МИНИСТЕРСТВА НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования «Самарский национальный исследовательский   
университет имени академика С.П. Королева»

(Самарский университет)

|  |  |
| --- | --- |
| Институт | информатики, математики и электроники |
| Факультет | информатики |
| Кафедра | геоинформатики и информационной безопасности |

КУРСОВОЙ ПРОЕКТ ПО ДИСЦИПЛИНЕ

«Алгоритмы и структуры данных»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

«Разработка программных средств автоматического резервного копирования данных пользователя»

|  |  |
| --- | --- |
| Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *(подпись)* | Р.Д. Данилов |
|  |  |
| Руководитель работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *(подпись)* | Е.В. Мясников |
|  |  |

САМАРА 2022

МИНИСТЕРСТВА НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования «Самарский национальный исследовательский   
университет имени академика С.П. Королева»

(Самарский университет)

|  |  |
| --- | --- |
| Институт | информатики, математики и электроники |
| Факультет | информатики |
| Кафедра | геоинформатики и информационной безопасности |

**ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ**

Студенту ***Данилову Роману Дмитриевичу*** группы 6312

Тема проекта: ***«***Разработка программных средств автоматического резервного копирования данных пользователя***»***

|  |  |  |
| --- | --- | --- |
| Планируемые результаты освоения образовательной программы (компетенции) | Планируемые результаты практики | Содержание задания |
| ОПК-3 ‑ способность применять языки, системы и инструментальные средства программирования в профессиональной деятельности | Знать: основные виды программных средств, технологию разработки алгоритмов и программ и методы их отладки, основы объектно-ориентированного подхода к программированию  Уметь: работать с современными системами программирования, самостоятельно осваивать новые программные средства  Владеть: языками процедурного и объектно-ориентированного программирования, навыками разработки и отладки программ. | 1. Изучить виды резервного копирования данных и выбрать те, которые будут присутствовать в программе.  2. Изучить аналоги программ, решающих такую же задачу.  3. Разработать алгоритм копирования данных пользователя.  4. Разработать понятный дизайн пользовательского интерфейса  5. Собрать программу и протестировать её. |

Дата выдачи задания 5 сентября 2022 г.

Срок представления на кафедру отчета о практике 21 декабря 2022 г.

Руководитель курсового проекта

доцент каф. ГИиИБ, к.т.н. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.В. Мясников

*(подпись)*

Задание принял к исполнению

студент группы № 100503D-6312 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Р.Д. Данилов

*(подпись)*

РЕФЕРАТ

**Пояснительная записка к курсовому проекту:** 36 c., 25 рисунков, 3 источника, 1 приложение.

АВТОМАТИЧЕСКОЕ РЕЗЕРВНОЕ КОПИРОВАНИЕ, РАБОТА С АРХИВАМИ, РАБОТА С ФАЙЛОВОЙ СИСТЕМОЙ

Цель работы – написание программного средства автоматического резервного копирования данных пользователя.

Мною был выполнен обзор существующих методов резервного копирования и программ, осуществляющих решение этой задачи. Благодаря этому, был составлен алгоритм автоматического копирования данных пользователя. Была разработана программа, имеющая понятный пользовательский интерфейс и использующая этот алгоритм как основу для своей работы. Собранная программа, кроме выполнения копирования, имеет способность работы в фоне. При том, что она использует несколько потоков в своей работе, производит нагрузку на ЦП менее 1%. Была введена некая защита от вмешательства в её работу путем изменения используемых файлов, программа не только уведомляет пользователя о проблеме, но и самостоятельно решает её в большинстве случаев. В тестах на работоспособность не было выявлено никаких сбоев. Автоматическое копирование проходит успешно без участия пользователя.

Содержание

[ВВЕДЕНИЕ 6](#_Toc122393498)

[1 АНАЛИЗ И РАЗРАБОТКА МЕТОДОВ РЕЗЕРВНОГО КОПИРОВАНИЯ 7](#_Toc122393499)

[1.1 Анализ методов резервного копирования 7](#_Toc122393500)

[1.2 Существующие программные средства резервного копирования 8](#_Toc122393501)

[1.3 Разработка алгоритма резервного копирования 9](#_Toc122393502)

[1.4 Выводы и результаты 11](#_Toc122393503)

[2 ОПИСАНИЕ РАЗРАБОТАННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 12](#_Toc122393504)

[2.1 Выбор языка, средств разработки, системных программных средств……………………………………………………………………………12](#_Toc122393505)

[2.2 Требования к разрабатываемой программе 12](#_Toc122393506)

[2.3 Структура программы 13](#_Toc122393507)

[2.3.1 Описание основных классов 14](#_Toc122393508)

[2.3.2 Описание основных функций резервного копирования 18](#_Toc122393509)

[2.4 Описание интерфейса пользователя 25](#_Toc122393510)

[2.5 Тестирование работоспособности программы 32](#_Toc122393511)

[2.6 Выводы 34](#_Toc122393512)

[ЗАКЛЮЧЕНИЕ 35](#_Toc122393513)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 36](#_Toc122393514)

[ПРИЛОЖЕНИЕ А 37](#_Toc122393516)

ВВЕДЕНИЕ

Актуальность темы резервного копирования не вызывает никаких сомнений. Физические носители информации имеют неприятное свойство выходить из строя или теряться. Кроме этого существует вредоносное ПО, которое также может привести к потере данных. Как правило, такое происходит в самый важный момент, и ценная информация может быть утеряна навсегда.

Сейчас существует более чем предостаточно способов застраховать себя от потери данных и каждый может выбрать для себя приемлемый вариант сохранения важных данных, будь то сервисы облачного хранения данных или отдельные физические носители. В настоящее время существует множество программ, помогающих делать резервные копии важных файлов без участия пользователя.

В данной работе задача состоит в написании программы для автоматического резервного копирования данных пользователя. В соответствии с поставленной целью были сформулированы и решены следующие задачи:

1. Изучить виды резервного копирования данных и выбрать те, которые будут присутствовать в программе.
2. Изучить аналоги программ, решающих такую же задачу.
3. Разработать алгоритм копирования данных пользователя.
4. Разработать понятный дизайн пользовательского интерфейса.
5. Собрать программу и протестировать её.

# АНАЛИЗ И РАЗРАБОТКА МЕТОДОВ РЕЗЕРВНОГО КОПИРОВАНИЯ

## Анализ методов резервного копирования

Резервное копирование данных — процесс создания копии данных пользователя.

Рассмотрим методы резервного копирования:

* Полное резервное копирование (Full backup)

Полное копирование обычно затрагивает всю систему и все файлы. Еженедельное, ежемесячное и ежеквартальное резервное копирование подразумевает создание полной копии всех данных.

* Дифференциальное резервное копирование (Differential backup)

При дифференциальном («разностном») резервном копировании каждый файл, который был изменён с момента последнего полного резервного копирования, копируется каждый раз заново.

* Инкрементное резервное копирование (Incremental backup)

При добавочном («инкрементном») резервном копировании происходит копирование только тех файлов, которые были изменены с тех пор, как в последний раз выполнялось полное или добавочное резервное копирование. Последующее инкрементное резервное копирование добавляет только файлы, которые были изменены с момента предыдущего.

