МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Измерительно-вычислительные комплексы»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

на лабораторную работу №8

по дисциплине «Алгоритмы и структуры данных»

|  |  |
| --- | --- |
| **Подп.и** |  |
| **Инв** |  |
| **Вза** |  |
| **Подп.и** |  |
| **Инв** |  |

**Объектно-ориентированная программа с графическим интерфейсом для работы с объектами программы (товар в магазине)**

Р.02069337. №23/748-Вариант 17 ТЗ-

Листов 13

Исполнитель:

студент гр. ИСТбд-23

Никольский Максим

Сергеевич

«\_\_» \_\_\_\_\_\_\_\_ 2024 г.

2024

**Введение**

Разрабатываемое приложение называется "Товары: Анализ и Сегментация". Условное обозначение приложения: TAS. Приложение предназначено для анализа и сегментации товаров на основе различных параметров, таких как категории и объемы продаж.

Приложение предоставляет следующие функциональные возможности:

**Загрузка данных из CSV-файла.** Пользователь может загрузить информацию о товарах из CSV-файлов, содержащих такие параметры, как название товара, категория, цена и количество.

**Сегментация товаров по категориям.** Приложение автоматически распределяет товары по категориям, указанным в исходном файле, и выводит статистику по каждой категории.

**Сегментация товаров по объемам продаж.** Приложение рассчитывает общий объем продаж каждого товара и группирует их по этому параметру.

**Визуализация сегментов.** Результаты сегментации отображаются в виде круговых диаграмм, что позволяет наглядно оценить распределение товаров по различным критериям.

**Интерфейс пользователя.** Программный продукт оснащен интуитивным графическим интерфейсом, позволяющим легко управлять процессом анализа и просмотра результатов.

# Проектная часть

* 1. **Постановка задачи на разработку приложения**

Детализируется в разработанном техническом задании.

* 1. **Математические методы**

**Математические модели для сегментации данных**

1. **Сегментация по категориям:**

* Данная операция представляет собой простую классификацию товаров по заранее определенным категориям. Для этого используется процедура группировки данных на основе значений одного из признаков (в данном случае — категории товара).
* Формально, пусть X — множество товаров, C — набор уникальных категорий. Тогда сегментация по категориям сводится к созданию множества подмножеств Xi⊆ X, где каждое подмножество Xi содержит товары, принадлежащие одной и той же категории ci​∈C.

1. **Сегментация по объемам продаж:**

* Объем продаж для каждого товара рассчитывается как произведение его цены (PP) на количество (QQ). Затем товары группируются по полученному объему продаж.
* Пусть S — объем продаж товара, тогда сегментация по продажам означает разбиение множества товаров X на подмножества Yj ​⊆X таким образом, что для любого товара xk​∈Yj​ выполняется условие S(xk)=Sj​, где Sj​ — некоторый диапазон значений объема продаж.

**Модель для построения круговых диаграмм**

Для визуализации результатов сегментации используется библиотека Matplotlib, которая предоставляет средства для построения различных типов графиков, включая круговые диаграммы (pie charts).

1. **Круговая диаграмма:**

* Круговая диаграмма представляет собой круг, разделенный на секторы, каждый из которых соответствует определенной доле в общем объеме данных. Площадь сектора пропорциональна величине соответствующего показателя.
* Для построения круговой диаграммы требуется вычислить доли каждого сектора. Пусть N — общее число товаров, а ni​ — количество товаров в сегменте i. Тогда доля pi​ для сегмента i определяется как:

pi=ni/N

* Далее эти доли используются для построения диаграммы с помощью функции plt.pie из библиотеки Matplotlib.

**Обоснование выбора моделей**

1. **Простота и эффективность:**

* Методы сегментации по категориям и объемам продаж просты в реализации и понимании. Они не требуют сложных вычислительных ресурсов и подходят для обработки небольших и средних объемов данных.

