Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Рязанский Государственный Радиотехнический Университет»

Кафедра ЭВМ

*К защите:*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

дата, подпись

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОЙ РАБОТЕ**

по дисциплине

**«Операционные системы и оболочки»**

**Тема**

**Разработка системной вспомогательной многопоточной программы в системе программирования Visual Studio 2017 Community с использованием средств WinAPI**

Выполнил студент группы 540

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Данилов И.Д.

дата сдачи на проверку, подпись

Руководитель

доцент Засорин С.В. \_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

оценка дата защиты, подпись

Рязань 2018 г.

Оглавление

[1. Введение 3](#_Toc515444791)

[Основная часть 4](#_Toc515444792)

[2. Техническое задание. 4](#_Toc515444793)

[2.1. Основания для разработки программы. 4](#_Toc515444794)

[2.2. Назначение разработки. 4](#_Toc515444795)

[2.3. Требования к программе. 4](#_Toc515444796)

[2.3.1. Требования к функциональным характеристикам. 4](#_Toc515444797)

[2.3.2. Требования к исходным кодам и языкам программирования. 4](#_Toc515444798)

[2.3.3. Требования к составу и параметрам технических средств 5](#_Toc515444799)

[2.4. Требования к надёжности 5](#_Toc515444800)

[2.5. Требования к программной документации 5](#_Toc515444801)

[2.6. Текст задания в соответствии с вариантом 5](#_Toc515444802)

[3. Описание программы. 6](#_Toc515444803)

[3.1. Общие сведения. 6](#_Toc515444804)

[3.2. Функциональное назначение 6](#_Toc515444805)

[3.2.1. Классы решаемых задач. 6](#_Toc515444806)

[3.2.2. Назначение программы. 6](#_Toc515444807)

[3.3. Описание логической структуры. 7](#_Toc515444808)

[3.3.1. Структура программы с описанием функций составных частей и связи между ними. 7](#_Toc515444809)

[3.3.2. Создание инсталлятора приложения 17](#_Toc515444810)

[3.3.3. Создание справочной системы приложения 21](#_Toc515444811)

[3.4. Используемые технические средства 24](#_Toc515444812)

[3.5. Вызов и загрузка 24](#_Toc515444813)

[3.6. Входные данные 24](#_Toc515444814)

[3.7. Выходные данные 26](#_Toc515444815)

[4. Программа и методика испытаний 27](#_Toc515444816)

[4.1. Объект испытаний 27](#_Toc515444817)

[4.1.1. Наименование испытуемой программы 27](#_Toc515444818)

[4.1.2. Область применения испытуемой программы 27](#_Toc515444819)

[4.2. Цель испытаний 27](#_Toc515444820)

[4.3. Требования к программе 27](#_Toc515444821)

[4.4. Требования к программной документации 27](#_Toc515444822)

[4.5. Средства и порядок испытания 28](#_Toc515444823)

[4.6. Методы испытаний 28](#_Toc515444824)

[5. Эксплуатационные документы 36](#_Toc515444825)

[5.1. Руководство программиста 36](#_Toc515444826)

[5.1.1. Общие сведения о программе 36](#_Toc515444827)

[5.1.2. Структура программы 36](#_Toc515444828)

[5.1.3. Настройка программы 37](#_Toc515444829)

[5.1.4. Проверка программы 37](#_Toc515444830)

[5.1.5. Дополнительные возможности программы 37](#_Toc515444831)

[5.1.6. Сообщения системному программисту 38](#_Toc515444832)

[5.2. Руководство пользователя (оператора) 38](#_Toc515444833)

[5.2.1. Назначение и условия применения программы 38](#_Toc515444834)

[5.2.2. Обращение к программе для запуска 39](#_Toc515444835)

[5.2.3. Входные и выходные данные 39](#_Toc515444836)

[5.2.4. Сообщения оператору 39](#_Toc515444837)

# Введение

Целью данной работы является разработка, отладка и тестирование приложения в объектно-ориентированной системе программирования Visual Studio 2018 с использованием средств Windows API.

В приложении должен быть предусмотрен пользовательский интерфейс.

**Наименование программы**

“NTFS2Threads”

Для разработки приложения был выбран объектно-ориентированный язык программирования C#. Это популярный в наше время язык, созданный компанией Microsoft для разработки приложений на платформе .NET Framework. Приоритетом этого языка является его простота. C# позволяет разработчику сосредоточиться на алгоритме, а не на деталях реализации, так как сложные конструкции в нём заключены в абстракции.

# Основная часть

# Техническое задание.

## Основания для разработки программы.

Разработка программы производится в соответствии с указанным в пункте 2.6 заданием.

## Назначение разработки.

Функциональным назначением программы NTFS2Threads является копирование структуры каталогов с одного логического диска на другой, а также анализ расположения группы файлов на диске по отрезкам.

## Требования к программе.

### Требования к функциональным характеристикам.

Программа NTFS2Threads должна обеспечивать возможность выполнения следующих функций:

* Выбор структуры каталогов на диске
* Выбор логического диска для вставки структуры каталогов
* Копирование выбранной структуры каталогов на другой логический диск
* Выдача сообщения о возможности копирования структуры
* Выдача сообщения о результате копирования
* Задание группы файлов
* Формирование таблицы номеров кластеров, занимаемых файлами в заданной группе
* Вывод данной таблицы номеров кластеров, занимаемых файлами в заданной группе

### Требования к исходным кодам и языкам программирования.

Программа должна быть разработана в объектно-ориентированной системе программирования с применением средств API используемой операционной системы.

### Требования к составу и параметрам технических средств

В состав технических средств должен входить персональный компьютер (ПЭВМ). Рекомендуемые технические характеристики:

* Операционная система: семейства Windows не ниже Windows® 7™
* Рекомендуемый процессор: не хуже Intel® Core™ i7-2630QM
* Не менее 256 мБ оперативной памяти
* Экран с разрешением не менее 800x600, 256 цветов.

## Требования к надёжности

Не имеется, так как обработка вводимой пользователем информации обеспечивается операционной системой Windows.

## Требования к программной документации

Программная документация должна быть выполнена в соответствии с ГОСТ ЕСПД. Программа должна содержать справочную систему, включающую всю необходимую информацию по работе с программой.

## Текст задания в соответствии с вариантом

Вариант 25.

Разработать двухпоточное приложение, выполняющие следующие операции:

Физическое копирование содержимого каталожной структуры с одного логического диска на другой.

Анализ расположения по отрезкам группы файлов на логическом диске.

**Исходные данные**

1. Тип файловой системы дисков - NTFS.
2. Имя диска для оригинала.
3. Имя диска для копии.
4. Полные имена файлов в группе.
5. Полное имя файла для вывода результатов.

# Описание программы.

## Общие сведения.

Разработанная прикладная программа имеет следующие атрибуты:

* Название исполняемого файла – NTFS2Threads.exe
* Размер исполняемого файла – 98 кБ
* Версия файла – 1.0.0.0
* Название продукта NTFS2Threads
* Язык – Русский

## Функциональное назначение

### Классы решаемых задач.

Данное приложение позволяет решать задачи, связанные с взаимодействием с файловой системой пользователя данной системы

### Назначение программы.

Основное назначение приложения NTFS2Threads – предоставление некоторой служебной аналитической информации о файловой системе.

В приложении должны быть реализованы следующие функции:

* Возможность выбора пользователем каталога, содержащего некоторую структуру каталогов на диске
* Возможность выбора логического диска для вставки каталога, содержащего структуру
* Проверка выбранного диска на соответствие файловой системе New Technology File System
* Копирование выбранной структуры каталогов на другой логический диск
* Возможность выбора пользователем группы файлов на диске
* Формирование таблицы номеров кластеров, занимаемых файлами в заданной группе
* Вывод сформированной таблицы номеров кластеров, занимаемых файлами в заданной группе

## Описание логической структуры.

### Структура программы с описанием функций составных частей и связи между ними.

Информационная система состоит из набора классов, список которых приведен на рисунке 1. Главным классом является класс Program, описание которого содержится в файле Program.cs. В нём описан главный метод программы – Main(), который вызывается при запуске приложения.