* Резервное копирование в виде образа

Образ — точная копия всего раздела или носителя (устройства), хранящаяся в одном файле.

* Резервное копирование в режиме реального времени

Резервное копирование в режиме реального времени позволяет создавать копии файлов, каталогов и томов, не прерывая работу, без перезагрузки компьютера.

* Клонирование

Клонирование позволяет скопировать целый раздел или носитель (устройство) со всеми файлами и каталогами в другой раздел или на другой носитель. Если раздел является загрузочным, то клонированный раздел тоже будет загрузочным.

## Существующие программные средства резервного копирования

Как говорилось ранее, на данный момент времени существует множество программ и сервисов, помогающих делать резервное копирование. Ниже мы рассмотрим несколько из них:

1. Cobian Backup

Приложение заточено под регулярное копирование информации на компьютере под Windows. Ее можно настроить на автоматическое копирование нужных папок. Можно выбрать резервное копирование со сжатием или без. Помимо этого, программа предлагает множество других функций, а также понятный русифицированный интерфейс.

1. Aomei Backupper

Имеет схожий функционал с Cobian Backup, но более современный дизайн интерфейса. В функции софта входит: резервирование и восстановление дисков Windows, разделов, отдельных файлов; полное, инкрементное, дифференциальное копирование; клонирование дисков/разделов; синхронизация файлов.

1. Handy Backup и Яндекс диск

Яндекс диск – это сервис, который позволяет пользователю хранить файлы на серверах Яндекса. Можно работать с файлами на Диске с любого устройства, подключенного к интернету.

Handy Backup – программа для резервного копирования файлов на Яндекс диск. Решение позволяет легко и удобно сохранять резервные копии различных данных с локальных и внешних дисков в облачное хранилище Яндекс Диск без использования дополнительных приложений или выделенных локальных папок.

Этот список на самом деле гораздо больше. Из этого можно сделать вывод, что для современных пользователей сейчас доступны множество решений, помогающих в нашей задаче.

## Разработка алгоритма резервного копирования

В работе был разработан алгоритм резервного копирования, он представлен на рисунке 1. Рассмотрим его подробнее.

Программа в фоновом режиме просматривает заранее созданный файл с текущими задачами, которые задает пользователь. Когда подходит время выполнения какой-либо задачи, программа вызывает функцию копирования, которая принимает такие данные: путь до директории, которую программа должна скопировать; путь до директории, куда будет происходить резервное копирование; индивидуальное имя задачи; интервалы времени, между резервными копиями; тип копирования (полное или дифференциальное). Основными данными здесь являются пути, которые собственно программа и проверяет на достоверность, а именно проверяет существование таких директорий на компьютере. Если директории не найдены, задача попросту удаляется из списка, и программа продолжает работать дальше. Если же проверка пройдена продолжается алгоритм копирования.

Узнав метод копирования, алгоритм выбирает какую функцию соответственно ему вызвать. Полное копирование может происходить с архивацией данных и без неё, дифференциальное копирование не имеет такой возможности.

При полном копировании с архивацией данных вызывается функция, использующая стороннюю подпрограмму 7-Zip, которая встроена в этот проект. Таким образом на выходе мы получаем архив, состоящий из полной копии директории, путь до которой получал алгоритм.

При полном копировании происходит вызов функции, которая рекурсивно обходит директорию и все ее подразделы, записывая в список пути до файлов.

При дифференциальном копировании происходит вызов аналогичной функции, с некоторыми изменениями. А именно, функция записывает в список не все файлы и подразделы, а только те, которые были изменены с момента последнего копирования. Такой метод позволяет ускорить процесс копирования, так как алгоритм будет делать копии только тех файлов, которые были изменены.

Программа, получив список, состоящий из путей до файлов, которые следует копировать, делает копию. Затем в список задач вносится изменение, меняется время выполнения задачи на системное время, и применив ранее полученные данные о временных интервалах, меняется время выполнения следующего копирования. На выходе мы соответственно получаем копии тех файлов, которые были в списке, по указанному пути резервной копии.

На этом алгоритм заканчивает свою работу.