1. **Широкая поддержка библиотек:**

* Библиотека Python, такая как Matplotlib, предоставляет мощные инструменты для работы с данными и построения графиков. Это значительно ускоряет процесс разработки и снижает необходимость написания собственного кода для базовых операций.
  1. **Архитектура и алгоритмы**
     1. Архитектура

**Основные структуры данных**

1. **CSV-файл:**

* Исходные данные о товарах хранятся в CSV-файлах. Каждая строка файла представляет собой запись о товаре, содержащую следующую информацию:

1. Название товара (name)
2. Категория товара (category)
3. Цена товара (price)
4. Количество товара (quantity*)*
5. **Объект Product:**

* Данные из CSV-файлов преобразуются в объекты класса Product, который хранит информацию о каждом товаре. Атрибуты объекта включают:

class Product:

def \_\_init\_\_(self, name, category, price, quantity):

self.name = name

self.category = category

self.price = float(price) if price else None

self.quantity = int(quantity) if quantity else None

1. **Списки и словари:**

* После загрузки данных из CSV, информация хранится в виде списка объектов Product. Дальнейшая обработка данных ведется с использованием списков и словарей:

products = [] # Список объектов Product

categories = {} # Словарь, где ключи - категории, значения - списки товаров

sales = {} # Словарь, где ключи - объемы продаж, значения - списки товаров

**Основные функции и их взаимодействие**

1. **Функция read\_csv\_file():**

* Загружает данные из CSV-файла и преобразует их в список объектов Product.
* Возвращаемое значение: список объектов products.

1. **Функции сегментации:**

* segment\_by\_categories(products):

Сегментирует товары по категориям, создавая словарь categories, где ключи — уникальные категории, а значения — списки товаров, относящихся к данной категории.

* segment\_by\_sales(products):

Рассчитывает общий объем продаж для каждого товара и группирует товары по объемам продаж, создавая словарь sales, где ключи — объемы продаж, а значения — списки соответствующих товаров.

1. **Функции визуализации:**

* visualize\_segmentation\_by\_categories(categories):

Строит круговую диаграмму, показывающую распределение товаров по категориям.

* visualize\_segmentation\_by\_sales(sales):

Строит круговую диаграмму, отображающую распределение товаров по объемам продаж.

1. **GUI-компоненты:**

* Интерфейс построен с использованием библиотеки Tkinter. Основные компоненты включают:
* Кнопки для загрузки данных, выполнения сегментации и визуализации.
* Элементы для отображения сообщений об ошибках и статусов выполнения.
* Окна для отображения круговых диаграмм.

**Взаимодействие компонентов**

1. Пользователь загружает CSV-файл через интерфейс.
2. Приложение вызывает функцию read\_csv\_file(), которая преобразует данные в список products.
3. Пользователь выбирает тип сегментации (по категориям или по продажам).
4. Соответствующая функция сегментации (segment\_by\_categories() или segment\_by\_sales()) вызывается, и создаются словари categories или sales.
5. Пользователь запрашивает визуализацию результатов, вызывая функции visualize\_segmentation\_by\_categories(categories) или visualize\_segmentation\_by\_sales(sales).
6. Результат отображается в виде круговой диаграммы.
   * 1. Алгоритм формирования диаграммы

Алгоритм формирования круговой диаграммы в вашем приложении можно описать следующим образом:

1. **Сбор данных:**

* Извлекаются данные из словаря categories или sales, где ключи — это категории или объёмы продаж, а значения — списки товаров.
* Формируются списки labels (категории/объёмы продаж) и sizes (количества товаров в каждой категории/объемах продаж).

1. **Вычисления:**

* Вычисляются проценты для каждого сектора диаграммы.
* Определяется порядок сортировки секторов для более аккуратного отображения.

1. **Построение диаграммы:**

* Создаются новые фигуры и оси для диаграммы с помощью библиотеки Matplotlib.
* Построение круговой диаграммы, где размеры секторов соответствуют долям, и добавляются подписи для каждого сектора.

1. **Отображение диаграммы:**

* Диаграмма выводится на экран.
  1. **Тестирование**

Результаты диаграммы по продажам

Результаты диаграммы по категориям

Круговая диаграмма продаж

Круговая диаграмма категорий

Визуализация данных

Функции сегментации

Кнопка сегментации

Модуль чтения данных

Кнопка «Загрузить Данные»

GUI

Словарь продаж

Сегментация по продажам

Словарь категорий

Сегментация по категориям

Функции сегментации

Список товаров

Класс Product

CSV-файл

Модуль чтения данных

* + 1. Описание отчета о тестировании

Отчет о тестировании программы анализа товаров предназначен для документирования процесса тестирования функциональности и надежности программного продукта, разработанной с использованием объектно-ориентированного программирования и библиотеки Tkinter.

* + 1. Цель тестирования

Цель тестирования — убедиться, что приложение работает корректно, соответствует требованиям и не содержит критических ошибок.

