МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет ИСТ

Кафедра «Измерительно-вычислительные комплексы»

Дисциплина «Алгоритмы и структуры данных»

**ЛАБОРАТОРНАЯ РАБОТА**

по дисциплине «Алгоритмы и структуры данных»

Тема «Разработка объектно-ориентированной программы с графическим интерфейсом»

Исполнитель:

студент гр. ИСТбд -22

Марочкина А.Д

« » 2024 г.

Ульяновск 2024 г.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет ИСТ

Кафедра «Измерительно-вычислительные комплексы»

Дисциплина «Алгоритмы и структуры данных»

**ЗАДАНИЕ НА ЛАБОРАТОРНУЮ РАБОТУ**

Студенту ИСТбд-22 Марочкина А.Д.

группа фамилия, инициалы

Тема работы «Разработка объектно-ориентированной программы с графическим интерфейсом»

**Исходные данные к работе:** описание задания по теме, утвержденной распоряжением деканата ФИСТ

**Рекомендуемая литература:** курс лекций по дисциплине «Алгоритмы и структуры данных», методические указания к лабораторным работам по дисциплине «Алгоритмы и структуры данных», интернет- источники.

**Содержание пояснительной записки** (перечень подлежащих разработке вопросов)

Первая глава. Описание объектно-ориентированного программирования.

Вторая глава. Представление руководства пользователя для разработанного проекта. Третья глава. Представление руководства программиста для разработанного проекта.

**Перечень графического материала** (с точным указанием обязательных чертежей) Скриншоты разработанного программного продукта.

**Записка к Лабораторной работе по теме:**

**«Разработка объектно-ориентированной программы с графическим интерфейсом»»**

**Дисциплина:**

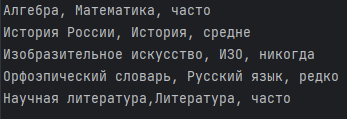
**«Алгоритмы и структуры данных»**

Глава 1. Описание ООП

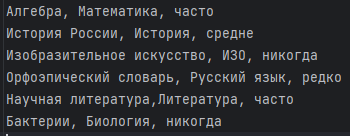
Объектно-ориентированное программирование (ООП) — это подход, при котором программа рассматривается как набор объектов, взаимодействующих друг с другом. У каждого есть свойства и поведение.

Проект представляет собой программу, в котором пользователи могут добавлять и удалять элементы из списка, визуализировать сегментацию полного списка книг по учебным дисциплинам в форме круговой диаграммы, а также визуализировать сегментацию полного списка книг по использованию (часто, средне, редко, никогда) в форме круговой диаграммы.

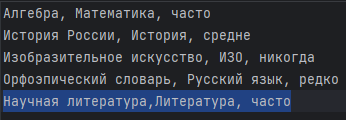
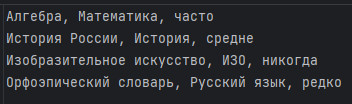
**Глава 2. Руководство пользователя**



1. **Добавление элемента в список**: Нажимаете клавишу Enter и вводите название предмета, учебную дисциплину и как часто вы используете книгу. Затем сохраните изменения с помощью комбинации клавиш Ctrl + C

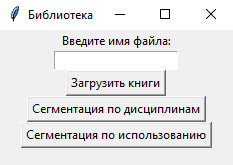
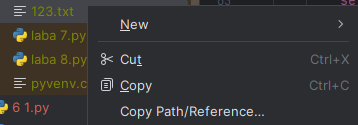


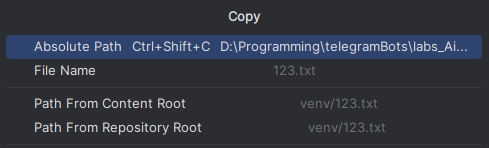
1. **Удаление элемента из списка**:
   * Выделяете нужную вам строчку и нажимаете клавишу Delete.

