Минобрнауки России

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Санкт-Петербургский государственный технологический институт

(технический университет)»

|  |  |  |  |
| --- | --- | --- | --- |
| **Направление подготовки** | | 09.03.03 Прикладная информатика в химии | |
| **Направленность** | | Прикладная информатика в химии | |
| **Факультет** | | Информационных технологий и управления | |
| **Кафедра** | | Систем автоматизированного проектирования и управления | |
| **Учебная дисциплина** | | Программирование | |
| **Курс** | 1 | **Группа** | 485 |
| **Студент** | |  | |

Курсовой проект

Тема: Разработка программного комплекса для анализа занятого и свободного места на жестком диске.

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (дата) гр.№485

Руководители

ст.преп. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.К. Федин

(подпись) (дата)

доц., к.т.н. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ И.Г. Корниенко

(подпись) (дата)

Санкт-Петербург

2019

Минобрнауки России

федеральное государственное бюджетное образовательное учреждение высшего образования

«Санкт–Петербургский государственный технологический институт

(технический университет)»

ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ

|  |  |  |  |
| --- | --- | --- | --- |
| **Направление подготовки** | | 09.03.03 Прикладная информатика в химии | |
| **Направленность** | | Прикладная информатика в химии | |
| **Факультет** | | Информационных технологий и управления | |
| **Кафедра** | | Систем автоматизированного проектирования и управления | |
| **Учебная дисциплина** | | Программирование | |
| **Курс** | 1 | **Группа** | 485 |
| **Студент** | |  | |

|  |  |
| --- | --- |
| **Тема:** | Разработка программного комплекса для анализа занятого и свободного места на жестком диске. |

|  |  |
| --- | --- |
| **Цель работы:** | Разработка программы для поиска файлов на диске и подсчёта занимаемого ими объёма на диске. Вывод общей информации о файлах. |

**Исходные данные по проекту:**

1. Страуструп, Б. Programming: Principles and Practice Using C++ / Б. Страуструп. – М. : Вильямс, 2011. – 1248 c.

2. Норенков, И.П. Основы автоматизированного проектирования / И.П. Норенков. – М. : МГТУ им. Н. Э. Баумана, 2009. – 430 c.

3. Вильямс, Я.Ш. C++: базовый курс / Я.Ш. Вильямс. – М. : Вильямс, 2014. – 624 с.

4. Липпман, C. Основы программирования на C++ / С. Липпман. – М. : Вильямс, 2014. – 1104 с.

5. Олифер, Н. А., Олифер, В. Г. Сетевые операционные системы / Н. А. Олифер В. Г. Олифер – СПб: Питер, 2009. – 669 с.

6. Майерс, С. Эффективное использование C++ / С. Майерс – СПб: ДМК Пресс, 2014. – 300 с.

7. Руссинович М., Соломон Д., Ионеску А. Внутреннее устройство Microsoft Windows. Основные подсистемы ОС / М. Руссинович, Д. Соломон, А. Ионеску – СПб: Питер СПБ, 2014. – 672 с.

**Перечень вопросов, подлежащих разработке:**

1 Аналитический обзор

1.1 Обзор и анализ процесса поиска файлов и подсчёта занимаемого ими объёма на диске. Сравнительная характеристика существующих систем-аналогов.

1.2 Общая характеристика и особенности процесса поиска файлов и подсчёта занимаемого ими объёма на диске.

1.3 Обзор и обоснование выбора инструментального программного обеспечения.

2 Цель и задачи курсового проекта.

3 Технологическая часть.

3.1 Формализованное описание процесса поиска файлов и подсчёта занимаемого ими объёма на диске как объекта обработки информации.

3.2 Постановка задач поиска файлов и подсчёта занимаемого ими объёма на диске.

3.3 Разработка функциональной структуры программного комплекса для поиска файлов и подсчёта занимаемого ими объёма на диске.

3.4 Создание алгоритма поиска файлов и подсчёта занимаемого ими объёма на диске.

3.5 Разработка структуры интерфейса для пользователя.

3.6 Описание структур данных и алгоритмов (формат представления данных в памяти и на внешних носителях).

3.7 Описание структуры программы (модули, основные функции, классы и т.д.).

3.8 Тестирование программного комплекса (на заданном примере).

3.9 Оформление документации (пояснительной записки, презентации) по проекту.

**Перечень графического материала:**

1 Формализованное описание процесса поиска файлов и подсчёта занимаемого ими объёма на диске.

2 Функциональная структура программного комплекса по поиску файлов и подсчёту занимаемого ими объёма на диске.

3 UML-диаграмма использования программы пользователем.

4 Блок–схемы алгоритмов поиска файлов и подсчёта занимаемого ими объёма на диске.

5 Тестовый пример работы программного комплекса.

6 Характеристика аппаратного и программного обеспечения.

**Требования к аппаратному и программному обеспечению:**

|  |  |
| --- | --- |
| Показатель | Значение |
| Тип ЭВМ | Персональный компьютер |
| Тактовая частота процессора, ГГц | 1,8 |
| Объем оперативной памяти, Гб | 8 |
| Объем внешней памяти, Гб | 256 |
| Состав и характеристика периферийных устройств ЭВМ | Клавиатура, мышь, монитор с диагональю 13′′ и разрешением 1920 точек |
| Операционная система | Windows 10 |
| Прикладное программное обеспечение, необходимое для функционирования программного комплекса | Программы пакета Microsoft Office, Microsoft Visual Studio 19 |

|  |  |
| --- | --- |
| **Дата выдачи задания:** |  |
| **Дата предоставления курсового проекта к защите:** |  |

|  |  |
| --- | --- |
| Заведующая кафедрой, проф. | Т. Б. Чистякова |
| Лектор, доц. | И. Г. Корниенко |
| Консультант, ст. преп. | А. К. Федин |
| Задание принял к выполнению |  |
|  |  |

**Оглавление**

[**1 АНАЛИТИЧЕСКИЙ ОБЗОР** 8](#_Toc26396046)

[**1.1 Сравнительная характеристика существующих систем-аналогов** 8](#_Toc26396047)

[**1.2 Общая характеристика и особенности процесса поиска файлов и подсчёта занимаемого ими объёма на диске.** 12](#_Toc26396048)

[**1.3 Обзор и обоснование выбора инструментального программного обеспечения** 13](#_Toc26396049)

[**1.3.1 Обзор языков программирования** 13](#_Toc26396050)

[**1.3.2 Обзор средств разработки** 15](#_Toc26396051)

[**2 ЦЕЛЬ И ЗАДАЧИ КУРСОВОГО ПРОЕКТА** 17](#_Toc26396052)

[**3. ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ** 18](#_Toc26396053)

[**3.1 Формализованное описание процесса разработки программного комплекса для анализа занятого и свободного места на жестком диске** 18](#_Toc26396054)

[**3.2 Постановка задач анализа занятого и свободного места на жестком диске** 18](#_Toc26396055)

[**3.3 Функциональная структура программного комплекса для анализа занятого и свободного места на жестком диске** 19](#_Toc26396056)

[**3.4 Алгоритм анализа занятого и свободного места на жестком диске** 20](#_Toc26396057)

[**3.4.1 Алгоритм поиска файлов** 20](#_Toc26396058)

[**3.4.2 Алгоритм вставки файла согласно расширению и категории** 21](#_Toc26396059)

[**3.5 Разработка структуры интерфейса для пользователя** 23](#_Toc26396060)

[**3.6 Описание структур данных** 26](#_Toc26396061)

[**3.7 Описание структуры программы** 26](#_Toc26396062)

[**3.8 Тестирование программного комплекса** 29](#_Toc26396063)

[**СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ** 36](#_Toc26396064)

**ВВЕДЕНИЕ**

Темой курсового проекта стала разработка программного комплекса для анализа занятого и свободного места на жестком диске.

На компьютерах многих пользователей, как опытных, так и начинающих, зачастую скапливается множество файлов, которые требуют структуризации в категории и вычисления занимаемого ими объема на диске. Тогда пользователь сможет решить какие файлы занимают больше всего места и стоит ли их удалять. Составление программы для решения этой проблемы обусловили цель и задачи этого курсового проекта.

Анализ занятого и свободного места на жестком диске – это процесс, поиска файлов и распределения их по выбранным пользователем категориям, а также вычисления общего объема, занимаемого ими на диске.

Данный курсовой проект представляет собой портативную программу, анализирующую файлы и вычисляющую занимаемый ими объем. Пользователь может выбрать путь для анализа, нужные ему категории и входящие в них расширения.

# **1 АНАЛИТИЧЕСКИЙ ОБЗОР**

**1.1 Сравнительная характеристика существующих систем-аналогов**

Систем-аналогов существует достаточно много. Большинство из них имеет схожий функционал, позволяющий искать файлы. Несколько таких программ будет описано ниже.

**CCleaner**

CCleaner - утилита для компьютеров под управлением Microsoft Windows, которая очищает «мусор», накапливающийся с течением времени: временные файлы, сломанные ярлыки и другие проблемы. Одним из разделов этой программы является анализ дисков. Интерфейс прост, понятен и удобен в использовании (рисунок 1).

