**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра Систем автоматизированного проектирования**

отчет

**по лабораторной работе №4**

**по дисциплине «Организация ЭВМ и систем»**

Тема: **Клавиатура IBM PC. Использование прерываний**

Вариант: 4

|  |  |  |
| --- | --- | --- |
| Студент гр. 1302 |  | Рождественский К.И. |
| Студент гр. 1302 |  | Новиков Г.В. |
| Студент гр. 1302 |  | Романова О.В. |
| Студент гр. 1302 |  | Безруков П.М. |
| Преподаватель |  | Костичев С.В. |

Санкт-Петербург

2022**Сведения о подсистеме ввода информации с клавиатуры, используемых прерываниях, буфере клавиатуры и функциях обслуживания ввода с клавиатуры:**

Общие положения:

Подавляющее большинство программ выполняют ввод информации с клавиатуры. Ввод информации в компьютер может быть выполнен на трех уровнях: обращением к функциям MS-DOS; обращением к функциям BIOS; физическим доступом к аппаратным средствам.

Ввод информации на уровне MS-DOS позволяет "пропустить" клавиатурный ввод через инсталлируемые драйверы, обеспечивает отслеживание нажатия комбинации клавиш Ctrl-C (Ctrl-Break), стандартную для MS-DOS обработку ошибок.

Доступ к клавиатуре на уровне BIOS позволяет программе отслеживать нажатие всех, а не только символьных клавиш, выполнять управление аппаратурой клавиатуры и пр. Интерфейсом Turbo С с BIOS является функция bioskey().

Непосредственный доступ к буферу клавиатуры резко повышает производительность программы. В некоторых случаях необходима имитация нажатий клавиш клавиатуры с записью кодов непосредственно в буфер. При этом физически нажатия клавиш не происходят. Так строятся многие демонстрационные программы, которые открывают или закрывают окна меню, выполняют необходимый выбор, показывают работу программы в "автоматическом" режиме и т.п. На том же самом принципе имитации нажатий клавиш построены программы, способные переносить одним нажатием клавиши целые куски текста из одной программы в любой текстовый редактор. Примером такой программы является входящая в Turbo С резидентная Help-система THELP.COM.

Аппаратные средства персонального компьютера для ввода информации с клавиатуры:

Клавиатура персонального компьютера содержит специальный встроенный микропроцессор. Он при каждом нажатии и отпускании клавиши определяет ее порядковый номер и помещает его в порт 60h специальной электронной схемы - программируемого периферийного интерфейса (ППИ). Далее этот код будем называть скэн-кодом. Скэн-код в первых 7 битах содержит порядковый номер нажатой клавиши, а восьмой бит равен 0, если клавиша была нажата (прямой скэн-код), и равен 1, если клавиша была отпущена (обратный скэн-код). Когда скэн-код записан в порт 60h, схема ППИ выдает сигнал "подтверждения", уведомляя микропроцессор клавиатуры о принятии кода.

Если клавиша остается нажатой дольше некоторого времени задержки (delay value), микропроцессор клавиатуры начинает генерировать с заданной частотой (typematic rate) прямой скэн-код нажатой клавиши. Значения задержки и частоты повторения могут устанавливаться в нужные значения либо через порты клавиатуры, либо через функцию АН = 03h прерывания 16h BIOS. Когда скэн-код принят схемой ППИ, аппаратура компьютера генерирует аппаратное прерывание с номером 9.

Стандартный обработчик прерывания 9 — это программа, входящая в состав BIOS (BIOS ISR). BIOS ISR анализирует скэн-код и по специальным правилам преобразует его. Отметим, что по скэн-коду всегда можно установить, вследствие чего ISR получила управление: из-за нажатия или из-за отпускания клавиши.

Состояние шифт- и триггерных клавиш

|  |  |
| --- | --- |
| Бит | Состояние шифт- и триггерных клавиш |
|  | Байт 40:17h |
| 0 | Нажата и не отпущена клавиша Right Shift |
| 1 | Нажата и не отпущена клавиша Left Shift |
| 2 | Нажата и не отпущена клавиша Ctrl |
| 3 | Нажата и не отпущена клавиша Alt |
| 4 | Зафиксирован скроллинг экрана (ScrollLock - включен) |
| 5 | Включена цифровая клавиатура (NumLock - включен) |
| 6 | Зафиксирован верхний регистр (CapsLock - включен) |
| 7 | Включен режим вставки (хотя клавиша Ins не является триггерной, BIOS-обработчик фиксирует каждое ее нажатие, а код клавиши помещается еще и в буфер клавиатуры) |
|  | Байт40:18h |
| 0 | Нажата и не отпущена клавиша Left Ctrl |
| 1 | Нажата и не отпущена клавиша Left Alt |
| 2 | Нажата клавиша System Request (System) |
| 3 | Включен режим Pause(Ctrl-NumLock) |
| 4 | Нажата и не отпущена клавиша ScrollLock |
| 5 | Нажата и не отпущена клавиша NumLock |
| 6 | Нажата и не отпущена клавиша CapsLock |
| 7 | Нажата и не отпущена клавиша Ins |

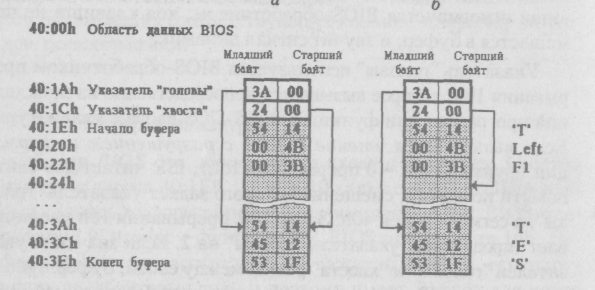
Буфер клавиатуры:

Буфер BIOS для записи кодов клавиш занимает 32 байта оперативной памяти с адреса 40:lEh до 40:3Eh. Запись информации в буфер выполняет ISR BIOS прерывания 9, чтение - функции ISR BIOS прерывания 16h. Буфер клавиатуры рассчитан на 15 нажатий клавиш, генерирующих двухбайтовые коды и поэтому имеет 30 байт для кодов клавиш и еще два дополнительных байта, которые резервируются под двухбайтовый код для клавиши ENTER.

