МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«Омский государственный технический университет»

**Лабораторная работа №5**

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

|  |  |
| --- | --- |
| Студента | Передериной Софьи Владимировны |
|  | фамилия, имя, отчество полностью |
| Курс | 2 Группа ФИТ-242 |
| Направление | 02.03.02 Фундаментальная информатика |
|  | и информационные технологии |
|  | код, наименование |
|  |  |

Омск-2025

**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.

Усложненный вариант требует вместо информации о указанном символе в месте щелчка мышью отобразить этот символ на своем месте, но в другом регистре (строчные буквы заменить на прописные и наоборот), одновременно перекрашивая указанные мышью символы в яркий цвет (ярко-красный или желтый). Координаты места выводить не нужно. (Дополнительное задание для максимального рейтинга).

**Базовый вариант задания**

#include <windows.h>

#include <stdio.h>

#include <string.h>

#define *MAX\_COUNT\_LINES* 1024

*int* *main*(){

    HANDLE input\_handle = *GetStdHandle*(*STD\_INPUT\_HANDLE*);

    HANDLE output\_handle = *GetStdHandle*(*STD\_OUTPUT\_HANDLE*);

    DWORD current\_console\_mode;

    INPUT\_RECORD input\_record;

*GetConsoleMode*(input\_handle, &current\_console\_mode);

*SetConsoleMode*(input\_handle, ~*ENABLE\_QUICK\_EDIT\_MODE* & (*ENABLE\_MOUSE\_INPUT* | *ENABLE\_EXTENDED\_FLAGS*));

    FILE \*file = *fopen*("input.txt", "r");

    if (!file){

*printf*("Opening file is failed\n");

        return 1;

    }

*char* line\_buffer[1024];

*int* line\_count = 0;

*int* line\_lengths[*MAX\_COUNT\_LINES*] = {0};

    while (*fgets*(line\_buffer, sizeof(line\_buffer), file)){

*int* curr\_len = (*int*)*strlen*(line\_buffer);

        if (line\_buffer[curr\_len - 1] *==* '\n'){

            line\_buffer[curr\_len-1] = '\0';

            curr\_len--;

        }

        line\_lengths[line\_count] = curr\_len;

*printf*("%s\n", line\_buffer);

        line\_count++;

    }

*fclose*(file);

    CONSOLE\_SCREEN\_BUFFER\_INFO console\_inf;

*GetConsoleScreenBufferInfo*(output\_handle, &console\_inf);

*int* message\_line = console\_inf.dwCursorPosition.Y + 1;

    DWORD count;

    while (1){

*ReadConsoleInput*(input\_handle, &input\_record, 1, &count);

        if (input\_record.EventType *!=* *MOUSE\_EVENT*) continue;

        MOUSE\_EVENT\_RECORD mouse\_event = input\_record.Event.MouseEvent;

        if (mouse\_event.dwButtonState *==* *FROM\_LEFT\_1ST\_BUTTON\_PRESSED*){

*char* symbol;

            COORD position = mouse\_event.dwMousePosition;

            if (position.Y *<* line\_count && position.X *<* line\_lengths[position.Y]){

                DWORD read;

                if (*ReadConsoleOutputCharacter*(output\_handle, &symbol, 1, position, &read)

                        && symbol *!=* ' '){

                            COORD write\_position = {0, message\_line};

*SetConsoleCursorPosition*(output\_handle, write\_position);

*printf*("Symbol: %c, Row: %d, Column: %d\n", symbol, position.Y, position.X);

                            message\_line++;

                        }

            }

        }

        if (mouse\_event.dwButtonState *==* *RIGHTMOST\_BUTTON\_PRESSED*) break;

    }

*SetConsoleMode*(input\_handle, current\_console\_mode);

    return 0;

}

Изображение выглядит как текст, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным.

