Минобрнауки России  
Федеральное государственное автономное образовательное  
Учреждение высшего образования  
«Санкт-Петербургский государственный электротехнический  
Университет им. В.И. Ульянова (Ленина)»  
(СПГЭТУ «ЛЭТИ»)  
Факультет компьютерных технологий и информатики

Кафедра вычислительной техники  
  
  
  
**Отчет по лабораторной работе №1**

**на тему: «Управление файловой системой»**

**по дисциплине «Операционные системы»**

Выполнил студент группы 9308: Семенов А.И.

Принял: к.т.н., доцент Тимофеев А.В.

Санкт-Петербург  
2021 г.

Оглавление

[Цель работы 3](#_Toc84108923)

[Управление дисками, каталогами и файлами 3](#_Toc84108924)

[Указания к выполнению 3](#_Toc84108925)

[Примеры работы программы 4](#_Toc84108926)

[Копирование файла с помощью операций перекрывающегося ввода-вывода 11](#_Toc84108927)

[Указания к выполнению 11](#_Toc84108928)

[Примеры выполнения программы 12](#_Toc84108929)

[Измерения времени на копирование и анализ 12](#_Toc84108930)

[Вывод 17](#_Toc84108931)

# Цель работы

Исследовать управление файловой системой с помощью Win32 API.

# Управление дисками, каталогами и файлами

## Указания к выполнению

Создайте консольное приложение с меню (каждая выполняемая функция и/или операция должна быть доступна по отдельному пункту меню), которое выполняет:

* вывод списка дисков (функции Win32 API – GetLogicalDrives, GetLogicalDriveStrings);
* для одного из выбранных дисков вывод информации о диске и размер свободного пространства (функции Win32 API – GetDriveType, GetVolumeInformation, GetDiskFreeSpace);
* создание и удаление заданных каталогов (функции Win32 API CreateDirectory, RemoveDirectory);
* создание файлов в новых каталогах (функция Win32 API – CreateFile)
* копирование и перемещение файлов между каталогами с возможностью выявления попытки работы с файлами, имеющими совпадающие имена (функции Win32 API – CopyFile, MoveFile, MoveFileEx);
* анализ и изменение атрибутов файлов (функции Win32 API – GetFileAttributes, SetFileAttributes, GetFileInformationByHandle, GetFileTime, SetFileTime).

## Примеры работы программы

При запуске программы представляется выбор из 2 пунктов, которые

отражают 2 пункта задания (рисунок 1). Для текущего пункта выбираем 1-й

пункт меню.



Рисунок . Главное меню

При открытии 1-го пункта главного меню открывается подменю, где представлены доступные действия (рисунок 2).

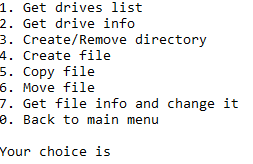


Рисунок . Меню 1-го задания

1. Получение списка дисков (рисунок 3). Используемые функции: GetLogicalDrives, GetLogicalDriveStrings.



Рисунок . Список дисков

2. Получение информации о каком-либо диске (рисунок 4). Для примера был выбран диск А. Используемые функции: GetDriveType, GetVolumeInformation, GetDiskFreeSpace.

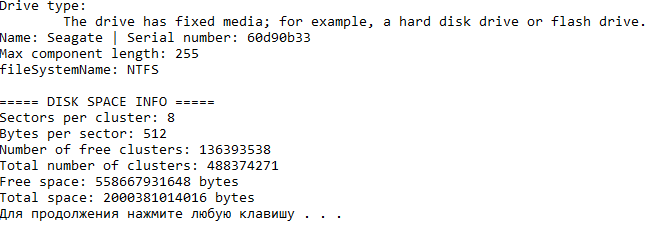


Рисунок . Информация о диске

3. Создание/удаление директории. При выборе этого пункта появляется подменю (рисунок 5), где можно выбрать либо создание директории, либо ее удаление.



Рисунок . Подменю создания/удаления директории

3.1. Создание директории. Для примера создадим директорию в текущей папке с названием “testDir”. Итог создания продемонстрирован на рисунке 7. Используемая функция: CreateDirectory.

