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 | Зензина Егора Николаевича |
|  | фамилия, имя, отчество полностью |
| Курс | 2 Группа ФИТ-222 |
| Направление | 02.03.02 Фундаментальная информатика |
|  | и информационные технологии |
|  | код, наименование |
| Руководитель | старший преподаватель |
|  | должность, ученая степень, звание |
|  | Карабцов Р.Д. |
|  | фамилия, инициалы |
| Выполнил | 03.05.2024 |
|  | 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**

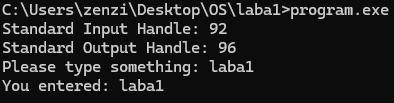


Рисунок 1. Ввод пользователя в Windows

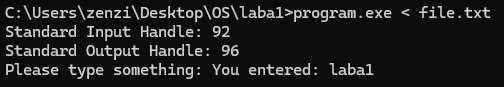


Рисунок 2. Перенаправленный ввод в Windows

1. В программах, работающих со стандартным вводом-выводом, файловые дескрипторы или хэндлы обычно находятся и используются внутри функций, отвечающих за операции ввода-вывода. Например, в Windows для получения хэндлов стандартных потоков используется функция GetStdHandle. Эти хэндлы затем передаются другим функциям, таким как ReadFile и WriteFile, для чтения и записи данных.

2. Перенаправление стандартного ввода-вывода изменяет источник ввода или место вывода программы, что позволяет ей работать с данными из файлов или других программ, а также сохранять результаты в файлы.

3. В программировании существует три основных вида стандартного ввода-вывода: стандартный ввод (stdin), который предназначен для подачи данных в программу, обычно с клавиатуры; стандартный вывод (stdout), служащий для вывода данных из программы, обычно на экран; и стандартный поток ошибок (stderr), предназначенный для вывода сообщений об ошибках, обычно на экран, но отдельно от обычного вывода.

4. Основное отличие в вызове программ между Windows и Linux заключается в способе указания пути к исполняемому файлу и перенаправлении потоков ввода-вывода. В Windows для путей используются обратные слэши (), а для перенаправления – символы ">" и "<". В Linux же применяются прямые слэши (/) для путей и символы "|", ">", "<" для перенаправления, предоставляя больше возможностей для комбинирования команд.

**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 accessing standard handles\n");

        return 1;

    }

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

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

    printf("Please type something: ");

    char buffer[100];

    DWORD bytesRead;

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

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

    } else {

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

        return 1;

    }

    getchar();

    return 0;

}