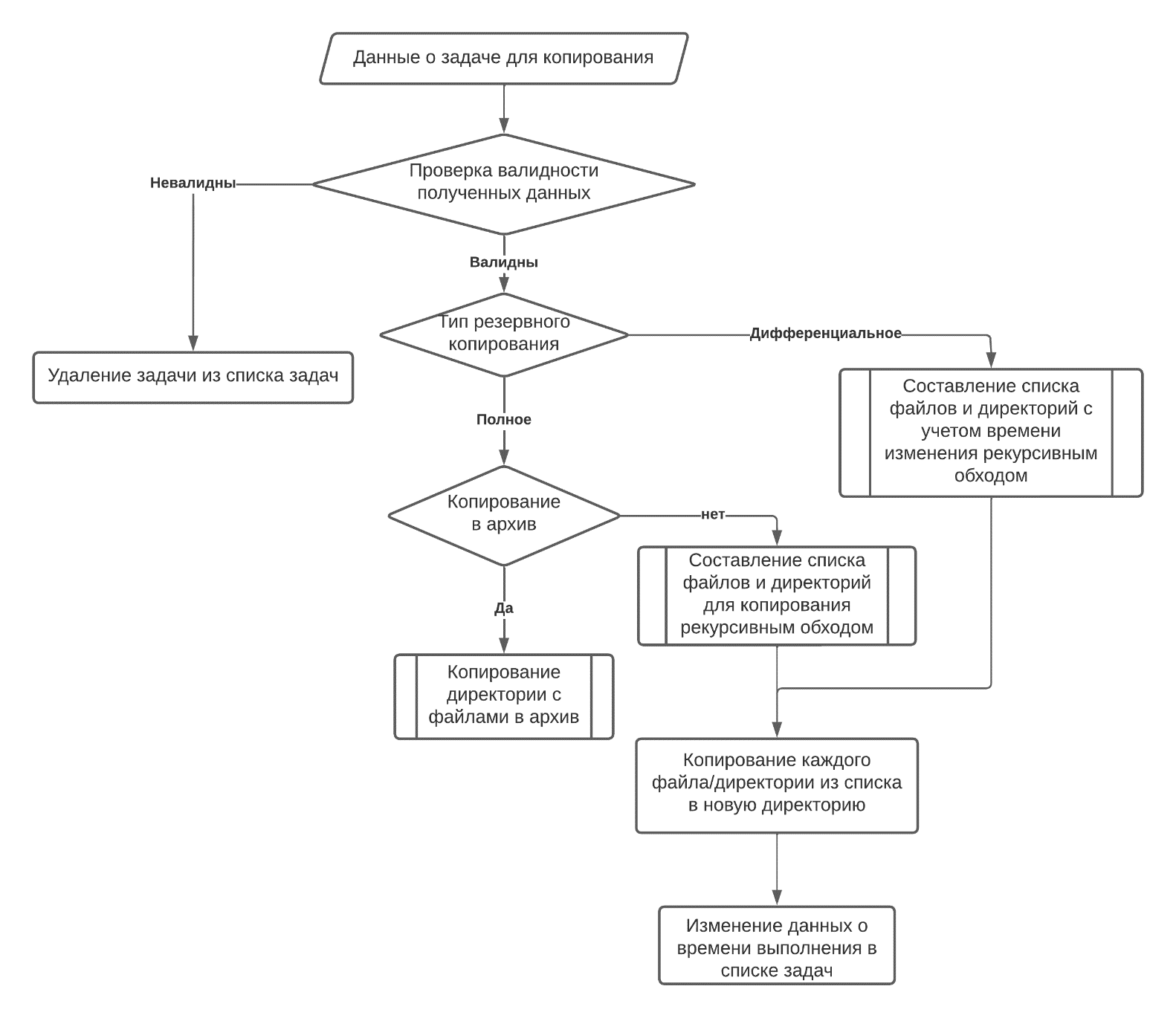


Рисунок 1 – Описание алгоритма резервного копирования в виде блок-схемы

## Выводы и результаты

Подробно изучив методы резервного копирования и существующие программные средства, мною были выбраны два вида резервного копирования для реализации в этой работе, а именно: полное и дифференциальное. Полное копирование – это основной метод резервного копирования, который подразумевает под собой любая подобная программа. А дифференциальное копирование существенно ускоряет этот процесс и экономит ресурсы компьютера при выполнении. В паре эти методы могут покрыть большинство требований обычного пользователя.

# ОПИСАНИЕ РАЗРАБОТАННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

## Выбор языка, средств разработки, системных программных средств

Для написания программы выбран язык С++ и для этого есть несколько причин:

* Быстрота исполнения кода
* Универсальность языка (Компиляторы C++ есть на каждой операционной системе, большинство программ легко переносится с платформы на платформу)
* Язык активно поддерживается (речь не только об обновлениях, но и об обширном сообществе и качественной литературе)

В качестве среды разработки был выбран Qt Creator. Благодаря встроенному функционалу по работе с графическим интерфейсом, задача написания программы дружелюбной для пользователя становится проще. Кроме этого в ней имеются: встроенные библиотеки с практичными классами, разработанными Qt; удобное межпроцессорное взаимодействие; обилие хорошей документации.

В программе используется дополнительное программное обеспечение 7-Zip. Это бесплатный файловый архиватор с открытым исходным кодом.

Для работы программы требуется компьютер под управлением Windows.

## Требования к разрабатываемой программе

Список функциональных требований к программе:

* Программа должна делать копии согласно заданному пользователем временному интервалу.
* Программа должна делать копии по запросу пользователя.
* Программа должна позволять пользователю добавлять/изменять/удалять задачи по резервному копированию.
* Программа должна уметь работать в фоне, до отключения ПК или до отключения программы пользователем.

Список нефункциональных требований к программе:

* Программа должна иметь понятный пользовательский интерфейс для удобного взаимодействия пользователя с ней.
* Программа должна поддерживать свою работу на операционной системе Windows.
* Программа должна уведомлять о сбоях в работе.

## Структура программы

Ниже мы рассмотрим исполняемые файлы, присутствующие в программе и их краткое описание.

* addtaskwindow.cpp: в файле хранится класс AddTaskWindow, который отвечает за открытие окна добавления задачи и соответствующий функционал.
* changetaskwindow.cpp: в файле хранится класс ChangeTaskWindow, который отвечает за открытие окна изменения задачи и соответствующий функционал.
* copyfunctions.cpp: в файле хранится глобальная функция резервного копирования, а также используемые ею функции составления списка копируемых элементов.
* deletetaskwindow.cpp: в файле хранится класс DeleteTaskWindow, который отвечает за открытие окна удаления задачи и соответствующий функционал.
* executenowwindow.cpp: в файле хранится класс ExecuteNowWindow, который отвечает за открытие окна выбора задачи, которую пользователь хочет выполнить сейчас.
* helpwindow.cpp: в файле хранится класс HelpWindow, который отвечает за открытие окна справки.
* main.cpp: основной файл программы, который открывает главное окно и запускает цикл. Этот цикл просматривает файл с текущими задачи и по необходимости вызывает функцию копирования. Также тут содержится код, который при старте программы проверяет наличие нужных для работы программы файлов, таких как: список задач и конфигурационный файл.
* mainwindow.cpp: файл содержащий класс MainWindow, который отвечает за открытие и обновление главного окна программы, а также создание log-файла журнала с событиями.
* settingswindow.cpp: в файле хранится класс SettingsWindow, который отвечает за открытие окна настроек и изменение конфигурационного файла.

### Описание основных классов

Ранее мы рассмотрели структуру программы и затронули классы, содержащиеся в исполняемых файлах. Рассмотрим класс MainWondow.

Он содержит множество методов. Основная задача этого класса обеспечить работу пользовательского интерфейса и связать всю программу воедино. В классе присутствуют методы для вызова всех окон содержащихся в программе.

Метод timerEvent (рисунок 2) вызывается каждую секунду работы программы, он сверяется с конфигурационным файлом и исходя из полученных данных строит пользовательский интерфейс, а также обновляет журнал событий и таблицу задач с помощью вызова методов log\_event\_fill и task\_table\_fill.



Рисунок 2 – Код метода timerEvent

Метод log\_event\_fill (рисунок 3) заполняет журнал событий исходя из данных полученных из log-файла. Это помогает пользователю понимать, что в данный момент делает программа.



Рисунок 3 – Код метода log\_event\_fill

Метод task\_table\_fill (рисунок 4) заполняет таблицу задач исходя из данных полученных из файла с текущими задачами. Таким образом пользователь может видеть полный список задач по копированию, которые будет выполнять программа.