Буфер организуется как кольцевая очередь, доступ к которой осуществляется с помощью указателя «головы» (head pointer), адрес которого 40:1Ah, и указателя «хвоста» (tail pointer), адрес которого 40:1Ch. Указатель "хвоста" задает смещение до слова, где будет записан обработчиком прерывания 9 код буферизуемой клавиши, т.е. первое свободное слово буфера. Указатель "головы" задает смещение слова, которое будет возвращено запросу буфе­ризованного ввода с клавиатуры, сделанного операционной системой или BIOSoм.

При каждом нажатии клавиши, для которой генерируется двухбайтовый код, ISR BIOS прерывания 9,'используя текущее значение указателя "хвоста", записывает в память образованный двухбайтовый код. После этого указатель "хвоста" увеличивается на 2. Если указатель "хвоста" перед доступом к буферу указывает на верхнюю границу буфера (на слово 40:3Eh), указатель после записи в буфер "перепрыгивает" на начало буфера, т.е. ему присваивается значение 40:1Eh. Поэ­тому значение указателя "хвоста" может быть и меньше значения указателя "головы". Это значит, что указатель "хвоста" "перескочил" назад к нижней границе буфера. Когда указатель "хвоста" догонит указатель "головы", наступит пере­полнение буфера. В этом случае указатель "хвоста" задает смещение до "холостой" позиции. Каждое новое нажатие клавиши игнорируется BIOS-обработчиком; код клавиши не помещается в буфер, и звучит сигнал динамика.

Указатель "головы" используется BIOS-обработчиком прерывания 16h, которое вызывается непосредственно из приклад­ной программы или функциями MS-DOS ввода с клавиатуры. Если выполняется чтение буфера с разрушением информации (функция АН = 0 прерывания 16h, ISR читает два байта памяти по адресу, смещение которого задает указатель "голо­вы", а сегмент равен 40h. Затем ISR прерывания 16h увеличи­вает (продвигает) указатель "головы" на 2. Если значения ука­зателей "головы" и "хвоста" равны между собой, буфер пуст. В этом случае ISR прерывания 16 выполняет бесконечный цикл ожидания, условием выхода из которого будет неравенство указателей. При нажатии клавиши, генерирующей двухбайтовый код, этот цикл будет прерван BIOS-обработчиком прерывания 9. В результате указатель "головы" продвинется на 2. После завершения ISR BIOS прерывания 9 возобновится выполнение ISR BIOS прерывания 16h. Так как буфер уже не пуст, произойдет выход из цикла и передача в точку вызова прочитанного в буфере двухбайтового кода.Если выполняется чтение буфера без разрушения информации (функция АН = 1 прерывания 16h), продвижение указателя "головы" не происходит, но прочитанный по этому указателю код передается в точку вызова, если только буфер не пуст. Если буфер пуст, ISR прерывания 16h не выполняет цикл ожидания. Вместо этого флаг переноса CF устанавливается в 1, и обработчик завершает свою работу.



На рис. приведены примеры пустого буфера клавиатуры и буфера после ввода с клавиатуры строки "TEST", нажатия клавиш Left и F1 при условии, что текущая программа не выполняла в этот момент ввод с клавиатуры.

Рис. Организация буфера клавиатуры: а - пустой буфер (значения указателей "готовы" и "хвоста" равны); б- тот же буфер после набора на клавиатуре строки TEST, нажатия специальных клавиш Left и F1 (текущая программа не выполняет ввод информации и указатель "головы" не продвигается)

Буфер клавиатуры - это классический пример использования кольцевого буфера для организации асинхронного взаимодействия двух программ по схеме "производитель-потребитель". Одна из программ (ISR BIOS прерывания 9) "производит" информацию или, как говорят, является процессом-производителем. Исполняемая программа через функцию АН= 00h прерывания 16h BIOS "потребляет" информацию или является процессом-потребителем. Асинхронность взаимодействия означает, что запись в буфер новой информации и чтение из него происходят в случайные, не связанные между собой моменты времени. Так как производитель постоянно анализирует наличие переполнения буфера, не происходит переопределения не прочитанных еще потребителем кодов клавиатуры. Другими словами, при переполнении буфера производитель блокируется до тех пор, пока потребитель не прочитает одно или несколько слов из буфера. Если же буфер пуст и выполняется попытка чтения информации, функция АН = 00h прерывания 16h BIOS переходит к бесконечному циклу, условием которого является неравенство между собой указателей "головы" и "хвоста". Фактически текущая программа, выполняющая ввод с клавиатуры, блокируется, не давая "потребить" несуществующую еще информацию.

Функции прерывания 21h MS-DOS для ввода информации с клавиатуры:

MS-DOS имеет целую группу функций прерывания 21h для выполнения ввода информации с клавиатуры. Последовательность действий системы при вводе с клавиатуры такова. Функция MS-DOS вызывает драйвер клавиатуры, передавая ему запрос на ввод одного символа из буфера клавиатуры. Драйвер, выполняя запрос, обращается к нужной функции прерывания 16h BIOS. ISR BIOS прерывания 16h читает из буфера клавиатуры нужное слово и передает в драйвер. Драйвер возвращает байт (обычно младший) в MS-DOS. Таким образом, функции MS-DOS и опирающиеся на них функции библиотеки Turbo С слабо зависят от особенностей аппаратуры, поскольку система от нее изолирована двумя слоями программного обеспечения - драйверами и BIOSом.

