

# Разработка и тестирование модели функционирования системы резервирования данных

Рабочие данные, хранимыми на жёстком диске, изменяются, стираются и переписываются. Поэтому, при неправильном изменении данных, важно иметь возможность вернуться к исходной точке их редактирования. Для этого существуют резервы.

Без резервирования данных, при ошибке с их работой, придется вручную откатывать прогресс. Самый простой пример — Ctrl + Z. Ситуация, когда нужно вернуть предыдущую операцию. Но если нужно возвратиться на множество операций назад, понадобится резерв начала предыдущей работы с данными.

Проект курсовой работы нацелен на создание программы, упрощающей копирование и перенос данных.

## Софт

Алгоритм заключается в запуске главного файла `main.py`, который импортирует функции файлов `reses.py` и `graphics.py`.

Через `reses.py` главный файл получает информацию о предыдущих сделанных резервах. Через `graphics.py` главный файл `main.py` создает класс `Window`, отображающий пользовательский интерфейс и передает полученную из `reses.py` информацию о предыдущих копированиях в пользовательское окно.

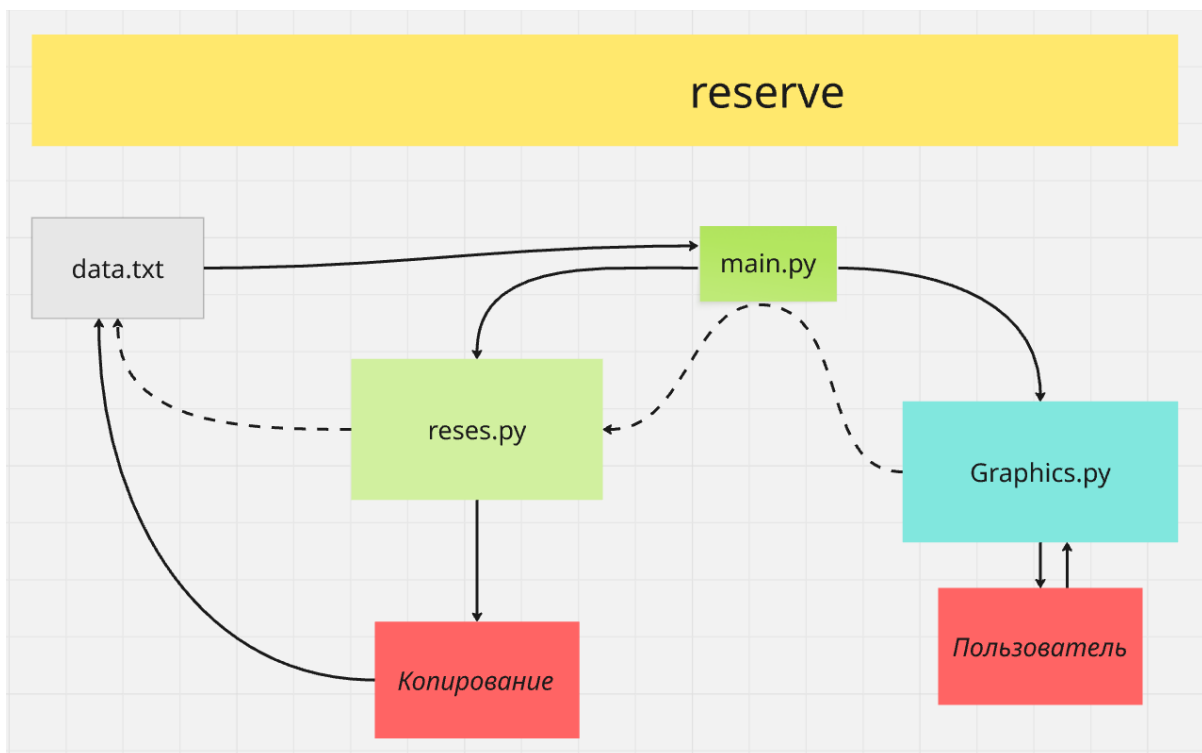
После заполнения данных в окне и его закрытия, информация передается обратно в файл `main.py`, который уже через `reses.py` создает класс `dir_i`. Этот

класс при инициализации получает информацию о путях резервирования и названия копии. Далее копирует данные.