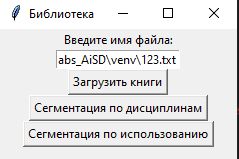
**Запуск программы и ввод необходимых данных**:

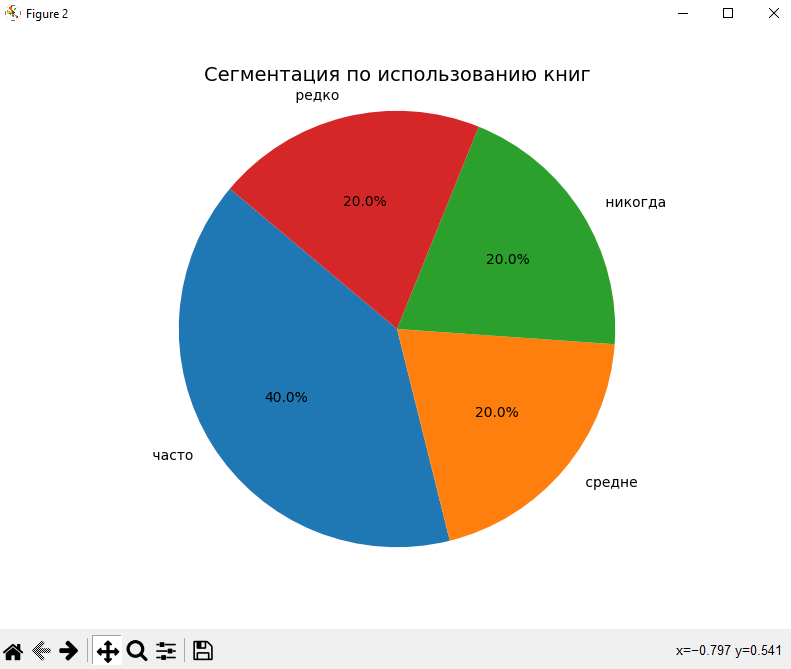
* Запускаете программу с помощью комбинации клавиш Shift + F10. Затем у вас появляется окно, далее в текстовом поле «Введите имя файла» необходимо вставить нужную нам информацию. Для этого правой кнопкой мыши нажимаете на 4 строчку, затем нажимаете на 1 строчку.

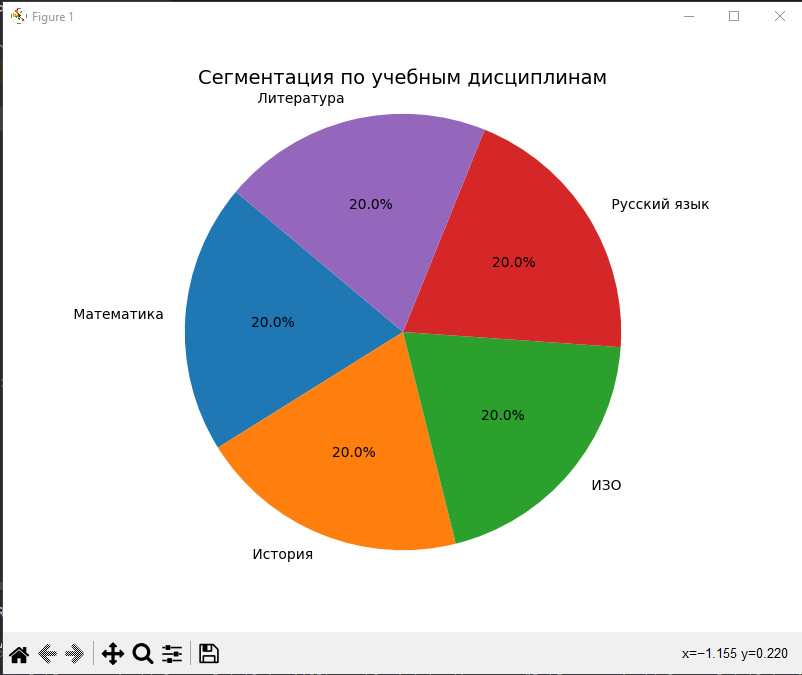




1. **Визуализация сегментации полного списка книг по 2 критериям**:



Затем нажимаете на кнопку «Загрузить книги», а далее на нужную вам сегментацию.



**Глава 3. Руководство программиста**

**Используемые библиотеки:**

tkinter для создания графического интерфейса пользователя (GUI)

matplotlib для визуализации данных

csv для работы с CSV-файлами.