Функции библиотеки С++:

int getch (void): выполняет ввод с клавиатуры через функцию MS-DOS АН=07h. Она не выполняет "эхо" вывода на экран. В этой связи полезна для организации интерфейса с пользователем, при котором нажатие той или иной клавиши вызывает немед­ленную реакцию программы без отображения введенного символа на экране.

int getche (void): выполняет небуферизуемый ввод с клавиатуры через функцию MS-DOS AH=07h, но в отличие от предыдущей функции обеспечивает вывод введенного символа на экран. Перевод строки происходит при достижении правой вертикальной границы текущего активного окна.

char \*getpass(char \* prompt): Выводит на экран ASCII-строку, на начало которой указывает prompt, a затем принимает с клавиатуры без "эха" строку символов. Вводимые символы (не более 7) помещаются во внутреннюю статическую память. Функция возвращает указатель на внутреннюю статическую строку, переопределяемую каждым новым обращением к функции. Основное назначение данной функции - ввод паролей в программе без отображения их на экран.

int kbhit (void): проверяет, пуст ли буфер клавиатуры. Если в буфере есть символы, функция возвращает ненулевое значение, в противном случае она возвращает 0. Использует функцию 0Bh MS-DOS. Является удобным средством предотвращения "зацикливания" при ожидании невозможного в данный момент события. Кроме того, при выполнении функции 0Bh осуществляется проверка нажатия комбинации клавиш Ctrl-Break, что позволяет выполнить аварийное завершение программы.

Функции для работы с вводом/выводом:

|  |  |
| --- | --- |
| getch (void) | Считать клавишу без «эха» |
| getche (void) | Считать клавишу |
| kbhit (void) | Проверка, пуст ли буфер |
| bioskey(int cmd) | Считывание через прерывания |

**Задание**

**1.Разработать, написать и отладить программу управления перемещением символа в пределах заданного на экране окна.** Для управления использовать клавиши из набора: "стрелка вверх" (СтВВ), "стрелка вниз" (СтВН), "стрелка вправо" (СтВП), "стрелка влево" (СтВЛ). Для ввода использовать стандартные функции языка C++ (getch, getche, kbhit, bioskey).

2.Заменить стандартные функции библиотеки C++ (getch, getche, kbhit, bioskey) своими. Для написания функций используйте заданное прерывание INT 21h или INT 16h. Если его возможностей недостаточно, то замените его по своему усмотрению.

****

**Алгоритмы и тексты отлаженных программ**

Первая программа:

#include <dos.h>

#include <conio.h>

#include <stdio.h>

int x1 = 25;

int y1 = 5;

int x2 = 55;

int y2 = 15;

int x = 1;

int y = 1;

int ch = 0;

int left = 0;

int down = 0;

int move = 0;

int main()

{

window(x1, y1, x2, y2);

textbackground(RED);

textcolor(YELLOW);

clrscr();

do {

clrscr();

if ((left == 0) && (down == 0)){

printf("exit - F1, start - any arrow");

}

else {

if (move == 0){

x = x;

y = y;

}

else {

if ((left == 1) && (down == -1) && ((x - 1) >= 0))

x --;

if ((left == 0) && (down == -1) && ((x + 1) <= (x2 - x1)))

x++;

if ((left == -1) && (down == 0) && ((y - 1) >= 0))

y--;

if ((left == -1) && (down == 1) && ((y + 1) <= (y2 - y1)))

y++;

}

gotoxy(x, y);

putch('\*');

delay(100);

}

ch = getch();

if (ch == 0 || ch == 224){

switch (getch()){

case 75:

move = 1;

left = 1;

down = -1;

break;

case 77:

move = 1;

left = 0;

down = -1;

break;

case 72:

move = 1;

left = -1;

down = 0;

break;

case 80:

move = 1;

left = -1;

down = 1;

break;

case 59:

return 0;

break;

default:

move = 0;

break;

