МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра вычислительной техники

ОТЧЕТ

по лабораторной работе №4

по дисциплине «Организация ЭВМ и систем»

Тема: КЛАВИАТУРА IBM PC. ИСПОЛЬЗОВАНИЕ ПРЕРЫВАНИЙ

Студентка гр. 3353 _____ Шинкарь К. Д.

Преподаватель _____ Гречухин М.Н.

Санкт-Петербург

2024

Цель работы: изучение возможностей работы с клавиатурой, ознакомление со стандартными средствами библиотеки C++ и средствами системы прерываний DOS и BIOS, обслуживающими клавиатуру.

Задание:

1) Разработать, написать и отладить программу управления перемещением символа (например, "*") в пределах заданного на экране окна. Для управления использовать клавиши из набора: "стрелка вверх" (СтВВ), "стрелка вниз" (СтВН), "стрелка вправо" (СтВП), "стрелка влево" (СтВЛ) или функциональные клавиши Fl - F12 (варианты см. в таблице 4.2). Для ввода использовать стандартные функции языка C++.
2) Изменить программу, заменив стандартные функции библиотеки C++ своими. Для написания функций используйте заданное прерывание (см. таблицу), если его возможностей достаточно. Если его возможностей недостаточно, то замените его по своему усмотрению. Сохраните отлаженную программу.

В моем случае координаты окна: (10; 5; 70; 15), движение символа постоянное, стрелка движется вверх и вниз, используется прерывание INT 21h.

Краткие сведения о подсистеме ввода информации с клавиатуры, используемых прерываниях, буфере клавиатуры и функциях обслуживания ввода с клавиатуры.

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 C 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 C резидентная Help-система THELP.COM.

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 AH = 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.

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 AH = 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 AH = 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.

int getch (void) выполняет ввод с клавиатуры через функцию MS-DOS AH=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, а 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.

Текст программы

1 вариант:

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>

#define ESCAPE 27
#define UP 72
#define DOWN 80

#define X1 10
#define X2 70
#define Y1 5
#define Y2 15

int get_key_press() {
   union REGS regs;
   regs.h.ah = 0x08;
   int86(0x21, &regs, &regs);
   return regs.h.al;
}
```

```
int main() {
    int x = (X2 - X1 + 1) / 2 + X1;
    int y = (Y2 - Y1 + 1) / 2 + Y1;
    int direction = 0;
    int ch;

    window(X1, Y1, X2, Y2);
    textbackground(2);
    textcolor(6);
    _setcursortype(_NOCURSOR);
    clrscr();

    gotoxy(x, y);
    printf("*");

    do {
        if (kbhit()) {
            ch = get_key_press();
            if (ch == 0 || ch == 224) {
                ch = get_key_press();
                if (ch == DOWN) {
                    direction = 1;
                } else if (ch == UP) {
                    direction = -1;
                }
            }
        }

        if (direction != 0) {
            delay(100);
            clrscr();
            y += direction;
```

```
            if (y > (Y2 - Y1 + 1)) {
                y = 1;
            } else if (y < 1) {
                y = (Y2 - Y1 + 1);
            }

            gotoxy(x, y);
            printf("*");
        }
    } while (ch != ESCAPE);