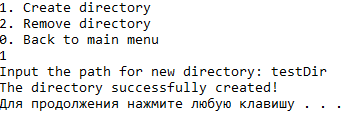


Рисунок . Пример создания директории в программе

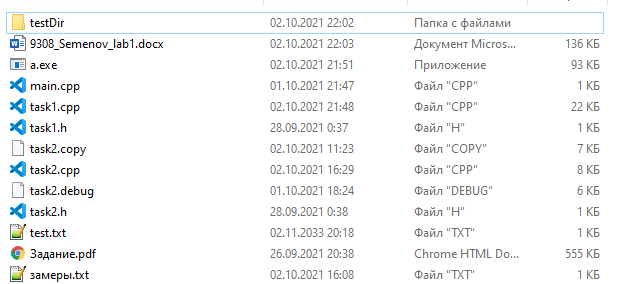


Рисунок . Созданная директория

3.2. Удаление директории. Удалим только что созданную директорию. Используемая функция: RemoveDirectory.

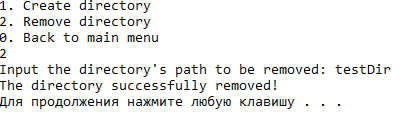


Рисунок . Пример удаления директории в программе

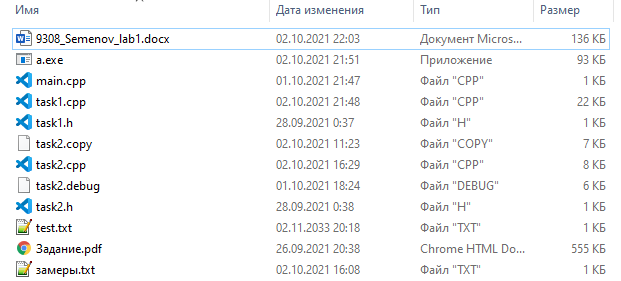


Рисунок . Итог удаления директории

4. Создание файла. Используемая функция: CreateFile

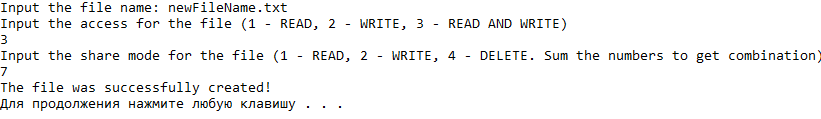


Рисунок . Создание файла

5. Копирование файла. Копируется файл, созданный в прошлом пункте. Перед копированием в файл был написан текст “some text here”. Используемая функция: CopyFile.

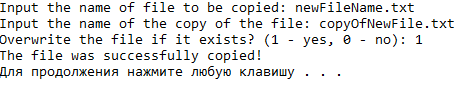


Рисунок . Копирование файла

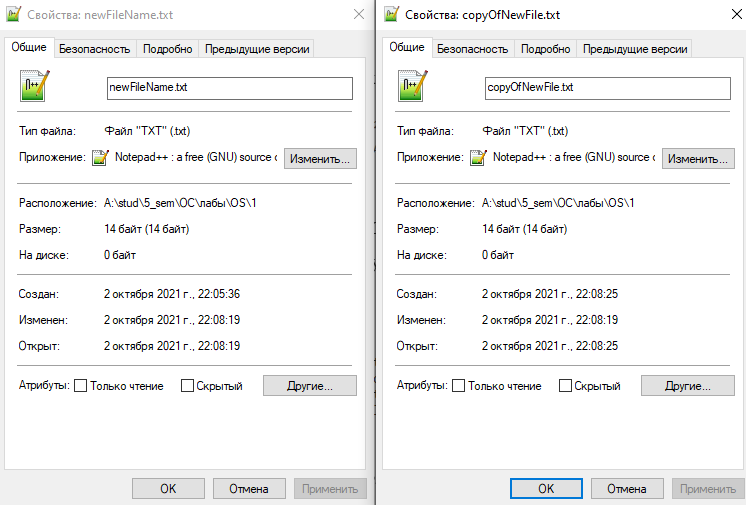


Рисунок . Свойства скопированного файла и копии

6. Перемещение файла. Переместим копию файла в предварительно созданную директорию testDir. Используемые функции: MoveFile, MoveFileEx.

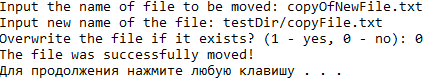


Рисунок . Перемещение файла в программе

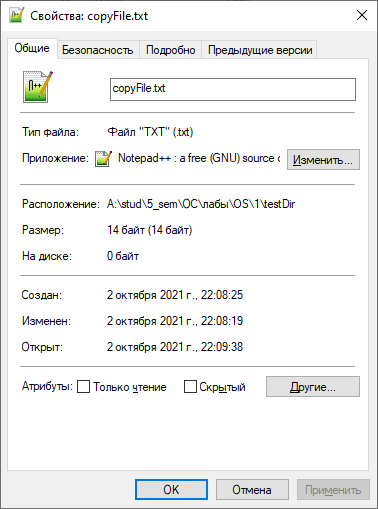


Рисунок . Свойства перемещенного файла

7. Получение атрибутов файла и их изменение. Используемые функции: GetFileAttributes, SetFileAttributes, GetFileInformationByHandle, GetFileTime, SetFileTime.