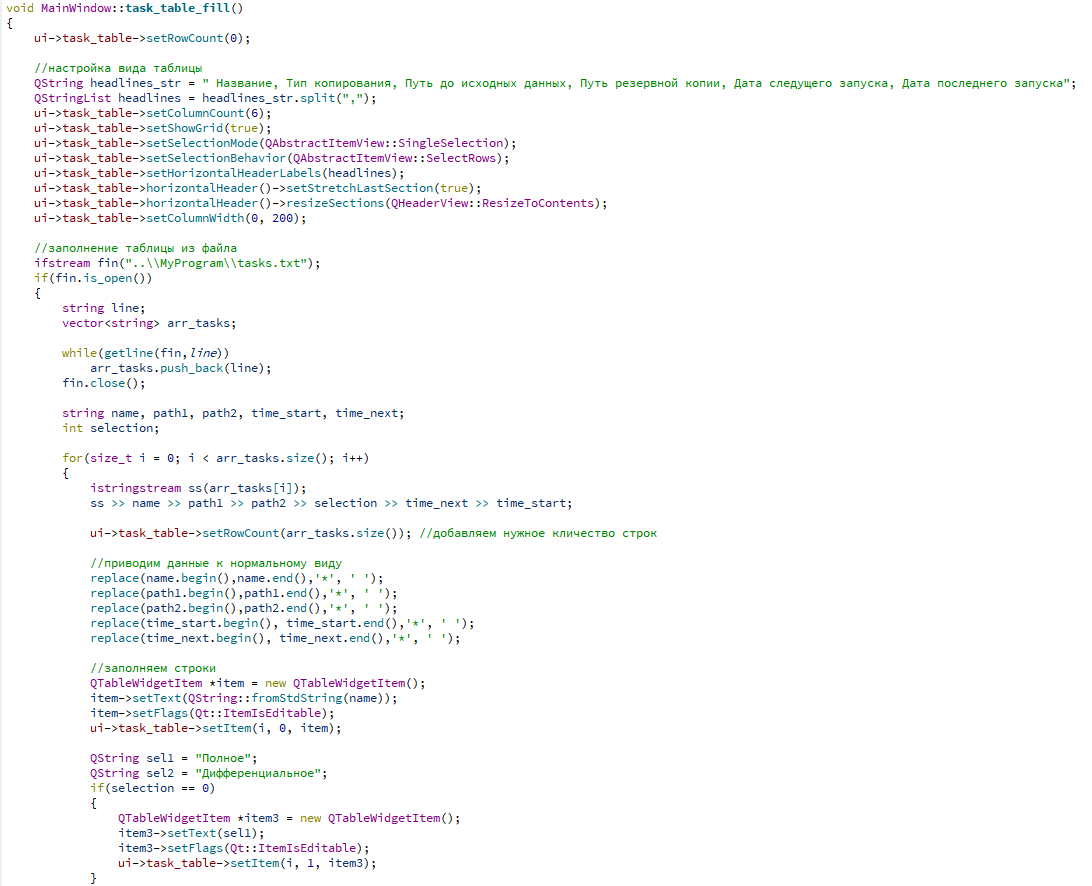


Рисунок 4 – Часть кода метода task\_table\_fill

Остальные методы, содержащиеся в классе, отвечают за открытие того или иного окна, когда пользователь нажимает на кнопку.

Рассмотрим класс AddTaskWindow. Особый интерес здесь представляет не менее важный метод добавления задачи.

Метод on\_add\_button\_clicked получает данные введенные пользователем в окне и проводит их проверку (рисунок 5). Когда проверка успешно пройдена, введенные данные обрабатываются, производится полная копия, затем данные записываются, в файл со списком задач. В журнал событий делается соответствующая пометка об этом.



Рисунок 5 – Часть кода метода on\_add\_button\_clicked (Добавление задачи)

Прочие классы отвечают за функционал кнопок в окнах, вызываемых из MainWindow, в основном методы, содержащиеся в этих классах, работают с файлами: журнала событий, списка задач и конфигурационным файлом.

### Описание основных функций резервного копирования

Функция Copy (рисунок 6) является основной в этом проекте. Она отвечает за резервное копирование данных одним из двух методов присутствующих в программе, выбранных пользователем.

На вход функция принимает такие параметры:

* path1 (путь до директории, из которой следует делать копирование)
* path2 (путь до директории, куда следует делать копирование)
* name (индивидуальное имя задачи)
* days, hours, minutes (интервалы времени между вызовами функции резервного копирования)
* type\_copy (метод резервного копирования, который следует применять при работе функции)

extern void **Copy**(string path1, string path2, string name, int days, int hours, int minutes, int type\_copy)

Рисунок 6 – Заголовок функции копирования

Используя параметр type\_copy, функция понимает какой вид копирования нужно применять к текущей задаче (рисунок 7).

При полном копировании программа сверяется с конфигурационным файлом, и если в нем есть соответствующая пометка о том, что копирование нужно производить в архив вызывается процедура system(), которая выполняет поставленную задачу, путем вызова подпрограммы 7-Zip.



Рисунок 7 – Код вызова функций заполнения списка копируемых файлов

Иначе же вызывается функция Recursive\_Content\_List, которая заполняет список content\_list, содержащий абсолютные пути до файлов, какие попадут на копирование. Работа Recursive\_Content\_List показана на рисунке 8.

В ней задается фильтр элементов, которые попадут в финальный список, а также метод их сортировки в списке, в данном случае по имени. Далее происходит заполнение списка. Так как функция вызывается при методе полного копирования, в список попадают все элементы, содержащиеся в директории и ее подразделах.

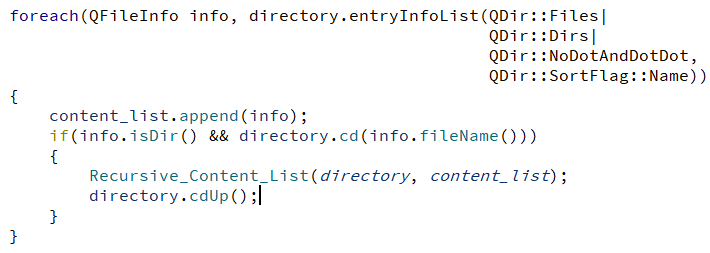


Рисунок 8 – Код функции Recursive\_Content\_List

При выборе метода дифференциального копирования вызывается функция Difference\_Content\_List. Пример кода, которой приведен на рисунке 9.

В этой функции составляется список различий между двумя директориями. В список могут попасть файлы и каталоги, которых нет в конечной директории или же файлы, которые были изменены. Изменения файлов отслеживаются с помощью сравнения времени последнего изменения.

Для начала мы ищем элементы, с похожими именами, а далее сравниваем их по времени последнего изменения. Если текущий элемент является каталогом или это файл и время его изменения меньше или равно времени изменения файла в директории, в которую производится копирование, значит данный элемент копировать не надо.

Если функция не находит элемент в директории, в которую производится копирование, она добавляет этот элемент в список.

Если же элемент – это директория, которая присутствует в конечной, программа должна сравнить элементы в этих директориях путем рекурсивного вызова этой же функции.

И конечно же, если элемент – это директория, которой нет в конечной, то программа вызывает функцию Recursive\_Content\_List, которая занесет в список все элементы этой директории.



Рисунок 9 – Код функции Difference\_Content\_List

Таким образом, после всех этих действий, список элементов, которые подлежат копированию, составлен. Остается только рассмотреть код, который и производит копирование с помощью заданного списка (рисунок 10).

В данном алгоритме создается локальная переменная backup, хранящая в себе путь файла источника, с заменой на путь папки резервного копирования.

Далее алгоритм проводит проверку, если элемент – это файл, то он копируется, если элемент – директория, то она попросту создается.

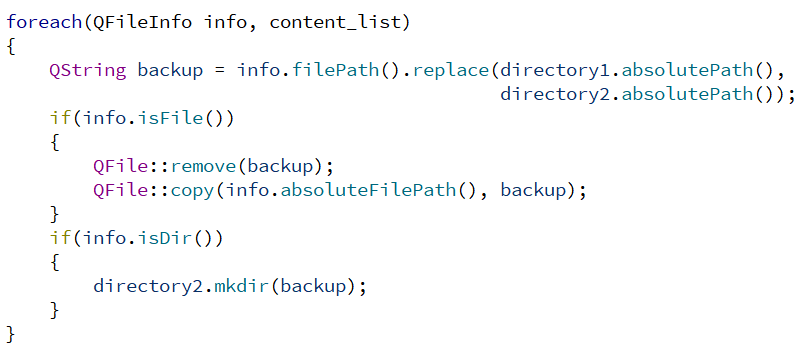


Рисунок 10 – Код производящий копирование файлов в соответствии со списком

Далее функция Copy, завершая свою работу, вносит изменения в файл с текущими задачами. Используя полученные параметры интервалов времени, изменяет время выполнения на текущее, и производит расчет времени следующего вызова функции, изменяя его соответственно.

