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**

Результат выполнения лабораторной работы должен состоять из двух программ для Linux. Первая программа должна создавать текстовый файл, вводя данные со стандартного ввода. (Более детально: открывает файл для записи, читает текст со стандартного ввода и выводит этот прочитанный текст в файл.) Вторая программа открывает тот же файл (созданный перед этим другой программой) для чтения и хэндл, полученный при этом открытии, запоминает в 1-й переменной для хэндла. Используя этот хэндл, далее с помощью функции dup() получается новое значение хэндла для доступа к тому же файлу (2-й хэндл). Еще раз открывается тот же файл, запоминая 3-е значение хэндла. С помощью первого хэндла программа позиционирует чтение для 10-й позиции файла от начала этого файла. Далее программа должна выводить числовые значения всех трех хэндлов на экран. Используя по очереди все 3 хэндла, из файла читаются по 7 символов и тут же эти три прочитанных текста выводятся на экран, каждая в своей строке. Результаты вывода объяснить

**Контрольные вопросы:**

1.Чем открытый файл в операционной системе отличается от неоткрытого, обосновать ответ.

2. Зачем закрывается файл, что происходит, если программист не указал явную функцию закрытия файла?

3. Что содержится в управляющем блоке файла?

4. Где относительно остальной части программы находится управляющий блок файла, используемый в этой?

**Решение**



Рисунок 1. Запись 1ой программы



Рисунок 2. Запись в файле

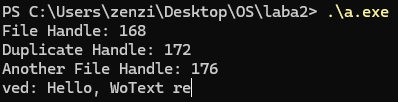


Рисунок 3. Результат 2ой программы

1) Открытый файл взаимодействует с программой, позволяя ей читать, записывать и изменять данные. Неоткрытый файл недоступен программе и не может быть использован для чтения или записи.

2) Закрытие файла необходимо для освобождения ресурсов и сохранения внесенных изменений. Если программист забывает явно закрыть файл, это может привести к утечке ресурсов или потере данных из-за несохранения.

3) Управляющий блок файла хранит в себе информацию о файле, такую как его название, объем, текущее местоположение в файле, права доступа и другие данные, которые необходимы для его обработки.

4) Обычно информация из управляющего блока файла хранится в оперативной памяти компьютера, и к ней нет прямого доступа из программы.

**Листинг программы**

First.c:

#include <windows.h>

int main() {

    char text[100] = "Text received: ";

    DWORD length, bytesRead;

    HANDLE hStdin, fileHandle;

    char filename[] = "output.txt";

    BOOL success;

    length = strlen(text);

    hStdin = GetStdHandle(STD\_INPUT\_HANDLE);

    if (hStdin == INVALID\_HANDLE\_VALUE) return 1;

    fileHandle = CreateFile(filename, GENERIC\_WRITE, 0, NULL, CREATE\_ALWAYS, FILE\_ATTRIBUTE\_NORMAL, NULL);

    if (fileHandle == INVALID\_HANDLE\_VALUE) return 1;

    success = ReadFile(hStdin, text + length, 80, &bytesRead, NULL);

    if (!success) return 1;

    WriteFile(fileHandle, text, length + bytesRead, &bytesRead, NULL);

    CloseHandle(fileHandle);

    return 0;

}

Second.c:

#include <windows.h>

#include <stdio.h>

int main() {

    char buffer[100] = "";

    DWORD bytesRead;

    HANDLE hStdout, fileHandle, duplicateHandle, anotherFileHandle;

    char filename[] = "output.txt";

    BOOL rc;

    hStdout = GetStdHandle(STD\_OUTPUT\_HANDLE);

    if (hStdout == INVALID\_HANDLE\_VALUE)

        return 1;

    fileHandle = CreateFile(filename, GENERIC\_READ, FILE\_SHARE\_READ, NULL, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, NULL);

    if (fileHandle == INVALID\_HANDLE\_VALUE)

        return 1;

    DuplicateHandle(GetCurrentProcess(), fileHandle, GetCurrentProcess(), &duplicateHandle, 0, FALSE, DUPLICATE\_SAME\_ACCESS);

    anotherFileHandle = CreateFile(filename, GENERIC\_READ, FILE\_SHARE\_READ, NULL, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, NULL);

    if (anotherFileHandle == INVALID\_HANDLE\_VALUE)

        return 1;

    printf("File Handle: %lu\nDuplicate Handle: %lu\nAnother File Handle: %lu\n", (unsigned long)fileHandle, (unsigned long)duplicateHandle, (unsigned long)anotherFileHandle);

    rc = SetFilePointer(fileHandle, 10, NULL, FILE\_BEGIN);

    rc = ReadFile(fileHandle, buffer, 7, &bytesRead, NULL);

    if (!rc)

        return 1;

    WriteFile(hStdout, buffer, bytesRead, &bytesRead, NULL);

    rc = ReadFile(duplicateHandle, buffer, 7, &bytesRead, NULL);

    if (!rc)

        return 1;

    WriteFile(hStdout, buffer, bytesRead, &bytesRead, NULL);

    rc = ReadFile(anotherFileHandle, buffer, 7, &bytesRead, NULL);

    if (!rc)

        return 1;

    WriteFile(hStdout, buffer, bytesRead, &bytesRead, NULL);

    CloseHandle(fileHandle);

    CloseHandle(duplicateHandle);

    CloseHandle(anotherFileHandle);

    getchar();

    return 0;

}