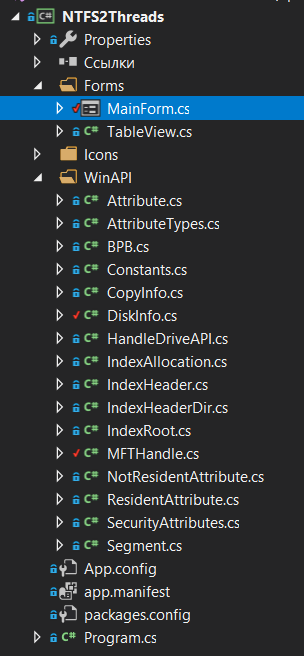


Рисунок 1. Список классов приложения NTFS2Threads

Обозначения, используемые на рисунке 1 и далее:

 – C# файл,

– файл формы,



– конфигурационный файл,



– файл манифеста приложения,



– свойства приложения



 – зависимости приложения

В каталоге WinAPI содержатся классы по работе с файловой системой NTFS. В классе HandleDriveAPI описаны обращения к системным функциям из библиотеки kernel32.dll, таким, как CreateFile, CreateDirectory, CopyFileEx, ReadFile и другим, а также системные перечисления и структуры. Класс Constants содержит необходимые для работы с файловой системой константы, такие, как количество байт в секторе, количество секторов в кластере и т.д. Класс DiskInfo содержит методы для работы с томом NTFS, такие, как чтение сектора по номеру, чтение кластера по номеру и т.д. В классе BPB определены методы по работе с блоком параметров BIOS, такие, как чтение и перевод в строчный формат содержащихся в BPB данных. Класс AttributeTypes содержит описание атрибутов записи в MFT. В свою очередь класс MFTHandle предназначен для работы с записями в MFT, загрузкой атрибутов записи, работой с индексными атрибутами, системные записи о которые представлены в виде классов IndexRoot ($INDEX\_ROOT), IndexAllocation($INDEX\_ALLOCATION), IndexHeader – стандартный заголовок индексного атрибута, IndexHeaderDir – заголовок каталога, содержащий адрес файловой записи в MFT. MFTHandle это объектное представление записи в MFT. Оно содержит поля, содержащиеся в файловой записи в MFT. Класс AttributeTypes содержит описание стандартных атрибутов файловой записи, содержащийся в $STANDARD\_INFORMATION. Классы NotResidentAttribute и ResidentAttribute описывают соответственно нерезидентный и резидентный атрибуты. Класс Segment описывает один отрезок.

В каталоге Forms находится главная форма приложения, а также класс TableView, предназначенный для вывода в табличном виде информации о MFT записях файлов на экран.

Общий алгоритм работы приложения представлен на рисунке 2.



Рисунок 2 – Общий алгоритм приложения

Продолжение см. на следующей странице



Рисунок 2 – Общий алгоритм приложения (продолжение)

Алгоритм, представленный на рисунке 2 реализуется следующим образом:

1. Пользователь посредством интерфейса выбирает задание, с которым намерен работать.
2. Если поток одного из заданий запущен, он приостанавливается и управление передается обработчику события нажатия кнопки пользователем. В этом обработчике запускается поток выбранного пользователем задания.
3. После того, как поток завершает работу, результирующие данные отображаются в программе.

Последовательность действий, выполняющаяся при выполнении обработки события выбора пользователем задания копирования структуры каталогов представлена на рисунке 3.



Рисунок 3 – Алгоритм выполнения копирования структуры каталогов

Продолжение см. на следующей странице



Рисунок 3 – Алгоритм выполнения копирования структуры каталогов (продолжение)

Алгоритм, представленный на рисунке 3 реализуется следующим образом:

1. Пользователь выбирает каталог для копирования и диск, на который хочет скопировать каталог.
2. Запускается поток копирования, в который передаются путь до каталога и выбранный для копирования диск.
3. После чтения блока параметров BIOS производится проверка – является ли выбранный для копирования диск томом NTFS, если нет, выводится соответствующее сообщение. Если же диск является томом NTFS, то программа успешно копирует на него структуру каталогов и выводит соответствующее сообщение.

Последовательность действий, выполняющаяся при выполнении обработки события выбора пользователем задания анализа группы файлов по отрезкам представлена на рисунке 4.



Рисунок 4 – Алгоритм анализа выбранной группы файлов по отрезкам

Алгоритм, представленный на рисунке 4 реализуется следующим образом:

1. Пользователь выбирает файлы для анализа.
2. Если пользователь не выбрал ни одного файла, выводится соответствующее сообщение.
3. Если пользователь выбрал файлы, то для каждого выбранного файла производится чтение пути файла и разбиение пути на строки, где каждая строка является именем каталога, который содержится в пути файла.
4. Для каждого каталога в пути к файлу считается номер записи в MFT. В конце концов мы доходим до номера записи в MFT о файле.
5. Производится вывод информации о данном файле.

Алгоритм чтения записи MFT по индексу записи представлен на рисунке 5.



Рисунок 5 – Алгоритм чтения записи MFT

Алгоритм, представленный на рисунке 5 реализуется следующим образом:

1. Производится подсчёт размера записи MFT и количества записей MFT в одном кластере.
2. Производится поиск первого сектора записи MFT путём перебора отрезков, в которых хранится нерезидентный атрибут DATA на соответствие.
3. Производится чтение найденного сектора
4. Данные сектора анализируются и приводятся к объектному представлению записи MFT для дальнейшего использования.

### Создание инсталлятора приложения

Создадим инсталлятор для приложения при помощи программы Inno Setup 5. Изначально создаётся скрипт с указанными пользователем параметрами, который будет откомпилирован пользователем программы в исполняемый файл-инсталлятор. При создании нового сценарного файла открывается окно, представленное на рисунке 6.



Рисунок 6 – Окно создания инсталлятора

После нажатия кнопки далее откроется окно, представленное на рисунке 7.

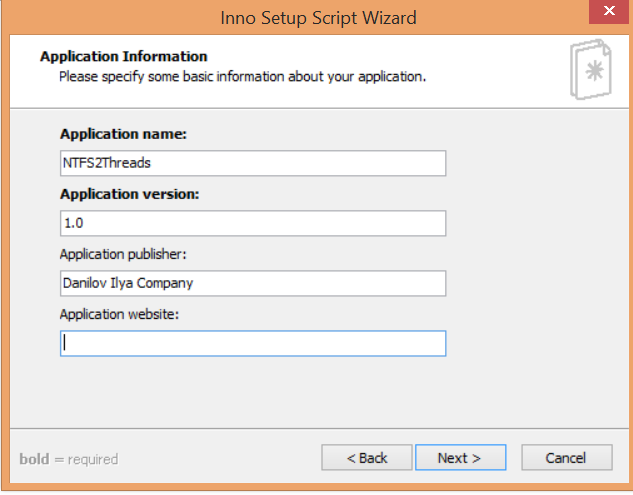


Рисунок 7 – Окно заполнения общей информации о приложении

Далее появится окно, содержащее информацию о стандартном каталоге, в который нужно установить приложение, а также стандартное название каталога. Приведено на рисунке 8.

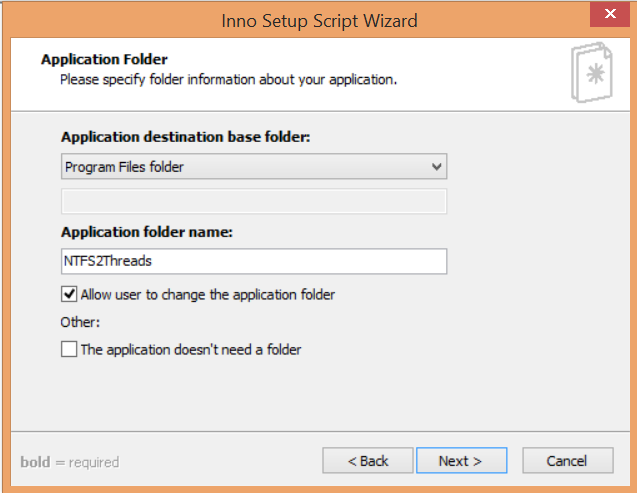


Рисунок 8 – Окно заполнения информации о каталоге приложения.

В следующем окне создания устанавливаются основной исполняемый файл проекта, а также дополнительные файлы приложения. Приведено на рисунке 9.