Рисунок 1. Пример запуска программы

С помощью define было объявлено максимальное количество строк, которые возможно загрузить из файла. Для получения хэндлов была использована функция GetStdHandle. Для настройки режима консоли для работы с мышью была использована функция GetConsoleMode для сохранения информации о текущем режиме консоли в переменную current\_console\_mode. С помощью SetConsoleMode были установлены настройки для работы с мышью (флаги ~Enable\_Quick\_Edit\_Mode, Enable\_Mouse\_Input b Enable\_Extended\_Flags*).* Через функцию fopen был открыт текстовый файл input.txt, с помощью fgets этот файл был прочитан построчно, для каждой строки была определена ее длина без учета управляющих символов. С помощью функции GetConsoleScreenBufferInfo была получена информация о текущем положении курсора в консоли, после этого была определена первая свободная строка для вывода информации о кликах мыши. Для получения информации о событии в виде клика мыши была использована функция ReadConsoleInput. В случае клика на левую клавишу информация о позиции клика курсора была получена с помощью поля dwMousePosition переменной mouse\_event. При правом клике происходит выход из цикла. С помощью функции SetConsoleMode консоль возвращается в первоначальное состояние.

**Дополнительное задание**

#include <windows.h>

#include <stdio.h>

#include <string.h>

#define MAX\_COUNT\_LINES 1024

int main(){

HANDLE input\_handle = GetStdHandle(STD\_INPUT\_HANDLE);

HANDLE output\_handle = GetStdHandle(STD\_OUTPUT\_HANDLE);

DWORD current\_console\_mode;

INPUT\_RECORD input\_record;

GetConsoleMode(input\_handle, &current\_console\_mode);

SetConsoleMode(input\_handle, ~ENABLE\_QUICK\_EDIT\_MODE & (ENABLE\_MOUSE\_INPUT | ENABLE\_EXTENDED\_FLAGS));

FILE \*file = fopen("input.txt", "r");

if (!file){

printf("Opening file is failed\n");

return 1;

}

char line\_buffer[1024];

int line\_count = 0;

int line\_lengths[MAX\_COUNT\_LINES] = {0};

while (fgets(line\_buffer, sizeof(line\_buffer), file)){

int curr\_len = (int)strlen(line\_buffer);

if (line\_buffer[curr\_len - 1] == '\n'){

line\_buffer[curr\_len-1] = '\0';

curr\_len--;

}

line\_lengths[line\_count] = curr\_len;

printf("%s\n", line\_buffer);

line\_count++;

}

fclose(file);

CONSOLE\_SCREEN\_BUFFER\_INFO console\_inf;

GetConsoleScreenBufferInfo(output\_handle, &console\_inf);

int message\_line = console\_inf.dwCursorPosition.Y + 1;

DWORD count;

while (1){

if (ReadConsoleInput(input\_handle, &input\_record, 1, &count)){

if (input\_record.EventType != MOUSE\_EVENT) continue;

MOUSE\_EVENT\_RECORD mouse\_event = input\_record.Event.MouseEvent;

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

char symbol;

COORD position = mouse\_event.dwMousePosition;

if (position.Y < line\_count && position.X < line\_lengths[position.Y]){

DWORD read;

if (ReadConsoleOutputCharacter(output\_handle, &symbol, 1, position, &read)

&& symbol != ' '){

if (islower(symbol)) symbol = toupper(symbol);

else if (isupper(symbol)) symbol = tolower(symbol);

SetConsoleTextAttribute(output\_handle, FOREGROUND\_INTENSITY | FOREGROUND\_RED);

SetConsoleCursorPosition(output\_handle, position);

DWORD written;

WriteConsole(output\_handle, &symbol, 1, &written, NULL);

SetConsoleTextAttribute(output\_handle, FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_BLUE);

}

}

}

if (mouse\_event.dwButtonState == RIGHTMOST\_BUTTON\_PRESSED) break;

}

}

SetConsoleMode(input\_handle, current\_console\_mode);

return 0;

}

При левом клике мыши с помощью функции ReadConsoleOutputCharacter соответствующий символ сохраняется в переменную symbol. Его регистр меняется на противоположный. С помощью SetConsoleTextAttribute меняется цвет текста на красный, и с помощью WriteConsole символ выводится на консоль.