МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Измерительно-вычислительные комплексы»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | |  | |  |  |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | |  | | Руководство программиста  на лабораторную работу  по дисциплине «Алгоритмы и структуры данных»  Тема «Управление банковскими вкладами» | | | | | | |
|  | | |  | |  | | | |
|  | | Исполнитель  студент гр. ИСТбд-21  Клянченков М.А.  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2024 г. | | | |
|  | | 2024 | | | | | | |

**1. Введение**

Данный документ предназначен для программистов, которые будут поддерживать или расширять функциональность приложения "Управление банковскими вкладами". Программа разработана на языке Python с использованием библиотеки Tkinter для графического интерфейса и библиотеки Matplotlib для построения диаграмм. Руководство описывает архитектуру проекта, детали реализации основных классов и методов, а также инструкции для запуска и тестирования.

**2. Среда разработки**

Для работы с проектом необходимы следующие инструменты и библиотеки:

1. **Python 3.x** – основной язык программирования.
2. **Tkinter** – для создания графического интерфейса (входит в стандартный дистрибутив Python).
3. **Matplotlib** – для построения круговых диаграмм. Устанавливается с помощью pip install matplotlib.

**3. Структура проекта**

Проект состоит из одного файла с кодом, который включает:

* Классы для хранения и управления банковскими вкладами.
* Логику загрузки данных из файла.
* Сегментацию данных по категориям.
* Построение круговых диаграмм для визуализации данных.

**4. Описание основных классов и модулей**

1. **Класс BankDeposit**

* Описание: Представляет данные об одном банковском вкладе.
* Основные атрибуты:
  + deposit\_id: уникальный идентификатор вклада.
  + deposit\_type: тип вклада (например, "депозит", "сберегательный" и т.д.).
  + amount: сумма вклада.
* Конструктор \_\_init\_\_: инициализирует объект с указанными значениями атрибутов.

**Класс DepositManager**

* Описание: Управляет коллекцией вкладов, реализует методы для сегментации и загрузки данных.
* Основные методы:
  + load\_from\_file(self, filename): загружает данные о вкладах из файла, проверяет корректность формата данных.
  + segment\_by\_type(self): сегментирует вклады по типу.
  + segment\_by\_amount(self): сегментирует вклады на категории "мелкие", "средние" и "крупные" в зависимости от суммы.

**Класс App**

* Описание: Основной класс для управления графическим интерфейсом.
* Основные методы:
  + load\_data(self): позволяет пользователю выбрать файл для загрузки данных о вкладах.
  + visualize\_by\_type(self): строит круговую диаграмму сегментации вкладов по типам.
  + visualize\_by\_amount(self): строит круговую диаграмму сегментации вкладов по категориям сумм.
* Основные элементы интерфейса:
  + Кнопка "Загрузить данные" для выбора файла.
  + Кнопки для построения диаграмм: "Сегментация по видам" и "Сегментация по типам"

**5. Детали реализации**

**Интерфейс загрузки данных**

* При нажатии на кнопку "Загрузить данные" вызывается метод load\_data. Он открывает окно выбора файла, загружает данные и передает их в DepositManager.
* Метод load\_from\_file проверяет корректность данных: каждая строка должна состоять из трех значений, разделенных запятыми. Если данные не соответствуют формату, выводится сообщение об ошибке.

**Сегментация данных**

* Метод segment\_by\_type формирует сегментацию данных по типам вкладов (например, "депозит", "накопительный" и т.д.), возвращая словарь с типами вкладов в качестве ключей.
* Метод segment\_by\_amount сегментирует вклады по суммам:
  + Вклады менее 20,000 считаются "мелкими".
  + Вклады от 20,000 до 50,000 считаются "средними".
  + Вклады свыше 50,000 считаются "крупными".

**Построение диаграмм**

* Для визуализации сегментированных данных используются круговые диаграммы (метод plt.pie из Matplotlib).
* visualize\_by\_type строит диаграмму распределения вкладов по типам, visualize\_by\_amount — по суммам.

**7. Тестирование и отладка**

**Тесты загрузки данных**

* Проверьте, что программа правильно обрабатывает файл с корректным и некорректным форматом данных.
* Убедитесь, что при ошибках ввода выводится соответствующее сообщение.

**Тесты сегментации**

* Проверьте, что сегментация по типу корректно распределяет вклады на основе их атрибута deposit\_type.
* Тестируйте сегментацию по сумме для значений, находящихся на границах категорий ("мелкие", "средние", "крупные").

**Тесты визуализации**

* Убедитесь, что построение диаграмм успешно работает при наличии данных и адекватно отображает распределение.
* Проверьте корректность названий категорий на диаграммах.

**8. Заключение**

Этот документ описывает основные компоненты и функции проекта "Управление банковскими вкладами". Классы и методы реализованы с учетом модульности и могут быть расширены для добавления новых функций, таких как сортировка вкладов или дополнительная фильтрация данных.