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

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.

```
1. Task one
2. Task two: copy file
0. Exit
```

Рисунок 1. Главное меню

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

```
1. Get drives list
2. Get drive info
3. Create/Remove directory
4. Create file
5. Copy file
6. Move file
7. Get file attributes
8. Set file attributes
9. Get file time
10. Set file time
0. Back to main menu

Your choice is
```

Рисунок 2. Меню 1-го задания

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

```
Drives available:
A(1) C(4) D(8) E(16) F(32)
Для продолжения нажмите любую клавишу . . .
```

Рисунок 3. Список дисков

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

```
Name: Seagate | Serial number: 60d90b33
Max component length: 255
fileSystemName: NTFS

The specified volume:
        - supports preserved case of file names when it places a name on disk
        - supports case-sensitive file names
        - supports file-based compression
        - supports named streams
        - preserves and enforces access control lists
        - supports the Encrypted File System
        - supports extended attributes
        - supports hard links
        - supports object identifiers
        - supports open by FileID
        - supports reparse points
        - supports sparse files
        - supports transactions
        - supports update sequence number journals
        - supports Unicode in file names as they appear on disk
        - supports disk quotas

==== DISK SPACE INFO =====
Sectors per cluster: 8
Bytes per sector: 512
Number of free clusters: 137527371
Total number of clusters: 488374271
Free space: 563312111616 bytes
Total space: 2000381014016 bytes
```

Рисунок 4. Информация о диске

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

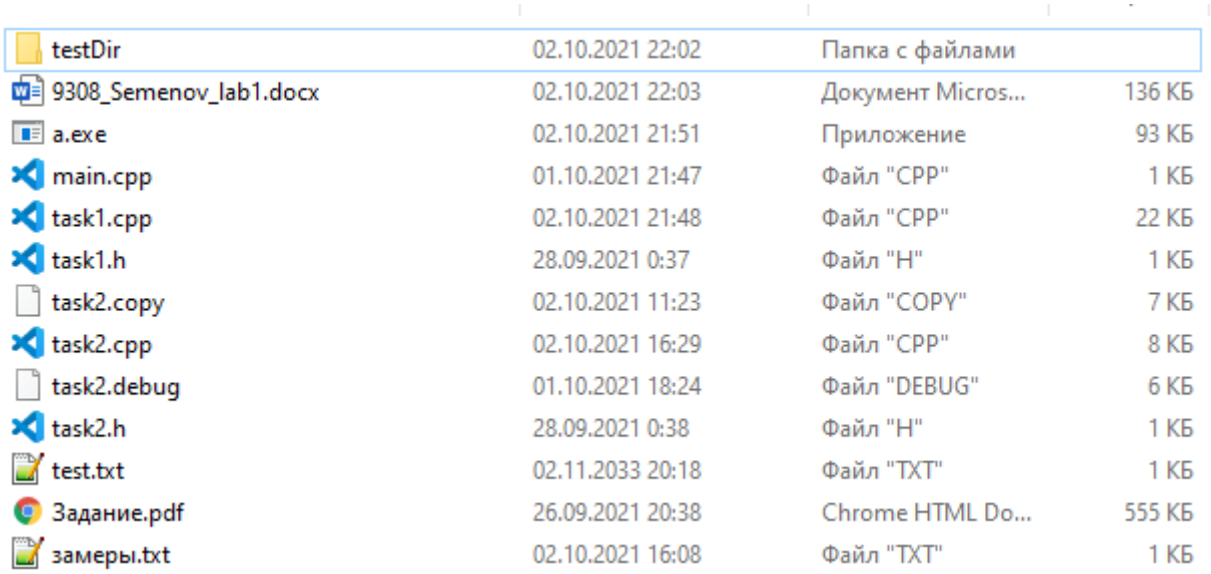
```
1. Create directory
2. Remove directory
0. Back to main menu
```

Рисунок 5. 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.

```
1. Create directory
2. Remove directory
0. Back to main menu
1
Input the path for new directory: testDir
The directory successfully created!
Для продолжения нажмите любую клавишу . . .
```

