# Содержание

[Содержание 2](#_Toc53701928)

[ВВЕДЕНИЕ 2](#_Toc53701929)

[Поиск операций с файлами 3](#_Toc53701930)

[Нахождение решения для реализации операций с файлами 3](#_Toc53701931)

[ВЫБОР ЯЗЫКА РАЗРАБОТКИ 3](#_Toc53701932)

[СОЗДАНИЕ ИСХОДНОГО ФАЙЛА С РЕШЕНИЯМИ 4](#_Toc53701933)

[Объединение всех решений и создание более понятного меню. 5](#_Toc53701934)

[ЗАКЛЮЧЕНИЕ 5](#_Toc53701935)

[Пример работы приложения 7](#_Toc53701936)

**Тема**: Помощь в работе с файлами на языке программирования Python.

**Цель**: Создание консольного приложения, помогающее при работе с файлами и файловыми структурами.

**Актуальность**: в современном мире приходиться работать с большими объёмами информации, а без вспомогательных программ это становиться сложным.

**Задачи**:

1. Найти операции с файлами, выполнение которых часто отнимает время

2. Найти решения для реализации операций с файлами на языке программирования

3. Создать приложение, объединяющее все решения

# ВВЕДЕНИЕ

Работа с информацией является целью компьютера. Их используют в простой, но рутинной работе по обработке информации. Файл основная ячейка, хранящая информацию самых разных типов. Следовательно, работа с файлами и есть применяемое нами в компьютере. Упрощение этого процесса является упрощением работы с компьютером, что сказывается не только на производительность пользователей, но и на здоровье. Наверное, некоторые замечали, что иногда приходиться трудно и нудно сидеть за компьютером “разгребая” файлы. В ПК мало встроенных функций по работе с файлами, для исправления этой проблемы мною была создана данная программа.

# Поиск операций с файлами

При работе с файлами у многих пользователей нет определённой файловой структуры, файлы разных типов разбросаны по всем каталогам, значит, требуется помощь в размещении файлов по их каталогам, а так же при длительной работе часто на компьютере остаются копии файлов, о которых пользователь не знает, следовательно, требуется удаление одинаковых файлов.

# Нахождение решения для реализации операций с файлами

## Выбор языка и среды разработки

Язык Python был выбран, потому что он имеет простую структуру, большое сообщество и множество различных технических плюсов, упрощающих написание кода. По сложности он сравним с преподаваемым в школах языком Pascal, но имеет больший набор как встроенных, так и подключаемых функций.

Мною часто используются Wing, IDLE[[1]](#footnote-1), текстовый редактор Notepad++ и PyCharm Community. В каждом из этих средств разработки есть свои плюсы и минусы.

PyCharm самая удобная и мощная среда разработки из представленных, она имеет больший функционал, но как следствие намного больший объём и длительный запуск. Wing для меня является облегчённой версией PyCharm. В нём есть почти всё что нужно, но его интерфейс и частые ошибки с кодировкой иногда препятствуют комфортному процессу написания кода. IDLE и Notepad++ мною используются одновременно. Notepad++ имеет самый удобный редактор кода и приятный интерфейс, но в нём отсутствует встроенная быстрая проверка и исполнение кода. При написании программы я пользовался поочерёдно всеми перечисленными средствами разработки.

## Создание исходного файла с решениями

Так как программе необходимы каталоги для правильной работы, при каждом запуске программы они проверяются на наличие. При отсутствии нужных каталогов, они создаются функцией *makedirs()*.

Для решения задачи по перемещению файлов в их каталоги будет служить функции *clearing()* и. В данной функции поочерёдно перебираются файлы директории, которым присевается класс *File*.

Класс *File* нужен для инициализации имени файла, его расширения и каталога, в который его требуется переместить из передаваемого перебиранием пути, с помощью функции *\_\_init\_\_()*. Его перемещение с переименованием, реализуется функцией *move()*, а удаление функцией *delete()*.

Для удаления одинаковых файлов будет служить функции *delete\_copies()*  и вспомогательная ей *remove()*. В ней идёт перебор всех файлов в каталоге и подкаталогах со всеми файлами в тех же каталогах и подкаталогах. Файлам, являющимся результатом перебора, присевается класс *File*. После чего идёт сравнение параметров файлов и проверка удалялся ли этот файл и существует ли ещё этот путь, чтобы не читать содержимое, если это не обходимо, далее читается содержимое файлов и сравнивается, если совпадает, программа предлагает, какой файл из двух удалить, и продолжает перебор.