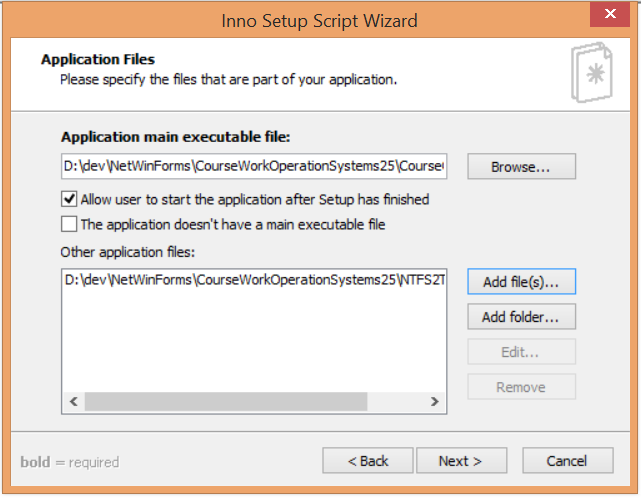


Рисунок 9 – Окно указания основных файлов и директорий приложения

В предпоследнем окне создания инсталлятора заполняется информация об имени скомпилированного инсталлятора, директории, в которой создастся инсталлятор, иконке инсталлятора и задании пароля на инсталляцию. Оно приведено на рисунке 10.

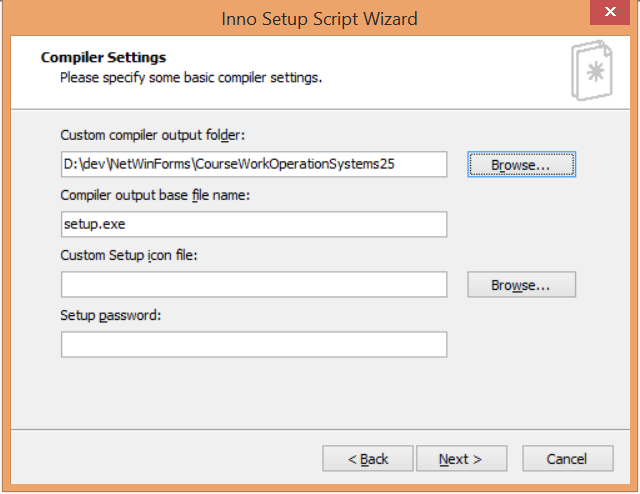


Рисунок 10 – Окно заполнения информации об исполняемом файле инсталлятора.

Последнее окно создания инсталлятора приведено на рисунке 11.

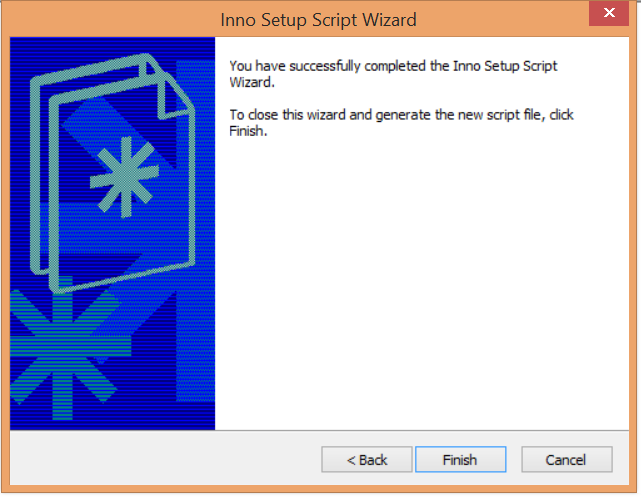


Рисунок 11 – Окно завершения создания инсталлятора

После выполнения выше перечисленных действий создастся файл-скрипт для создания инсталлятора. После его компиляции на компьютере появится исполняемый файл инсталлятора setup.exe.

### Создание справочной системы приложения

Справочную систему для приложения создадим при помощи программы HelpNDoc 5.7. После запуска приложения, достаточно выбрать в панели быстрого доступа задачу New Project (рисунок 12).

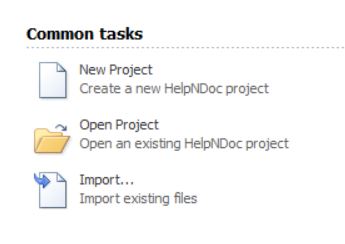


Рисунок 12 – Фрагмент главного окна приложения HelpNDoc.

Откроется окно создания новой справочной системы (рисунок 13).

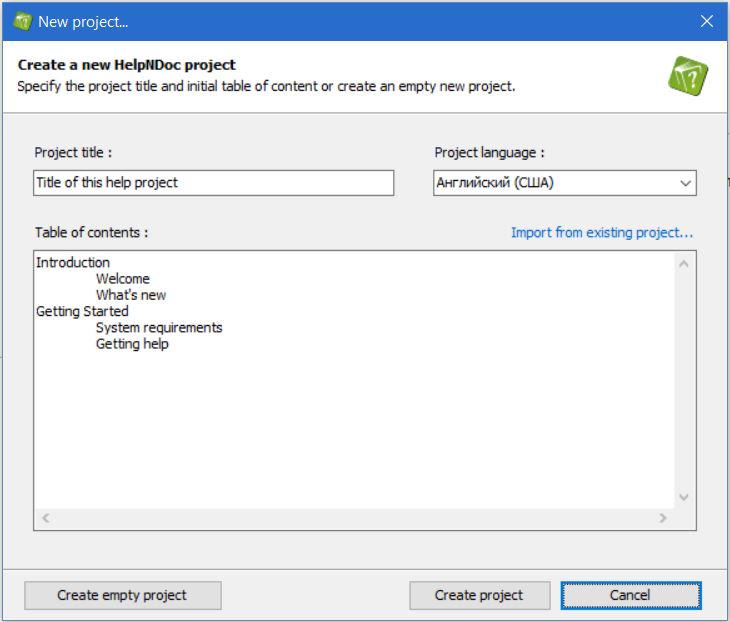


Рисунок 13 – Окно создания справочной системы

Необходимо нажать на кнопку “Create empty project”, после чего откроется окно редактирования справочной системы. Вид окна после заполнения справочной информацией представлен на рисунке 14.

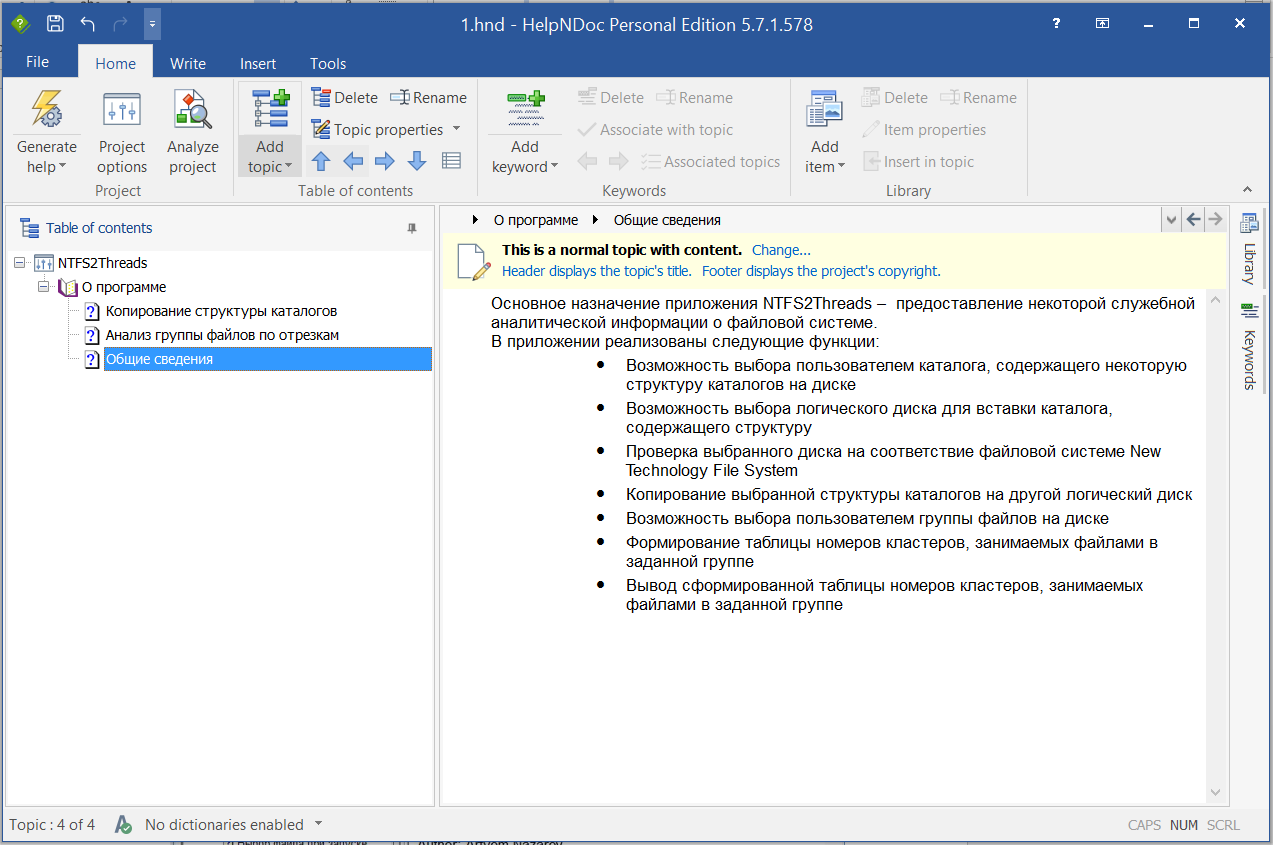


Рисунок 14 – Окно редактирования справочной системы

В разработанном приложении вызов справки осуществляется выбором пункта меню **Помощь > Справка** или нажатием кнопки F1. После вызова справки появляется окно, показанное на рисунке 15.