**Класс Book**

Класс, который содержит конструктор с названием книги, предмета и частотой использования

**Методы:**

* **\_\_str\_\_**(self): возвращает книги в виде строки.
* **\_\_init\_\_**(self, title, discipline, usage): инициализация книги с переданными параметрами

**Класс Library**

Класс для управления списком книг и их сегментацию

**Методы:**

* **load\_books**: Загружает книги из CSV-файла и проверяет правильность формата данных.
* \_\_init\_\_(self):инициализирует пустой список книг.
* segment\_by\_discipline(self:сегментирует книги по дисциплинам, возвращая словарь с количеством книг для каждой дисциплины.
* segment\_by\_usage(self) — сегментирует книги по типу использования.
* visualize\_data(self, data, title) — создает круговую диаграмму на основе переданных данных и заголовка.

**Класс LibraryApp**

Класс, отвечающий за создание и управление графическим интерфейсом.

**Методы:**

* \_\_init\_\_(self, master) — инициализирует интерфейс приложения, включая окна, кнопки и ввод данных.
* center\_window(self, width, height) — центрирует окно приложения на экране.
* load\_books(self) — вызывает метод для загрузки книг.
* segment\_by\_discipline(self) — вызывает метод для сегментации книг по дисциплинам и их визуализации.
* segment\_by\_usage(self) — вызывает метод для сегментации книг по использованию и их визуализации.

**Глава 4. Тестирование**

1. **Описание отчета**

В данном отчете представлены результаты тестирования программы

на основе разработанных test-case и чек-листов, статического тестирования документации и программного кода. Описаны проведенные тесты, их результаты и обнаруженные дефекты.

**2**. **Цель тестирования**

Целью тестирования является проверка соответствия ПО предъявляемым требованиям, а также выявление возможных багов. По результатам тестирования следует исправление выявленных багов.

**3. Методика тестирования**

Тестирование проводилось с использованием следующих методов:

– Статическое тестирование: анализ и проверка кода без его запуска, выявление ошибок в технической документации.

– Ручное тестирование: запуск пользовательских сценариев

программы с различными входными данными и проверка корректности

полученных результатов.

1. **Проведенные тесты**

В ходе тестирования были проведены следующие тесты:

**4.1) Статическое тестирование**

Количество обнаруженных и исправленных ошибок в документации:

[количество ошибок].

Количество обнаруженных и исправленных ошибок в программном

коде: [количество ошибок].

**4.2) Ручное тестирование.**

Написаны и проведены следующие тест-кейсы и чек-листы:

**ТК1. Отработка используемого файла.**

Предварительные шаги:

Создать txt файл с данными: Алгебра, Математика, часто  
История России, История, средне  
Изобразительное искусство, ИЗО, никогда  
Орфоэпический словарь, Русский язык, редко  
Научная литература, Литература, часто

Шаги:

1. Запустить программу.
2. В окне имя файла ввести путь файла.
3. Нажать на кнопку «Загрузить книги»

Ожидаемый результат:

Компилятор не выявит никакой ошибки.

Фактический результат:

Пользователь не получил сообщения об ошибке.

**ТК2. Использование пустого файла или списка с двумя столбцами.**

Предварительные шаги:

Изменить txt файл.

История России, средне  
Изобразительное искусство, никогда  
Орфоэпический словарь, редко  
Научная литература, часто

Шаги.

1. Запустить программу.
2. В окне имя файла ввести путь файла.
3. Нажать на кнопку «Загрузить книги»

Ожидаемый результат:

Пользователь получит сообщение об ошибке.

Фактический результат:

Пользователь получил сообщение об ошибке.

Чек лист

|  |  |  |
| --- | --- | --- |
| Описание | Пример | Результат |
| Проверка поля «Имя файла» | | |
| Корректный список | Алгебра, Математика, часто История России, История, средне Изобразительное искусство, ИЗО, никогда Орфоэпический словарь, Русский язык, редко Научная литература, Литература, часто | Отсутствие сообщения об ошибке |
| Пустой файл |  | Ошибка |
| Количество столбцов 2 | История России, средне Изобразительное искусство, никогда Орфоэпический словарь, редко Научная литература, часто | Ошибка |

**5. Вывод**

На основе проведенных тестов сделаны следующий вывод:

– Программа успешно прошла все тесты и работает корректно.

