МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение
высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Измерительно-вычислительные комплексы»

лабораторная работа

по дисциплине «Алгоритмы и структуры данных»

Тема лабораторная работа №9

Пояснительная записка

Р.02069337.<23/742 >-<19> ТЗ-<2-зн. номерредакции>

Листов (5)

Исполнитель:
студент гр. ИСТбд-23
Романов И.Н
«____»_____2024 г.

Подп. и	
Инв	
Вза	
Подп. и	
Инв.	

2024

Введение

Данный проект представляет собой разработку приложения для управления договорами. Основная цель приложения – автоматизация работы с договорами, их сегментация по заказчикам и месяцам, а также визуализация данных с помощью круговых диаграмм. Для этого используется библиотека Python Tkinter для графического интерфейса и Matplotlib для построения диаграмм.

1. Проектная часть

1.1. Постановка задачи на разработку приложения

Определяется заданием на лабораторную работу.

1.2. Математические методы

Для отображения статистики о суммах, заключенных с заказчиками или по месяцам, используется круговая диаграмма. Данные суммируются для каждой категории, и на основе этой информации рассчитываются углы для каждого сектора диаграммы. Угол сектора для каждого элемента пропорционален доле его суммы в общей сумме.

Формула для расчета угла сектора: (Сумма элемента / Общая сумма)×360

1.3. Архитектура и алгоритмы

1.3.1 Архитектура

Приложение разделено на несколько классов:

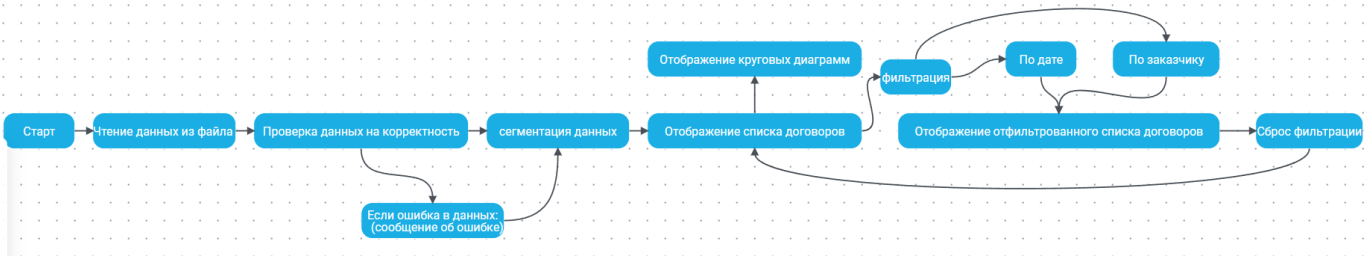
1. **Класс Contract** – хранит информацию о каждом договоре (заказчик, дата, сумма).
2. **Класс Contract1** – управляет договорами, загружает данные из файла и выполняет сегментацию данных по заказчикам и месяцам.
3. **Класс App** – управляет графическим интерфейсом, отображает данные в таблице и позволяет пользователю работать с фильтрацией и визуализацией данных.

1.3.2 Алгоритм формирования диаграммы

Для формирования диаграммы выполняются следующие шаги:

1. **Сегментация данных.** Данные группируются по выбранным категориям: заказчикам или месяцам.
2. **Подсчет сумм.** Для каждой категории вычисляется сумма всех договоров, относящихся к данной категории.
3. **Построение диаграммы.** На основе подсчитанных сумм строится круговая диаграмма с помощью библиотеки Matplotlib.

1.4. Тестирование



1.4.1 Описание отчета о тестировании

В ходе тестирования приложения проверялись основные функции, такие как загрузка данных из файла, фильтрация по заказчикам и датам, а также корректность отображения диаграмм.

1.4.2 Цель тестирования

Цель тестирования – убедиться в корректной работе всех функций приложения, правильности отображения данных, а также в корректности расчета и отображения диаграмм.

1.4.3 Методика тестирования

Тестирование проводилось с использованием следующих методов:

- **Модульное тестирование**:
  Проверка отдельных компонентов приложения, таких как функции сегментации данных, загрузки файла и построения диаграмм, на корректность работы.
- **Тестирование границ**:
  Проверка работы приложения при минимальных, максимальных и граничных значениях данных, например, пустой файл, данные с очень большими или отрицательными суммами.

- **Интеграционное тестирование**:
  Проверка взаимодействия между различными модулями приложения, такими как загрузка данных из файла, отображение данных в таблице и построение диаграмм.

- **Динамическое тестирование**:
  Оценка поведения приложения во время его выполнения, включая обработку событий, ввод данных и построение диаграмм.

1.4.4 Проведенные тесты:

Тестирование границ:

Пустой файл

- **Описание теста**: Файл без данных.
- **Ожидаемый результат**: ошибка
- **Результат теста**: Приложение корректно обрабатывает пустой файл.

Один корректный договор

- **Описание теста**: Файл с одной строкой, содержащей корректные данные
- **Ожидаемый результат**: Строка добавляется в список договоров без ошибок.
- **Результат теста**: Данные загружаются корректно.

Один некорректный договор

- **Описание теста**: Файл с одной строкой, содержащей некорректные данные:
- **Ожидаемый результат**: Приложение выводит предупреждение о неверном формате строки.
- **Результат теста**: Приложение корректно идентифицирует ошибку.

Динамическое тестирование:

Загрузка данных из файла

- **Сценарий**: Пользователь запускает приложение, и данные автоматически загружаются из CSV-файла.
- **Проверка**:
  - Корректная обработка файлов с правильными и неправильными строками.
  - Отображение предупреждений об ошибках в формате данных.
- **Результат**:
  Данные корректно загружаются, ошибки в строках выводятся в виде предупреждений.

Работа с большими данными

- **Сценарий**: Загрузка файла с большим количеством строк (>10,000).
- **Проверка**:
  - Корректная обработка данных.
  - Устойчивость интерфейса при отображении большого объема информации.
- **Результат**:
  Приложение устойчиво к большому объему данных

1.4.5 Чек лист для фильтрации по дате

Описание	Пример	Результат
фильтрации по дате		
Запрос года и месяца для фильтрации	Ввод года: 2024, месяца: 04	Программа фильтрует контракты за апрель 2024 года и отображает их в таблице.
Ввод некорректного года	Ввод года: 202, месяца: 04	Появляется ошибка, что год введен неверно (менее 4 цифр). Фильтрация не применяется.
Ввод некорректного месяца	Ввод года: 2024, месяца: 13	Появляется ошибка, что месяц введен неверно (число больше 12). Фильтрация не применяется.
Отмена фильтрации (нажатие на кнопку "Отмена")	Нажатие кнопки "Отмена"	Появляется ошибка, что месяц не существует. Фильтрация не применяется.

1.4.6 Вывод

Приложение для управления договорами успешно решает задачи по загрузке, фильтрации и визуализации данных.

Источники, использованные при разработке

https://metanit.com/python/tkinter/