Информация о текущем резерве через `reses.py` добавляется в текстовый файл `data.txt`, который сохраняет всю историю копирований. Текстовый файл находится в одной папке и `main.py`. После каждой записи информации в файл тот пересоздается.

Алгоритм удобен тем, что каждый файл имеет свою собственную область применения. Их функционал не пересекается, в связи с чем код эффективен, лаконичен и упорядочен.

- Файл `main.py` — координирующий
- Файл `graphics.py` — отображающий
- Файл `reses.py` — резервирующий, сохраняющий



Класс `graphics.Window` запускает пользовательское окно, которое будет выводить поля:

1. Название Резерва
2. Путь резервирования
3. Путь исходных данных
4. История предыдущих резервов

WINDOW

Имя резерва:


Путь резервирования Место для резервирования

Путь исходных данных файлы, которые резервируем

Резервы есть

Выбрать

Файл `reses.py` будет считывать данные предыдущих резервов и передавать в `graphics.py` через `main.py`.

 data – Блокнот

Файл Правка Формат Вид Справка

---

-> uuuuuuuuuu

C:\Users\Egor\3D Objects

C:\Users\Egor\OneDrive\Рабочий стол\ТАНКИ

5

-> qqqqqqqqqqqqqqqqq

C:\Users\Egor\Music

C:\Users\Egor\OneDrive\Изображения\Снимки экрана

5

-> RESERVE\_3

C:\Users\Egor\Music

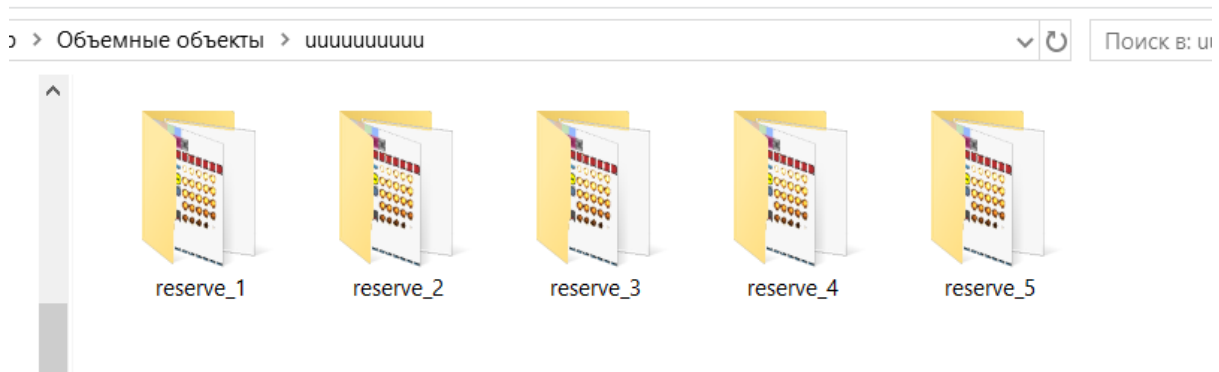
C:\Users\Egor\OneDrive\Изображения\Снимки экрана

