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**

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.

Выполнить то же задание для операционной системы Windows, используя для дублирования хэндла файла системную функцию DublicateHandle, вместо dup в Linux.

**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**

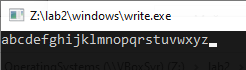


Рисунок 1. Запись 1ой программы

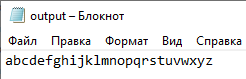


Рисунок 2. Запись в файле

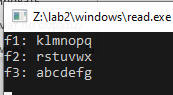


Рисунок 3. Результат 2ой программы

1)Открытый файл в операционной системе отличается от неоткрытого тем, что после открытия программой файла, операционная система создает структуры данных для отслеживания состояния этого открытого файла. В управляющем блоке файла (File Control Block, FCB) хранится информация о файле, такая как текущая позиция в файле, режим доступа, указатель на следующую доступную запись и другие параметры. Неоткрытый файл не имеет связанных с ним структур данных и операционная система не отслеживает его состояние.

2)Закрытие файла важно для освобождения ресурсов, связанных с открытым файлом, таких как дескриптор файла. Если программист не указал явную функцию закрытия файла, некоторые операционные системы могут автоматически закрыть файл при завершении программы, но это не гарантировано. Неявное закрытие файла может привести к утечкам ресурсов и другим проблемам.

3)Управляющий блок файла (FCB) содержит различные атрибуты файла, такие как имя файла, расширение, атрибуты доступа, размер файла, дата создания и другие метаданные. Эти данные используются операционной системой для управления файлом.

4)Управляющий блок файла, как правило, находится в системных структурах данных операционной системы. Он не является частью программы пользователя, а скорее представляет собой структуру, управляемую операционной системой для каждого открытого файла в системе. Он может находиться в ядре операционной системы или в специальной области памяти, управляемой файловой системой.

**Листинг программы**

read\_from\_file.c

#include <stdio.h>

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

#include <windows.h>

int main()

{

DWORD len;

HANDLE f1, f2, f3, hout;

char buffer[100];

char \*b = buffer;

f1 = CreateFile(

"output", GENERIC\_READ, FILE\_SHARE\_READ, 0,

OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, 0

);

DuplicateHandle(

GetCurrentProcess(), f1, GetCurrentProcess(), &f2,

0, FALSE, DUPLICATE\_SAME\_ACCESS

);

f3 = CreateFile(

"output", GENERIC\_READ, FILE\_SHARE\_READ, 0,

OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, 0

);

hout = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetFilePointer(f1, 10, 0, FILE\_BEGIN);

printf("f1: ");

fflush(stdout);

ReadFile(f1, buffer, 7, &len, NULL);

WriteFile(hout, buffer, 7, NULL, NULL);

putchar('\n');

printf("f2: ");

fflush(stdout);

ReadFile(f2, buffer+7, 7, &len, NULL);

WriteFile(hout, buffer+7, 7, NULL, NULL);

putchar('\n');

printf("f3: ");

fflush(stdout);

ReadFile(f3, buffer+14, 7, &len, NULL);

WriteFile(hout, buffer+14, 7, NULL, NULL);

putchar('\n');

getchar();

}

write\_to\_file.c

#include <stdio.h>

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

#include <windows.h>

int main()

{

char buffer[100];

DWORD actlen;

HANDLE hin, f\_handle;

BOOL rc;

hin = GetStdHandle(STD\_INPUT\_HANDLE);

if (hin == INVALID\_HANDLE\_VALUE)

return 1;

f\_handle = CreateFile(

"output", GENERIC\_WRITE, 0, 0,

CREATE\_ALWAYS, FILE\_ATTRIBUTE\_NORMAL, 0

);

if (f\_handle == INVALID\_HANDLE\_VALUE)

return 1;

rc = ReadFile(hin, buffer, 100, &actlen, NULL);

if (!rc)

return 1;

WriteFile(f\_handle, buffer, actlen, NULL, NULL);

CloseHandle(f\_handle);

}