    return 0;
}
```

2 вариант:

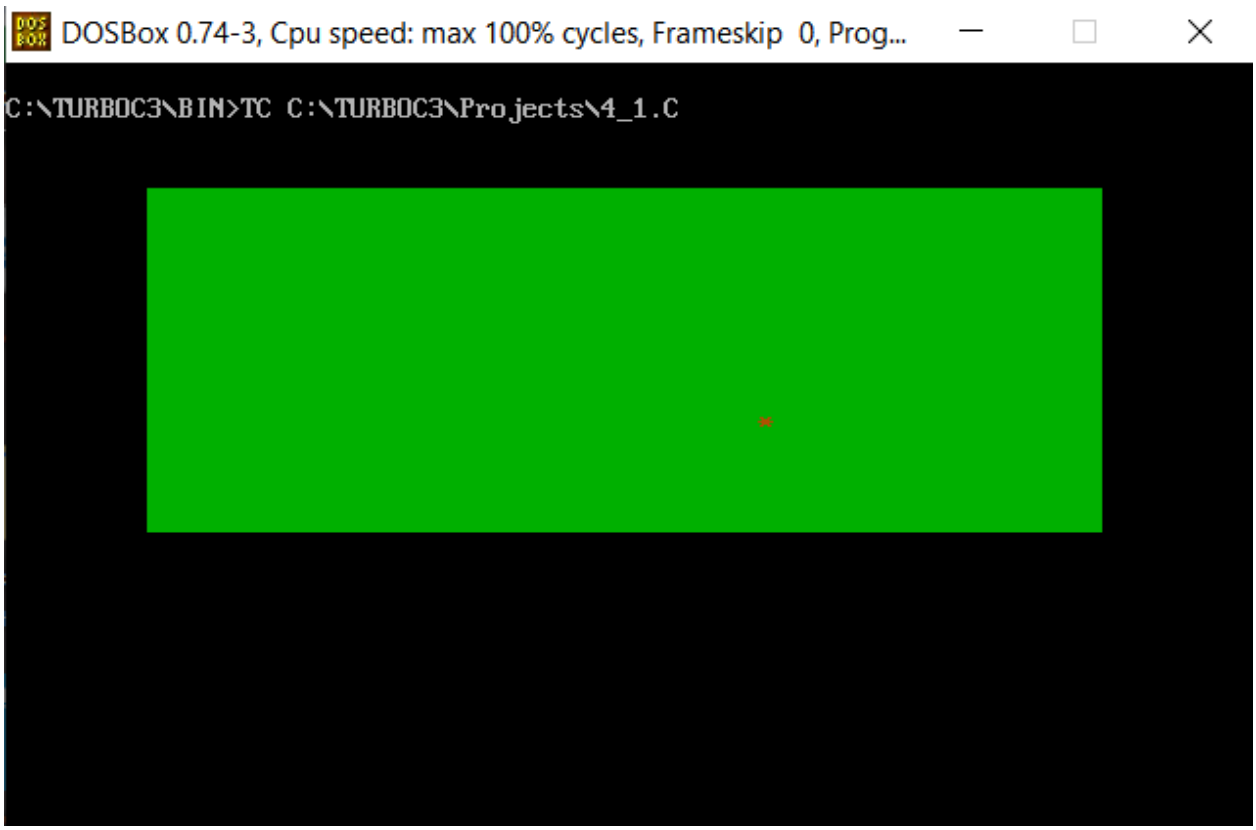
```
#include <conio.h>
#include <bios.h>
#include <dos.h>