|

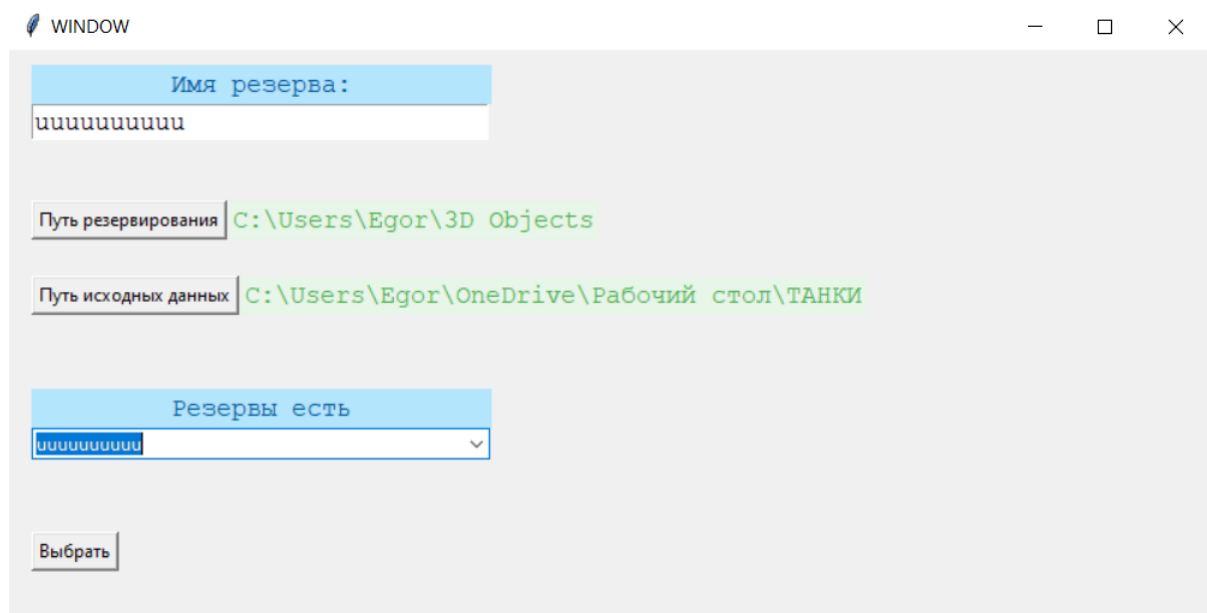
Где строки:

1. -> Название резерва
2. Путь резервирования
3. Путь исходных данных
4. Номер копирования выбранных данных — i

Резервы индексируются: reserve\_{i}. При вторичном копировании одних и тех же данных: reserve\_2, reserve\_3 и т.д.



При выборе варианты предыдущего резервирования все поля ввода в окне graphics.py заполняются автоматически, исходя из того, какие данные были заполнены ранее при создании этой копии. То есть перенесутся из файла data.txt прямо в поля ввода в пользовательском окне Window



Библиотеки

Файл `reses.py`:

- Импортирует библиотеку `shutil` для создания нового пути (древа директорий) и копирования файлов — `shutil.copytree()`
- Импортирует библиотеку `os` для объединения выбранного пути резервирования, названия резерва и номера резерва `{i}` — `os.path.join()`

Файл `graphics.py`:

- Импортирует библиотеку `tkinter` для создания диалогового окна с пользователем, сообщений и кнопок
- Импортирует `from tkinter import filedialog` для выбора путей резервирования и исходных данных через вкладку проводника
- Импортирует `from tkinter import ttk` для создания в диалоговом окне меню с историей предыдущих резервирований

В файле `reses.py` имеется класс `dir_i` и отдельная функция для чтения файла `data.txt`.

## Классы

1. Файл `main.py` с помощью `reses.py` считывает информацию из `data.txt`, если такое возможно, и создает список.

2. Файл `main.py` инициализирует класс `Window` в файле `graphics.py`, запускает диалоговое окно и передает в него созданный список истории резервов.
3. Файл `graphics.py` ждет заполнения пользователем необходимых полей и нажатия кнопки “Принять”
4. Файл `main.py` получает заполненные данные из `graphics.py`.
5. Файл `main.py` инициализирует класс `dir_i` в файле `reses.py` и передает заполненные данные.
6. Файл `reses.py` резервирует данные, о которых передавалась информация в класс `dir_i`.

## Ошибки и исключения

- Пользователь заполнил не все данные

Тогда в окне `Window` появится сообщение о нехватки информации:

Имя резерва:

резерв

Путь резервирования Место для резервирования

Путь исходных данных C:/Users/Egor/OneDrive/Документы/STAR WARS Battlefront II

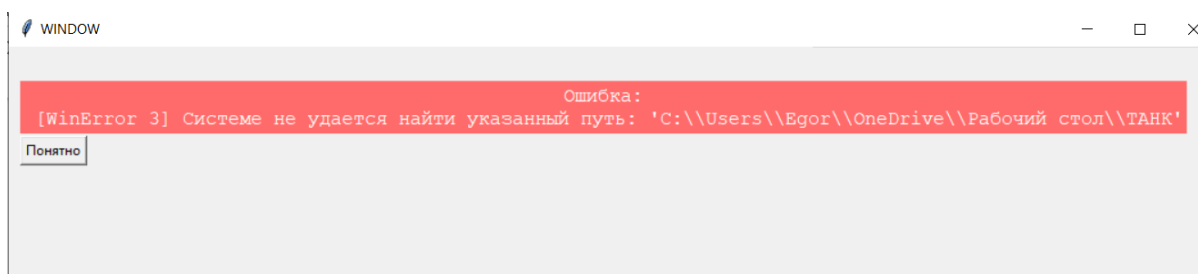
Резервы есть

Выбрать Информации не достаточно

- Ошибки при записи и копировании файлов.

