****

Федеральное государственное бюджетное образовательное учреждение  
высшего образования

**«РОССИЙСКАЯ АКАДЕМИЯ НАРОДНОГО ХОЗЯЙСТВА И ГОСУДАРСТВЕННОЙ СЛУЖБЫ ПРИ ПРЕЗИДЕНТЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»**

КОЛЛЕДЖ МНОГОУРОВНЕВОГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

ОТЧЕТ

По учебной практике УП.01.01 Разработка программных модулей

специальность 09.02.07 Информационные системы и программирование

Выполнила студентка:

Макарова А.С.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(подпись)

Гусятинер Л. Б.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(оценка)

Москва, 2023

СОДЕРЖАНИЕ

[**Раздел 1. Проектирование консольного приложения** 1](#_Toc152661839)

[1.1 Общая информация: 1](#_Toc152661840)

[1.2 Изучение входной и выходной документации 3](#_Toc152661841)

[1.3 Разработка требований к проекту. 4](#_Toc152661842)

[Построение диаграммы использования. 4](#_Toc152661843)

[1.4 Разработка сценария проекта 6](#_Toc152661844)

[1.5 Построение диаграммы классов 7](#_Toc152661845)

[**Раздел 2. Разработка консольного приложения** 8](#_Toc152661846)

[2.1 Разработка главного модуля 8](#_Toc152661847)

[2.2 Разработка входящих модулей 12](#_Toc152661848)

[2.3 Тестирование и отладка 14](#_Toc152661849)

[2.4 Дневник 15](#_Toc152661850)

[Использованные источники 15](#_Toc152661851)

[**Приложение 1. Руководство программиста** 16](#_Toc152661852)

[**Приложение 2. Руководство пользователя** 17](#_Toc152661853)

[**Приложение 3. Листинги** 18](#_Toc152661854)

ПРОЕКТИРОВАНИЕ КОНСОЛЬНОГО ПРИЛОЖЕНИЯ

РАЗДЕЛ 1

## 1.1 Общая информация:

Проект представляет собой консольное приложение на языке Python.

*Описание бизнес-процесса*

Клиент обращается с запросом, ссылаясь на необходимые требования. Менеджер принимает запрос, записывает данные о клиенте и его запросе, затем проверяет наличие абонемента и подтверждает запрос. После согласования между менеджером и клиентом обсуждается момент получения карты лояльности, к которой будет привязан абонемент. Сотрудники центра привязывают приобретенный клиентом абонемент к новой карте лояльности, затем карта отправляется клиенту или готовится к передаче клиенту в батутном центре. Менеджер отслеживает выполнение запроса и собирает данные о времени выполнения. После выполнения запроса менеджер составляет отчет о результатах и предоставляет его клиенту. Затем обсуждаются дальнейшие шаги и обратная связь с клиентом о качестве предоставленных услуг.

*Структура базы данных*

1. Таблица "Клиент":

- Идентификатор

- ФИО

- id Лояльности

- Персональная скидка

2. Таблица "Лояльность":

- Идентификатор

- Персональный номер

- id Абонемент

3. Таблица "Абонемент":

- Идентификатор

- Количество занятий

- Идентификатор Инструктор

- Дата занятия

- Стоимость

4. Таблица "Инструктор":

- Идентификатор

- ФИО

- Направление

- Стаж

- Режим работы

5. Таблица "История":

- Идентификатор

- Идентификатор Клиент

- Идентификатор Абонемент

- Дата и время

*Запросы к базе данных:*

1. Активные абонементы

2. Режим работы инструктора ДША

В данном проекте мы работаем в основе с библиотекой “tkinter”.

Библиотека Tinter установлена в Python в качестве стандартного модуля, поэтому нам не нужно устанавливать что-либо для его использования.

Чтобы импортировать его, пропишем

import tkinter

## 1.2 Изучение входной и выходной документации

*Запрос: Получить данные об Активных абонементах*

Входной файл

⋅ «Абонементы»

Id Кол-во занятий id Инструктор День недели и время Стоимость

1 12 Бобырев Ю.А. Пятница/13:00 6000

2 8 Гоблинов Р.Б. Вторник/15:00 4000

3 4 Бобров А.С. Понедельник/10:00 3000\

· «Клиенты»