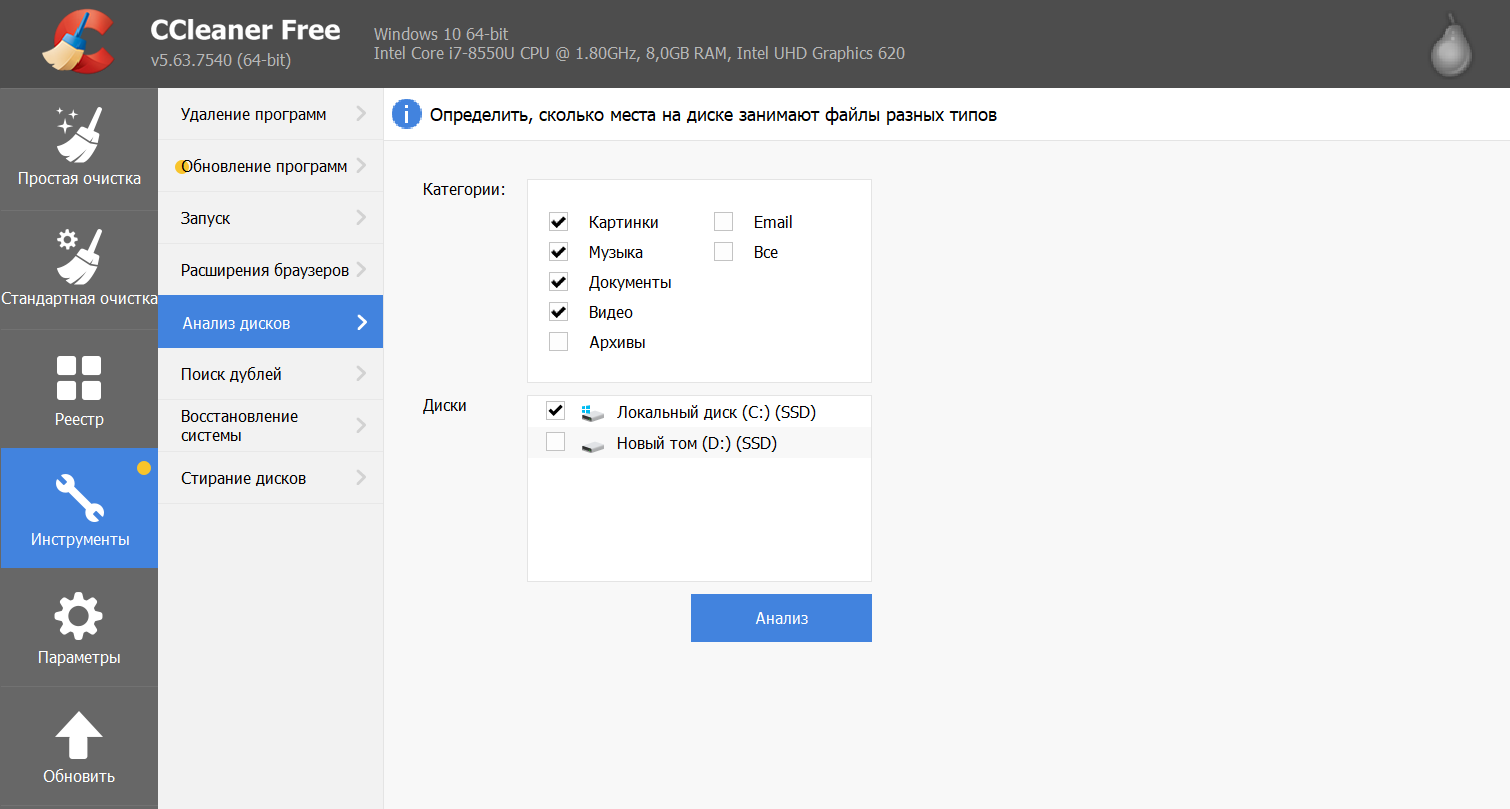


Рисунок 1 – Интерфейс раздела «Анализ дисков» утилиты CCleaner

На выбор пользователю предоставляются категории и диски, по которым и где будет производится поиск.

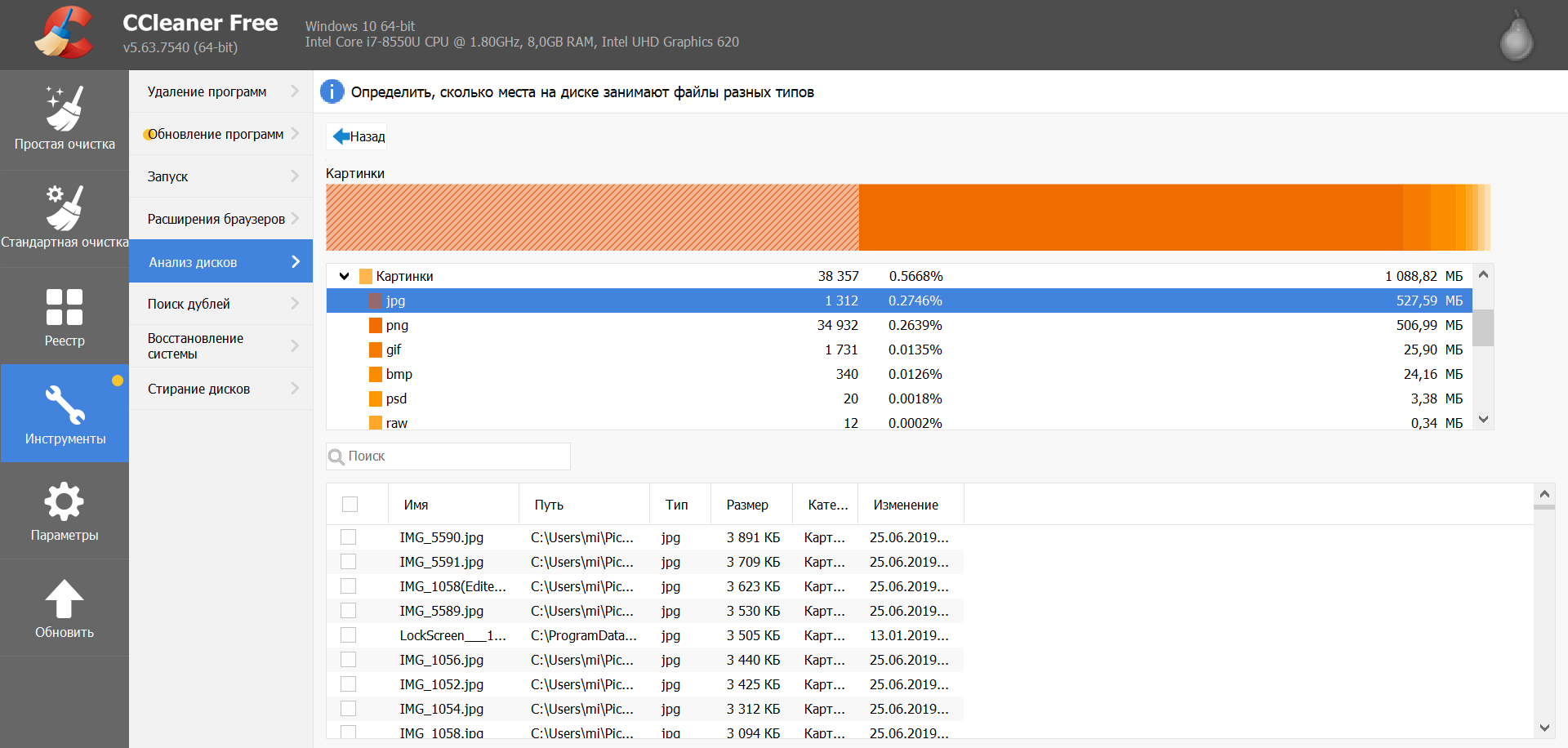
CCleaner позволяет после полного анализа диска просмотреть, какие файлы есть в той или иной категории. Также можно узнать более детальную информацию, а именно: имя, путь, размер, категорию и дату последнего изменения каждого файла. Кроме того, категории показывают объем занимаемого ими места на диске в Мб и в процентном соотношении. Ниже приведен результат работы программы, всего за 25 секунд было найдено 346707 файла (Рисунок 2). 

Рисунок 2 – Результаты анализа диска

CCleaner ищет все файлы, которые присутствуют на диске и относит каждый к соответствующему расширению и категории. Если файл не попал ни в одну, то он заносится в раздел «Другие файлы».

Следующей системой-аналогом стала Disk Savvy. Её интерфейс не менее удобен и прост в использовании, он представлен на рисунке 3.