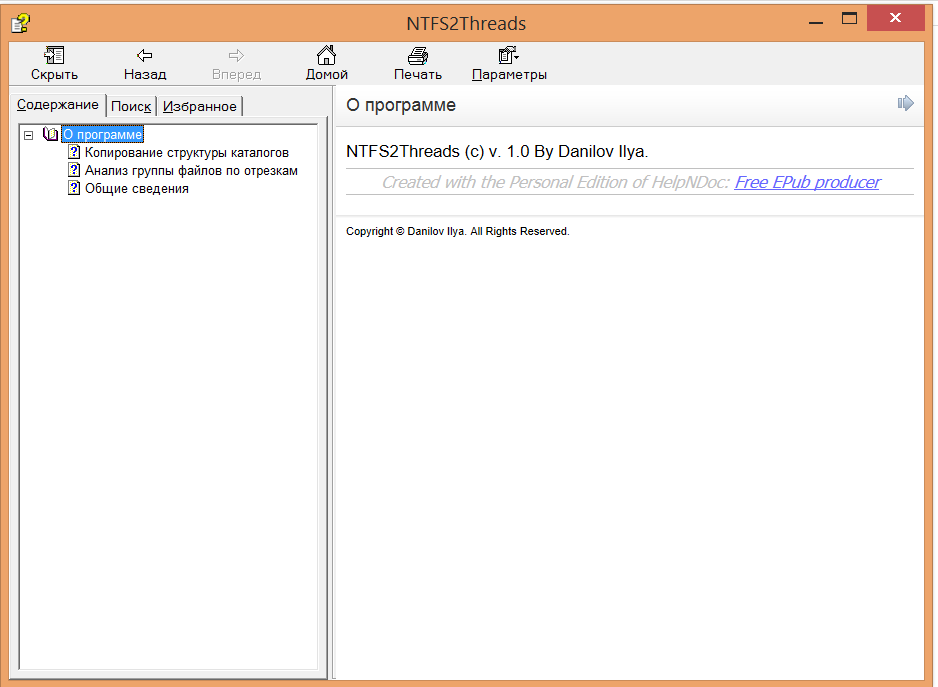


Рисунок 15 – Справка приложения

## Используемые технические средства

* Операционная система: Windows® 7™
* Рекомендуемый процессор: Intel® Core™ i7-2630QM
* ОЗУ: 4 гБ
* Тип системы 64-разрядная ОС
* Мышь
* Клавиатура

## Вызов и загрузка

Установка программы осуществляется при помощи инсталлятора setup.exe (см. п. 3.3.2). После успешной инсталляции, программу можно запустить при помощи меню **Пуск > NTFS2Threads**. Можно также перейти в каталог установленного приложения и открыть исполняемый файл NTFS2Threads.exe.

## Входные данные

При запуске приложения открывается его главная форма (рисунок 16).

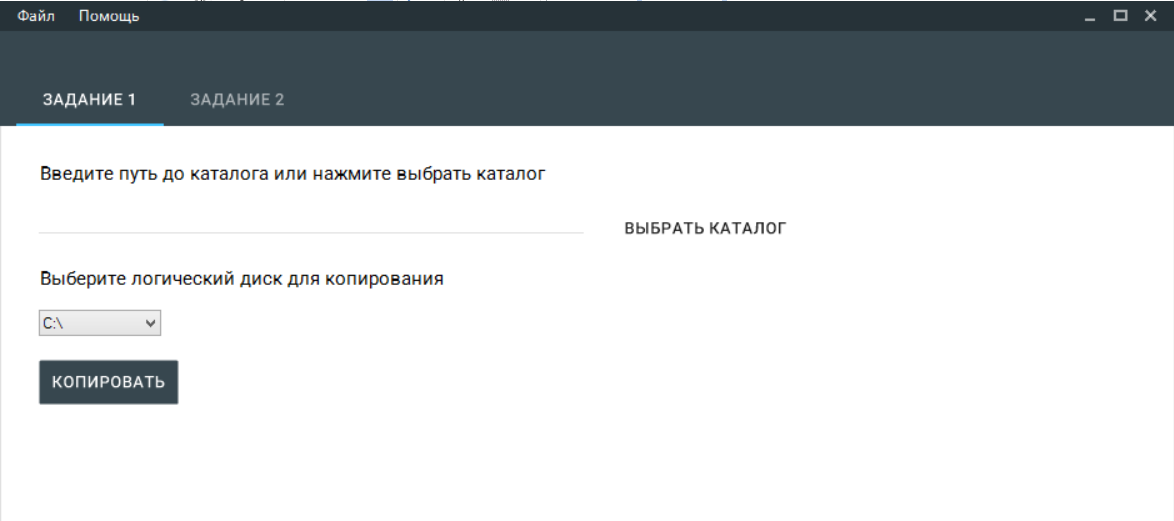


Рисунок 16 – Главное окно приложения, вкладка Задание 1

Для получения данных, требуются действия пользователя, такие, как ввод пути к каталогу и выбор логического диска для копирования.

Для выполнения анализа группы файлов также требуется выбор этих файлов. При нажатии пользователем на кнопку “Выбрать…” открывается диалоговое окно выбора файлов (рисунок 18).

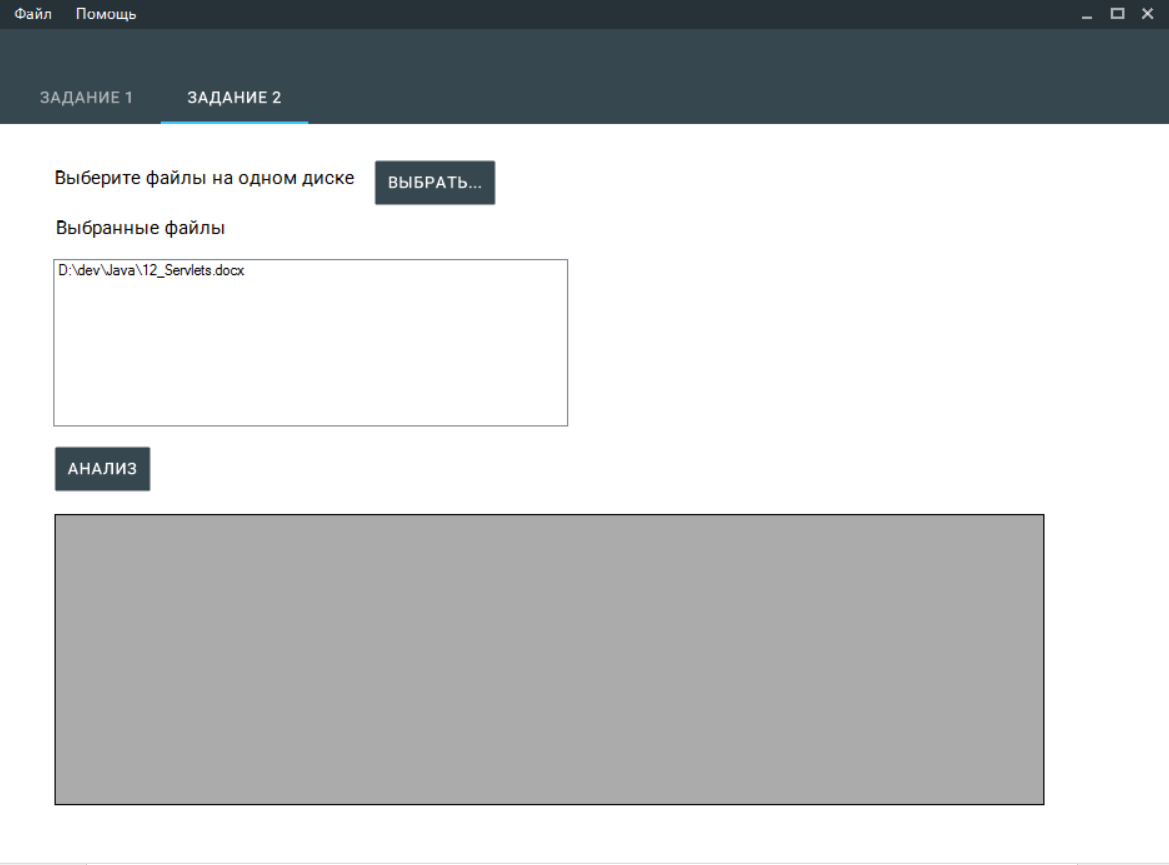


Рисунок 17 – Главное окно приложения, вкладка "Задание 2"