Функция ничего не возвращает, по окончанию работы делается соответствующая запись в log-файл с пометкой времени о том, что задача успешно скопирована, либо произошла ошибка (рисунок 11).

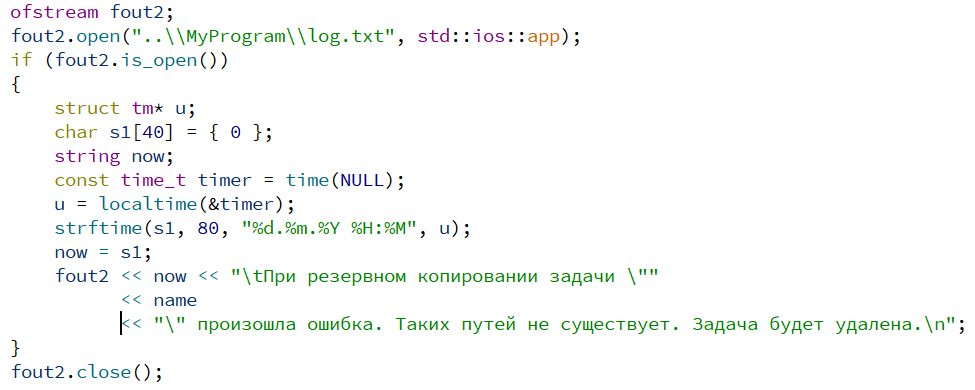


Рисунок 11 – Пример кода записи данных о текущем состоянии программы в log-файл

Рассмотрим функцию Work (рисунок 12), которая вызывается в самом начале программы с помощью процедуры thread и работает до ее закрытия. Она содержит тот самый бесконечный цикл, вызывающий основную функцию Copy по времени.

void **Work**()

Рисунок 12 – Заголовок функции Work

Код содержащийся в функции Work показан на рисунке 13.

В первую очередь программа получает все строки из файла с текущими задачами. Если она не находит файл, то программа принудительно закрывается, при следующем запуске этот файл будет создан.

Каждая полученная строка разбивается на параметры: имя, тип копирования, пути и т. д. Запрашивается текущее время системы. Затем происходит проверка времени, если полученное из файла время выполнения задачи подошло, либо уже прошло, вызывается функция Copy, описанная выше.

Для того, чтобы программа не «съедала» много ресурсов процессора, временная задержка между выполнением цикла составляет секунду. Это позволяет значительно снизить нагрузку на ЦП.



Рисунок 13 – Пример кода функции Work

## Описание интерфейса пользователя

Интерфейс главного окна представлен на рисунке 14. При работе с программой нас встречают: кнопки быстрых действий «2-7»; таблица задач «1», в которой будут отображаться введенные пользователем задачи; журнал событий «8», в нем будут отображаться заметки программы по ходу ее работы.

При первом открытии программы пользователя встречает приветственное окно с описанием самой программы и ее возможностями (рисунок 15). При последующих открытиях это окно не появляется, но его всегда можно вызвать, нажав на кнопку справки «7» (рисунок 14).

Для начала работы с программой нужно добавить задачу. Сделать это можно нажатием кнопки «2» или нажатием соответствующего пункта меню «9» (рисунок 14).

В открывшемся окне (рисунок 16) можно создать новую задачу. Заполнив все нужные строки, выбрав тип копирования, и интервалы времени, как часто делать копии, можно нажать на кнопку «Добавить». Если всё введено корректно, то задача попадет в список и будет немедленно произведено полное копирование. Если данные введены неправильно, то программа сообщит об этом и предложит проверить корректность данных.

Вернемся к главному окну (рисунок 14). Кнопка «4» вызывает окно изменения задачи (рисунок 17).

Из появившегося списка нужно выбрать задачу, которую пользователь хочет изменить. Когда задача выбрана, кнопка «Изменить» становится активной и при ее нажатии вид окна меняется (рисунок 18).

Окно приобретает вид окна добавления за исключением того, что название задачи изменить нельзя. Зато все остальные данные менять можно. При нажатии кнопки «Внести изменения» введенные данные проходят все те же проверки, как при добавлении, и если все введено корректно, то измененные данные вносятся в файл со списком задач и соответственно отображаются в таблице «Текущие задачи».

Кнопка «5» на рисунке 14 открывает окно удаления задачи (рисунок 19).

В этом окне можно не только удалять задачи по отдельности, выбрав их из списка и нажав на кнопку «Удалить выбранный элемент», так и удалить все существующие задачи нажатием кнопки «Удалить все задачи».

Кнопка «3» на рисунке 14 открывает окно «Выполнить сейчас» (рисунок 20).

Данный функционал добавлен для удобства пользователя. К примеру, если копию нужно произвести, не дожидаясь времени следующего копирования, можно вызвать это окно и выбрать из списка нужную задачу. При нажатии кнопки «Выполнить выбранную задачу» будет вызвана функция копирования.

На рисунке 14 кнопка «6» вызывает окно настроек (рисунок 21).

В программу добавлена некоторая кастомизация, т. е. ее внешний вид может быть изменен пользователем. Кнопки быстрых действий и журнал событий можно скрыть, поставив соответствующие галочки настроек, тогда в главном окне (рисунок 14) останется только таблица.

«Очистить журнал событий» - данная кнопка собственно очищает журнал событий.

Также в настройках предусмотрены специальные возможности, которые также можно настроить, нажав соответствующие пункты и применив настройки нажатием кнопки «Применить».

Таким образом весь интерфейс программы достаточно понятен и дружелюбен для среднестатистического пользователя.