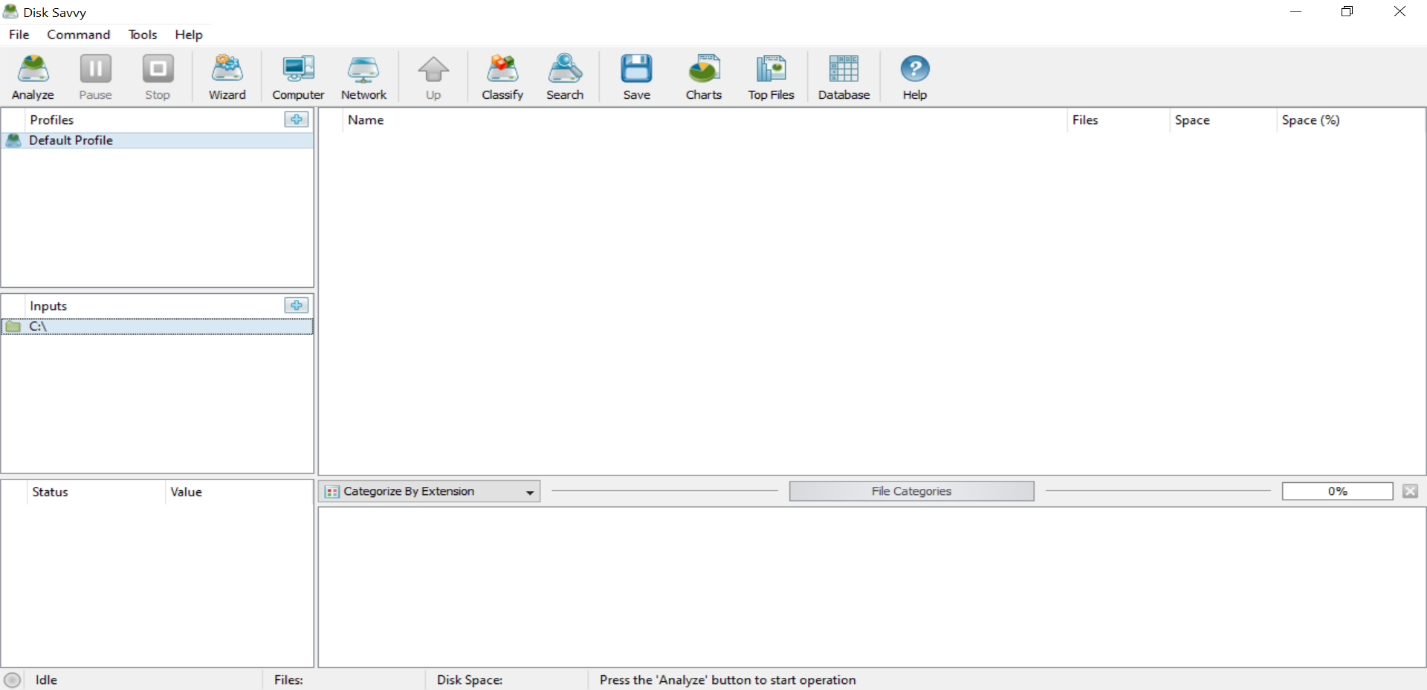


Рисунок 3 – Интерфейс программы Disk Savvy

Как видно, интерфейс этой утилиты более расширенный и содержит в себе больший функционал. В отличие от CCleaner, он позволяет выбирать конкретные папки для анализа и сохранять их в профили. Также можно выбирать сетевые расположения для анализа. Результаты поиска на диске С представлены ниже (Рисунок 4).

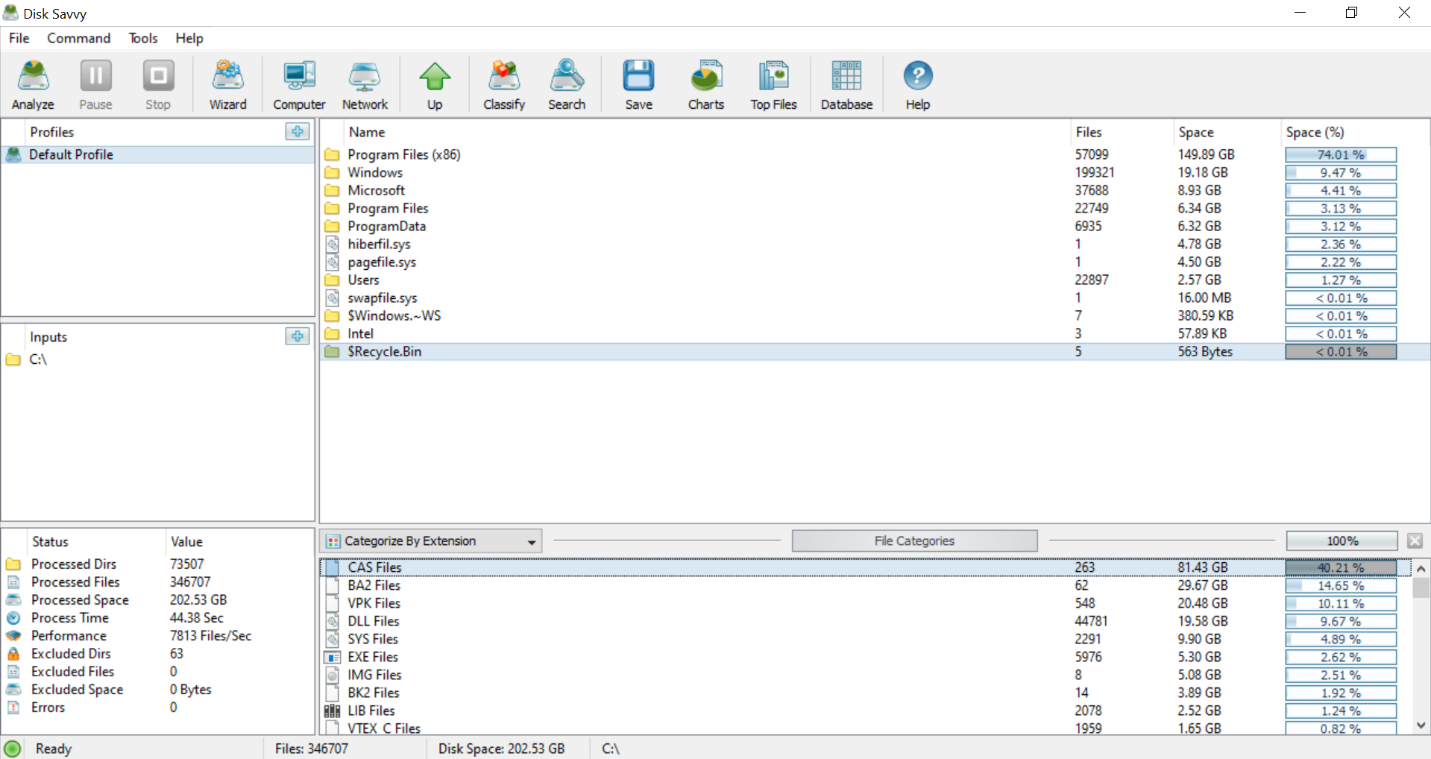


Рисунок 4 – Результаты поиска на диске С

Результаты поиска сортированы по возрастанию занимаемого объема на диске папками. Также внизу представлена таблица, сортированная по тому же принципу, однако, вместо папок файлы собираются по расширениям. Время поиска на том же диске составило 44 секунды, что почти в 2 раза больше, чем у CCleaner. Это говорит о меньшем быстродействии программы. Помимо этого, отображена информация по произведенному поиску, а именно: количество найденных файлов, папок, время поиска и средняя скорость обработки файлов.

Результат поиска можно конвертировать в базу данных или представить в виде диаграмм (Рисунок 5) и (Рисунок 6).

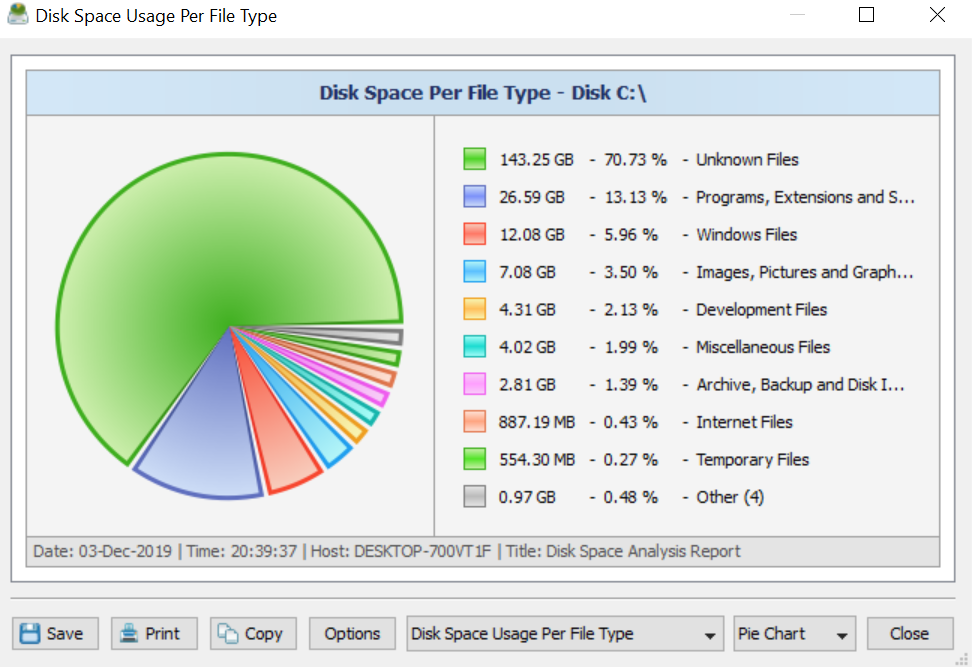


Рисунок 5 – Результаты поиска на диске С, представленные в виде диаграммы по категориям файлов

