**Руководство программиста**

**1. Обзор программы**

Программа представляет собой графический интерфейс (GUI) для управления списком факультативов. Она позволяет:

* Загружать данные о факультативах из файла "facultatives.txt".
* Сохранять данные о факультативах в файл "facultatives.txt".
* Просматривать список факультативов.
* Добавлять новые факультативы.
* Выполнять сегментацию факультативов по студентам и преподавателям.
* Отображать результаты сегментации в виде диаграммы.

**2. Структура программы**

Программа состоит из двух классов:

1. Facultative: Представляет собой объект факультатива. Имеет следующие атрибуты:

* name: Название факультатива.
* student: Имя студента, посещающего факультатив.
* teacher: Имя преподавателя факультатива.

1. FacultativeManager: Управляет списком факультативов. Имеет следующие методы:

* init(): Инициализирует менеджер, загружает данные из файла и создает GUI.
* load\_data(): Загружает данные о факультативах из файла.
* save\_data(): Сохраняет данные о факультативах в файл.
* segment\_by\_student(): Выполняет сегментацию по студентам.
* segment\_by\_teacher(): Выполняет сегментацию по преподавателям.
* create\_gui(): Создает GUI программы.
* show\_student\_data(): Отображает результаты сегментации по студентам.
* show\_teacher\_data(): Отображает результаты сегментации по преподавателям.
* add\_facultative(): Добавляет новый факультатив.
* update\_facultative\_listbox(): Обновляет список факультативов в GUI.
* delete\_facultative(): Удаляет выбранный факультатив.
* edit\_facultative(): Редактирует выбранный факультатив.
* visualize\_data(data, title): Отображает данные в виде диаграммы.

**3. Алгоритм работы программы**

1. Инициализация:

* При запуске программы создается экземпляр класса FacultativeManager.
* FacultativeManager загружает данные из файла "facultatives.txt".
* FacultativeManager создает GUI программы.

2. Взаимодействие пользователя:

* Пользователь может нажать на кнопки "Сегментация по студентам", "Сегментация по преподавателям", "Добавить факультатив", "Обновить список".
* При нажатии на кнопку "Добавить факультатив" открывается диалоговое окно для ввода информации о новом факультативе.
* При нажатии на кнопку "Удалить факультатив" удаляется выбранный в списке факультатив.
* При нажатии на кнопку "Редактировать факультатив" открывается диалоговое окно для редактирования выбранного факультатива.

3. Обработка данных:

* При нажатии на кнопки "Сегментация по студентам" или "Сегментация по преподавателям" вызывается соответствующий метод FacultativeManager ( segment\_by\_student() или segment\_by\_teacher()).
* Методы segment\_by\_student() и segment\_by\_teacher() возвращают словарь с количеством факультативов для каждого студента или преподавателя.
* Результаты сегментации используются для создания диаграммы.

4. Отображение результатов:

* Результаты сегментации отображаются в виде диаграммы.
* Список факультативов отображается в Listbox в GUI.

**4. Описание кода**

facultative.py:

class Facultative:

def init(self, name, student, teacher):

self.name = name

self.student = student

self.teacher = teacher

def str(self):

return f"{self.name} - Student: {self.student}, Teacher: {self.teacher}"

1. Класс Facultative:

* Метод init(self, name, student, teacher): Инициализирует объект Facultative с заданными параметрами.
* Метод str(self): Возвращает строковое представление объекта Facultative.

facultative\_manager.py:

import tkinter as tk

from tkinter import ttk

import matplotlib.pyplot as plt

from matplotlib.backends.backend\_tkagg import FigureCanvasTkAgg

from tkinter import messagebox

import re

# ... (код класса FacultativeManager)

1. Класс FacultativeManager:

* Метод init(self):
* Инициализирует список facultatives.
* Вызывает load\_data().
* Вызывает create\_gui().
* Метод load\_data(self):
* Загружает данные о факультативах из файла "facultatives.txt".
* Обрабатывает возможные ошибки, например, если файл не найден.
* Метод save\_data(self):
* Сохраняет данные о факультативах в файл "facultatives.txt".
* Метод segment\_by\_student(self):
* Выполняет сегментацию по студентам.
* Возвращает словарь с количеством факультативов для каждого студента.
* Метод segment\_by\_teacher(self):
* Выполняет сегментацию по преподавателям.
* Возвращает словарь с количеством факультативов для каждого преподавателя.
* Метод create\_gui(self):
* Создает главное окно root.
* Устанавливает заголовок окна.
* Определяет размеры окна и позицию на экране.
* Создает рамку main\_frame для кнопок и отображения данных.
* Создает кнопки "Сегментация по студентам", "Сегментация по преподавателям", "Добавить факультатив".
* Создает рамку chart\_frame для диаграммы.
* Создает Listbox для отображения списка факультативов.
* Создает кнопки "Обновить список" и "Удалить".
* Метод show\_student\_data(self):
* Вызывает segment\_by\_student().
* Отображает результаты сегментации в виде диаграммы.
* Метод show\_teacher\_data(self):
* Вызывает segment\_by\_teacher().
* Отображает результаты сегментации в виде диаграммы.
* Метод add\_facultative(self):
* Создает диалоговое окно для ввода информации о новом факультативе.
* Добавляет новый факультатив в список facultatives.
* Обновляет список факультативов в GUI.
* Метод update\_facultative\_listbox(self):
* Обновляет список факультативов в GUI.

**5. Рекомендации по использованию**

* Программа использует файл "facultatives.txt" для хранения данных. Файл должен быть создан вручную и содержать информацию о факультативах в формате:

(Название факультатива, Имя студента, Имя преподавателя)

* Для добавления нового факультатива необходимо заполнить диалоговое окно с информацией о факультативе.
* Для сегментации по студентам или преподавателям достаточно нажать соответствующую кнопку.

**6. Дополнительные возможности**

* Добавить возможность удаления факультативов.
* Добавить возможность редактирования данных о факультативах.
* Добавить возможность фильтрации списка факультативов.
* Добавить возможность вывода результатов сегментации в файл.
* Добавить возможность использования других типов диаграмм.

**7. Тестирование**

* Тестирование программы должно включать:
* Проверку загрузки данных из файла.
* Проверку сохранения данных в файл.
* Проверку добавления новых факультативов.
* Проверку сегментации по студентам и преподавателям.
* Проверку корректности отображения результатов сегментации.

**8. Документация**

* Данное руководство программиста является основным документом по программе.
* Дополнительную документацию можно добавить в виде комментариев к коду.
* Важно использовать стандартные соглашения по наименованию переменных и функций.
* Документация должна быть четкой, понятной и доступной для пользователя.

**9. Замечания**

* Программа использует библиотеку tkinter для создания GUI.
* Программа использует библиотеку matplotlib для создания диаграмм.
* Программа использует кодировку cp1251 для чтения и записи данных из файла.

**10. Дополнительные замечания**

* Программа может быть расширена и улучшена.
* Программа может быть использована как основа для создания более сложных приложений.
* Важно следовать принципам модульности, повторного использования кода и ясности написания кода.