}

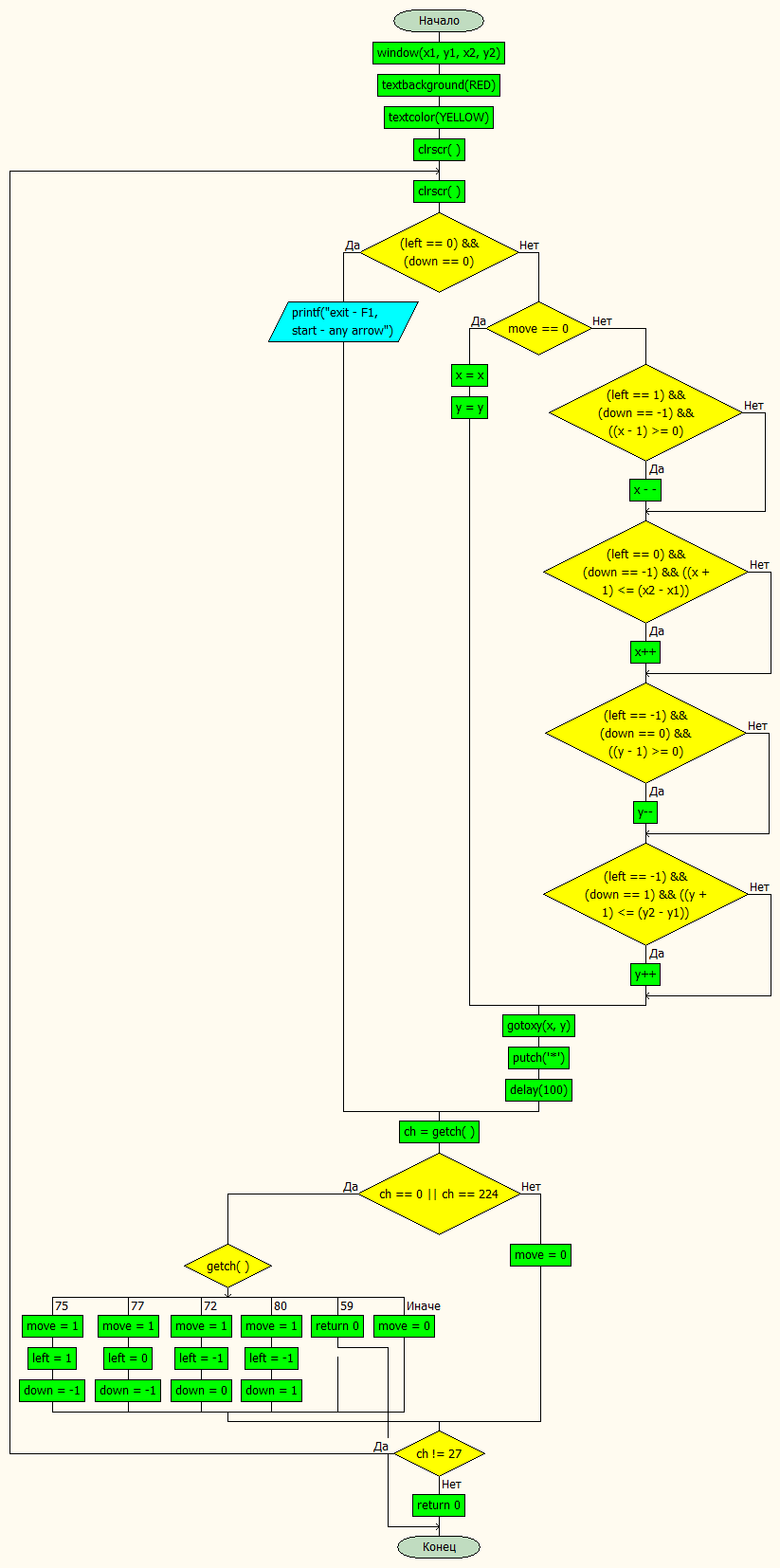
}

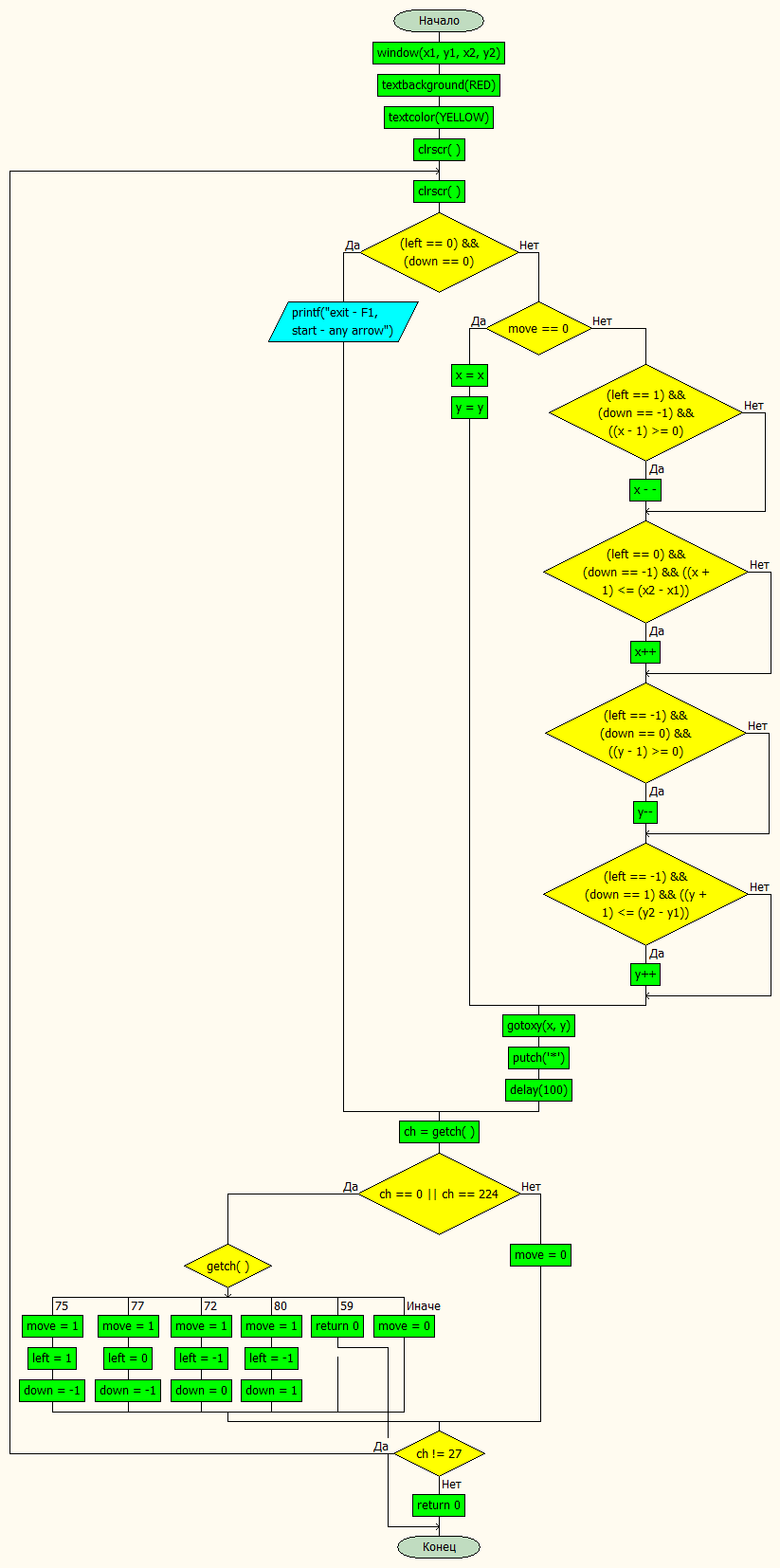
else

move = 0;

} while (ch != 27);

return 0;

}



Вторая программа:

#include <dos.h>

#include <conio.h>

#include <stdio.h>

int x1 = 25;

int y1 = 5;

int x2 = 55;

int y2 = 15;

int x = 1;

int y = 1;

int ch = 0;

int left = 0;

int down = 0;

int move = 0;

int get\_code(){

union REGS r;

r.h.ah = 0x7;

int86(0x21, &r, &r);

return r.h.al;

}

int main()

{

window(x1, y1, x2, y2);

textbackground(RED);

textcolor(YELLOW);

clrscr();

do {

clrscr();

if ((left == 0) && (down == 0)){

printf("exit - F1, start - any arrow");

}

else {

if (move == 0){

x = x;

y = y;

}

else {

if ((left == 1) && (down == -1) && ((x - 1) >= 0))