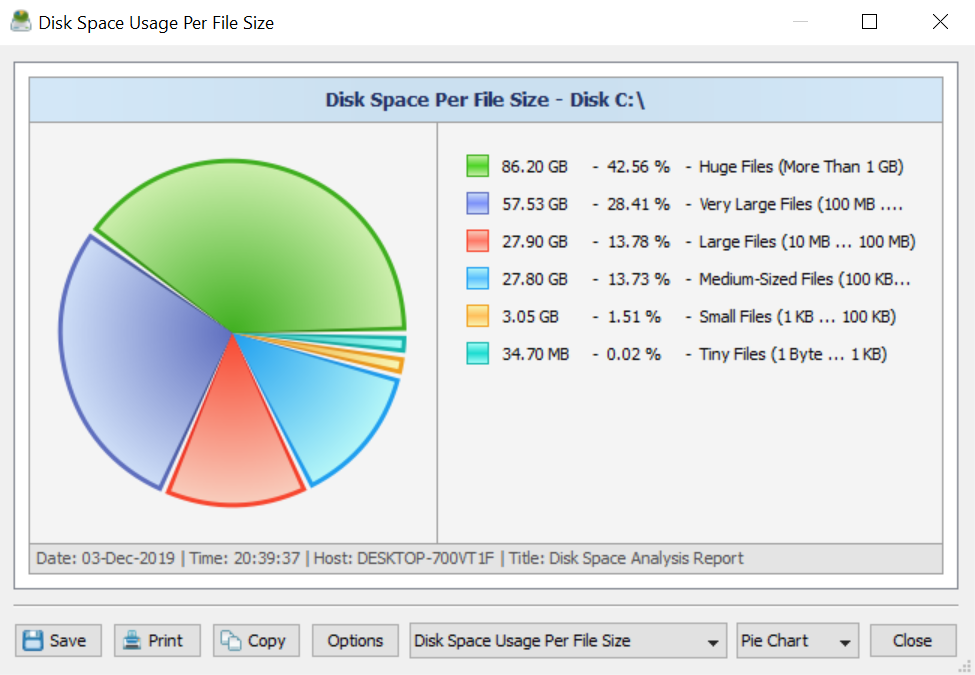


Рисунок 5 – Результаты поиска на диске С, представленные в виде диаграммы по величине файлов

В итоге программа CCleaner из-за большего количества инструментов по очистке компьютера имеет меньший функционал в опции анализа диска. Это скорее универсальная утилита, лишь имеющая функционал рассматриваемой темы, но не специализирующаяся на ней. Disk Savvy же имеет намного более расширенный инструментарий, позволяющий представлять полученные результаты в разных видах.

Таблица 1 — Сравнительная таблица систем-аналогов

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Программ­ный про­дукт | Про­стой ин­терфейс | Тонкая настрой-ка | Возмож­ность со­хранения результа­тов | Слож­ность в освоении | Эффек­тив­ность поиска | Время по­иска |
| CCleaner | + | - | - | - | + | + |
| Disk Savvy | + | + | + | - | + | - |

Исходя из плюсов и минусов представленных выше аналогов, было решено составить программу, имеющую преимущества и избегающую недостатки обоих утилит.

## **1.2 Общая характеристика и особенности процесса поиска файлов и подсчёта занимаемого ими объёма на диске.**

Поиск файлов и подсчёт занимаемого ими объёма на диске предоставляет пользователю возможность задать категории и путь. Это те параметры, которые будут учитываться при поиске и сортировке файлов.

Кроме того, пользователь может выбрать типы файлов, включаемых в категории. Это позволяет дифференцировать нужные для пользователя файлы от тех, в поиске которых нет необходимости. Например, рядовой пользователь вряд ли захочет производить поиск среди типов, файлы которых нужны для функционирования системы. Чаще всего люди интересуются занимаемым объемом файлами музыки, видео, картинок.

Более того, пользователь в праве указать папку, в которой будет производиться поиск файлов. Это сокращает время операции и выводит только нужную пользователю информацию.

После указания всех этих параметров производится процесс поиска файлов, сохранение информации о них по расширениям, прибавление их размера к общему объему занимаемому на диске категории.

## **1.3 Обзор и обоснование выбора инструментального программного обеспечения**

### **1.3.1 Обзор языков программирования**

Для разработки программы, производящей анализ занятого и свободного места на жестком диске, был рассмотрен ряд языков программирования: С++, Java, Python.

Первым рассмотрим С++. Поддерживает разные парадигмы программирования, но, в сравнении с его предшественником — языком Си, — наибольшее внимание уделено поддержке объектно-ориентированного и обобщённого программирования.

Таблица 2 – Плюсы и минусы C++

|  |  |
| --- | --- |
| Плюсы | Минусы |
| Управление памятью; большое количество доступной литературы и документации. | Высокий порог вхождения; необходимость уделять внимание мелким деталям, из-за чего трудно реализовать некоторые простые задачи; относительно бедная стандартная библиотека. |

Для Java характерен фокус на абстракциях; сильная статическая типизация; среда выполнения; ограничения на прямой доступ к памяти.

Таблица 3 — Плюсы и минусы Java

|  |  |
| --- | --- |
| Плюсы | Минусы |
| Автоматическое управление памятью, многопоточность, стабильное сообщество, простой синтаксис. | Повышенное требование к оперативной памяти. |

Среди характеристик Python можно отметить: сильное абстрагирование; динамическая, слабая типизация; полностью независимое управление памятью и наличие среды выполнения.

Таблица 4 — Плюсы и минусы Python

|  |  |
| --- | --- |
| Плюсы | Минусы |
| Абстракции существенно облегчают выполнение даже сложных задач; большие стандартные библиотеки; простой синтаксис. | Динамическая типизация может стать источником проблем в больших проектах; низкая производительность (рисунок 8). |



Рисунок 6 – Сводная таблица времени выполнения тестовых программ в секундах.

Тестовые программы состояли из 3 частей:

1. Загрузка данных из файла CSV.
2. Вычисления.
3. Вывод результатов.

Для данного проекта был выбрал язык С++, поскольку он обладает большим функционалом. Также я уже имею некоторый опыт разработки на этом языке с занятий в ВУЗе.

### **1.3.2 Обзор средств разработки**

Для создания программного комплекса были рассмотрены следующие среды разработки: Visual Studio, Code::Blocks , Eclipse.

Visual Studio — среда разработки от Microsoft, известна для написания приложений, включающих в себя .NET. Это полный набор инструментов, позволяющий произвести точную отладку и настройку приложения.

Таблица 5 — Плюсы и минусы Visual Studio

|  |  |
| --- | --- |
| Плюсы | Минусы |
| Поддержка других языков, таких как C++, C#, Python, JavaScript; удобный отладчик, позволяющий пошагово проследить выполнение программы и исправлять ошибки; регулярные обновления, расширяющие функционал, повышающие производительность. | Для работы требуются значительные ресурсы; высокий порог вхождения, без использования дополнительной литературы. |

Code::Blocks — среда разработки с открытым исходным кодом. Code::Blocks также ориентирован на C и C++.

Таблица 6 — Плюсы и минусы Code::Blocks

|  |  |
| --- | --- |
| Плюсы | Минусы |
| Поддержка большинства компиляторов, интегрированный список задач, отладчик, GUI; простота интерфейса; кроссплатформенность | Редкость официальных релизов; нет подсветки ошибок в редакторе; |

Eclipse – среда разработки, изначально ориентированная на работу с Java, прославилась большим количеством внешних модулей, существенно расширяющих её функциональность (в том числе, это касается количества поддерживаемых языков).

Таблица 7 — Плюсы и минусы Eclipse

|  |  |
| --- | --- |
| Плюсы | Минусы |
| Множество пакетных решений, обеспечивающих многоязычную поддержку; интерфейс, ориентированный на задачи, включая уведомления в системном трее; автоматическое создание отчетов об ошибках. | Нехватка документации; отсутствие единого сообщества разработчиков. |

Подводя итог, далее будет представлена сравнительная таблица ключевых критериев средств разработки.

Таблица 8 — Сравнительная таблица сред разработки от 1 до 5

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий | Visual Studio | Eclipse | Code::Blocks |
| Функциональность | 5 | 4 | 4 |
| Скорость компиляции | 5 | 4 | 4 |
| Простота интерфейса | 3 | 4 | 5 |
| Портативность | Средняя | Хорошая | Хорошая |
| Доступность | Есть бесплатная версия | Бесплатно | Бесплатно |

Выбранной средой разработки стала Visual Studio, так как она многофункциональна, удобна в использовании.

# **2 ЦЕЛЬ И ЗАДАЧИ КУРСОВОГО ПРОЕКТА**

Целью данного курсового проекта является разработка программного комплекса для анализа занятого и свободного места на жестком диске, а также вывод общей информации о файлах.

