МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«Омский государственный технический университет»

Факультет информационных технологий и компьютерных систем

Кафедра «Прикладная математика и фундаментальная информатика»

**Домашнее задание**

|  |  |  |
| --- | --- | --- |
| по дисциплине | Операционные системы |  |

|  |  |
| --- | --- |
| Студента | Иванютин Сергей Александрович |
|  | фамилия, имя, отчество полностью |
| Курс | 2 Группа ФИТ-221 |
| Направление | 02.03.02 Фундаментальная информатика |
|  | и информационные технологии |
|  | код, наименование |
| Руководитель | ассистент |
|  | должность, ученая степень, звание |
|  | Карабцов Р.Д. |
|  | фамилия, инициалы |
| Выполнил | 04.06.2024 |
|  | дата, подпись студента |
| баллы |  |
|  | дата, подпись руководителя |

Омск-2024

**Задание**

Программа для Windows должна открыть текстовый файл для чтения и все его содержимое вывести в текстовую консоль (текстовое окно на экране). Затем в программе запускается опрос мыши в консольном режиме. По щелчке мышью на любом слове, находящемся в текстовой консоли, в последней строке этого текстового окна выводится буква, на которой осуществлен щелчок, и координаты позиции, на которой он произошел, выраженные в виде номера строки и столбца текстового режима. Эта информация выводится только для не пустого позиции (для пробелов выводить ничего не нужно). Такие действия программ может осуществлять многократно. Завершение программы осуществляется по нажатию правой клавиши мыши.

1. Чем событийное программирование отличается от жестко алгоритмического, опишите сферу применений того и другого?

2. Чем отличается программная обработка нажатий на управляющую клавишу от аналогичной обработки нажатий на алфавитно-цифровую клавишу при использовании функции ReadConsoleInput в Windows?

3. Почему в современных ОС для получения информации от мыши в прикладных программах используется событийно организованное программирование, а для получения информации от клавиатуры - не всегда?

**Решение**

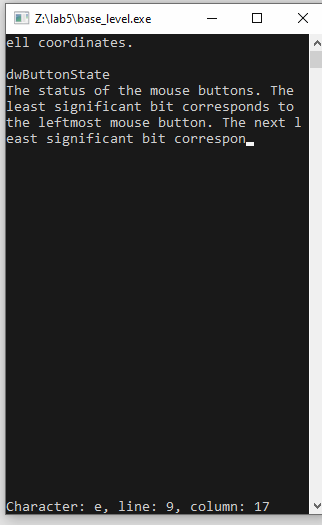
****

Рисунок 1. Вывод программы

1.Отличия между событийным и жестко алгоритмическим программированием:

Событийное программирование основано на обработке событий, таких как ввод пользователя, сетевые события и другие внешние события. Программа реагирует на события, вызывая соответствующие обработчики. Применяется в графических интерфейсах, веб-приложениях, играх и других областях, где важна реакция на динамичные внешние изменения.

Жестко алгоритмическое программирование предполагает четкий и строгий порядок выполнения операций. Алгоритмы заранее определены, и программа следует им, не зависимо от внешних событий. Применяется в вычислительных задачах, обработке данных, алгоритмических задачах, где порядок выполнения важен.

2.Обработка нажатий клавиш в Windows:

При использовании функции ReadConsoleInput в Windows, нажатия на управляющие клавиши (например, клавиши Shift, Ctrl, Alt) обрабатываются как события KEY\_EVENT. В структуре KEY\_EVENT\_RECORD устанавливается флаг dwControlKeyState, который указывает на состояние управляющих клавиш.

Нажатия на алфавитно-цифровые клавиши также обрабатываются как события KEY\_EVENT, но состояние управляющих клавиш в dwControlKeyState будет равно нулю.

3.Использование событийно организованного программирования для мыши и клавиатуры:

Для получения информации от мыши часто используется событийная модель, потому что действия пользователя (клики, перемещения) считываются как события. Это позволяет реагировать на динамичные и непредсказуемые действия пользователя.

В случае клавиатуры подходы могут различаться. Для некоторых сценариев, особенно где важен строгий порядок выполнения, жестко алгоритмическое программирование может быть предпочтительным. Однако, с ростом сложности пользовательских интерфейсов, даже клавиатурные вводы часто обрабатываются с использованием событий для более гибкого управления и улучшенной реакции на действия пользователя.

**Листинг программы**

#include <stdio.h>

#include <windows.h>

#include "/usr/x86\_64-w64-mingw32/include/windows.h"

void display\_at\_bottom(HANDLE h, char \*text, DWORD len);

int main()

{

HANDLE hin, hout, fh;

DWORD len, actlen, default\_mode, mode;

char buffer[300];

INPUT\_RECORD inpbuf;

MOUSE\_EVENT\_RECORD mouse\_record;

int stop = 0;

hin = GetStdHandle(STD\_INPUT\_HANDLE);

if (hin == INVALID\_HANDLE\_VALUE)

{

return 1;

}

hout = GetStdHandle(STD\_OUTPUT\_HANDLE);

if (hin == INVALID\_HANDLE\_VALUE)

{

return 1;

}

fh = CreateFile("file", GENERIC\_READ, 0, 0, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, 0);

if (fh == INVALID\_HANDLE\_VALUE)

{

return 1;

}

ReadFile(fh, buffer, 300, &actlen, NULL);

CloseHandle(fh);

WriteFile(hout, buffer, actlen, &actlen, NULL);

GetConsoleMode(hin, &default\_mode);

mode = ENABLE\_EXTENDED\_FLAGS | ENABLE\_WINDOW\_INPUT | ENABLE\_MOUSE\_INPUT;

SetConsoleMode(hin, mode);

while (!stop)

{

ReadConsoleInputW(hin, &inpbuf, 1, &actlen);

switch (inpbuf.EventType)

{

case MOUSE\_EVENT:

mouse\_record = inpbuf.Event.MouseEvent;

if (mouse\_record.dwButtonState == RIGHTMOST\_BUTTON\_PRESSED)

{

stop = 1;

}

if (mouse\_record.dwButtonState == FROM\_LEFT\_1ST\_BUTTON\_PRESSED)

{

char c;

ReadConsoleOutputCharacter(hout, &c, 1, mouse\_record.dwMousePosition, &actlen);

if (actlen && c != ' ')

{

len = sprintf(buffer, "Character: %c, line: %d, column: %d", c, mouse\_record.dwMousePosition.Y, mouse\_record.dwMousePosition.X);

display\_at\_bottom(hout, buffer, len);

}

}

break;

}

}

SetConsoleMode(hin, default\_mode);

}

void display\_at\_bottom(HANDLE h, char \*text, DWORD len)

{

CONSOLE\_SCREEN\_BUFFER\_INFO csbi;

COORD pos, temp\_pos;

DWORD actlen;

GetConsoleScreenBufferInfo(h, &csbi);

pos.X = 0;

pos.Y = csbi.srWindow.Bottom;

FillConsoleOutputCharacter(h, ' ', csbi.srWindow.Right, pos, &actlen);

WriteConsoleOutputCharacter(h, text, len, pos, &actlen);

if (actlen == 0)

{

printf("No output in bottom line\n");

}

}