x --;

if ((left == 0) && (down == -1) && ((x + 1) <= (x2 - x1)))

x++;

if ((left == -1) && (down == 0) && ((y - 1) >= 0))

y--;

if ((left == -1) && (down == 1) && ((y + 1) <= (y2 - y1)))

y++;

}

gotoxy(x, y);

putch('\*');

delay(100);

}

ch = get\_code();

if (ch == 0 || ch == 224){

switch (get\_code()){

case 75:

move = 1;

left = 1;

down = -1;

break;

case 77:

move = 1;

left = 0;

down = -1;

break;

case 72:

move = 1;

left = -1;

down = 0;

break;

case 80:

move = 1;

left = -1;

down = 1;

break;

case 59:

return 0;

break;

default:

move = 0;

break;

}

}

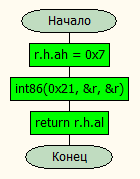
else

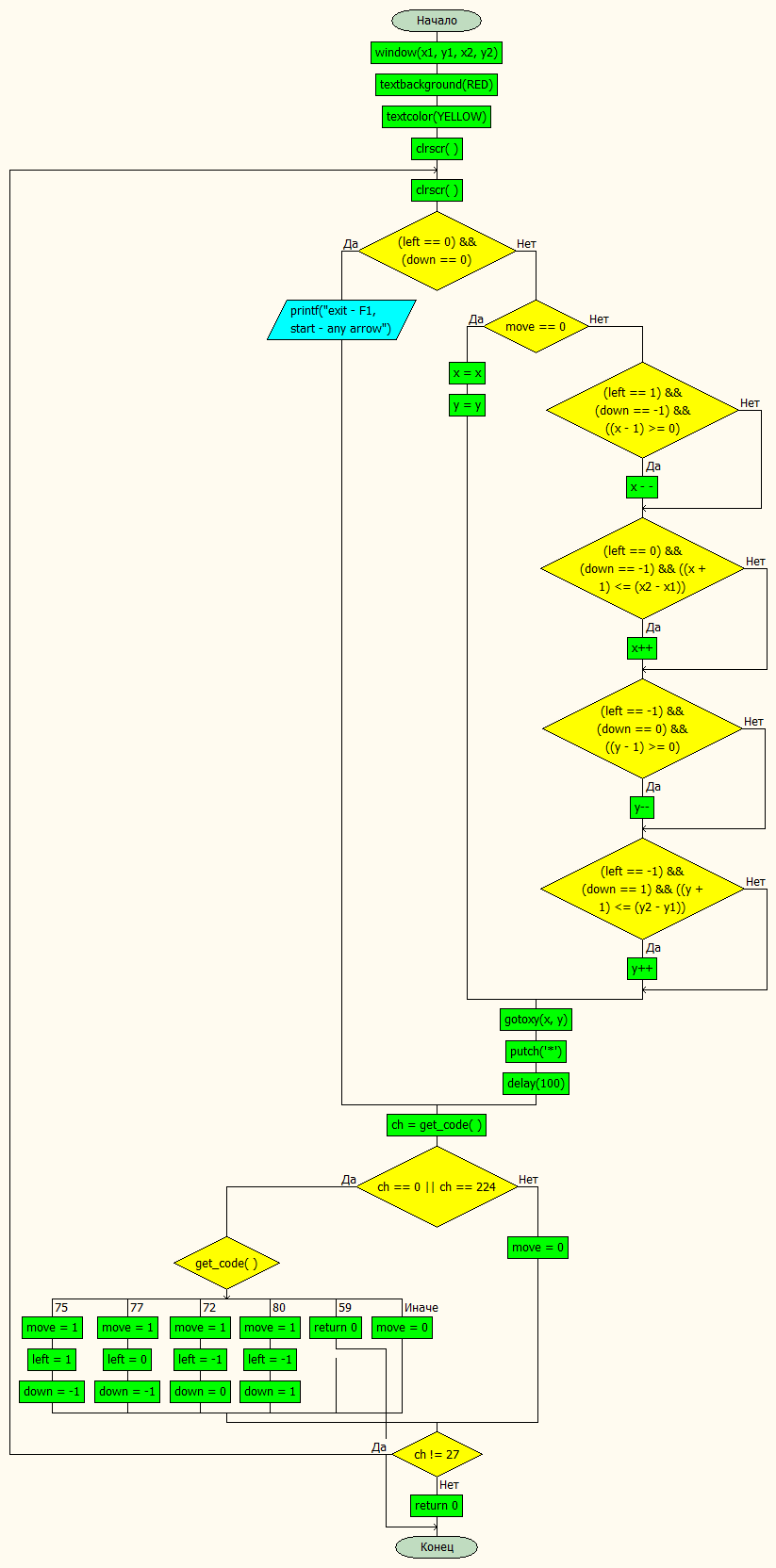
move = 0;