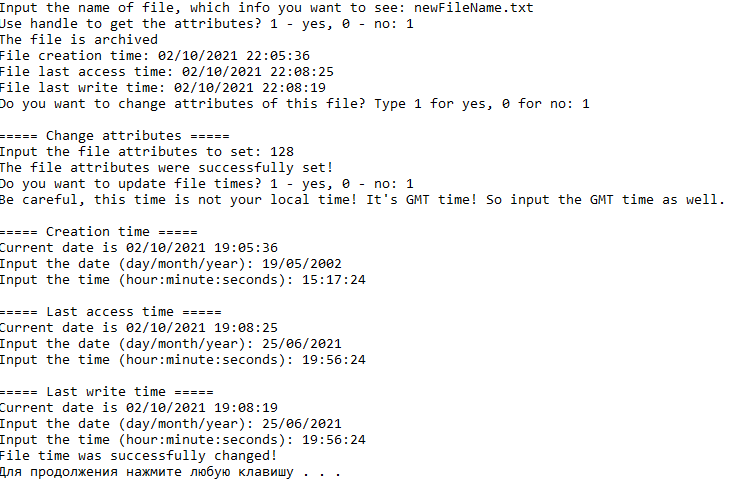


Рисунок . Получение и установление атрибутов файла

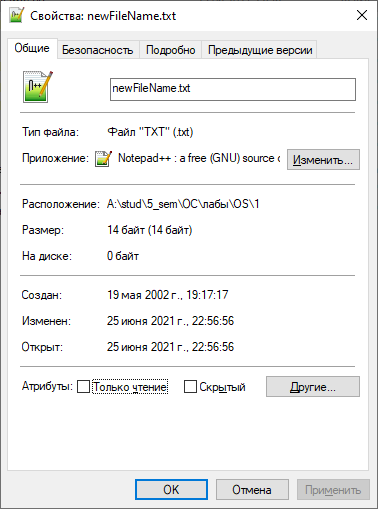


Рисунок . Свойства измененного файла

**Вывод**

Были изучены различные функции, позволяющие работать с файловой системой операционной системы Windows с помощью Win32 API. Данные функции открывают доступ к получению различной информации о носителях данных и о самих данных, к изменению этих самых данных, включая атрибуты файла.

# Копирование файла с помощью операций перекрывающегося ввода-вывода

## Указания к выполнению

Создайте консольное приложение, которое выполняет:

* открытие/создание файлов (функция Win32 API – CreateFile, обязательно использовать флаги FILE\_FLAG\_NO\_BUFFERING и FILE\_FLAG\_OVERLAPPED);
* файловый ввод-вывод (функции Win32 API – ReadFileEx, WriteFileEx) блоками кратными размеру кластера;
* ожидание срабатывания вызова функции завершения (функция Win32 API – SleepEx);
* измерение продолжительности выполнения операции копирования файла (функция Win32 API – TimeGetTime).

## Примеры выполнения программы

Для выполнения копирования был выбран видео-файл размером в 4.17 Гбайт. В качестве примера были выбраны буфер размера 512\*32 байт и количество операций, равное 8.

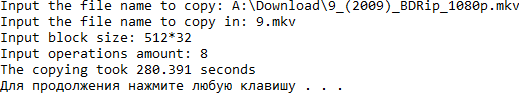


Рисунок . Пример копирования файла

Сравнение оригинального файла и полученной копии подтверждает, что файл был скопирован полностью и без ошибок.



Рисунок . Сравнение копируемого файла и копии

## Измерения времени на копирование и анализ

Все операции копирования были проведены на диске Seagate ST200DM008. Некоторые характеристики диска представлены в таблице 1.

Таблица 1. Характеристики диска

|  |  |
| --- | --- |
| Интерфейс | SATA III |
| Объем накопителя | 2048 ГБ |
| Скорость вращения шпинделя | 7200 об/мин |
| Буферная память | 64 МБ |
| Форм-фактор накопителя (физический) | 3.5 |

Результаты замеров копирования представлены в таблицах 2 и 3. Таблица 2 отражает изменение времени от изменения количества операций перекрывающего ввода-вывода при одном размере буфера – 512\*128 байт.

Таблица 2. Результаты замера при одном размере буфера.

|  |  |
| --- | --- |
| Затраченное время, с | Количество операций |
| 280.25 | 1 |
| 231.593 | 2 |
| 182.765 | 4 |
| 135.39 | 8 |
| 133.969 | 16 |

Таблица 3 отражает изменение времени от изменения размера буфера при одном количестве операция перекрывающего ввода-вывода – 8.