Для достижения поставленных целей были определены следующие задачи:

* Сравнение систем-аналогов.
* Разработка формализованного описания.
* Разработка функциональной структуры.
* Разработка блок-схем алгоритма поиска файлов и анализа занятого и свободного места на жестком диске.
* Разработка UML диаграммы.
* Разработка интерфейса.
* Разработка структуры программного обеспечения.
* Определение требований к аппаратному и программному обеспечению.

# **3. ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ**

## **3.1 Формализованное описание процесса разработки программного комплекса для анализа занятого и свободного места на жестком диске**

Ниже представлена диаграмма формализованного описания процесса анализа занятого места на жестком диске.

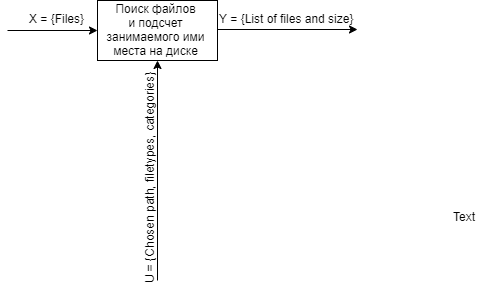


Рисунок 7 – Формализованное описание процесса анализа занятого места на жестком диске

Формализованное описание данного процесса состоит из одного основного блока: «Поиск файлов и подсчет занимаемого ими места на диске». Х — вектор входных данных, {Files} — файлы, считанные с диска. Y — вектор выходных данных, {List of files and size} — таблица файлов и объем, занимаемый ими на диске, а также информации о них. U — вектор управляющих воздействий, {Chosen path - заданный пользователем путь, filetypes – заданные пользователем типы файлов, categories – заданные пользователем категории}.

## **3.2 Постановка задач анализа занятого и свободного места на жестком диске**

В процессе написания программы была поставлена задача анализа занятого и свободного места на жестком диске.

Необходимо в заданном пользователем каталоге и его подкаталогах произвести поиск файлов, соответствующий указанным пользователем типам. Для этого нужно разработать алгоритм поиска файлов в каталоге и его подкаталогах, алгоритм определения соответствия типа файла тем, которые указал пользователь, также подсчет занимаемого ими места на диске. Необходимо в случае непопадания файла ни в одну из категорий отнести его к «Other», где будут храниться все оставшиеся файлы.

## **3.3 Функциональная структура программного комплекса для анализа занятого и свободного места на жестком диске**

Функциональная структура программного комплекса представляет собой четыре рабочих модуля: модуль выборов параметров поиска, модуль файловой системы, модуль поиска файлов и формирования таблицы, а также модуль отображения данных.

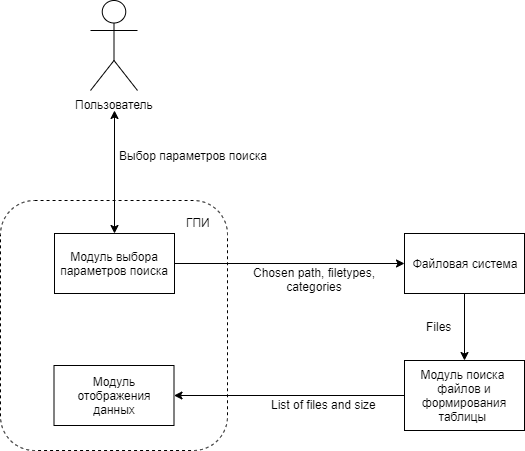


Рисунок 8 – Функциональная структура программного комплекса для анализа занятого и свободного места на жестком диске

Модуль выбора параметров поиска и модуль отображения данных находятся на графическом интерфейсе пользователя.

Модуль выбора параметров поиска получает выбранные пользователем путь, типы, категории файлов и передаёт их в модуль файловой системы.

Модуль файловой системы ищет в указанном каталоге и его подкаталогах файлы и передаёт их по одному в модуль поиска файлов и формирования таблицы.

Модуль поиска файлов и формирования таблицы считывает данные найденного файла, добавляет в специальный объект для хранения нужные данные, сохраняет в список файлов соответствующего расширения. По окончании сохранения данных о файле формируется таблица категорий и вложенных расширений, выбрав одно из которых можно увидеть список сохраненных файлов.

## **3.4 Алгоритм анализа занятого и свободного места на жестком диске**

### **3.4.1 Алгоритм поиска файлов**

Для анализа занимаемого объема файлами нужно сначала найти их в каталоге и подкаталогах. Для этого был создан следующий алгоритм:

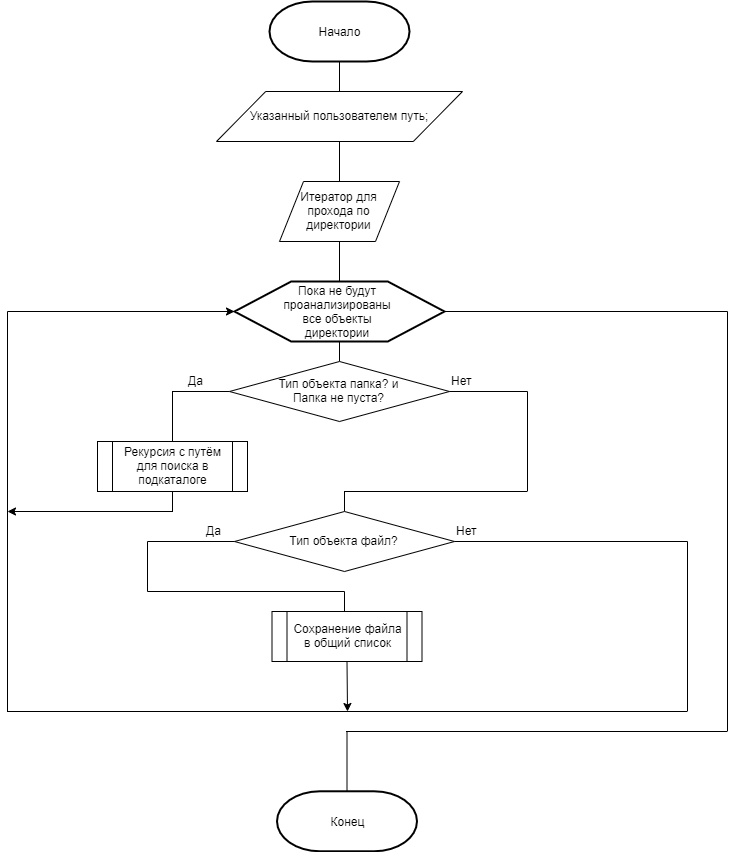


Рисунок 9 – Блок-схема алгоритма поиска файлов на диске

На этой блок-схеме (рисунок 9) представлен алгоритм поиска файлов в каталоге и подкаталогах. Сначала заводится итератор, проходящий по всем объектам заданного каталога. В случае если итератор наткнулся на папку, и она не пуста, происходит рекурсия с путём подкаталога. Если же тип объекта – файл, вызывается функция занесения файла в список соответствующего расширения и категории.

### **3.4.2 Алгоритм вставки файла согласно расширению и категории**

На представленной ниже блок-схеме представлен алгоритм вставки файла согласно расширению и категории.