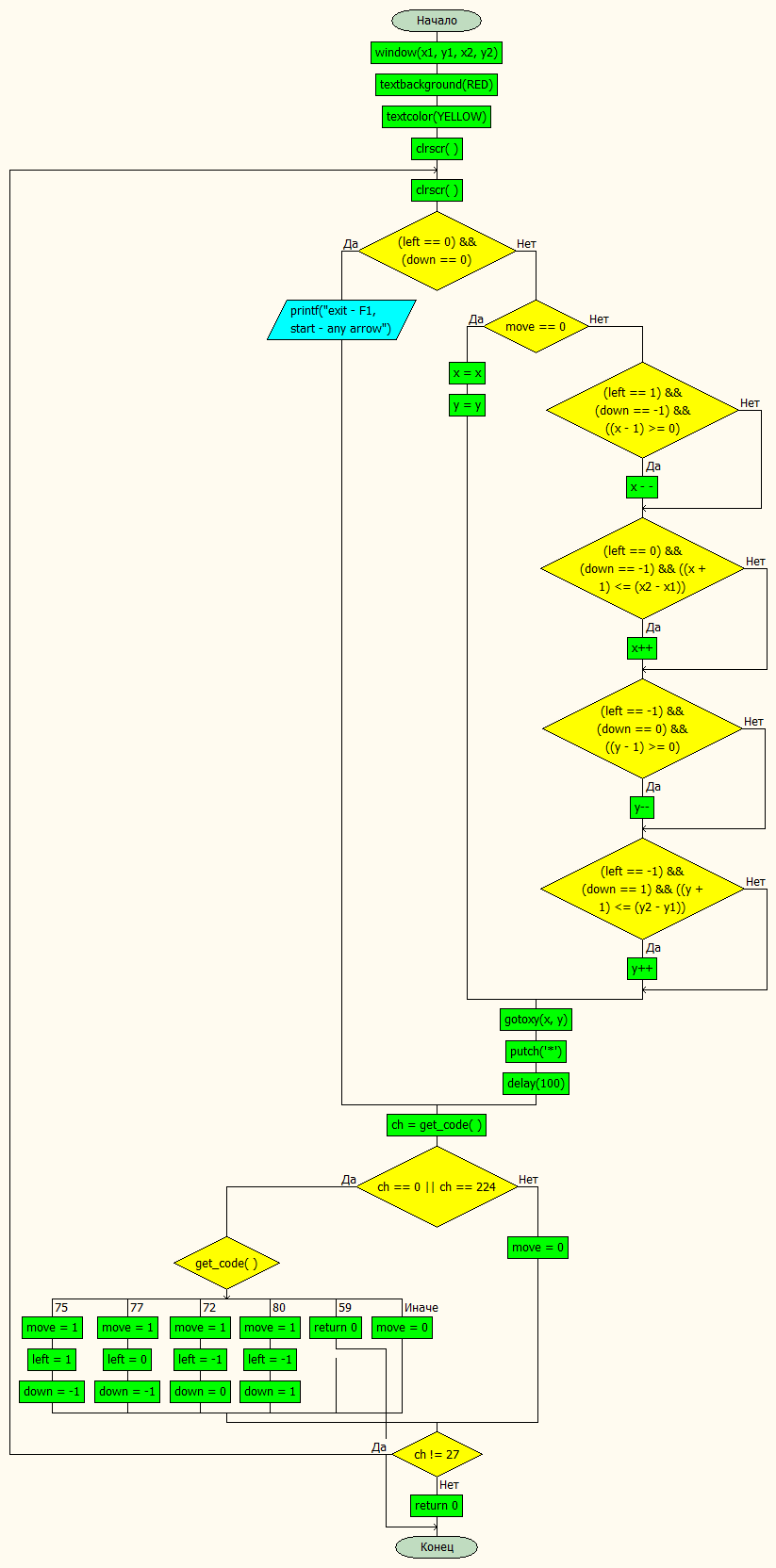
} while (ch != 27);

return 0;

