при определении цены, заключается не в том, сколько получит лицензиар за свою продукцию, а в том насколько выгодно ее применение лицензиату.

Иными словами, проблема состоит не в том, сколько стоит идея, а в том, сколько за нее готовы заплатить. (Иногда к оценке стоимости идеи пытаются подойти с точки зрения затрат лицензиара на ее разработку или возможных затрат лицензиата на создание альтернативной идеи. Такой подход нельзя признать продуктивным. Заинтересованность лицензиата в покупке идеи проистекает не из затрат лицензиара, а из выгоды, которую эта сделка принесет ему). Поэтому определение размера компенсации наиболее

целесообразно производить исходя из доли лицензиата в прибыли лицензиара, т.е. оплата лицензии должна производиться исходя из предполагаемого конечного результата ее использования.

Ценообразование в этой области осложняется еще и тем, что идеи могут еще не иметь адекватного материального выражения. Практическое использование их, как правило, требует финансовых вложений. Кроме того, будущая рентабельность неизвестна, да и вообще проблематична. Как ни однозначна ситуация с использованием идей, рыночная ситуация еще более непонятна, спрос на новый продукт может быть загадкой, размеры рынка неизвестны. Тем не менее, несмотря на данные трудности, продавец и покупатель должны достичь соглашения, установив приемлемую для обоих цену.

Итак, коль скоро мы установили, что размер компенсации за эксплуатацию идеи (разработки) должен зависеть от прибыли покупателя, то разумно и стоимость приобретения лицензии (разработки) определять исходя из доли лицензиара в будущей прибыли лицензиата и выражать ее в виде процента от прибыли.

Этот процент от прибыли называется СТАВКОЙ РОЯЛТИ. Обычно этот процент составляет от 0,5 до 15% реальной прибыли, а от чего он зависит мы рассмотрим немного позже.

Конечно получать оплату за свою разработку в виде процента от прибыли не очень удобно. Прежде всего Вы рискуете нарваться на организацию, которая не приложит сил к продвижению Вашей идеи и Вы ничего не получите, зато они, затормозив Вашу разработку, приложат максимум усилий к продвижению своих, устранив тем самым опасность, что Вы продадите свою идею ближайшим конкурентам.

Рискуете Вы также тем, что Ваша разработка как бы гениальна она не была, может просто не пойти на рынке, хотя бы потому, что рынок не готов к ее приему. Тогда лицензиар потерпит убытки, а Вы не получите за свою идею ни копейки.

Гораздо удобнее разработчику получить компенсацию за идею сразу и целиком, не дожидаясь прибыли. Конечно, в этом случае общий размер суммы должен быть существенно ниже, хотя бы за исключение риска не получить ничего, но и не только за это. Об этом мы тоже поговорим позже.

Такая единовременная выплата при приобретении вашей идеи называется ПАУШАЛЬНОЙ СУММОЙ.

Конечно, договор о продаже может быть комбинированным и предполагать некоторую единовременную ПАУШАЛЬНУЮ сумму и последующие отчисления роялти.

Дело не в том, в какой форме будет выплачиваться вознаграждение, суть в том, что при любой форме единственный способ прикинуть каким-то образом размер этой ПАУШАЛЬНОЙ СУММЫ (или ее первого взноса) можно тоже только на основе ставки РОЯЛТИ и расчеты и в том и в другом случае проводятся от определения этой ставки.

Применение в договоре и той и другой формы оплаты имеют свои достоинства и недостатки. Перечислим их хотя бы кратко, Вы конечно понимаете, что достоинства с точки зрения лицензиара могут быть недостатками с точки зрения лицензиата и наоборот. Чтобы как-то согласовать их интересы мы будем считать достоинствами те факторы, которые способствуют справедливому заключению договора, а недостатками те, которые ему препятствуют. Тем самым мы приводим противоположные интересы сторон к общему для них интересу - заключению справедливого контракта.