Для получения или дальнейшей записи данных используется наследуемый класс *Oper*, он при себе имеет функцию получения значения *get()* и функцию записи значения *set()*.

Для работы функций чтения, редактирования, удаления и нахождения файлов импортируется две необходимые библиотеки *os* и *shutil.*

Функция *path\_reset()* и служит для изменения путей перемещения файлов, которые с помощью библиотеки *json*, которая импортируется с остальными библиотеками,записываются в конфигурационный файл. Записывается словарь, состоящий из поименованных списков, которые образуют типовую единицу для сортировки. Эти списки в свою очередь состоят из строки, содержащей путь к данному типу файлов, и списка строк, содержащих расширения файлов для данного типа.

Функция является *path\_reset()* аналогом *path\_reset\_d()* , за исключением того, что она применяется для изменения рабочей директории.

Идущий вместе со всем проектом файл *Path\_error.py* является средством устранения ошибок с директориями, при изменении файлов *conf.txt* и *conf2.txt,* а так же применяется для сброса настроек по умолчанию.

# Объединение всех решений и создание более понятного меню.

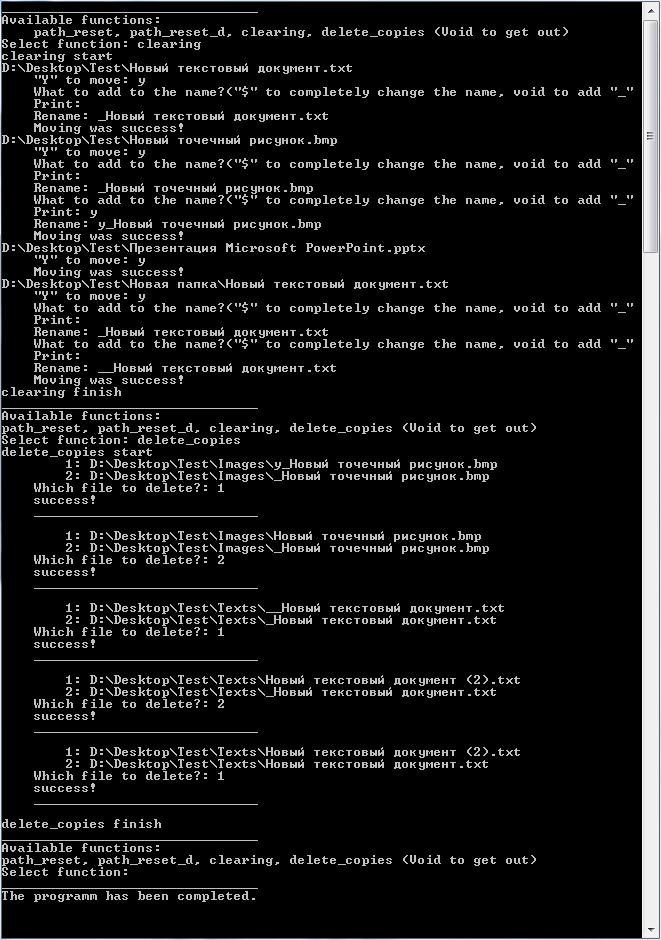
При создании данного консольного приложения мною использовались объектно-ориентированная и функционально-ориентированная парадигмы программирования. Они применялись для достижения наилучшего результата по оптимизации работы кода и реализации более правильного подхода к написанию программ, чтобы в дальнейшем была возможность развивать проект, дописывая код для новых возможностей приложения и убирая потенциальные сбои в работе приложения.

Меню приложения было написано на английском языке, поскольку английский язык является основным в программировании и принято писать программы на нём, а так же английский язык более универсален в плане файловой кодировки, и он менее восприимчив к ошибкам с неправильной кодировкой. Для разделения логических блоков я использовал табуляции и строку нижних подчёркиваний, это предало программе более приятный внешний вид. Так же в функции были добавлены ввод и вывод информации, по работе конкретной функции.

# ЗАКЛЮЧЕНИЕ

Данное приложение регулярно используется мной, для очистки файлов, в отличие от других программ, здесь реализована максимальная простота и понятность работы. Отсутствие рекламы и запутанного интерфейса позволяет быстро и эффективно использовать приложение. У программы есть будущее за счёт применения парадигм программирования и дальнейшей возможности создание графического интерфейса, который бы так же ускорил работу.

# Пример работы приложения



1. Встроенная в Python интегрированная среда разработки [↑](#footnote-ref-1)