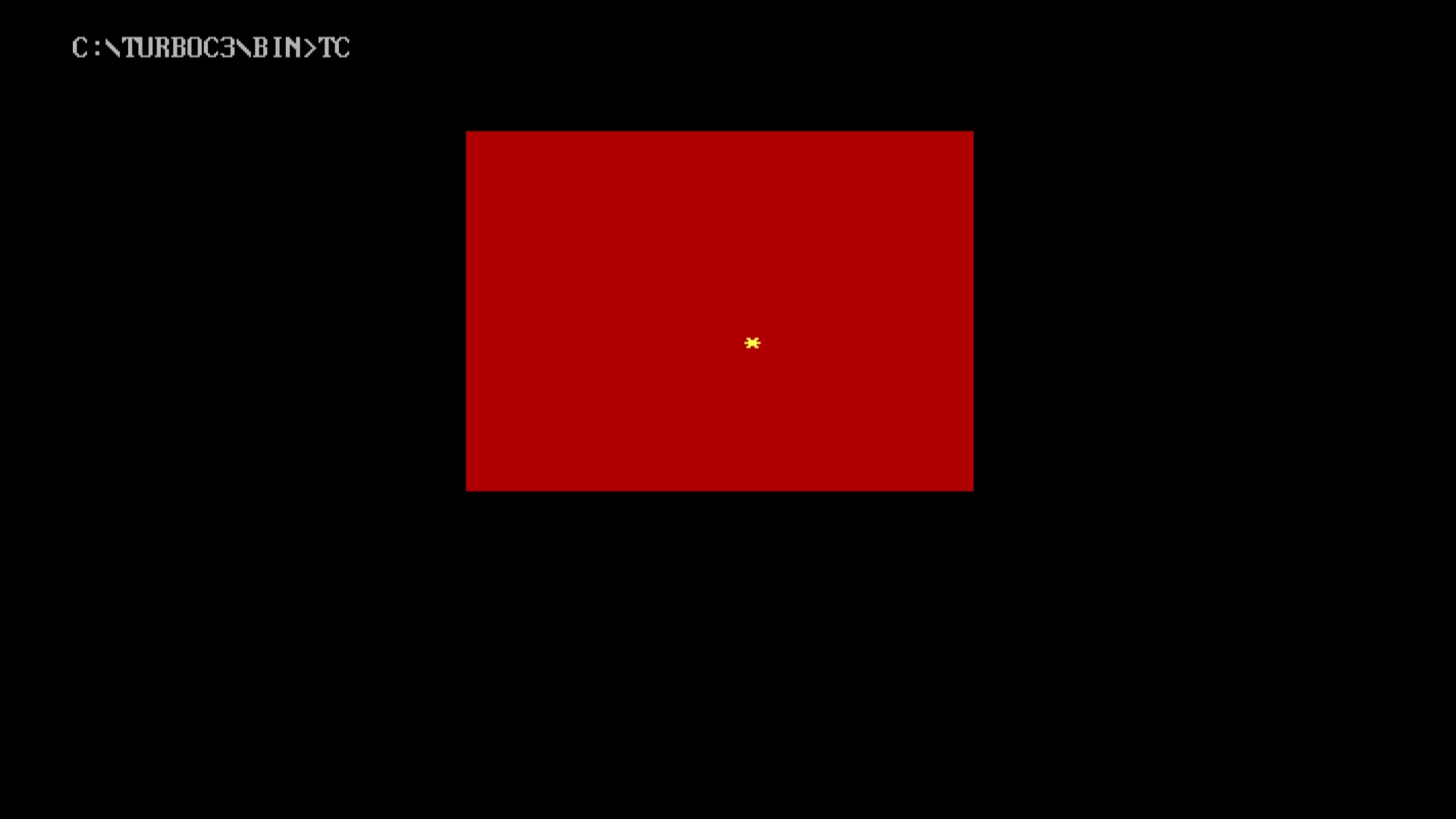
}

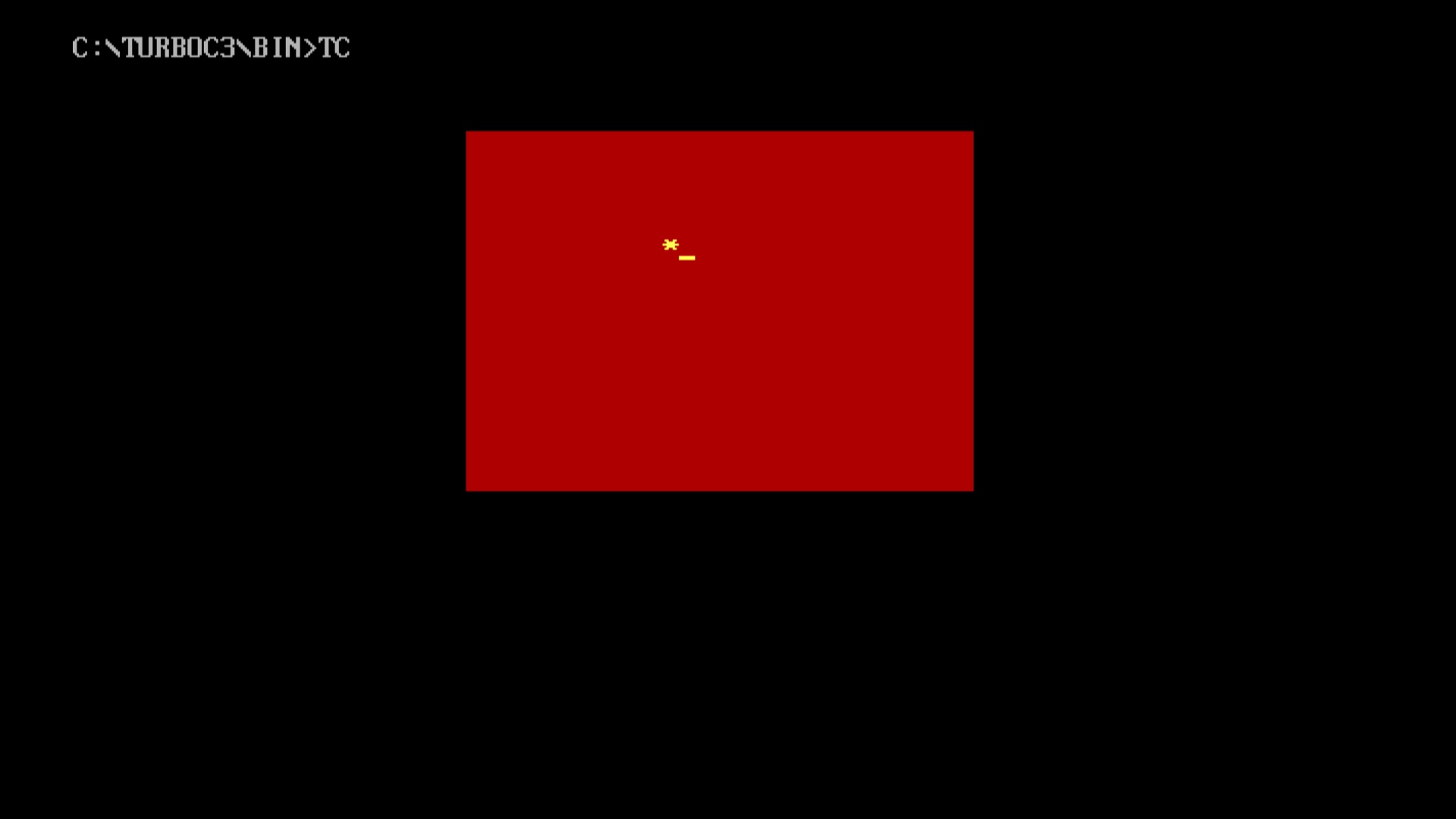
int get\_code()