Рисунок 6. Пример создания директории в программе



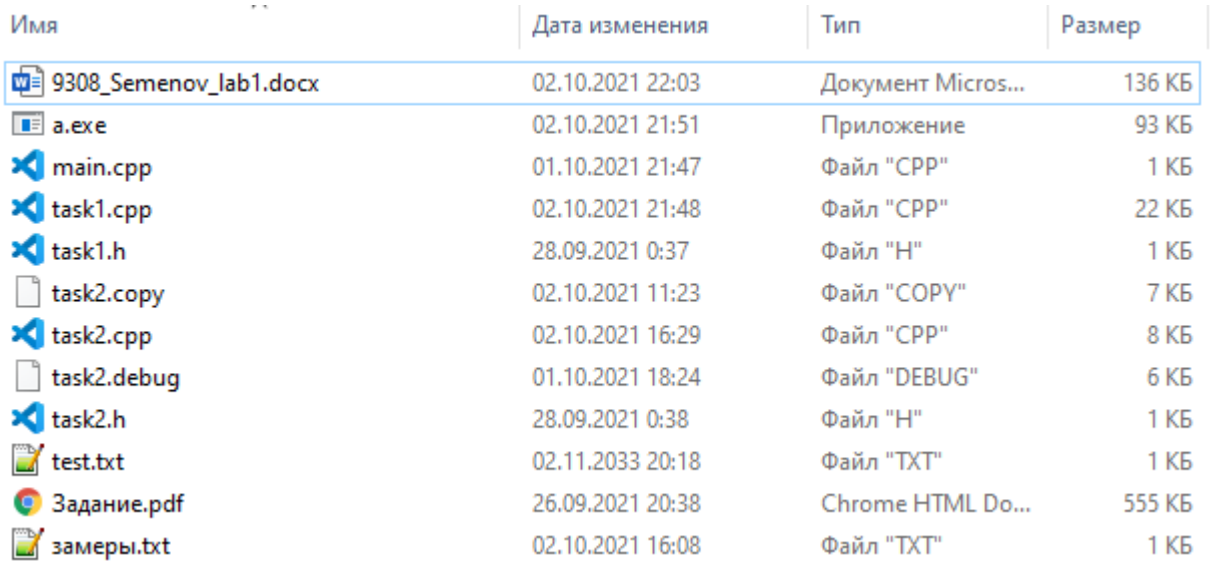
testDir	02.10.2021 22:02	Папка с файлами	
9308_Semenov_lab1.docx	02.10.2021 22:03	Документ Micros...	136 КБ
a.exe	02.10.2021 21:51	Приложение	93 КБ
main.cpp	01.10.2021 21:47	Файл "CPP"	1 КБ
task1.cpp	02.10.2021 21:48	Файл "CPP"	22 КБ
task1.h	28.09.2021 0:37	Файл "H"	1 КБ
task2.copy	02.10.2021 11:23	Файл "COPY"	7 КБ
task2.cpp	02.10.2021 16:29	Файл "CPP"	8 КБ
task2.debug	01.10.2021 18:24	Файл "DEBUG"	6 КБ
task2.h	28.09.2021 0:38	Файл "H"	1 КБ
test.txt	02.11.2033 20:18	Файл "TXT"	1 КБ
Задание.pdf	26.09.2021 20:38	Chrome HTML Do...	555 КБ
замеры.txt	02.10.2021 16:08	Файл "TXT"	1 КБ

Рисунок 7. Созданная директория

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

```
1. Create directory
2. Remove directory
0. Back to main menu
2
Input the directory's path to be removed: testDir
The directory successfully removed!
Для продолжения нажмите любую клавишу . . .
```

Рисунок 8. Пример удаления директории в программе



Имя	Дата изменения	Тип	Размер
9308_Semenov_lab1.docx	02.10.2021 22:03	Документ Micros...	136 КБ
a.exe	02.10.2021 21:51	Приложение	93 КБ
main.cpp	01.10.2021 21:47	Файл "CPP"	1 КБ
task1.cpp	02.10.2021 21:48	Файл "CPP"	22 КБ
task1.h	28.09.2021 0:37	Файл "H"	1 КБ
task2.copy	02.10.2021 11:23	Файл "COPY"	7 КБ
task2.cpp	02.10.2021 16:29	Файл "CPP"	8 КБ
task2.debug	01.10.2021 18:24	Файл "DEBUG"	6 КБ
task2.h	28.09.2021 0:38	Файл "H"	1 КБ
test.txt	02.11.2033 20:18	Файл "TXT"	1 КБ
Задание.pdf	26.09.2021 20:38	Chrome HTML Do...	555 КБ
замеры.txt	02.10.2021 16:08	Файл "TXT"	1 КБ