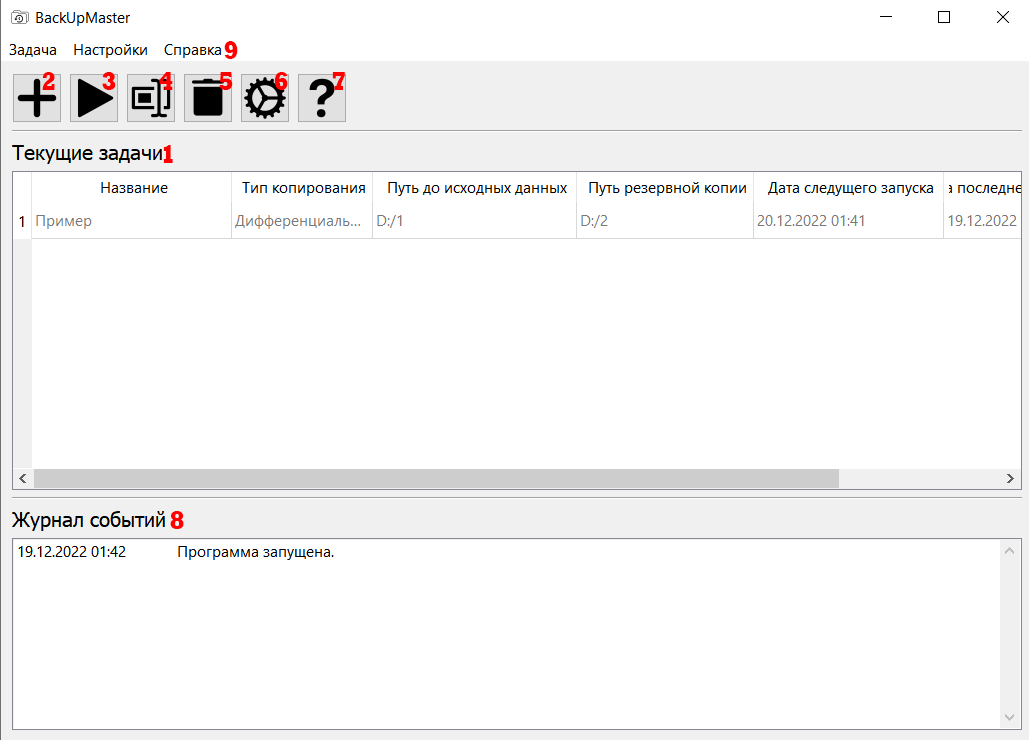


Рисунок 14 – Интерфейс главного окна программы

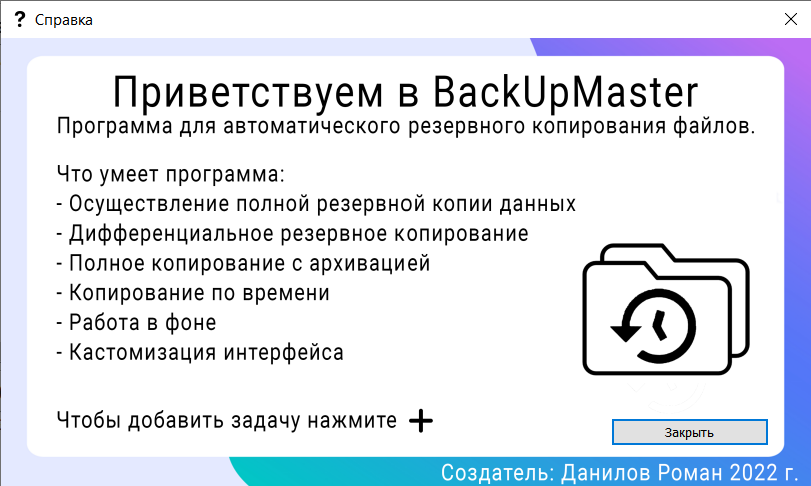


Рисунок 15 – Интерфейс окна справки

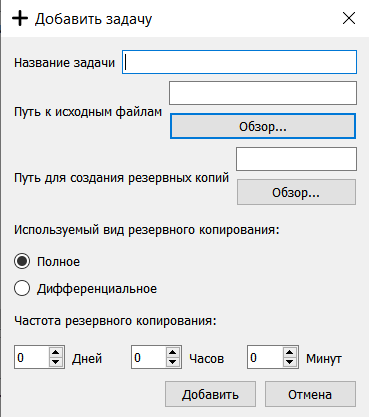


Рисунок 16 – Интерфейс окна добавления задачи

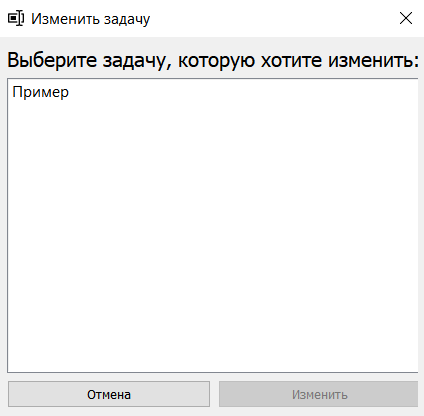


Рисунок 17 – Интерфейс окна изменения задачи (список)

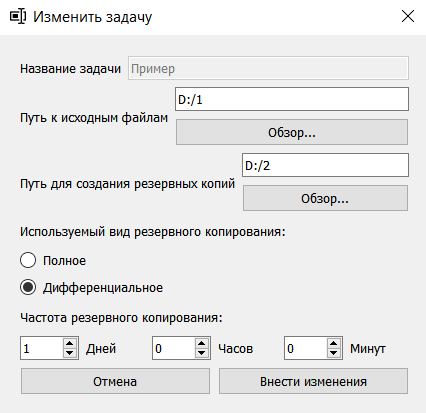


Рисунок 18 – Интерфейс окна изменения задачи (изменение)

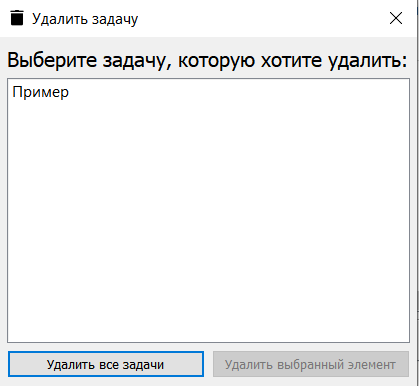


Рисунок 19 – Интерфейс окна удаления задачи

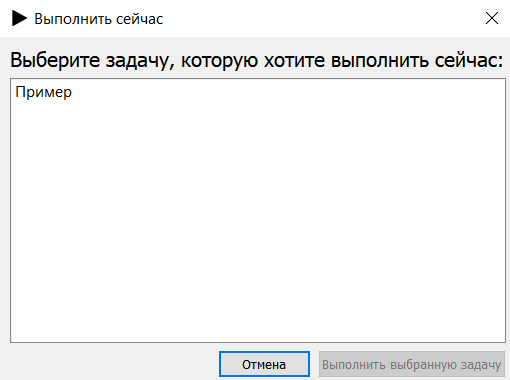


Рисунок 20 – Интерфейс окна «Выполнить сейчас»

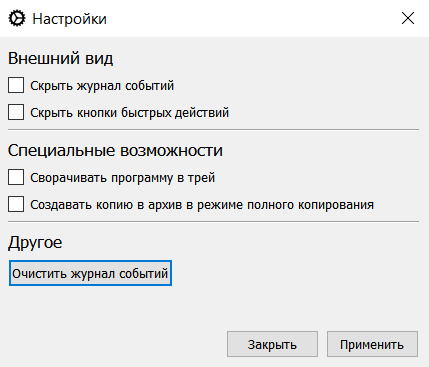


Рисунок 21 – Интерфейс окна настроек

## Тестирование работоспособности программы

Рассмотрим работу программы на примере полного резервного копирования. При добавлении задачи в список (рисунок 22), в журнале событий делается соответствующая пометка и происходит первичное полное копирование. На рисунке 23 можно увидеть папку 1, из которой делалось копирование, и папку 2, в которую это копирование произошло.