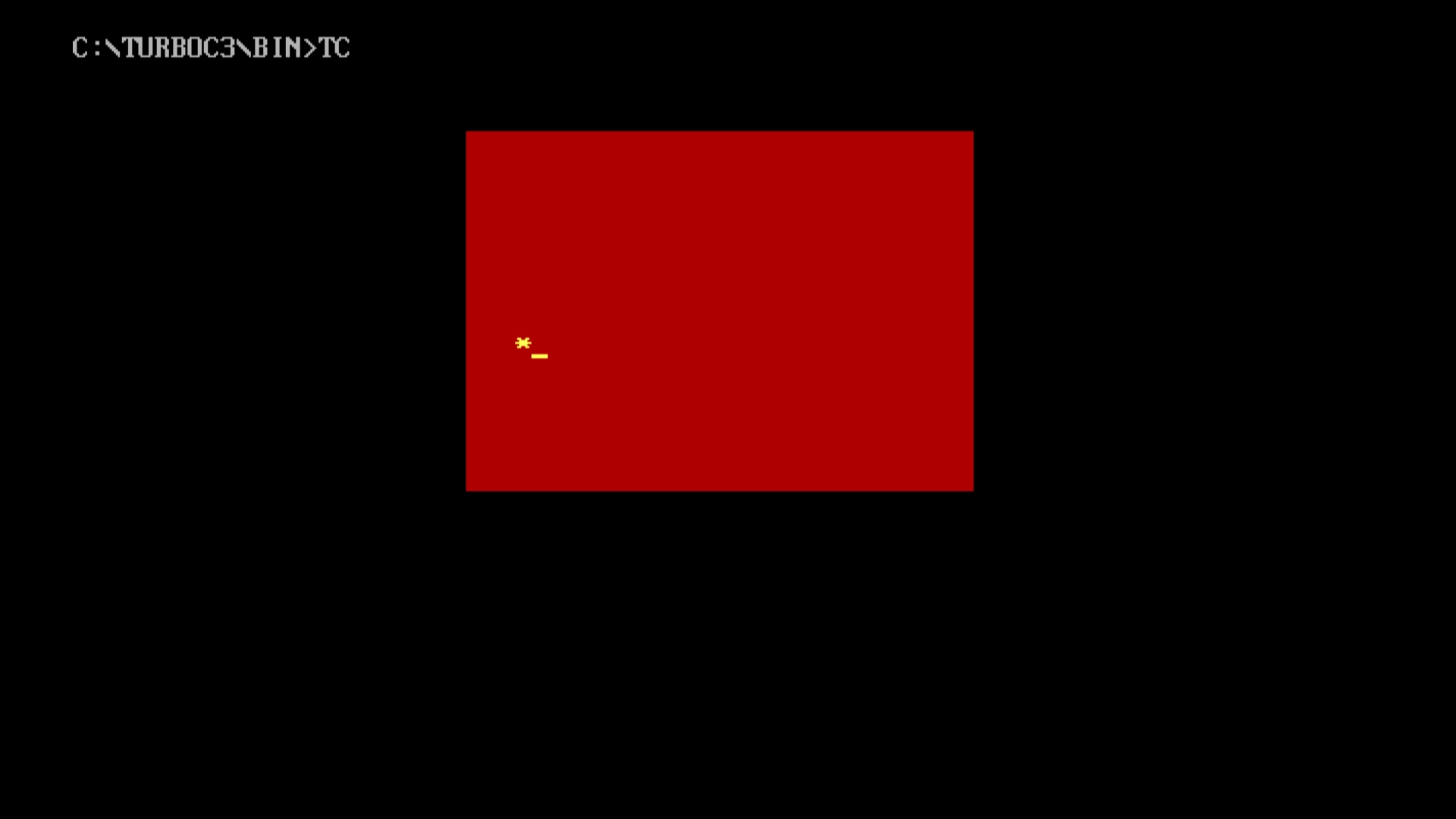




Результаты работы:







Вывод:

Была написана программа управления перемещением символа в пределах заданного на экране окна. Изучены стандартные функции языка С++, такие как getch, getche, kbhit, bioskey.