В случае возникновения такой ошибки, после закрытия окна Window, появится еще одно, сообщающее, какая именно ошибка возникла (исходя из внутренних обозначений ошибок Python)

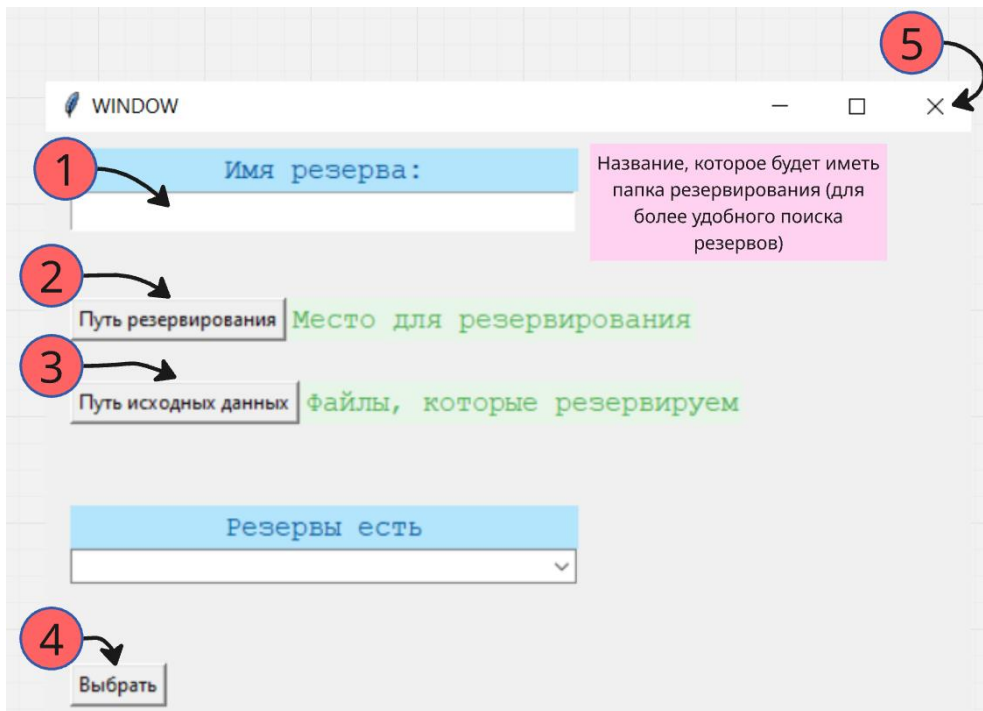
Например, путь исходных данных не существует:





## Руководство для пользователя

1. Открыть файл main.py. Выведется окно Window
2. Заполнить информацию о копировании
3. Нажать “Принять”



## Итоги

В заключении можно сказать, что программа представляет собой возможность для пользователя упростить копирование больших объемов данных.

Так же софт можно доработать до автоматического резервирования. Например, используя библиотеку APSchedul. Данные будут автоматически резервироваться через определенный срок, устанавливаемый пользователем. В таком случае понадобится отдельный файл для автоматизации, который должен автоматически запускаться при включении ПК

## Литература

- Работа библиотеки shutil:

<https://docs.python.org/3/library/shutil.html>

<https://www.book2s.com/tutorials/python-shutil.html>

- Работа с библиотекой tkinter:

<https://pythonru.com/uroki/obuchenie-python-gui-uroki-po-tkinter>

<https://habr.com/ru/articles/133337/>

<https://ultrapythonic.com/file-dialog-boxes-in-tkinter/>

<https://metanit.com/python/tkinter/2.1.php>

- ООП в Python:

<https://devpractice.ru/python-lesson-14-classes-and-objects/>

<https://proglib.io/p/python-oop>