**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**

|  |  |  |
| --- | --- | --- |
| Студент гр. 3311 |  | Сапронов 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.

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

Подсистема ввода информации с клавиатуры в MS-DOS представляет собой многоуровневую систему, которая обеспечивает взаимодействие пользователя с компьютером. Ввод с клавиатуры может осуществляться через три уровня: посредством функций MS-DOS, через BIOS или напрямую с использованием аппаратных средств.

На уровне MS-DOS обеспечивается стандартная обработка ввода, включая взаимодействие с драйверами и управление базовыми операциями. Этот уровень удобен для большинства задач, так как включает встроенные механизмы обработки ошибок. На уровне BIOS осуществляется более низкоуровневое взаимодействие с аппаратурой клавиатуры, что позволяет считывать нажатия всех клавиш, включая специальные, а также управлять аппаратными особенностями устройства. Самый низкий уровень представляет собой прямое обращение к аппаратным средствам, что обеспечивает максимальную производительность и возможность имитации нажатий клавиш через программное управление.

Для обработки ввода используется аппаратное прерывание с номером 9, которое обрабатывается стандартным обработчиком BIOS. Этот обработчик преобразует коды, поступающие от клавиатуры, в понятный для системы формат. При нажатии клавиши клавиатура отправляет скан-код в процессор, который затем фиксируется и передается в соответствующие регистры. BIOS анализирует эти коды, формирует ASCII-символы или коды специальных клавиш и помещает их в буфер клавиатуры для последующего использования приложениями.

Такая многоуровневая структура позволяет системе обрабатывать как стандартные команды пользователя, так и более сложные сценарии, например, перехват и управление вводом на низком уровне. Это делает подсистему ввода с клавиатуры универсальной и эффективной в рамках операционной системы MS-DOS.

**Задание (вариант 18)**

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

2. Изменить программу, **заменив стандартные функции библиотеки C++(**getch, getche, kbhit, bioskey) **своими.**  Для написания функций используйте заданное прерывание INT 21h. Если его возможностей не достаточно, то замените его по своему усмотрению.

**Описание решения**

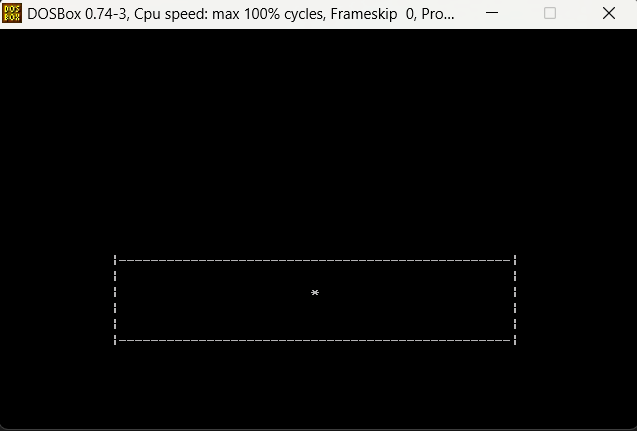
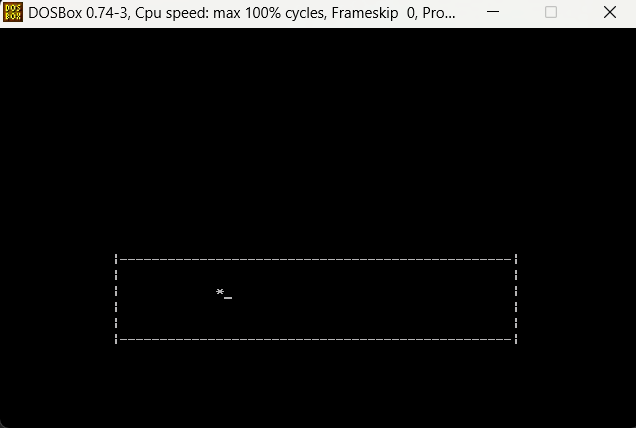
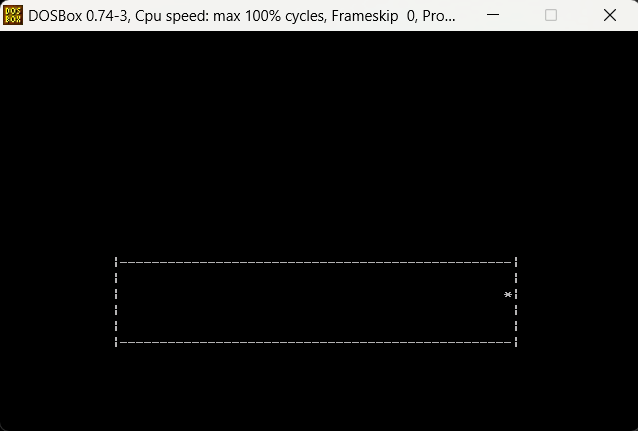
Для выполнения задания были созданы структура Point, хранящая текущие координаты символа, а также функции clear и render – первая очищает старую позицию, а вторая отрисовывает символ в новой.