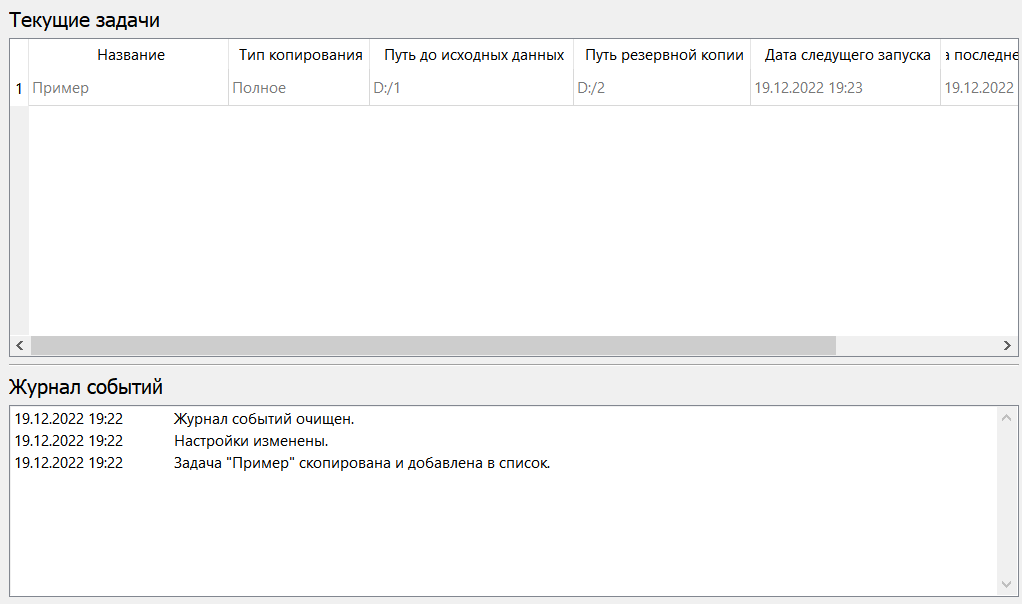


Рисунок 22 – Добавление задачи в список на копирование.

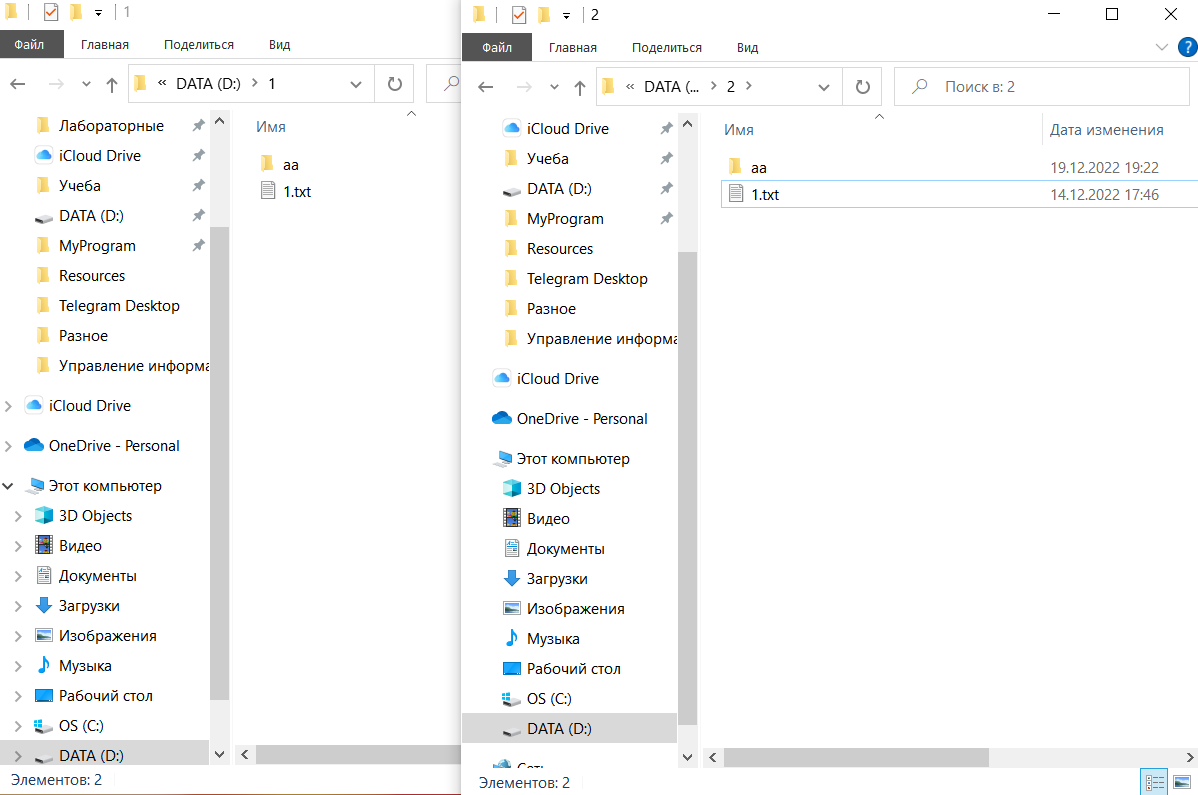


Рисунок 23 – Пример работоспособности полного резервного копирования

Применив параметр в настройках программы «Создавать копию в архив при режиме полного копирования», можно заметить, что в папке 2 появился архив, содержащий в себе элементы папки 1 (рисунок 24).