Рисунок 9. Итог удаления директории

4. Создание файла. Используемая функция: CreateFile

```
Input the file name: newFileName.txt
Input the access for the file (1 - READ, 2 - WRITE, 3 - READ AND WRITE)
3
Input the share mode for the file (1 - READ, 2 - WRITE, 4 - DELETE. Sum the numbers to get combination)
7
The file was successfully created!
Для продолжения нажмите любую клавишу . . .
```

Рисунок 10. Создание файла

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

```
Input the name of file to be copied: newFileName.txt
Input the name of the copy of the file: copyOfNewFile.txt
Overwrite the file if it exists? (1 - yes, 0 - no): 1
The file was successfully copied!
Для продолжения нажмите любую клавишу . . .
```

Рисунок 11. Копирование файла

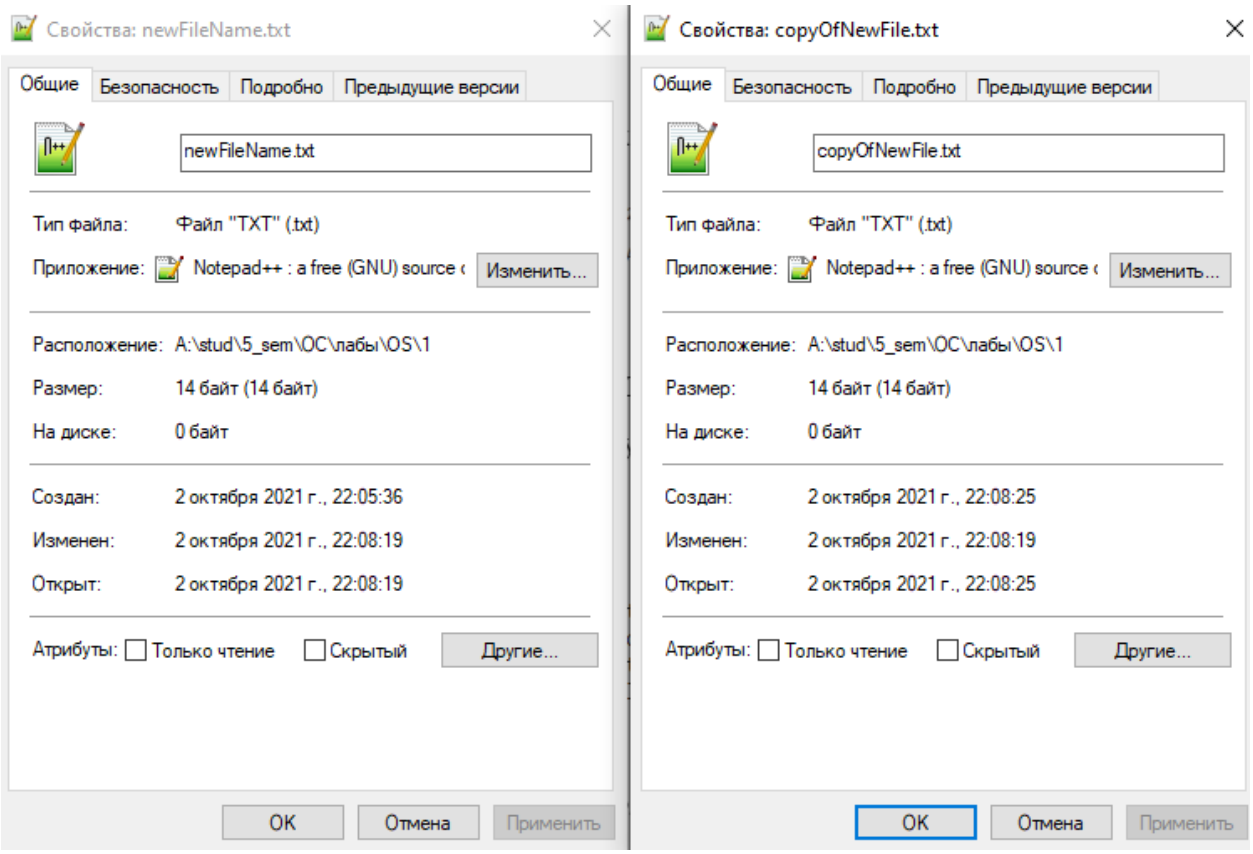


Рисунок 12. Свойства скопированного файла и копии

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

```
Input the name of file to be moved: copyOfNewFile.txt
Input new name of the file: testDir/copyFile.txt
Overwrite the file if it exists? (1 - yes, 0 - no): 0
The file was successfully moved!
Для продолжения нажмите любую клавишу . . .
```