**Заключение**

В ходе выполнения данной лабораторной работы было разработано объектно-ориентированная программа с графическим интерфейсом и основных операций, таких как добавление элемента и его удаление, а также визуализация сегментации полного списка книг по 2 критериям. Результатом этой работы стало создание инструмента, который обеспечивает пользователю понятный и наглядный способ ознакомления с функциональностью алгоритма.

Визуализация позволяет улучшить понимание принципов работы этого подхода. Это особенно важно для студентов и начинающих программистов, которые изучают структуры данных и стремятся улучшить свои навыки программирования.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Шишкин В.В., Афонин Д.С. Разработка логических компьютерных игр с графическим интерфейсом в среде Питон. - УлГТУ, 2023
2. Объектно-ориентированное программирование с помощью Python / Ирв Кальб ; [перевод с английского М. А. Райтмана]. — Москва : Эксмо, 2024. — 512 с; [электронный ресурс]

Программа. Листинг

import tkinter as tk  
from tkinter import messagebox  
import matplotlib.pyplot as plt  
import csv  
from collections import defaultdict  
  
class Book:  
 def \_\_init\_\_(self, title, discipline, usage):  
 self.title = title  
 self.discipline = discipline  
 self.usage = usage  
  
 def \_\_str\_\_(self):  
 return f"{self.title}, {self.discipline}, {self.usage}"  
  
class Library:  
 def \_\_init\_\_(self):  
 self.books = []  
  
 def load\_books(self, filename):  
 try:  
 with open(filename, 'r', encoding='utf-8') as file:  
 reader = csv.reader(file)  
 for row in reader:  
 if len(row) == 3:  
 title, discipline, usage = row  
 self.books.append(Book(title, discipline, usage))  
 else:  
 raise ValueError("Неверный формат данных в файле.")  
 except Exception as e:  
 messagebox.showerror("Ошибка", str(e))  
  
 def segment\_by\_discipline(self):  
 discipline\_count = defaultdict(int)  
 for book in self.books:  
 discipline\_count[book.discipline] += 1  
 return discipline\_count  
  
 def segment\_by\_usage(self):  
 usage\_count = defaultdict(int)  
 for book in self.books:  
 usage\_count[book.usage] += 1  
 return usage\_count  
  
 def visualize\_data(self, data, title):  
 plt.figure(figsize=(8, 6))  
 title\_obj = plt.title(title, fontsize=14)  
 title\_obj.set\_position([0.5, 2])  
 labels = data.keys()  
 sizes = data.values()  
 plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=140)  
 plt.subplots\_adjust(top=0.9, bottom=0.1, left=0.1, right=0.9, hspace=0.8)  
 plt.axis('equal')  
 plt.show()  
  
  
class LibraryApp:  
 def \_\_init\_\_(self, master):  
 self.master = master  
 self.master.title("Библиотека")  
  
 self.center\_window(400, 140)  
 self.library = Library()  
 self.label = tk.Label(master, text="Введите имя файла:")  
 self.label.pack()  
 self.filename\_entry = tk.Entry(master)  
 self.filename\_entry.pack()  
 self.load\_button = tk.Button(master, text="Загрузить книги", command=self.load\_books)  
 self.load\_button.pack()  
 self.segment\_discipline\_button = tk.Button(master, text="Сегментация по дисциплинам", command=self.segment\_by\_discipline)  
 self.segment\_discipline\_button.pack()  
 self.segment\_usage\_button = tk.Button(master, text="Сегментация по использованию", command=self.segment\_by\_usage)  
 self.segment\_usage\_button.pack()  
  
 def center\_window(self, width, height):  
 screen\_width = self.master.winfo\_screenwidth()  
 screen\_height = self.master.winfo\_screenheight()  
 x = (screen\_width // 2) - (width // 2)  
 y = (screen\_height // 2) - (height // 2)  
 self.master.geometry(f"{width}x{height}+{x}+{y}")  
  
 def load\_books(self):  
 filename = self.filename\_entry.get()  
 self.library.load\_books(filename)  
  
 def segment\_by\_discipline(self):  
 data = self.library.segment\_by\_discipline()  
 self.library.visualize\_data(data, "Сегментация по учебным дисциплинам")  
  
 def segment\_by\_usage(self):  
 data = self.library.segment\_by\_usage()  
 self.library.visualize\_data(data, "Сегментация по использованию книг")  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 root = tk.Tk()  
 app = LibraryApp(root)  
 root.mainloop()