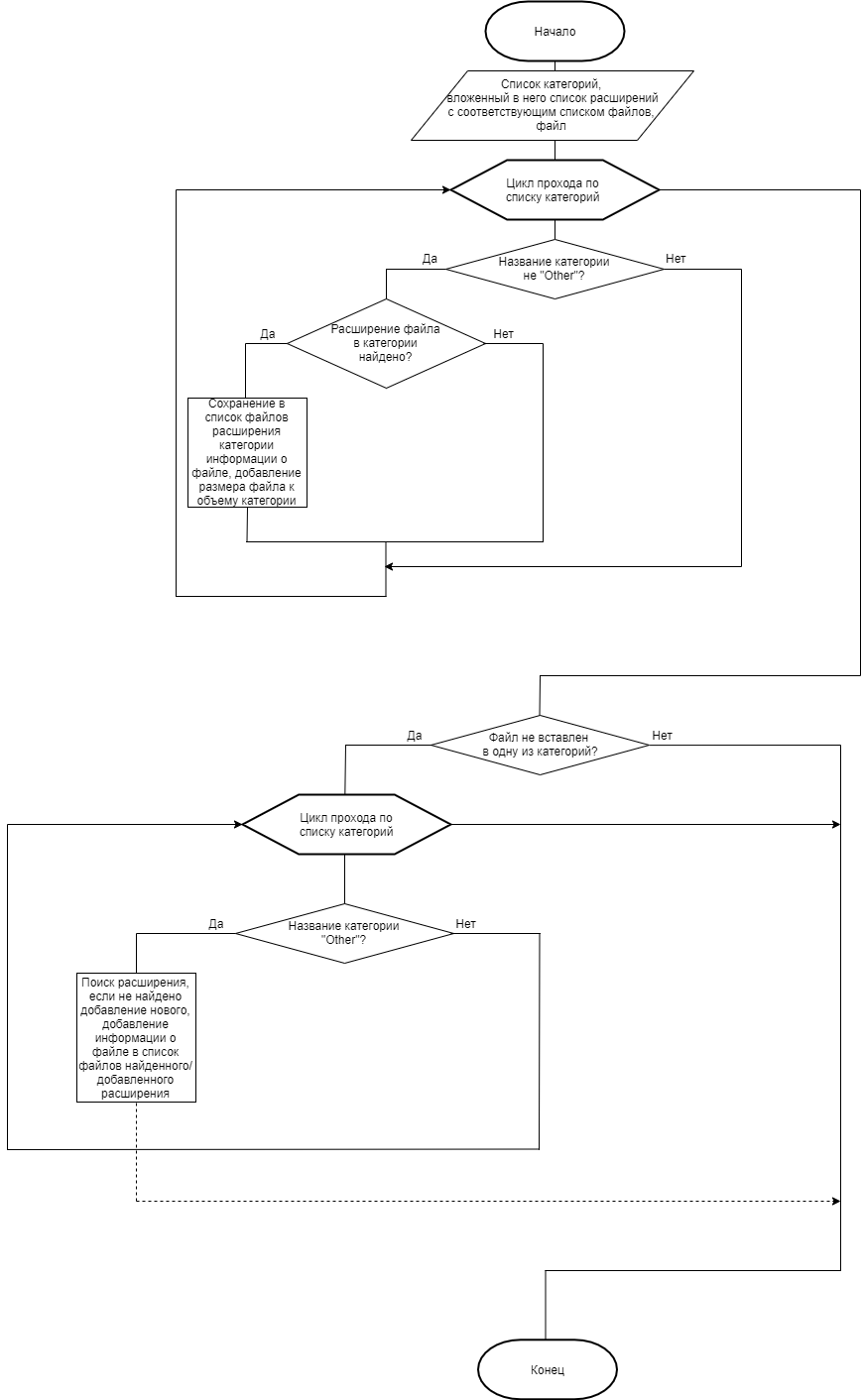


Рисунок 10 – Блок-схема алгоритма вставки файла согласно расширению и категории

Данный алгоритм начинается с перебора всех категорий в списке. Категория «Other» служит как «запасная», так как принимает в себя файлы, расширения которых не вошли в категории, заданные пользователем, поэтому в первую попытку сохранить информацию о файле, она не входит. Если было найдено расширение, соответствующее одной из пользовательских категорий, то информация о файле сохраняется в список файлов расширения, а размер файла прибавляется к общему размеру категории. Если же файл не вошел ни в одну из категорий, то среди них находится категория «Other», в ней ищется расширение файла. Если расширение найдено, то информация о файле вставляется в конец списка файлов расширения. Если оно не было найдено, в «Other» создается новое расширение со списком файлов, содержащим текущий.

## **3.5 Разработка структуры интерфейса для пользователя**

Одной из задач этого курсового проекта стала разработка интуитивно понятного и простого интерфейса, в котором пользователь без затруднений сможет найти нужные функции.

Сначала было решено разработать главную форму, где отображались бы все элементы, для взаимодействия пользователя с программой. Например, кнопка для выбора пути анализа, кнопка для выбора категорий и расширений, а также для анализа (Рисунок 11).

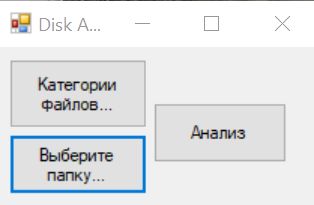


Рисунок 11 – Интерфейс, показываемый пользователю при запуске программы

Далее были добавлены элементы, отвечающие за отображение информации о результатах анализа, а именно: список категорий и расширений (Рисунок 12), диаграмма объемов категорий (Рисунок 13), список файлов для выбранного расширения (Рисунок 14).

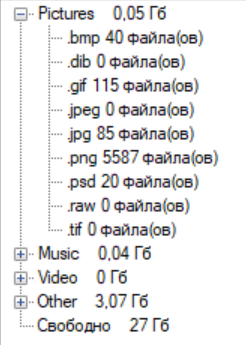


Рисунок 12 – Список категорий и расширений



Рисунок 13 – Диаграмма объемов категорий

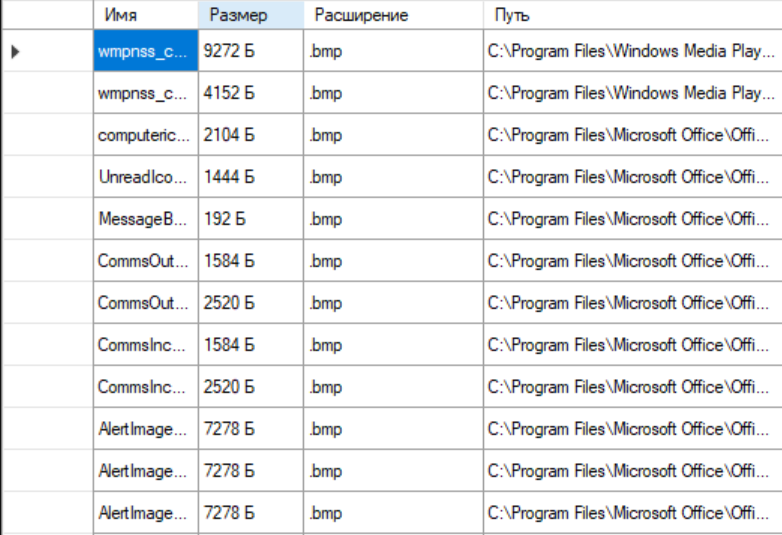


Рисунок 14 – Список файлов

После нажатия пользователем кнопки «Анализ» форма принимает свой конечный вид (Рисунок 15).

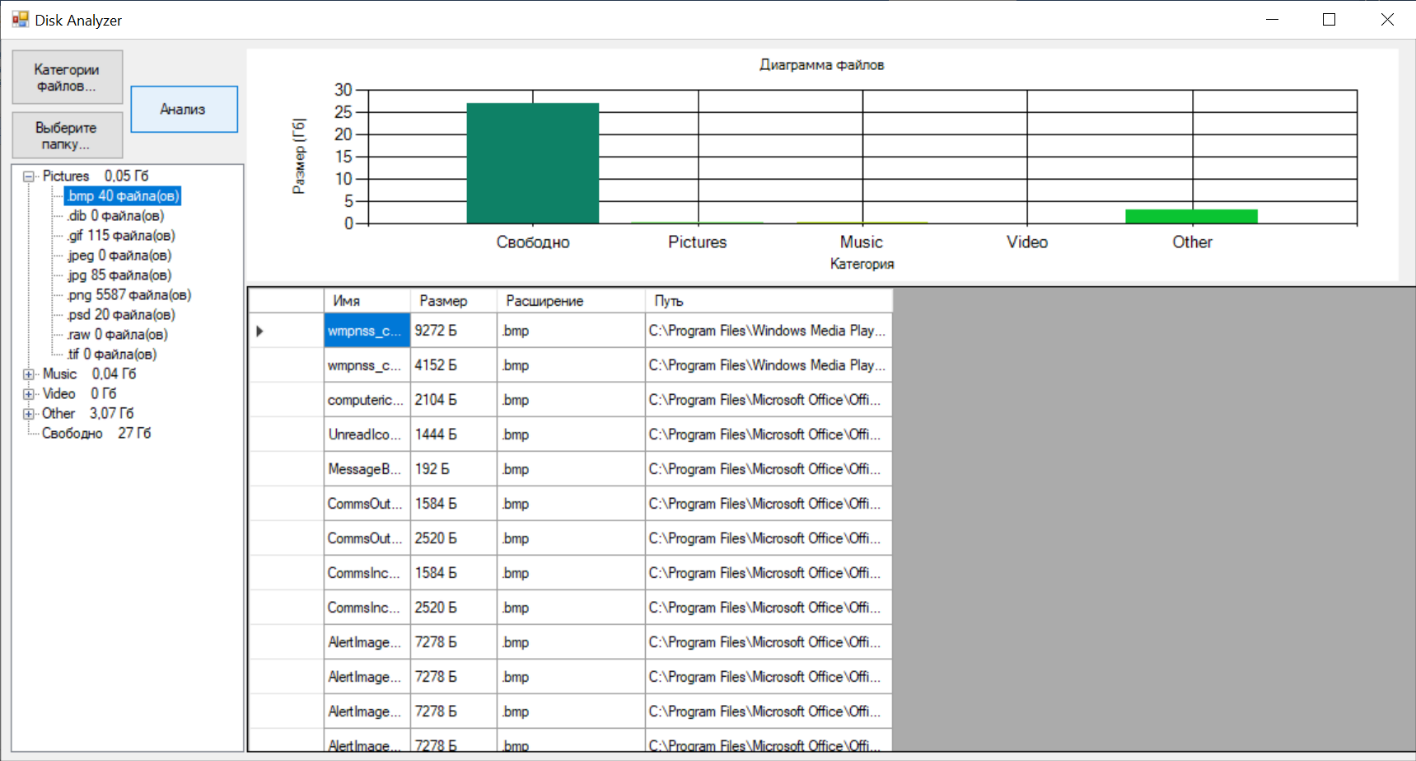


Рисунок 15 – Полный интерфейс программы

Для редактирования списка категорий и расширений была разработана отдельная форма. Она позволяет добавлять/удалять категории и расширения. Также существует возможность вернуться к стандартному списку категорий и расширений.