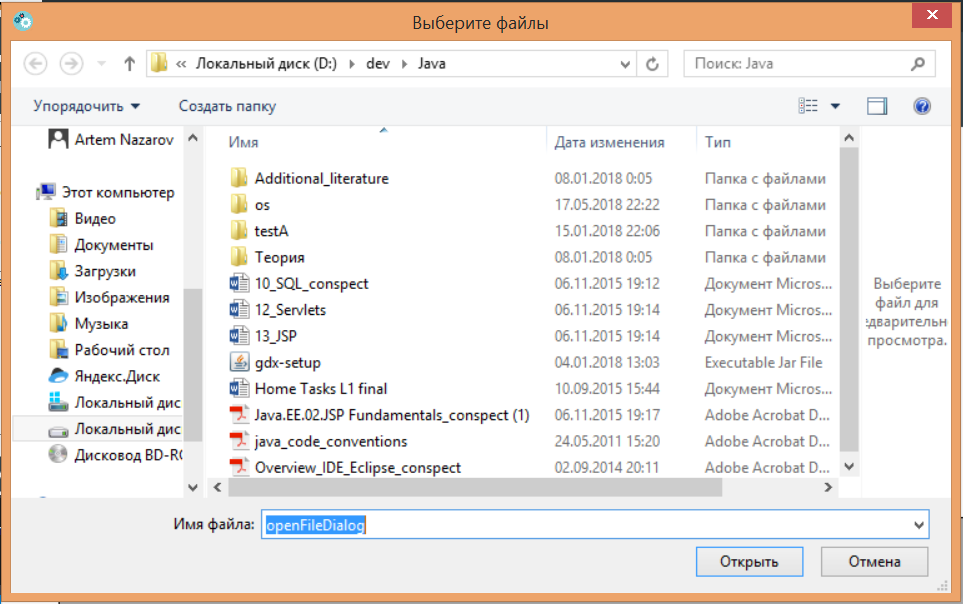


Рисунок 18 – Диалоговое окно выбора группы файлов

## Выходные данные

Выходными данными приложения являются сообщения, информирующие об успешности или неудачи выполнения осуществляемых пользователем действий, а также таблица результатов анализов группы файлов (приведена на рисунке 19).

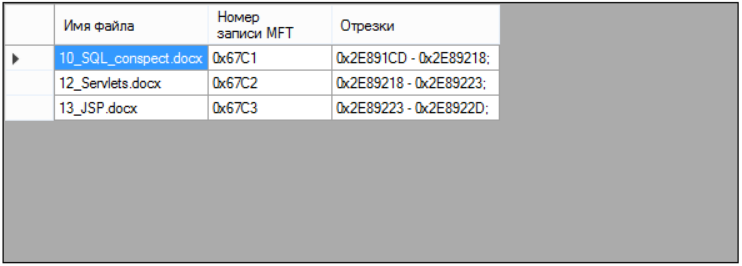


Рисунок 19 – Таблица проанализированных файлов

# Программа и методика испытаний

## Объект испытаний

### Наименование испытуемой программы

Наименование испытуемой программы – NTFS2Threads.

### Область применения испытуемой программы

Программа служит для анализа файлов в файловой системе NTFS и проведения операций с ними.

## Цель испытаний

Испытание данной программы проводится с целью проверки правильности функционирования созданной программы при вводимых данных

## Требования к программе

В ходе тестирования необходимо проверить соответствие программы требованиям, изложенным в п. 2.3, а именно:

* Корректность выбора структуры каталогов на диске
* Безошибочность выбора логического диска для вставки структуры каталогов
* Корректное копирование выбранной структуры каталогов на другой логический диск
* Выдача сообщения о возможности копирования структуры
* Выдача сообщения о результате копирования
* Возможность задания группы файлов
* Правильность формирования таблицы номеров кластеров, занимаемых файлами в заданной группе
* Корректный вывод данной таблицы номеров кластеров, занимаемых файлами в заданной группе

## Требования к программной документации

Программная документация выполнена в соответствии с ГОСТами ЕСПД. В приложении содержится справочная система.

## Средства и порядок испытания

Технические средства, используемые во время испытаний:

* Операционная система: Windows® 7™
* Используемый процессор: Intel® Core™ i7-2630QM
* ОЗУ: 4 гБ
* Тип системы 64-разрядная ОС

## Методы испытаний

Испытание программы проводится с целью проверки работоспособности созданной программы. В соответствии с этим приведены следующие контрольные примеры.

Главное окно приложения приведено на рисунке 20.

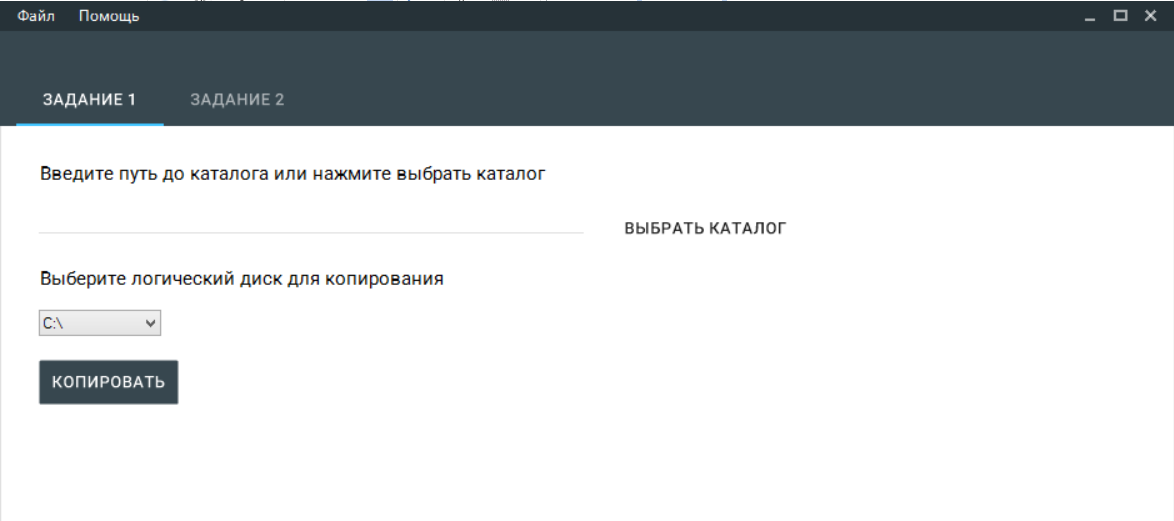


Рисунок 20 – Главное окно приложения

При запуске приложения открывается первая вкладка, на которой содержатся компоненты для копирования каталога. Прежде всего это путь до каталога, который можно выбрать с помощью диалогового окна, которое открывается при нажатии на кнопку “Выбрать каталог”. Выберем каталог D:\dev\NetWinForms\check (рисунок 21).