Таблица 3. Результаты замера при одинаковом количестве операций.

|  |  |
| --- | --- |
| Затраченное время, с | Размер буфера, байт |
| 280.391 | 512\*32 |
| 135.39 | 512\*128 |
| 118.047 | 512\*256 |
| 106.703 | 512\*512 |
| 89.75 | 512\*1024 |
| 88.64 | 512\*2048 |
| 87.657 | 512\*4096 |

Графики, отражающие таблицы 2 и 3 приведены на рисунках

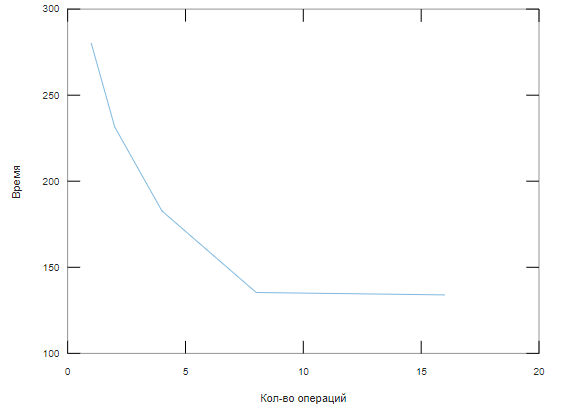


Рисунок . Зависимость времени от количества

операций при одном размере буфера

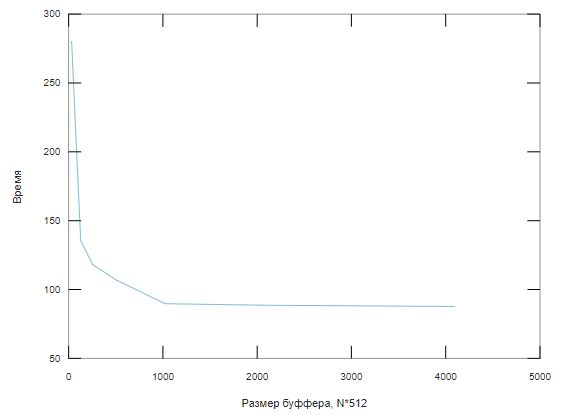


Рисунок . Зависимость времени от изменения размера

буфера при одинаковом числе операций

На данном этапе предполагается следующее: при данной реализации и выбранном диске оптимальное количество операций – 8 (при 16 операциях разница во времени не столь существенна), оптимальный размер буфера – 512\*1024 байт (при большем значении разница во времени не столь существенна).

Для подтверждения предположения были проведены дополнительные замеры на других файлах. Результаты замеров приведены в таблице 4.

Таблица 4. Результаты дополнительных замеров

|  |  |  |  |
| --- | --- | --- | --- |
| Размер файла | Размер буфера, байт | Кол-во операций | Затраченное время, с |
| 1.17 МБ | 512 | 1 | 21.937 |
| 1.17 МБ | 512 | 8 | 1.328 |
| 1.17 МБ | 512 | 16 | 1.11 |
| 1.17 МБ | 512\*32 | 1 | 0.094 |
| 1.17 МБ | 512\*32 | 8 | 0.031 |
| 1.16 ГБ | 512\*32 | 1 | 273.812 |
| 1.16 ГБ | 512\*32 | 8 | 95.954 |
| 1.16 ГБ | 512\*32 | 16 | 89.828 |
| 1.16 ГБ | 512\*64 | 8 | 59.25 |
| 1.16 ГБ | 512\*512 | 8 | 29.14 |
| 1.16 ГБ | 512\*1024 | 8 | 23.625 |
| 1.16 ГБ | 512\*2048 | 8 | 23.418 |
| 1.16 ГБ | 512\*2048 | 16 | 20.891 |

По приведенным данным предположение об оптимальном размере буфера и количестве операций на практике подтвердилось: при количестве операций свыше 8 или размере буфера свыше 512\*1024 байт разница во времени стримится к минимальной.

## Вывод

Был изучен механизм асинхронного копирования файла, при котором было использовано переменное количество как самих блоков для копирования, так и размеров этих блоков.

Исходя из полученных данных, можно сделать следующие выводы:

* изменение параметров (размер буфера, количество операций перекрывающего ввода-вывода) сильно сказывается на скорости копирования.
* оптимальное количество операций перекрывающего ввода-вывода – 8 (разница между 8 и 16 операциями не сильно отражаются на времени)