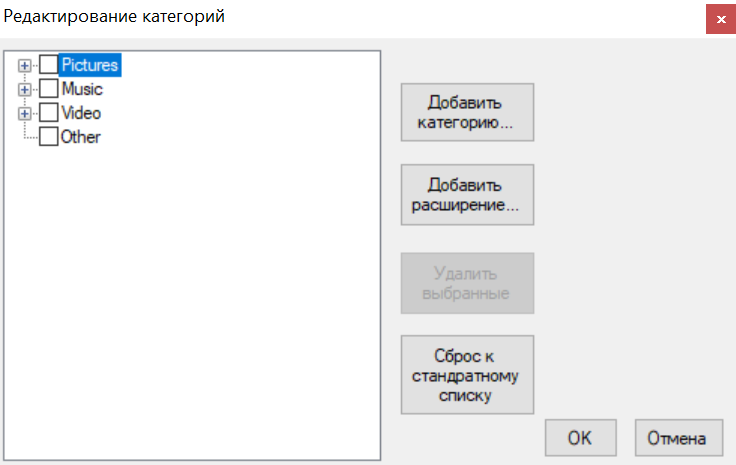


Рисунок 16 – Форма редактирования списка категорий и расширений

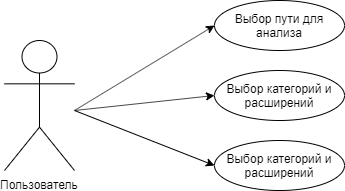


Рисунок 17 – UML-диаграмма прецедентов использования

## **3.6 Описание структур данных**

Эта программа использует две структуры данных. Первая – list. В ней хранятся категории файлов «Category» - структура, содержащая в себе объем, занимаемый на диске, название категории и список расширений, для которого используется контейнер map. Он хранит в себе пару ключ-значение. В виде ключа здесь используется расширение в формате данных string. Значение – ещё один list, но уже со структурой «File», содержащей название файла, размер, путь.

## **3.7 Описание структуры программы**

SearchAlgorithm – основной модуль программы, в котором содержатся функции поиска файлов, вставки информации о файле в категорию.

Таблица 9 — Функции, составляющие модуль SearchAlgorithm

|  |  |
| --- | --- |
| Функция | Комментарий |
| Search | Поиск файлов по каталогу и подкаталогам |
| Insert | Вставка информации о файле в категорию |

Structs – модуль, содержащий структуры данных, для хранения и сортировки информации о файлах. «Category» – структура, содержащая в себе название, объем, занимаемый на диске категорией, список расширений и файлов. «File» – структура для хранения информации о файле.

Interface – модуль, отвечающий за интерфейс главной формы. Он содержит в себе кнопки для взаимодействия пользователя с программой, структуры, отображающие результаты анализа. Также в нем содержатся функции, отвечающие за взаимодействие интерфейса программы и алгоритма анализа и взаимодействие с другими формами.

Таблица 9 — Функции, составляющие модуль Interface

|  |  |
| --- | --- |
| Функция | Комментарий |
| InitializeCatsList | Инициализация списка категорий и расширений |
| CategoriesChooseButton\_Click | Открытие формы для редактирования списка категорий и расширений |
| FolderChooseButton\_Click | Открытие проводника для выбора пути анализа |
| AnalyzeButton\_Click | Вызов алгоритма анализа, занесение результатов анализа в структуры формы |
| CatsAndExtsTree\_AfterSelect | Отображение файлов выбранного расширения в соответствующем списке |

CatsChoose – модуль, в котором пользователь может добавить/удалить категорию или расширение или сбросить их список на стандартный.

Таблица 10 — Функции, составляющие модуль CatsChoose

|  |  |
| --- | --- |
| Функция | Комментарий |
| InitializeTreeView | Инициализация списка категорий и расширений |
| deleteCatsOrExtsButton\_Click | Удаление отмеченных категорий/расширений |
| treeView\_AfterCheck | Для активация/дезактивации кнопки удаления категории или расширения |
| RemoveElemsFromTreeView | Удаление элемента из списка |
| resetButton\_Click | Сброс к стандартному списку |
| okButton\_Click | Сохранение списка |
| AbortButton\_Click | Выход из формы без сохранения списка |
| TreeView\_AfterSelect | Для активация/дезактивации кнопки добавления категории или расширения |
| AddCategoryButton\_Click | Добавление категории в список |
| AddExtButton\_Click | Добавление расширения в список |

AddCatOrExt – модуль, отвечающий за ввод пользователем названия категории/расширения, его проверку и возврат в список.

Таблица 11 — Функции, составляющие модуль AddCatOrExt

|  |  |
| --- | --- |
| Функция | Комментарий |
| OkButton\_Click | Проверка введенного названия, его возврат в список, закрытие формы |
| AbortButton\_Click | Закрытие формы |
| AddCategory | Задает маску, характерную для категории файлов |
| AddExtension | Задает маску, характерную для расширений файлов |
| textBox\_KeyPress | Эквивалент функции OkButton\_Click, в случае, если пользователь нажал Enter |
| ExportCatOrExt | Вывод расширения/категории в список |

## **3.8 Тестирование программного комплекса**

Для тестирования программного комплекса на рабочий стол был скопирован файл 1.bmp размером 1,51 Гб. Его размер был прибавлен к общему объему категории «Pictures», содержащей расширение bmp.

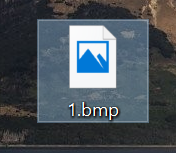


Рисунок 18 – Файл 1.bmp на рабочем столе

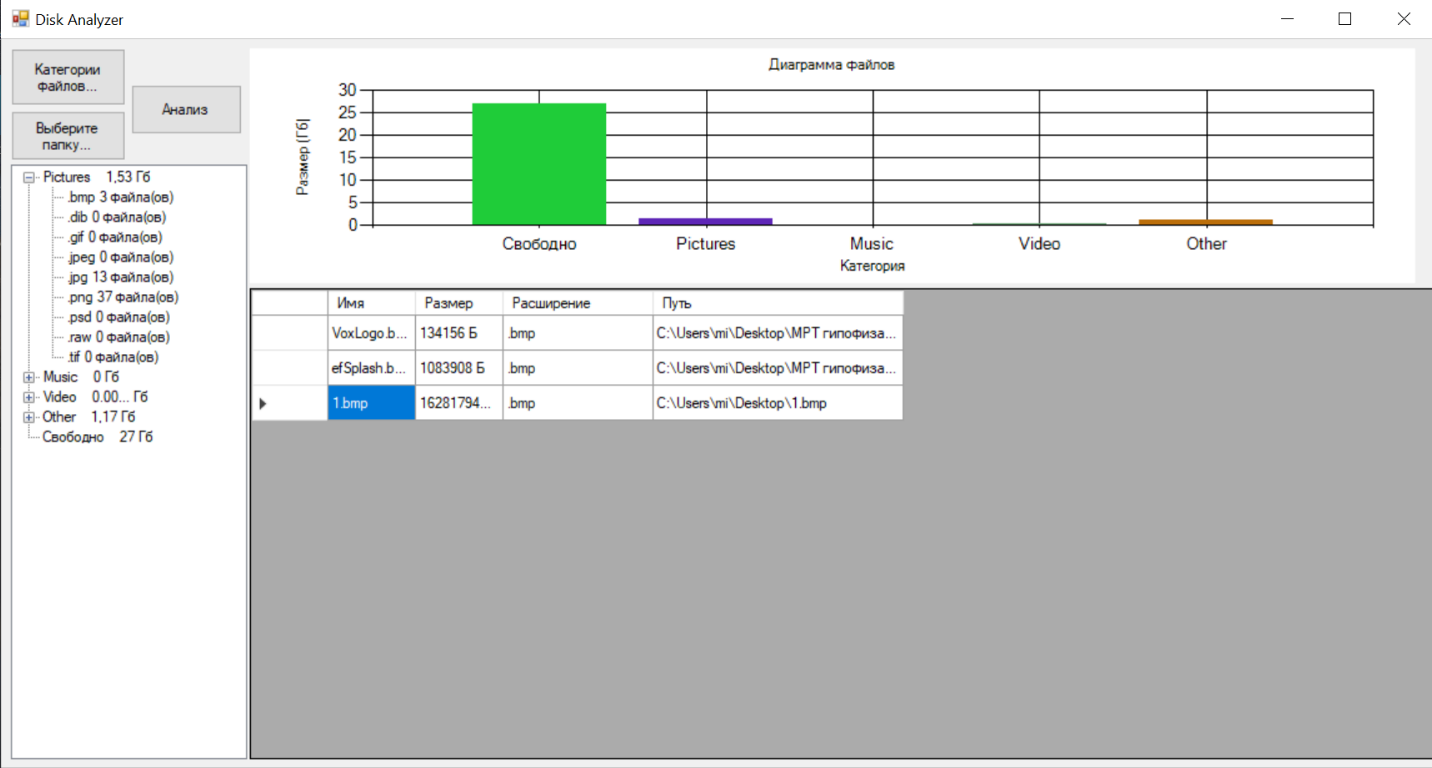


Рисунок 19 – Результат анализа рабочего стола с файлом 1.bmp

Как видно из результата выше, программа нашла файл 1.bmp и занесла его в расширение «.bmp» категории «Pictures».