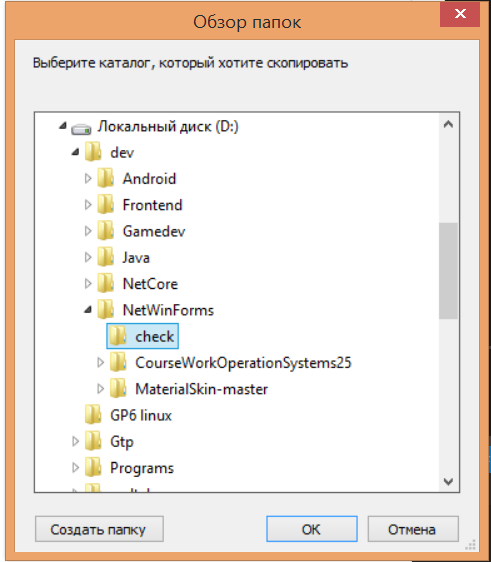


Рисунок 21 – Диалоговое окно выбора каталога для копирования

Выбранный каталог отобразился на форме в строке пути (рисунок 22). Полученная строка пути выделена на рисунке красным цветом.

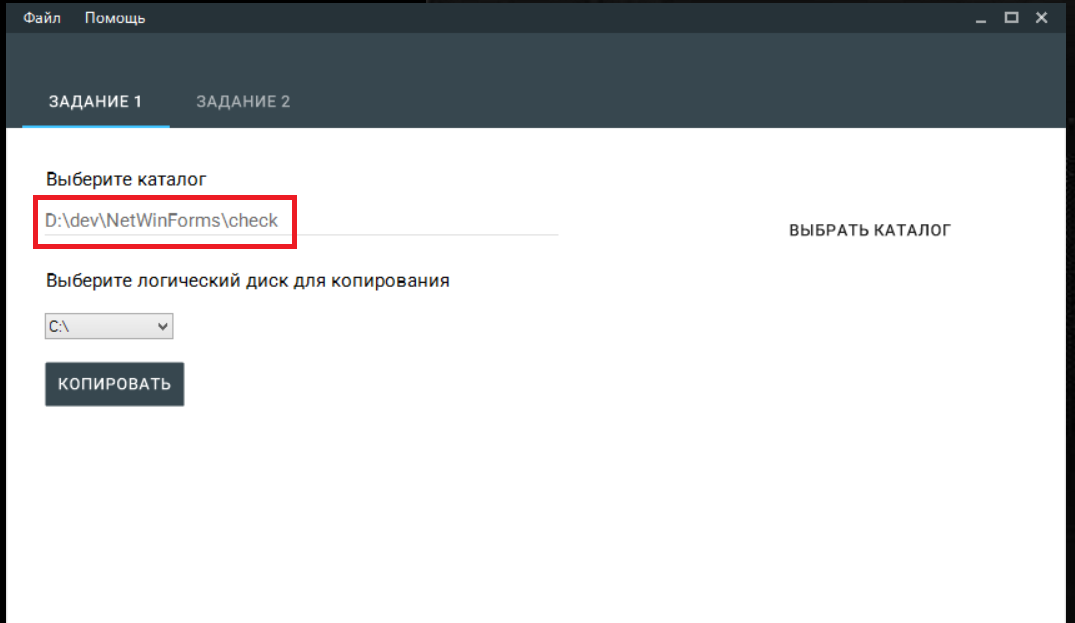


Рисунок 22 – Главное окно приложения, вкладка “Задание 1”

Попробуем скопировать выбранную структуру каталогов. Исходная структура каталогов представлена на рисунке 23.

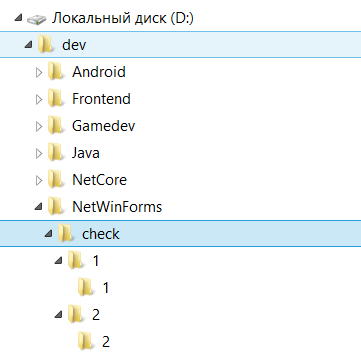


Рисунок 23 – Копируемая структура каталогов

После нажатия на кнопку “Копировать”, появится окно с сообщением об успешном копировании каталога (рисунок 24).

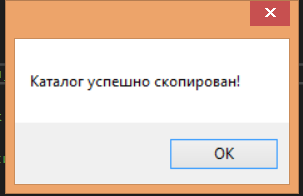


Рисунок 24 – Сообщение об успешном копировании каталога

На диске C проверим результат копирования (рисунок 25).

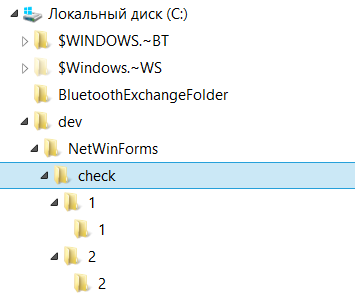


Рисунок 25 – Вставленный каталог

Как видно из рисунка, выбранный каталог успешно скопирован.

Если же в программе попытаться скопировать каталог на тот же диск, на котором он находится, то появится сообщение, представленное на рисунке 26.

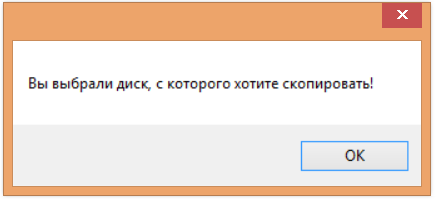


Рисунок 26 – Сообщение о некорректности введенных данных

Если пользователь попытается, не выбрав каталог, нажать кнопку “Копировать”, появится сообщение, приведенное на рисунке 27.

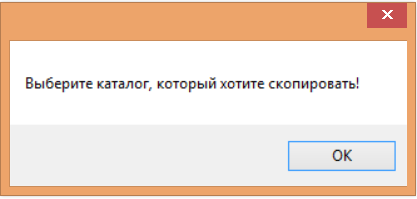


Рисунок 27 – Сообщение о некорректности введенных данных

Если будет произведена попытка копирования на диск, не являющийся NTFS, то будет выведено сообщение, приведенное на рисунке 28.

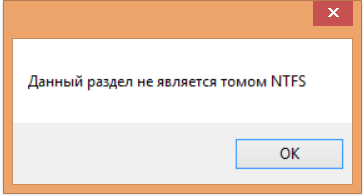


Рисунок 28 – Сообщение о некорректности введенных данных

При нажатии на вкладку “Задание 2”, главное окно приложения примет вид, приведённый на рисунке 29.

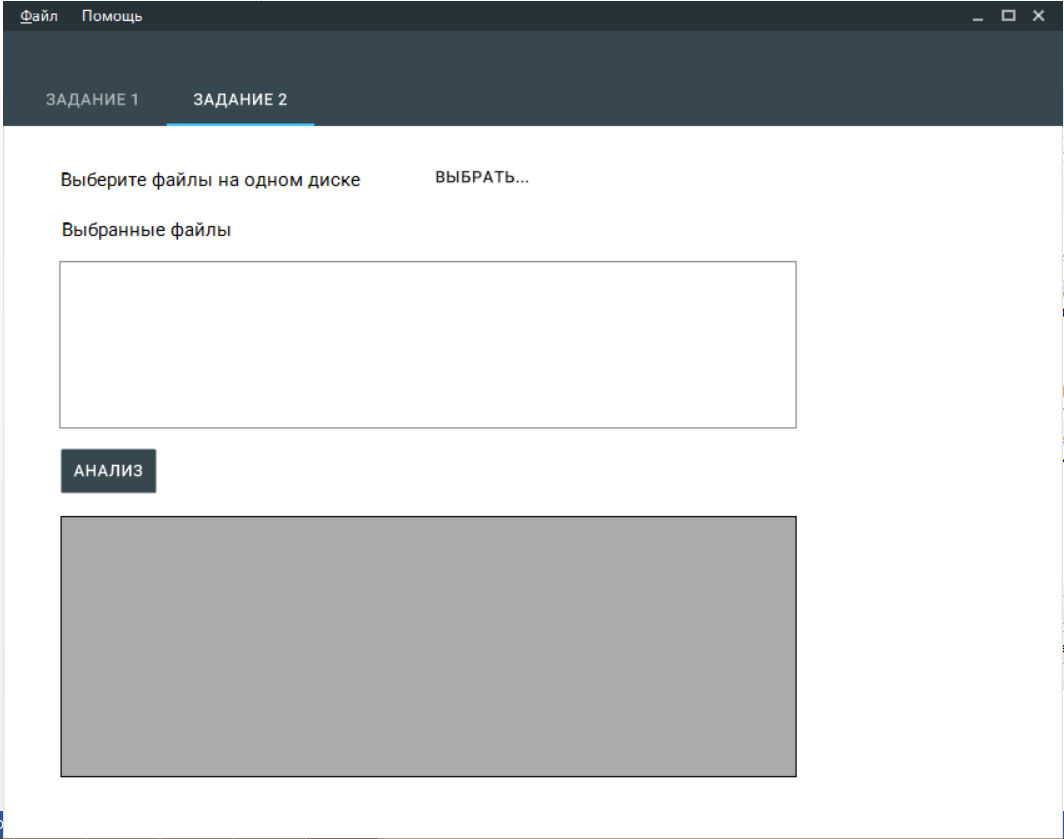


Рисунок 29 – Главное окно приложения при выбранной вкладке "Задание 2"

Попробуем выбрать файлы на диске для анализа. Нажмём на кнопку “Выбрать” и в открывшемся диалоговом окне открытия файла выберем несколько файлов для анализа (рисунок 30).