Рисунок 13. Перемещение файла в программе

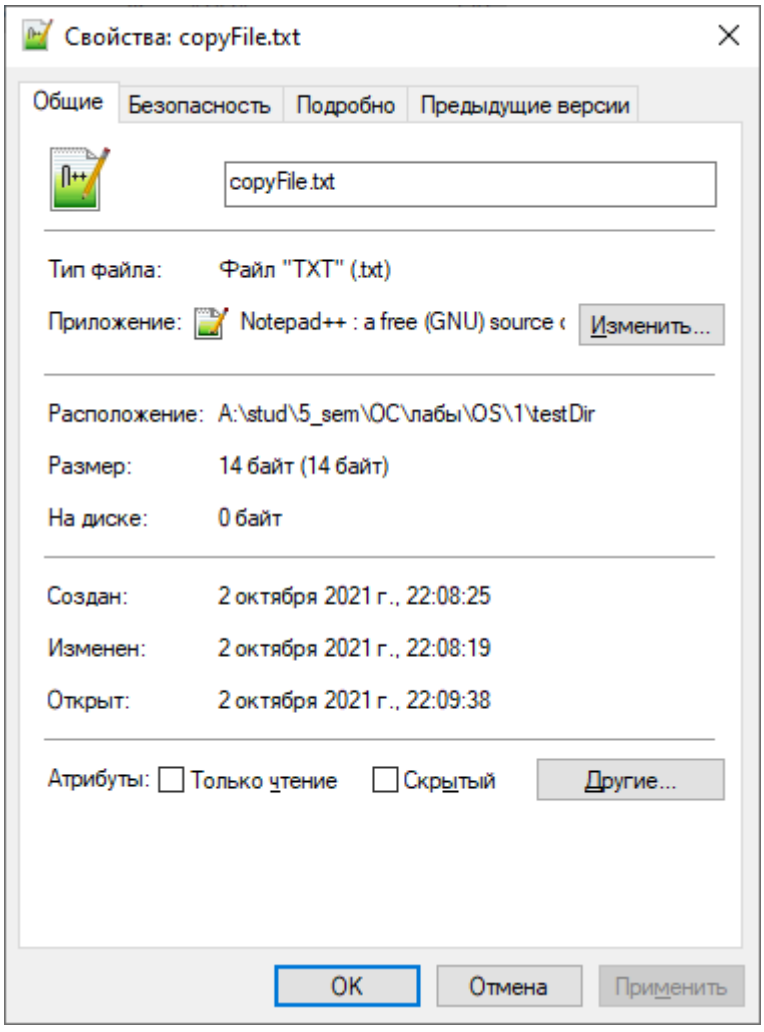


Рисунок 14. Свойства перемещенного файла

7. Получение атрибутов файла. Используемая функция: GetFileAttributes.

```
Input the name of file, which attributes you want to see: newFileName.txt
Use handle to get the attributes? 1 - yes, 0 - no: 1
The file attributes:
        archived
Для продолжения нажмите любую клавишу . . .
```

Рисунок 15. Получение атрибутов файла

8. Установка атрибутов файла. Свойства измененного файла представлены на рисунке 17. Используемая функция: SetFileAttributes.

```
Input the name of file, which attributes you want to set: newFileName.txt

===== Change attributes =====
Input the file attributes to set in one line (separated by space). 0 is end of attributes.
List of available attributes:
1. archived
2. compressed
3. directory
4. encrypted
5. hidden
6. normal
7. read-only
8. system
9. temporary
10. virtual
Input the attributes here: 1 5 0
The file attributes were successfully set!
```