Id ФИО Номер телефона idЛояльность Персональная скидка idИстория

1 Макарова А.С. 8987898989 1234 20% 8 посещений

2 Топчубаева А.С. 89878646377 2345 10% 10 посещений

3 Руднева А.А. 89354637263 3456 8% 12 посещений

Выходной файл

Id клиента ФИО id лояльность

1 Макарова А.С. 1234

2 Топчубаева А.С. 2345

*Запрос: Получить данные о Режиме работы инструктора ДША*

Входной файл

«Инструктор»

Id ФИО Направление Режим работы

1 Иванов И.И. Паркур 10:00-22:00

2 Кузнецов И.Н. Скалодром 12:00-20:00

3 Бобров О.А. ДША 12:00-18:00

Выходной файл

Id ФИО Режим работы

3 Бобров О.А, 12:00-18:00

## 1.3 Разработка требований к проекту.

## Построение диаграммы использования.

⋅ В таблицах "Клиент", "Абонемент", "Инструктор" и "Лояльность" не должно быть двух записей с одинаковым идентификатором.

⋅ Поля в таблицах должны корректно ссылаться на другие.

⋅ Дата покупки должна быть в определенном диапазоне.

Диаграмма использования приведена на Рисунке 1.



Рисунок 1. Диаграмма использования

## 1.4 Разработка сценария проекта

В данном разделе приведен сценарий использования программы пользователем (Рисунок 2).

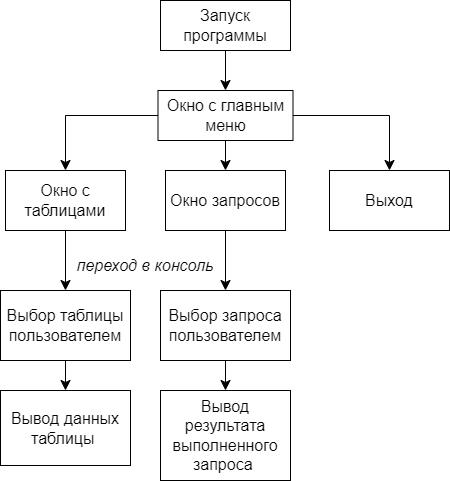


Рисунок 2. Сценарий проекта

РАЗДЕЛ 2

## 2.1 Разработка главного модуля

Главный модуль состоит из класса Trade.Выполняется при запуске программы. Его также можно назвать главным меню.

Листинг 1. Главный модуль Trade

class Trade:

    def \_\_init\_\_(self):

        self.clients = []

        self.loyalty = []

        self.abonements = []

        self.history = []

        self.instructors = []

    def main\_menu(self):

        while True:

            print("1. Добавить клиента")

            print("2. Добавить лояльность")

            print("3. Добавить абонемент")

            print("4. Добавить историю")

            print("5. Добавить инструктора")

            print("6. Вывести информацию")

            print("0. Выход")

            choice = input("Введите номер операции: ")

            if choice == "1":

                self.add\_client()

            elif choice == "2":

                self.add\_loyalty()

            elif choice == "3":

                self.add\_abonement()

            elif choice == "4":

                self.add\_history()

            elif choice == "5":

                self.add\_instructor()

            elif choice == "6":

                self.print\_info()

            elif choice == "0":

                break

            else:

                print("Некорректный ввод. Попробуйте снова.")

    def add\_client(self):

        # Логика добавления клиента

        pass

    def add\_loyalty(self):

        # Логика добавления лояльности

        pass

    def add\_abonement(self):

        # Логика добавления абонемента

        pass

    def add\_history(self):

        # Логика добавления истории

        pass

    def add\_instructor(self):

        # Логика добавления инструктора

        pass

    def print\_info(self):

        # Вывод информации о клиентах, лояльности, абонементах, истории, инструкторах

        Pass

    2.2 Разработка входящих модулей

Всего разработано 2 модуля:

* Класс Trade
* Класс Manager

Описание модуля:

Код содержит описание двух модулей (классов): "Manager" и "Trade".

Описание класса "Manager":

- Класс "Manager" имеет конструктор init, в котором инициализируется пустой список "self.clients" для хранения клиентов.

- В классе "Manager" определен метод "add\_client", который принимает имя клиента и добавляет его в список "self.clients". После добавления клиента выводится сообщение об успешном добавлении.

- В классе "Manager" также определен метод "print\_clients", который выводит список всех клиентов.

Описание класса "Trade":

- Класс "Trade" имеет конструктор init, в котором инициализируются пустые списки "self.loyalty" для хранения лояльности, "self.abonements" для хранения абонементов, "self.history" для хранения истории и "self.instructors" для хранения инструкторов. Также создается объект класса "Manager" и сохраняется в переменной "self.manager".