Преимущества оплаты лицензии паушальной суммой (а также оплаты в течение непродолжительного времени) следующие:

- цена разработки известна заранее;
- предотвращается инспектирование лицензиаром коммерческих операций лицензиата (а ведь они могут содержать коммерческую тайну);
- непредвиденное увеличение цен не повысит размер выплат;
- цены различных идей становятся более сопоставимыми, т.к. отпадает необходимость оценки возможного объема и времени производства и продаж;
- паушальная сумма, как правило, бывает ниже приведенных роялти, т.к. все бремя

риска берет на себя лицензиат.

Но есть и недостатки:

- оплата производится до получения результата;
- лицензиар оказывается не заинтересованным в успехе лицензиата, т.к. уже получил всю причитающуюся ему сумму заранее; лицензиар может продать аналогичную лицензию конкуренту по другой цене, что приведет к снижению прибыли лицензиата, а сама мысль об этом может сорвать заключение договора.

Соответственно, преимущества текущей ставки роялти заключаются в следующем:

- бремя риска распределяется между лицензиаром и лицензиатом;
- снижая ставку роялти, лицензиар может помочь лицензиату захватить большую часть рынка;
- если лицензиар не выполняет каких-то обязательств, выплата роялти может быть приостановлена. Прекращение платежа также наступит при ликвидации лицензиатом контракта;
- текущие ставки облегчают финансово-кредитное положение лицензиата;
- возможна дифференциация ставок роялти в зависимости от места (продажа в своем регионе или в соседнем) и времени продаж;
- текущая ставка в любой момент может быть с согласия лицензиара преобразована в паушальную сумму;
- ставки роялти могут быть изменены при обнаружении недостатков в приобретенной разработке.

Недостатки текущих ставок следующие:

- повышение цен в результате региональной инфляции или регулирующей политики властей приводит к увеличению выплат лицензиару без каких-либо усилий с его стороны;
- доход лицензиара за период контракта невозможно определить с необходимой точностью.

Установление приемлемой цены.

На этот вопрос нет легкого ответа. Соглашение должно обеспечить адекватную награду лицензиару за его творческий потенциал, усилия и затраты. Но цена не должна быть настолько высокой, чтобы не быть не выгодной, не угрожать получению приемлемой прибыли лицензиатом.

Разработчики новых технологий, авторы идей часто считают возможным запрашивать за них довольно высокие цены, т.к. они считают, что если бы не их усилия, то этих идей или технологий, а следовательно, и возможностей их применения не было бы вообще.

Лицензиаты, наоборот, считают, что т.к. на них по сравнению с изобретателем (автором идеи) ложится основное финансовое бремя, т.е. и риск неокупаемости вложенных средств, то они имеют право на большую долю дохода. Обе позиции обоснованы, но могут быть причиной одинаковых ошибок, являющихся результатом переоценки своего положения.

Необходимость, а не жадность является основной силой, направляющей заключение сделки, однако этот принцип, хотя и является важным, еще не приводит к установлению приемлемой для обеих сторон цены. Необходим учет многих других факторов.

Факторы определения цен.

Стоимость новой идеи, научной разработки в наиболее полной степени определяется предполагаемым доходом, который может получить владелец в течение их жизненного цикла. Этот доход в свою очередь определяется рыночным спросом на данную разработку и уровнем роялти, уплачиваемой лицензиару в течение всего срока лицензионного соглашения. Поэтому вопрос об уровне роялти является очень важным. Обычно ставки роялти составляют от 0.5 до 15 процентов от прибыли с продажи данной разработки. Оценить ставку роялти в этих пределах можно исходя из девяти перечисленных ниже

факторов, в той или иной мере присутствующих во всех сделках по продаже научных разработок.

Каждый из этих факторов влияет на коммерческую привлекательность разработки для потенциальных лицензиатов и поэтому должен оказывать влияние на ставку роялти. В каждом факторе мы постарались учесть только одну из значимых переменных, характеризующих коммерческую выгодность идеи.

В следующем выпуске мы рассмотрим эти факторы, приведем конкретные практические примеры из нашей жизни, разберем возможные расчетные формулы и дадим ряд практических советов для успешного продвижения Ваших идей и разработок.

А пока желаем Вам успешной работы над Вашими проектами.

ИНФОРКОМ

(Окончание в следующем выпуске)

Содержание