Рисунок 16. Установка атрибутов файла

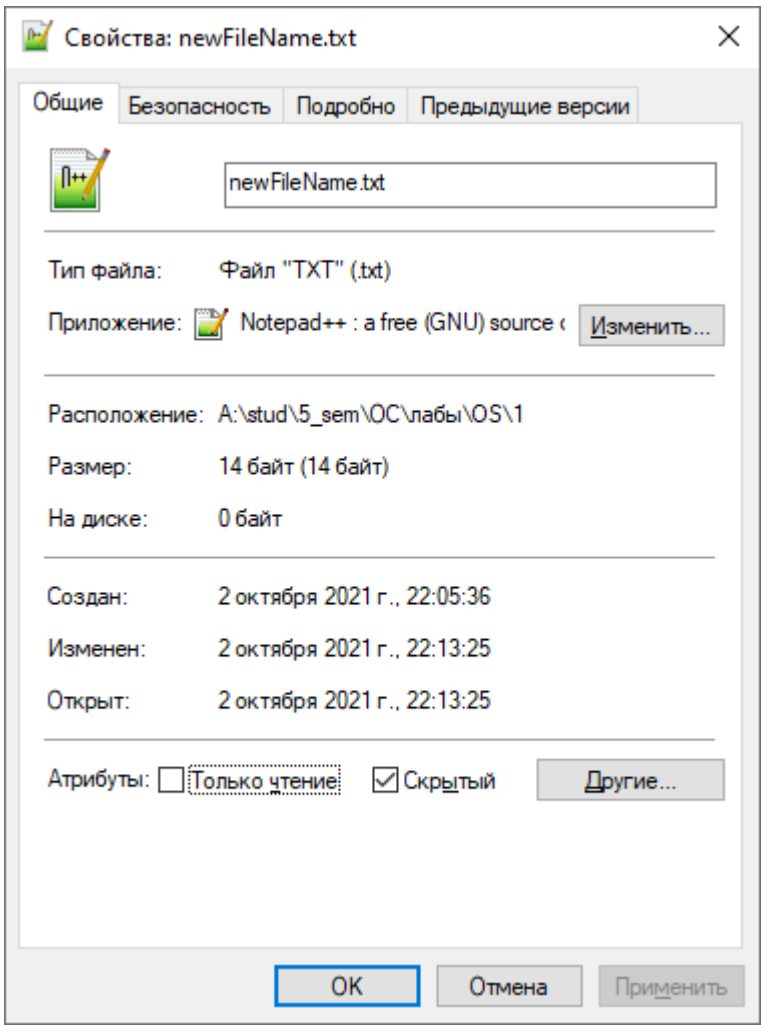


Рисунок 17. Свойства измененного файла

9. Получение времени создания, изменения и открытия файла. Используемая функция: GetFileTime.

```
Input the name of file, which time you want to set: newFileName.txt

Creation time is 02/10/2021 22:05:36
Last access time is 05/10/2021 20:50:16
Last write time is 02/10/2021 22:13:25
Для продолжения нажмите любую клавишу . . .
```

Рисунок 18. Времена файла

10. Изменение времени создания, изменения и открытия файла. Используемая функция: SetFileTime.

```
Input the name of file, which time you want to set: newFileName.txt

===== Creation time =====
Input the date (day/month/year): 19/05/2002
Input the time (hour:minute:seconds): 16:17:24

===== Last access time =====
Input the date (day/month/year): 25/06/2021
Input the time (hour:minute:seconds): 19:56:24

===== Last write time =====
Input the date (day/month/year): 25/06/2021
Input the time (hour:minute:seconds): 19:56:24
File time was successfully changed!
Для продолжения нажмите любую клавишу . . .
```