* + 1. Методика тестирования

Тестирование проводилось с использованием следующих методов:

1. Функциональное тестирование — проверка основных функций, таких как чтение данных из CSV, сегментация товаров и отображение результатов.
2. Тестирование пользовательского интерфейса (UI) — оценка удобства и доступности элементов интерфейса.
3. Нагрузочное тестирование — измерение производительности при большой нагрузке (например, с множеством товаров).
4. Регрессионное тестирование — повторное выполнение ключевых тестов после изменений в коде.
5. Юнит-тестирование — проверка отдельных частей кода (методов, функций) на корректность работы.
6. Интеграционное тестирование — проверка взаимодействия разных модулей приложения.
   * 1. Проведённые тесты
7. **Функциональное тестирование**

Сценарий: Загрузка данных из CSV-файла.

* + Ожидаемый результат: Приложение успешно загружает данные, и создается список объектов Product.
  + Реальный результат: Данные были успешно загружены, и список объектов создан.

Сценарий: Сегментация товаров по категориям.

* Ожидаемый результат: Товары правильно распределяются по категориям, без ошибок.
* Реальный результат: Сегментация прошла успешно, все товары распределены по правильным категориям.

Сценарий: Вывод результатов сегментации в виде круговой диаграммы.

* Ожидаемый результат: Диаграмма отображается корректно, с верными данными.
* Реальный результат: Диаграммы отображаются корректно и содержат правильные данные.

1. **Тестирование пользовательского интерфейса (UI)**

Сценарий: Проверка доступности кнопок.

* Ожидаемый результат: Все кнопки активны и доступны для кликов.
* Реальный результат: Кнопки активны, но одна кнопка ("Сохранить") неактивна при отсутствии данных.

Сценарий: Тестирование реакции на неверные данные.

* Ожидаемый результат: При попытке открыть несуществующий файл появляется сообщение об ошибке.
* Реальный результат: Сообщение об ошибке появляется, как ожидалось.

1. **Нагрузочное тестирование**

Сценарий: Работа с большим объемом данных (например, 5000 товаров).

* Ожидаемый результат: Время выполнения сегментации и визуализации остается приемлемым (< 30 секунд).
* Реальный результат: Процесс занял около 25 секунд, что соответствует ожиданиям.

1. **Регрессионное тестирование**

Сценарий: Повторное выполнение ключевых тестов после добавления новой функции.

* Ожидаемый результат: Ранее работавшие функции продолжают работать корректно.
* Реальный результат: Некоторые старые функции работали неправильно из-за конфликта с новым кодом. Проблемы устранены.

1. **Автоматизированное тестирование**

Сценарий: Юнит-тест метода validate\_price\_and\_quantity.

* Ожидаемый результат: Метод возвращает правильные значения для валидных данных и None для некорректных.
* Реальный результат: Тест пройден успешно, метод работает корректно.

1. **Юнит-тестирование**

Сценарий: Проверка конструктора класса Product на создание экземпляра.

* Ожидаемый результат: Экземпляр класса создается с правильными значениями свойств.
* Реальный результат: Конструктор работает корректно, экземпляр создается с нужными свойствами.

1. **Интеграционное тестирование**

Сценарий: Совместная работа модуля чтения данных и функции сегментации.

* Ожидаемый результат: Данные из CSV корректно обрабатываются и сегментированы.

Реальный результат: Модули работают вместе корректно, данные обрабатываются правильно.

* + 1. Чек-лист для проверки файла

1. **Общая структура файла**

* Файл имеет расширение .csv.
* Первая строка файла содержит заголовки столбцов.
* Количество столбцов одинаково для каждой строки.
* Нет пустых строк в середине файла.

1. **Проверка содержимого**

* Столбец *Название товара* заполнен для всех строк.
* Значения в столбце *Категория* соответствуют допустимым категориям (например: Электроника, Одежда, Продукты питания).
* В столбце *Цена* указаны только числовые значения.
* В столбцах *Количество* и *Продано* содержатся только целые положительные числа.
* Дата в столбце *Дата поступления* указана в формате ДД.ММ.ГГГГ.

1. **Логическая целостность данных**

* Сумма значений в столбцах *Цена* и *Количество* соответствует общей стоимости товара.
* Даты поступления товара (*Дата поступления*) находятся в диапазоне допустимых дат (например: последние 12 месяцев).
* Все уникальные идентификаторы товаров (*ID товара*) уникальны и не дублируются.

1. **Специальные случаи**

* Проверка на наличие специальных символов (например: ", \n, ;), которые могут нарушить парсинг файла.
* Наличие комментариев или примечаний в конце файла (если требуется, должны быть удалены).
* Кодировка файла UTF-8 (или другая требуемая кодировка).