Далее у этого же файла было изменено расширение на «.avi».

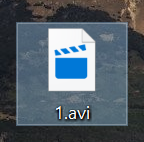


Рисунок 20 – Файл 1.avi на рабочем столе

После был произведён анализ, в результате которого объем категории «Pictures» был уменьшен на 1,51 Гб, а файл 1.bmp пропал из списка файлов расширения «.bmp». Файл 1.avi был занесен в категорию «Video» (Рисунок 21).

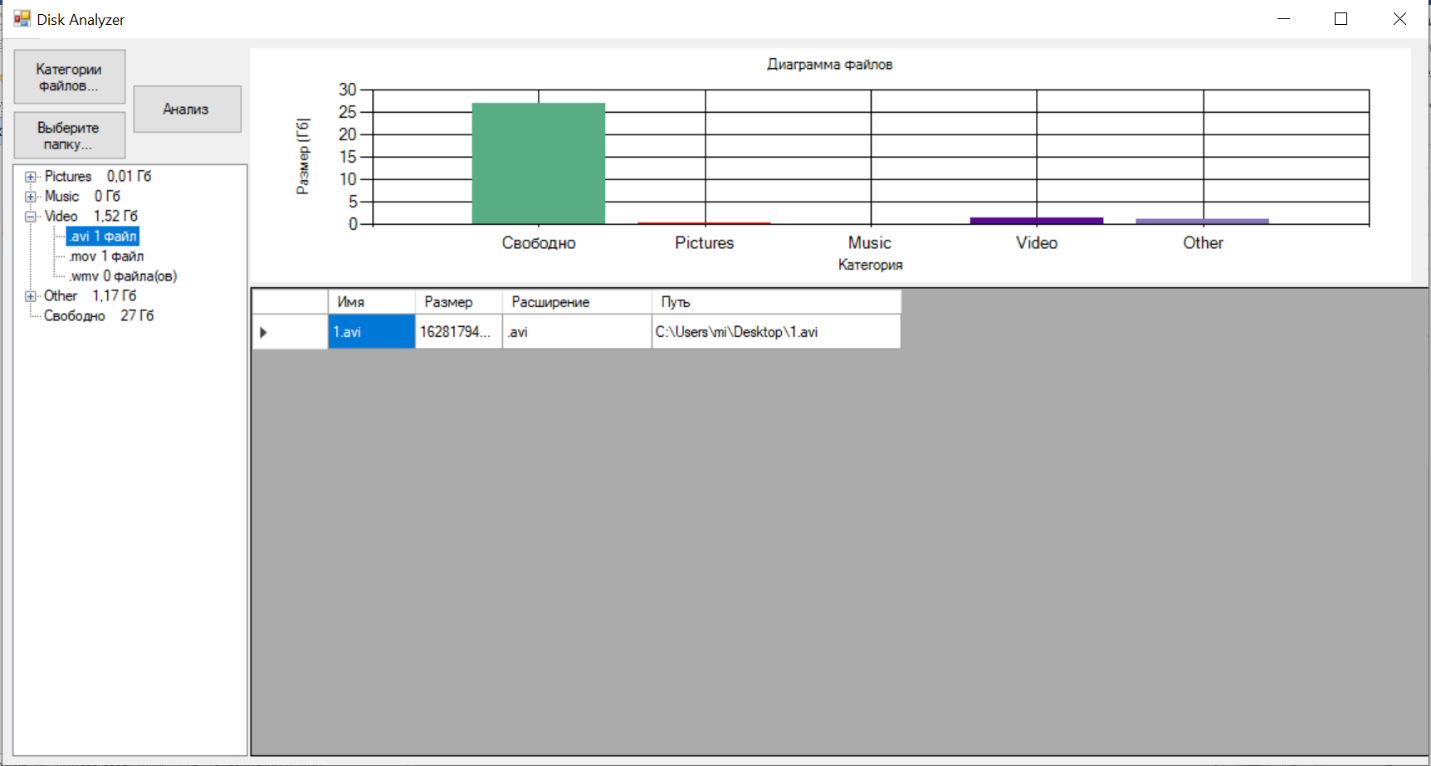


Рисунок 21 – Результаты анализа после изменения расширения файла 1 на «.avi»

**3.9 Характеристика программного и аппаратного обеспечения**

Программный комплекс разработан под управлением ОС Windows 10 на платформе .Net Framework 4.8. Средой разработки является Microsoft Visual Studio 2019.

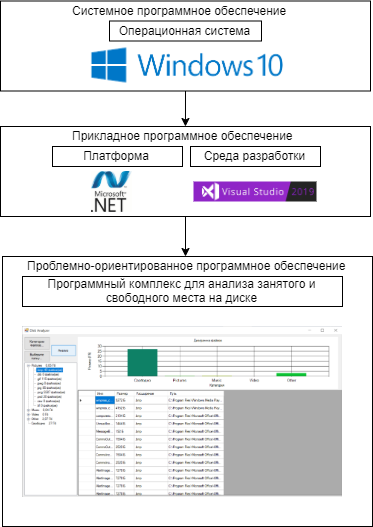
****

Рисунок 22 – Структура программного обеспечения

Таблица 12 – Минимальные системные требования

|  |  |
| --- | --- |
| Показатель | Значение |
| Тип ЭВМ | Персональный компьютер |
| Тактовая частота процессора, ГГц | 1,8 |
| Объём оперативной памяти, Гб | 8 |
| Объём внешней памяти, Гб | 256 |
| Состав и характеристика периферийных устройств ЭВМ | Клавиатура, мышь, монитор с диагональю 13.3′′ и разрешением 1920×1080 точек |
| Операционная система | Windows 10 |
| Прикладное программное обеспечение | .NET Framework 4.8 |

Таблица 13 – Характеристика программного обеспечения

|  |  |
| --- | --- |
| Показатель | Значение |
| Среда разработки | Microsoft Visual Studio 2019 |
| Технология программирования | ООП |
| Язык программирования | C++ |
| Количество входных переменных | 2 |
| Количество выходных переменных | В зависимости от результатов поиска |
| Количество классов | 3 |
| Количество функций | 22 |
| Размер исполняемого файла, Кб | 768 |
| Время обработки данных и визуализации результатов | В зависимости от количества файлов |

**ВЫВОДЫ**

Результатом проделанной работы стал программный комплекс для анализа занятого и свободного места на жестком диске, с указанными путем анализа, категориями и расширениями.

В ходе разработки данного программного комплекса были выполнены следующие задачи:

* Был произведен обзор языков программирования, в результате которого был выбран такой язык, как C++.
* Был произведен обзор сред разработки, подходящих для данного языка, в результате которого в качестве среды разработки была выбрана Visual Studio.
* Было разработано формализованное описание данного процесса, состоящее из одного основного блока: «Поиск файлов и подсчет занимаемого ими места на диске». Входными данными являются заданные пользователем путь, категории и расширения файлов. Выходные данные - таблица файлов и информации о них.
* Была разработана функциональная структура программного комплекса, представляющая собой четыре рабочих модуля: модуль выборов параметров поиска, модуль файловой системы, модуль поиска файлов и формирования таблицы, а также модуль отображения данных.
* Была разработана UML-диаграмма прецедентов использования.
* Были построены блок-схемы алгоритма поиска файлов на диске и алгоритма вставки файла согласно расширению и категории.
* Было проведено тестирование программного комплекса, в ходе которого была продемонстрирована работа программы с разными расширениями файла.

В будущем хотелось бы улучшить разработанную программу, а именно:

* Ускорить работу программы.
* Добавить возможность предпросмотра файлов.
* Сделать интерфейс более эстетически приятным.
* Добавить индикаторы прогресса процесса анализа поиска.

# **СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

1. Страуструп, Б. Programming: Principles and Practice Using C++ / Б. Страуструп. – М. : Вильямс, 2011. – 1248 c.

2. Норенков, И.П. Основы автоматизированного проектирования / И.П. Норенков. – М. : МГТУ им. Н. Э. Баумана, 2009. – 430 c.

3. Вильямс, Я.Ш. C++: базовый курс / Я.Ш. Вильямс. – М. : Вильямс, 2014. – 624 с.

4. Липпман, C. Основы программирования на C++ / С. Липпман. – М. : Вильямс, 2014. – 1104 с.

5. Олифер, Н. А., Олифер, В. Г. Сетевые операционные системы / Н. А. Олифер В. Г. Олифер – СПб: Питер, 2009. – 669 с.

6. Майерс, С. Эффективное использование C++ / С. Майерс – СПб: ДМК Пресс, 2014. – 300 с.

7. Руссинович М., Соломон Д., Ионеску А. Внутреннее устройство Microsoft Windows. Основные подсистемы ОС / М. Руссинович, Д. Соломон, А. Ионеску – СПб: Питер СПБ, 2014. – 672 с.

8. cppreference.com [Электронный ресурс] URL: <https://cppreference.com/> (дата обращения: 01.12.2019)