**Руководство программиста.**

**Руководство программиста на разработку программы "Договоры на тех. обслуживание станков".**

**1. Обзор программы**

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

* Загружать данные о факультативах из файла "Код.txt".
* Сохранять данные о факультативах в файл "Код.txt".
* Добавлять новые факультативы.
* Выполнять сегментацию списков договоров по видам станка и именам заказчиков.
* Отображать результаты сегментации в виде диаграммы.

**2. Структура программы**

Программа состоит из двух классов:

1. Contract: Представляет собой объект списка договоров. Имеет следующие атрибуты:

* machine\_type: Виды станков.
* customer: Заказчик.
* contract: Номер договора

1. ContractManager: Управляет списком договоров. Имеет следующие методы:

* init(): Инициализирует менеджер, загружает данные из файла и создает GUI.
* load\_data(): Загружает данные о договорах из файла.
* segment\_by\_machine\_type(): Выполняет сегментацию по видам станка.
* segment\_by\_customer(): Выполняет сегментацию по именам заказчика.
* visualize\_data(): Отображает данные в виде диаграммы.

**3. Алгоритм работы программы**

1. Инициализация:

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

2. Взаимодействие пользователя:

* Пользователь может нажать на кнопки "Сегментация по видам станков", "Сегментация по заказчикам".

3. Обработка данных:

* При нажатии на кнопки "Сегментация по видам станков" или "Сегментация по заказчикам" вызывается соответствующий метод ContractManager. (segment\_by\_machine\_type() или segment\_by\_customer()).
* Методы segment\_by\_machine\_type() и segment\_by\_customer() возвращают списки с количеством договоров в связи с условием.
* Результаты сегментации используются для создания диаграммы.

4. Отображение результатов:

* Результаты сегментации отображаются в виде диаграммы.

**4. Описание кода**

class Contract:

def \_init\_(self, machine\_type, customer, contract):

self.machine\_type = machine\_type

self.customer = customer

self.contract = contract

1. Класс Contract:

* Метод init(self, machine\_type, customer, contract): Инициализирует объект Contract с заданными параметрами.

import tkinter as tk

from tkinter import filedialog, messagebox

import matplotlib.pyplot as plt

from collections import Counter

from tkinter import messagebox

import csv

import os

#(код класса ContractManager)

1. Класс ContractManager:

* Метод init(self):
* Инициализирует список Код.
* Вызывает load\_data().
* Вызывает visualize\_data().
* Метод load\_data(self):
* Загружает данные о факультативах из файла "Код.txt".
* Обрабатывает возможные ошибки, например, если файл не найден.
* Метод segment\_by\_machine\_type(self):
* Выполняет сегментацию по видам станков.
* Метод segment\_by\_customer(self):
* Выполняет сегментацию по именам заказчиков
* Метод visualize\_data(self, data\_counter, title):
* 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 видам станка", "Сегментация по именам заказчика", "Загрузить данные из файла".
* Создает рамку chart\_frame для диаграммы.

**5. Рекомендации по использованию**

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

(Номер договора, Вид станка, Имя заказчика)

* Для добавления нового договора необходимо добавить в текстовый файл информацию о новом договоре.
* Для сегментации по видам станка и имени заказчика достаточно нажать соответствующую кнопку.

**6. Дополнительные возможности**

* Добавить возможность удаления договоров
* Добавить возможность редактирования данных о договорах.
* Добавить возможность фильтрации списка договоров.
* 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 добавления новых договоров.
* Проверку сегментации по видам станков и имени заказчиков.
* 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 csv 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.