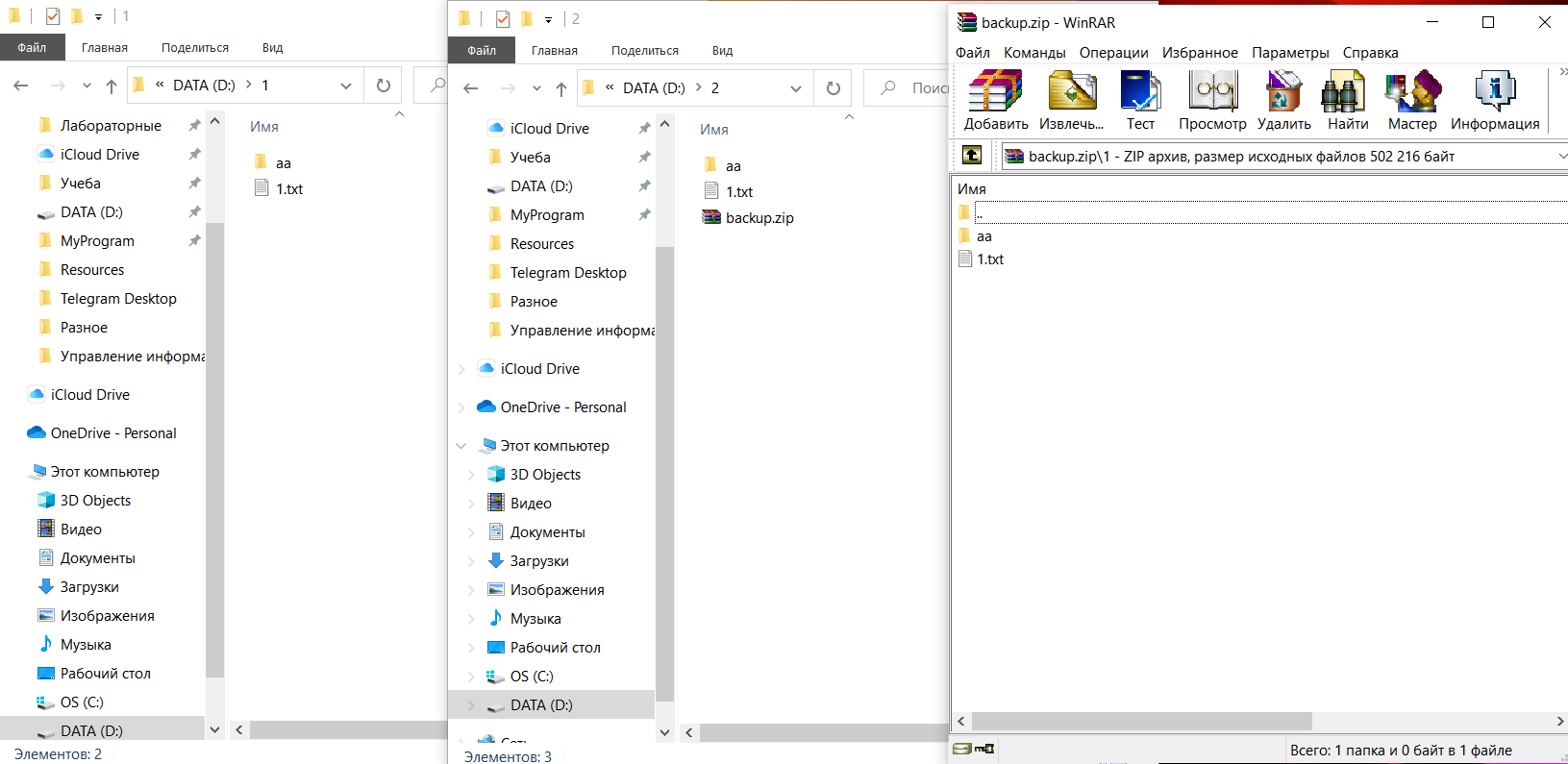


Рисунок 24 – Пример работоспособности архивации при полном резервном копировании

Видно, что, когда время копирования подошло, программа самостоятельно делает копию, уведомляя об этом пользователя в журнале событий (рисунок 25).

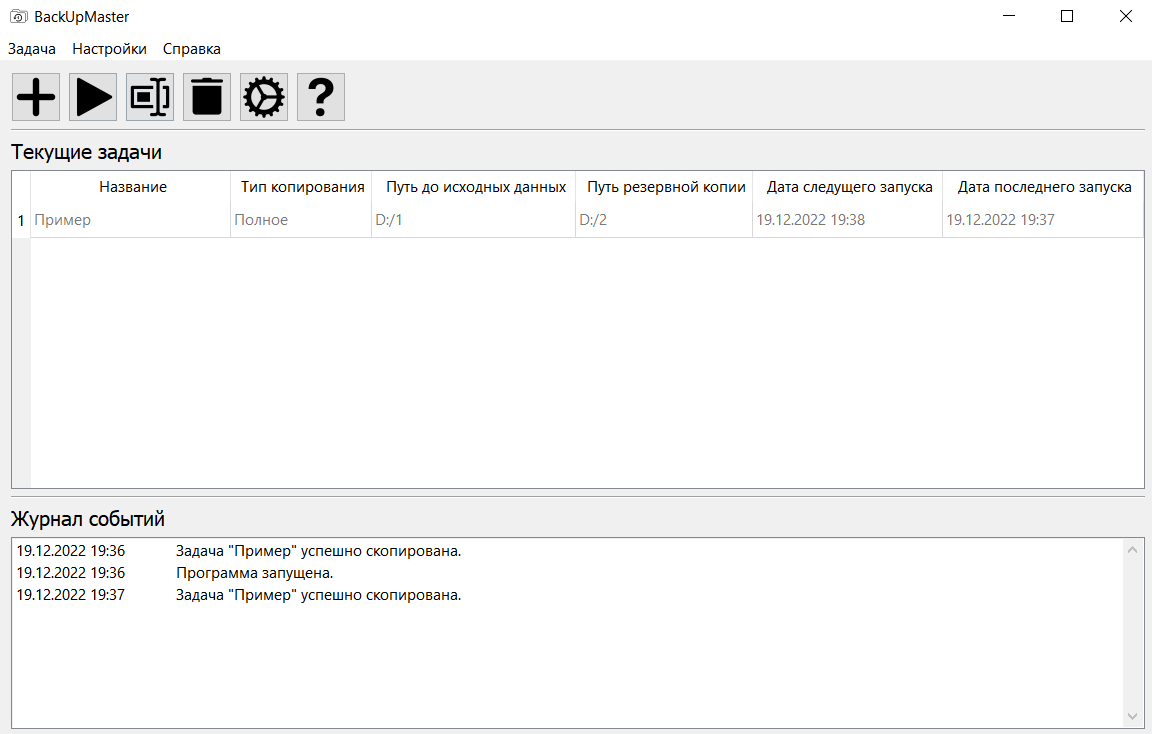


Рисунок 25 – Пример работоспособности копирования по времени

## Выводы

Используя документацию по классам Qt, с их помощью были разработаны алгоритмы полного и дифференциального копирования, которые полностью выполняют поставленную задачу.

Подробно изучив функционал Qt Creator, удалось разработать дружелюбный для пользователя интерфейс и добавить множество возможностей и функций, для удобной работы с программой.

ЗАКЛЮЧЕНИЕ

Разработанная программа является полностью работоспособной и выполняет поставленную задачу. Классы, методы и функции, содержащиеся в проекте, работают исправно.

Обо всех ошибках и проблемах, которые могут произойти в ходе ее работы, она сообщает пользователю и самостоятельно пытается бороться с ними.

Программа помимо выполнения своей прямой задачи также имеет удобный и понятный пользовательский интерфейс, с возможностью его настройки, множество функций для работы со списком задач, отзывчивость в виде комментариев по ходу работы в журнале событий, работу с архивами и работу в фоне.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Резервное копирование // Виды резервного копирования URL: <https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B7%D0%B5%D1%80%D0%B2%D0%BD%D0%BE%D0%B5_%D0%BA%D0%BE%D0%BF%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5> (Дата обращения: 18.12.2022).

2 Топ программ для резервного копирования URL: https://www.moyo.ua/news/programmy\_dlya\_rezervnogo\_kopirovaniya\_top-10\_luchshikh.html (Дата обращения: 18.12.2022).

# Документация Qt // Основные классы Qt URL: http://doc.crossplatform.ru/qt/4.5.0/mainclasses.html (Дата обращения: 20.10.2022).

ПРИЛОЖЕНИЕ А

Код программы

Код программы на приложенном физическом носителе.