Рисунок 19. Установка времен файла

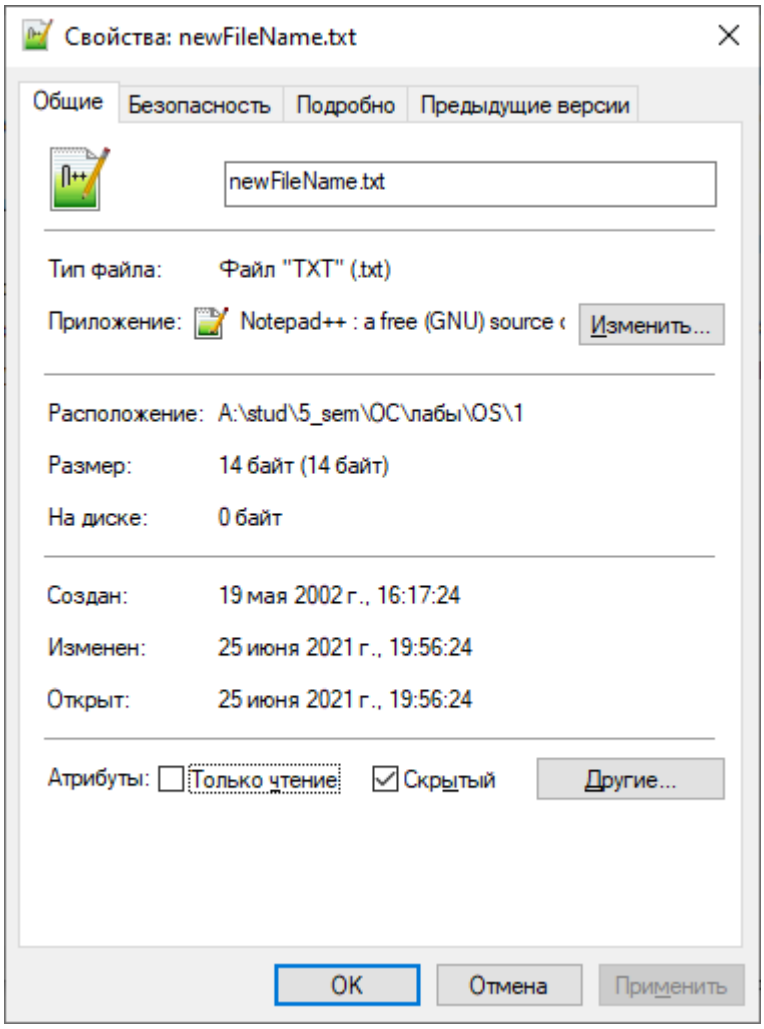


Рисунок 20. Свойства измененного файла

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.

```
Input the file name to copy: A:\Download\9_(2009)_BDRip_1080p.mkv
Input the file name to copy in: 9.mkv
Input block size: 512*32
Input operations amount: 8
The copying took 280.391 seconds
Для продолжения нажмите любую клавишу . . .
```

Рисунок 21. Пример копирования файла

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

```
A:\stud\5_sem\ОС\лабы\OS\1>fc A:\Download\9_(2009)_BDRip_1080p.mkv 9.mkv
Сравнение файлов A:\DOWNLOAD\9_(2009)_BDRip_1080p.mkv и 9.MKV
FC: различия не найдены
```