- В классе "Trade" определен метод "main\_menu", который содержит бесконечный цикл для отображения меню и обработки введенных пользователем операций.

- В методе "main\_menu" выводится список доступных операций. В зависимости от выбора операции, вызываются различные методы добавления информации в соответствующие списки.

- Метод "print\_info" выводит содержимое списков лояльности, абонементов, истории и инструкторов.

Модуль trade представляет собой класс Trade, который предоставляет функционал для управления информацией о клиентах, лояльности, абонементах, истории и инструкторах. При создании объекта класса Trade, инициализируются пустые списки clients, loyalty, abonements, history, instructors, которые будут использоваться для хранения соответствующих данных.

Рассмотрим функцию для взаимодействия менеджера с клиентом

Листинг 2. Функция взаимодействия:

class Manager:

    def \_\_init\_\_(self):

        self.clients = []

    def add\_client(self, name):

        self.clients.append(name)

        print("Клиент успешно добавлен.")

    def print\_clients(self):

        print("Список клиентов:")

        for client in self.clients:

            print(client)

    def interact\_with\_clients(self):

        while True:

            print("1. Взаимодействовать с клиентом")

            print("0. Вернуться в главное меню")

            choice = input("Введите номер операции: ")

            if choice == "1":

                self.interact\_with\_client()

            elif choice == "0":

                break

            else:

                print("Некорректный ввод. Попробуйте снова.")

    def interact\_with\_client(self):

        client\_name = input("Введите имя клиента: ")

        if client\_name in self.clients:

            print(f"Вы взаимодействуете с клиентом {client\_name}.")

            # здесь можно добавить логику взаимодействия с клиентом

        else:

            print("Клиент не найден.")

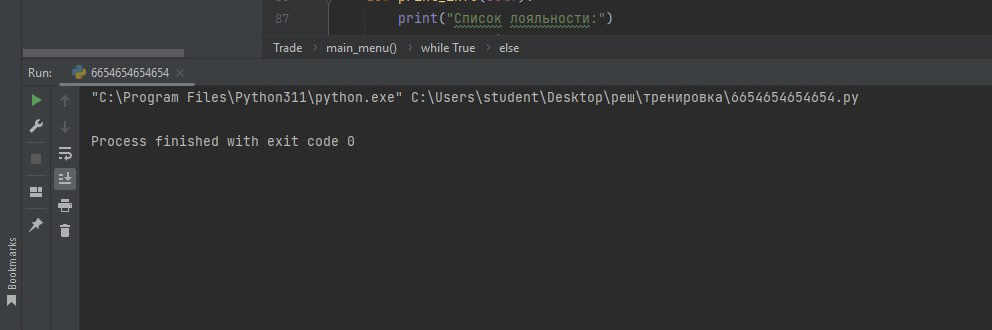
Изображение выглядит как диаграмма, текст, зарисовка, План

Автоматически созданное описание

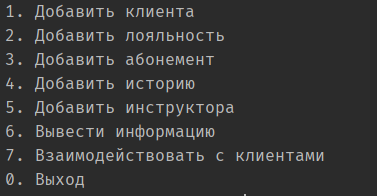
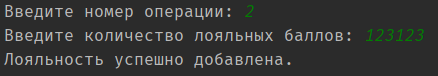
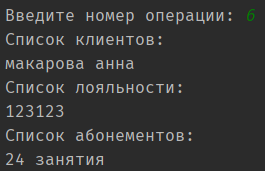
Рисунок 4. Блок-схема функции get\_manager\_info\_for\_client

## 2.3 Тестирование и отладка

В ходе написания проекта при итоговом тестирование выявилось данное сообщение (см. рис.1)

рисунок 1 - «итоговое тестирование»

При редактировании кода и исправления элементов, удалось получить должный результат (см. рис.2)

## 2.4 Дневник

|  |  |  |
| --- | --- | --- |
| Дата | Содержание работ | Отметка о выполнении |
| 22.09 | Техническое задание | 5- |
| 29.09 | Спецификация | 4/4/4 |
| 01.11 | Выполнение работ за месяц | 3- |
| 06.12 |  | 2 |
| 13.12 |  | 2 |

## Использованные источники

1. <https://habr.com/ru/amp/publications/709102/>
2. Stepik
3. Видео на ютубе о разделах, модулях и классах языка Python