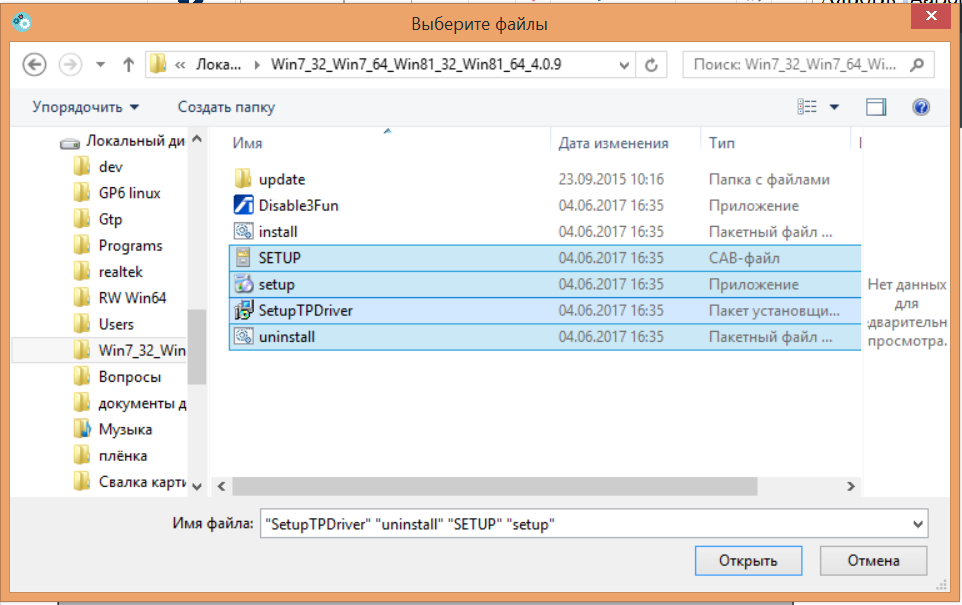


Рисунок 30 – Диалоговое окно открытия группы файлов

После нажатия на кнопку “Открыть” главное окно приложения примет вид, приведённый рисунке 31.

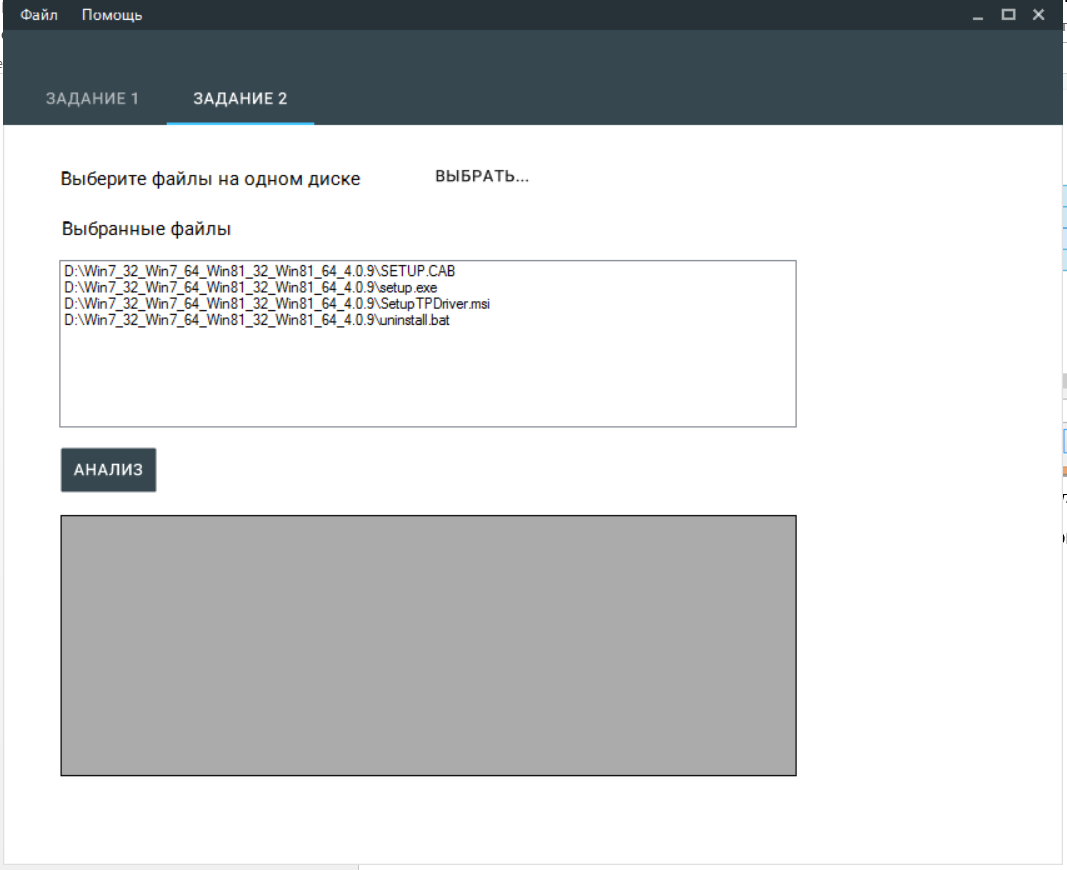


Рисунок 31 – Главное окно приложения после выбора файлов для анализа

После нажатия на кнопку “Анализ”, в нижней таблице главного окна появится информация о проанализированных файлах (рисунок 32).

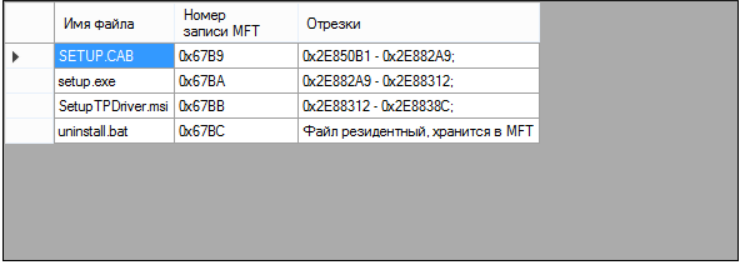


Рисунок 32 – Проанализированные файлы

Если диск, на котором были выбраны файлы не является томом NTFS, будет выведено сообщение, приведенное на рисунке 33.

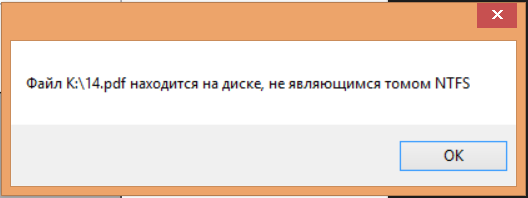


Рисунок 33 – Сообщение о неправильно выбранном для анализа файле

Если пользователь не выбрал ни одного файла для анализа, то будет выведено сообщение, представленное на рисунке 34.

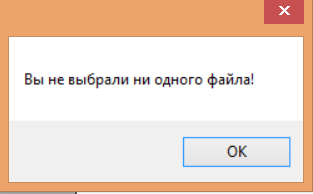


Рисунок 34 – Сообщение о некорректности введенных данных пользователем

# Эксплуатационные документы

## Руководство программиста

### Общие сведения о программе

Приложение NTFS2Threads является двухпоточной программой, осуществляющей копирование каталога с одного диска на другой, а также осуществляющей анализ группы файлов на диске NTFS по отрезкам. Приложение рассчитано на средний уровень квалификации пользователя. Приложение написано на объектно-ориентированном языке программирования C# и разработано в среде разработки Visual Studio Community 2017®.

### Структура программы

Приложение содержит 18 разработанных классов, которые сгруппированы по тематическим каталогам. Список классов и каталогов, в которых они содержатся приведены на рисунке 35.