Рисунок 22. 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

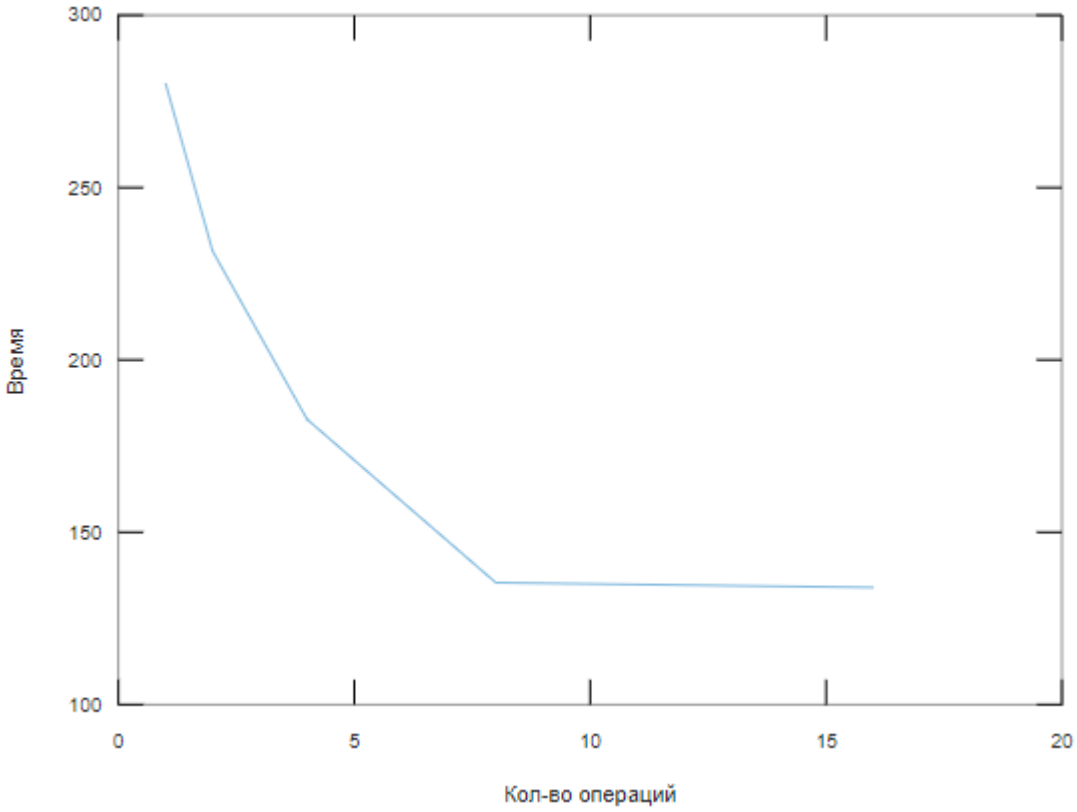


Рисунок 23. Зависимость времени от количества операций при одном размере буфера

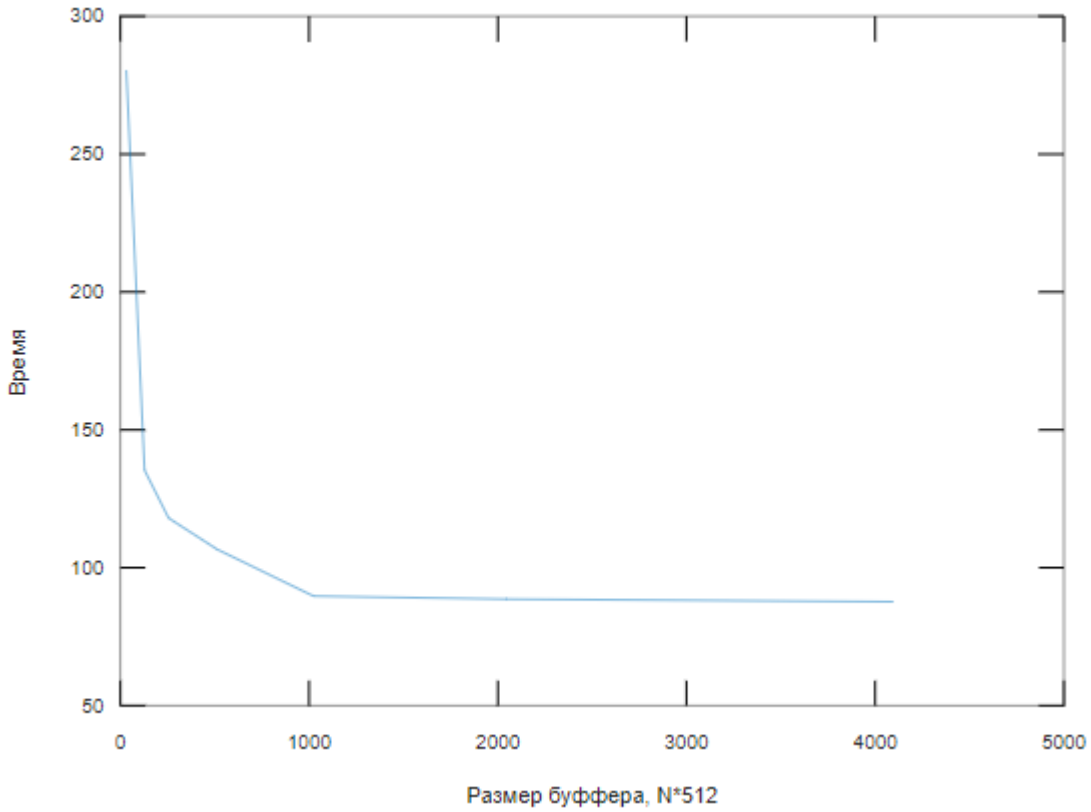


Рисунок 24. Зависимость времени от изменения размера буфера при одинаковом числе операций

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: несколько операций перекрывающего ввод-вывода показывают результат много лучше, чем при одной операции; бОльший размер буфера ускоряет копирование файлов. Однако как увеличение количества операций, так и размера буфера имеет свою границу, отличную для каждого ПК в силу разности характеристик комплектующих.

В ходе проведения замеров было выявлено, что для ПК, который был использован для замеров, оптимально использовать:

- количество операций перекрывающего ввода-вывода – 8 (разница между 8 и 16 операциями не сильно отражаются на времени)
- размер блока буфера – 512*1024 байт (разница между этим размером и размерами выше несущественна)