В функции update\_position производится проверка скан кода нажатой клавиши – он записывается в переменную ch. Если ch = 27 (ESC), то программа прекращает работу. Если ch = 0, то была нажата клавиша, использующая расширенный скан код формата “ 0 / код ”, поэтому снова производится чтение кода в переменную ch (уже второй части расширенного скан кода), и если это стрелка влево или вправо (75 и 77), то координаты символа уменьшаются или увличиваются соответственно.

В функции main работает цикл, который выполняет функцию update\_position до тех пор, пока не будет нажата клавиша ESC.

Стандартные функции getch и kbhit были заменены на my\_getch и my\_kbhit с использованием прерываний 21h. Первая функция считывает ввод без отображения символа на экране с помощью прерывания AH = 07h. Возвращает значение регистра AL, в который был записан скан код прочитанного символа. Вторая функция проверяет состояние ввода с помощью прерывания AH = 0Bh. Возвращает 0, если клавиша не была нажата.

**Пример запуска программы**

**Выводы**

Использование прерываний для обработки ввода с клавиатуры позволяет более эффективно выполнять узконаправленные задачи на низком уровне, но требует глубокого понимания особенностей аппаратной архитектуры, так как работа с такими вызовами может быть менее универсальной и при неосторожности приводить к сбоям и некорректному поведению.

**Текст программы**

#include <dos.h>

#include <stdio.h>

#include <conio.h>

typedef struct {

    int x;

    int y;

} Point;

int my\_getch() {

    union REGS in, out;

    in.h.ah = 0x07; // Read symbol

    int86(0x21, &in, &out);

    return out.h.al;

}

int my\_kbhit() {

    union REGS in, out;

    in.h.ah = 0x0B; // Keyboard input

    int86(0x21, &in, &out);

    return out.h.al != 0;

}

void draw\_window() {

    int i;

    for (i = 15; i <= 65; i++) { // Draw top and bottom borders

        gotoxy(i, 15);

        printf("-");

        gotoxy(i, 20);

        printf("-");

    }

    for (i = 15; i <= 20; i++) { // Draw left and right borders

        gotoxy(15, i);

        printf("|");

        gotoxy(65, i);

        printf("|");

    }

}

void render(Point p) {

    gotoxy(p.x, p.y);

    printf("\*");

}

void clear(Point p) {

    gotoxy(p.x, p.y);

    printf(" ");

}

int update\_position(Point \*p) {

    int ch = my\_getch();

    if (ch == 27) { // Exit on ESC

        return 0;

    }

    clear(\*p);

    if (ch == 0) {

        ch = my\_getch(); // Read second part of scan code

        if (ch == 75 && p->x > 16) { // Left arrow

            p->x--;

        } else if (ch == 77 && p->x < 64) { // Right arrow

            p->x++;

        }

    }

    render(\*p);

    return 1;

}

int main() {

    Point point = {

        40,

        17

    };

    clrscr();

    draw\_window();

    render(point);

    while (1) {

        if (my\_kbhit()) { // Check if a key is pressed

            if (!update\_position(&point)) { // Exit on ESC

                break;

            }

        }

    }

    clrscr();

    return 0;

}