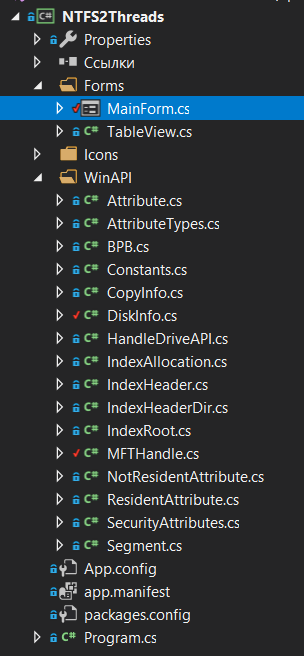


Рисунок 35 – Список классов информационной системы

В каталоге WinAPI содержатся классы по работе с файловой системой NTFS. В классе HandleDriveAPI описаны обращения к системным функциям из библиотеки kernel32.dll, таким, как CreateFile, CreateDirectory, CopyFileEx, ReadFile и другим, а также системные перечисления и структуры. Класс Constants содержит необходимые для работы с файловой системой константы, такие, как количество байт в секторе, количество секторов в кластере и т.д. Класс DiskInfo содержит методы для работы с томом NTFS, такие, как чтение сектора по номеру, чтение кластера по номеру и т.д. В классе BPB определены методы по работе с блоком параметров BIOS, такие, как чтение и перевод в строчный формат содержащихся в BPB данных. Класс AttributeTypes содержит описание атрибутов записи в MFT. В свою очередь класс MFTHandle предназначен для работы с записями в MFT, загрузкой атрибутов записи, работой с индексными атрибутами, системные записи о которые представлены в виде классов IndexRoot ($INDEX\_ROOT), IndexAllocation($INDEX\_ALLOCATION), IndexHeader – стандартный заголовок индексного атрибута, IndexHeaderDir – заголовок каталога, содержащий адрес файловой записи в MFT. MFTHandle это объектное представление записи в MFT. Оно содержит поля, содержащиеся в файловой записи в MFT. Класс AttributeTypes содержит описание стандартных атрибутов файловой записи, содержащийся в $STANDARD\_INFORMATION. Классы NotResidentAttribute и ResidentAttribute описывают соответственно нерезидентный и резидентный атрибуты. Класс Segment описывает один отрезок.

В каталоге Forms находится главная форма приложения, а также класс TableView, предназначенный для вывода в табличном виде информации о MFT записях файлов на экран.

### Настройка программы

Программа не требует дополнительной настройки.

### Проверка программы

Результаты проверки программы приведены в разделе 4.6.

### Дополнительные возможности программы

Программа не имеет дополнительных возможностей.

### Сообщения системному программисту

В программе предусмотрена защита от некорректно введенных данных пользователем. Для этого программой выводятся сообщения. Сообщение, приведенное на рисунке 36, возникает при попытке записи копирования каталога на диск, не являющийся томом NTFS.

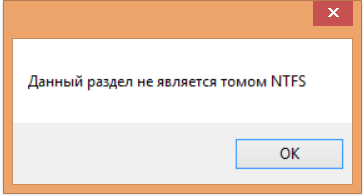


Рисунок 36 – Сообщение о некорректности файловой системы, на которую планируется копирование каталога

Если диск, на котором были выбраны файлы, не является томом NTFS, будет выведено сообщение, приведенное на рисунке 33.

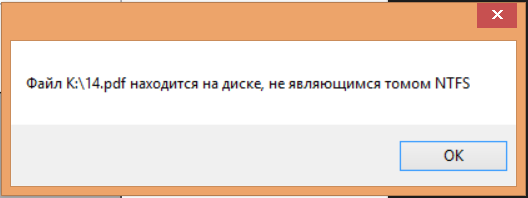


Рисунок 37 – Сообщение о некорректности файловой системы, в которой находится файл

## Руководство пользователя (оператора)

### Назначение и условия применения программы

Программа NTFS2Threads должна обеспечивать возможность выполнения следующих функций:

* Выбор структуры каталогов на диске
* Выбор логического диска для вставки структуры каталогов
* Копирование выбранной структуры каталогов на другой логический диск
* Выдача сообщения о возможности копирования структуры
* Выдача сообщения о результате копирования
* Задание группы файлов
* Формирование таблицы номеров кластеров, занимаемых файлами в заданной группе
* Вывод данной таблицы номеров кластеров, занимаемых файлами в заданной группе

Программа будет работать только на операционной системе Windows не ниже Windows 7 ®.

### Обращение к программе для запуска

Программа требует установки. Создание инсталлятора описано в п. 3.3.2.

* Операционная система: семейства Windows не ниже Windows® 7™
* Рекомендуемый процессор: не хуже Intel® Core™ i7-2630QM
* Не менее 256 мБ оперативной памяти
* Экран с разрешением не менее 800x600, 256 цветов.

Запуск программы можно осуществить 3 способами:

* Щёлкнуть по ярлыку программы на рабочем столе левой кнопкой мыши 2 раза
* открыть меню **Пуск,** выбрать среди программ **NTFS2Threads**
* перейти в каталог установленного приложения и открыть исполняемый файл NTFS2Threads.exe.

### Входные и выходные данные

Для получения данных, требуются действия пользователя, такие, как ввод пути к каталогу и выбор логического диска для копирования.

Для выполнения анализа группы файлов также требуется выбор этих файлов. При нажатии пользователем на кнопку “Выбрать…” открывается диалоговое окно выбора файлов.

Выходными данными приложения являются сообщения, информирующие об успешности или неудачи выполнения осуществляемых пользователем действий, а также таблица результатов анализов группы файлов.

### Сообщения оператору

Все введенные пользователем данные проходят проверку. Если при выполнении задания 1 пользователь, не выбрав каталога, нажмёт на кнопку копирования каталога, появится сообщение, приведенное на рисунке 38.

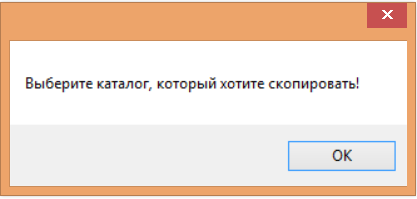


Рисунок 38 – Сообщение о необходимости выбора каталога для копирования

Если пользователь попробует скопировать данные на тот же диск, на котором находится копируемый файл, появится сообщение, приведенное на рисунке 39.

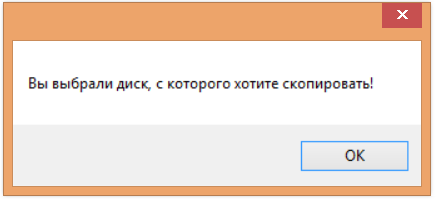


Рисунок 39 – Сообщение о неверно выбранном диске

Таким образом, производится защита от некорректной работы программы.

При успешном выполнении копирования каталогов, выводится сообщение, приведенное на рисунке 40.

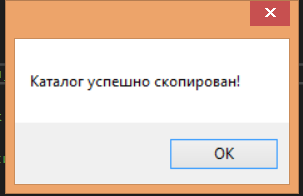


Рисунок 40 – Сообщение об успешном копировании каталога

При выполнении анализа группы файлов, также производится проверка наличия выбранных пользователем файлов для анализа. Если пользователь не выбрал ни одного файла, выполнение прекращается и выводится сообщение, приведенное на рисунке 41.

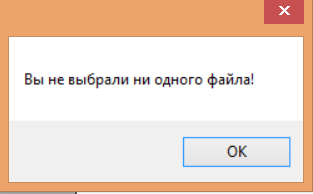


Рисунок 41 – Сообщение о необходимости выбора файлов пользователем

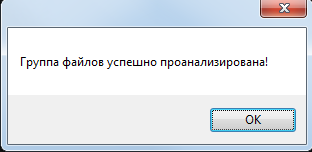


Рисунок 42 – Сообщение об успешном анализе группы файлов

# Заключение

В данной курсовой работе была поставлена задача разработки приложения,

реализующего копирование структуры каталогов с одного логического диска на другой, а также анализ расположения группы файлов на диске по отрезкам с помощью средств WinAPI.

Были разработаны алгоритмы, логическая структура программы и сама программа, произведена экспериментальная проверка программы, разработано руководство оператора и описано выполнение программы, также предусмотрен контроль над ошибками.

Для организации взаимодействия пользователя и программы разработан пользовательский интерфейс, предусмотрен контроль над корректностью введенных данных.

Созданная программа отвечает требованиям, заявленным как в индивидуальном, так и в общем задании.