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 | Абрамова Егора Владимировича |
|  | фамилия, имя, отчество полностью |
| Курс | 1 Группа ФИТ-221 |
| Направление | 02.03.02 Фундаментальная информатика |
|  | и информационные технологии |
|  | код, наименование |
| Руководитель | старший преподаватель |
|  | должность, ученая степень, звание |
|  | Карабцов Р. Д. |
|  | фамилия, инициалы |
| Выполнил | 8.02.2024 |
|  | дата, подпись студента |
| баллы |  |
|  | дата, подпись руководителя |

Омск-2024

**Задание**

Разработать в Windows программу, которая получает хэндлы стандартного ввода и вывода, выводит числовые с комментариями значения этих хэндлов, затем, используя стандартный ввод системными функциями небуферизованного ввода-вывода ReadFile или WrirwFile, делает приглашение для ввода, вводит любой текст и выводит его с предуведомлением, что он предварительно введен в программу. Предуведомление должно представлять вывод поясняющего текста с помощью отдельного оператора на основе вызова функции системного вывода. Продемонстрировать работу программы, запуская ее как с использованием стандартного ввода вывода по умолчанию, так и с переназначением этого ввода на файл для ввода исходных данных и вывода данных вместо экрана. (Базовый вариант задания).

Контрольные вопросы:

1. Где в ходе построения исходной программы, использующей стандартный ввод-вывод, записываются файловые дескрипторы или хэндлы, в каких частях или функциях?

2. Зачем используется переадресация стандартного ввода-вывода ?

3. Сколько существует разновидностей стандартного ввода-вывода?

4. Чем отличается вызов исполняемой программы в командной строке ОС Windows от аналогичного вызова исполняемой программы с тем же именем в командной строке Linux

**Решение**

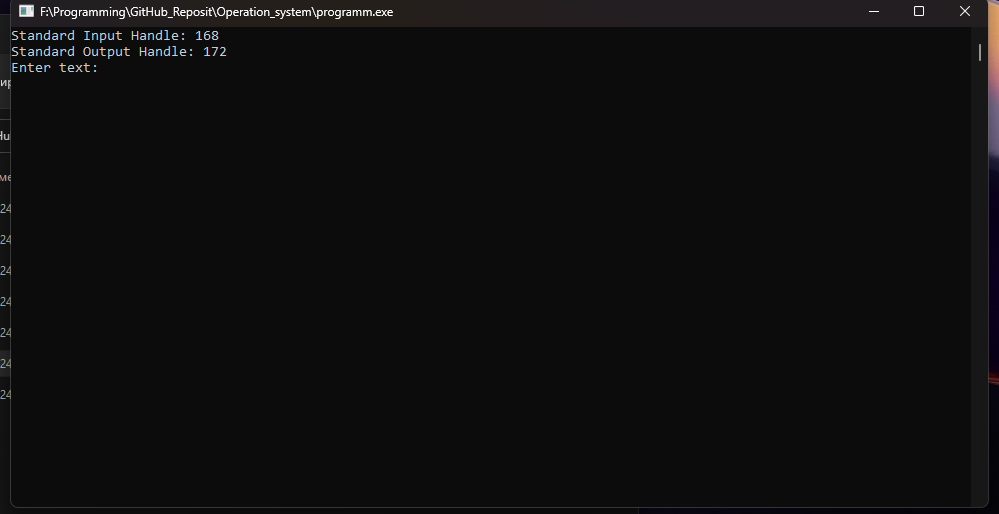


Рисунок 1. Старт работы программы

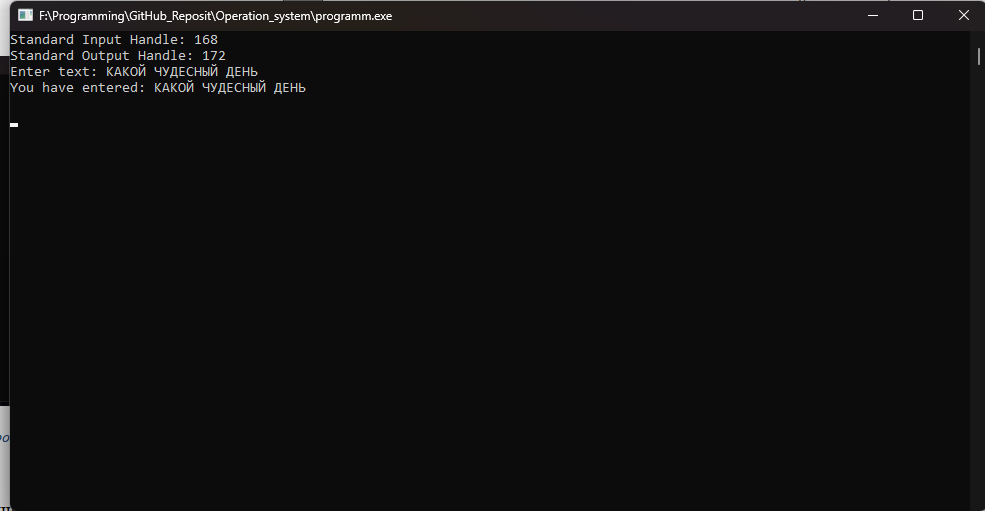


Рисунок 2. Результат работы программы

1. В ходе построения программы, использующей стандартный ввод-вывод, файловые дескрипторы или хэндлы записываются в функциях, которые работают с файлами или устройствами ввода-вывода. Например, в Windows для получения хэндлов стандартного ввода и вывода используется функция GetStdHandle.

2. Переадресация стандартного ввода-вывода используется для изменения источника ввода или места вывода программы. Это позволяет программе работать с данными из файлов или других программ, а также сохранять вывод в файлы.

3. Существует три разновидности стандартного ввода-вывода:

- Стандартный ввод (stdin) - поток для ввода данных в программу.

- Стандартный вывод (stdout) - поток для вывода данных из программы.

- Стандартный поток ошибок (stderr) - поток для вывода сообщений об ошибках.

4. Вызов исполняемой программы в командной строке ОС Windows отличается от аналогичного вызова в командной строке Linux прежде всего тем, что в Windows используются обратные слэши \ для указания пути к исполняемому файлу и аргументам, в то время как в Linux используются прямые слэши /. Кроме того, в Windows для перенаправления стандартного ввода-вывода используются операторы > и <, а в Linux операторы |, >, <.

**Листинг программы**

#include <windows.h>

#include <stdio.h>

int main()

{

    HANDLE hstdin = GetStdHandle(STD\_INPUT\_HANDLE);

    HANDLE hstdout = GetStdHandle(STD\_OUTPUT\_HANDLE);

    if (hstdin == INVALID\_HANDLE\_VALUE || hstdout == INVALID\_HANDLE\_VALUE)

    {

        fprintf(stderr, "Error handler\n");

        return 1;

    }

    printf("Standard Input Handle: %lu\n", hstdin);

    printf("Standard Output Handle: %lu\n", hstdout);

    printf("Enter text: ");

    char buffer[100];

    DWORD bytesRead;

    if (ReadFile(hstdin, buffer, sizeof(buffer), &bytesRead, NULL))

    {

        printf("You have entered: %.\*s\n", bytesRead, buffer);

    }

    else

    {

        fprintf(stderr, "Failed to read input\n");

        return 1;

    }

    getchar();

    return 0;

}