#define ESCAPE 27
#define UP 72
#define DOWN 80

#define X1 10
#define X2 70
#define Y1 5
#define Y2 15

int presskey() {
    union REGS r;
    r.h.ah = 0x0B;
    int86(0x21, &r, &r);
```

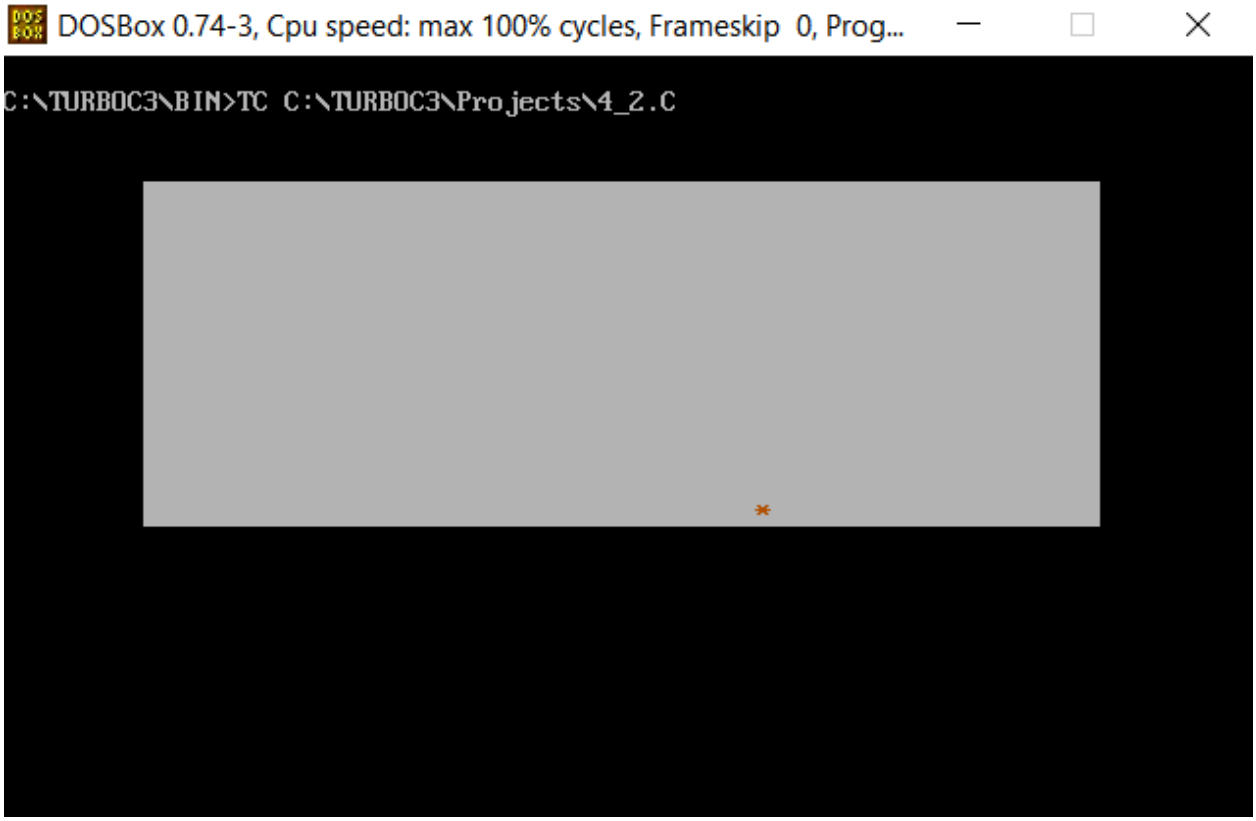
```
    return r.h.al;
}

int getKey() {
    union REGS r;
    r.h.ah = 0x07;
    int86(0x21, &r, &r);
    return r.h.al;
}

void putchar(char ch, int x, int y) {
    gotoxy(x, y);
    putch(ch);
}

int main() {
    int x, y, ch = 0;
    int direction = 0;

    x = (X2 - X1 + 1) / 2 + X1;
    y = (Y2 - Y1 + 1) / 2 + Y1;

    window(X1, Y1, X2, Y2);
    textbackground(7);
    textcolor(6);
    _setcursortype(_NOCURSOR);
    clrscr();

    putchar('*', x, y);

    do {
        if (presskey()) {
            ch = getKey();
            if (ch == 0 || ch == 224) {
                ch = getKey();
```

```
            switch (ch) {
                case DOWN:
                    direction = 1;
                    break;
                case UP:
                    direction = -1;
                    break;
            }
        }
    }

    if (direction != 0) {
        clrscr();
        y += direction;

        if (y > (Y2 - Y1 + 1)) {
            y = 1;
        } else if (y < 1) {
            y = (Y2 - Y1 + 1);
        }

        putchar('*', x, y);
        delay(100);
    }
  } while (ch != ESCAPE);

  return 0;
}
```

Вывод

В ходе выполнения лабораторной работы 2 способами была разработана программа, которая выводит на экран символ «*». Символ перемещается вверх и вниз при нажатии соответствующих клавиш стрелок. Результаты выполнения одинаковы для обоих вариантов.


DOSBox 0.74-3, Cpu speed: max 100% cycles, Frameskip 0, Prog...
C:\TURBOC3\BIN>TC C:\TURBOC3\Projects\4_1.C


DOSBox 0.74-3, Cpu speed: max 100% cycles, Frameskip 0, Prog...
C:\TURBOC3\BIN>TC C:\TURBOC3\Projects\4_2.C