

**КУРСОВОЙ ПРОЕКТ**

**по дисциплине: МДК 01.01 Разработка программных модулей**

**Тема: Разработка системы классов для приложения «Языковые курсы»**

**Специальность 09.02.07 «Информационные системы и программирование»**

|  |  |  |
| --- | --- | --- |
| Выполнила студент(ка) группы 309ИС-22 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Б.Б. Анаркулова |
| Руководитель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Л. Б. Гусятинер |

**Москва 2024**

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc185494614)

[1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 6](#_Toc185494615)

[1.1 Информационное обеспечение задачи 6](#_Toc185494616)

[1.2 Обзор и анализ существующих программных решений 8](#_Toc185494617)

[1.3 Постановка задачи. Структура входной и выходной информации 12](#_Toc185494618)

[2. ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА СИСТЕМЫ КЛАССОВ 15](#_Toc185494619)

[2.1 Построение модели системы 15](#_Toc185494620)

[2.2 Описание главного модуля 16](#_Toc185494621)

[2.3 Описание спецификаций к модулям 17](#_Toc185494622)

[2.4 Описание модулей 17](#_Toc185494623)

[2.5 Расчет сложности алгоритма 23](#_Toc185494624)

[3. ТЕСТИРОВАНИЕ И ОТЛАДКА СИСТЕМЫ КЛАССОВ 25](#_Toc185494625)

[3.1 Описание тестовых наборов модулей 25](#_Toc185494626)

[3.2 Описание применения средств отладки 25](#_Toc185494627)

[3.2.1 Тестирование класса Students 25](#_Toc185494628)

[3.2.2 Тестирование класса Teachers 26](#_Toc185494629)

[3.2.3 Тестирование класса Managers 27](#_Toc185494630)

[3.2.4 Тестирование основного модуля 28](#_Toc185494631)

[ЗАКЛЮЧЕНИЕ 29](#_Toc185494632)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 31](#_Toc185494633)

[РИЛОЖЕНИЕ 1 33](#_Toc185494634)

[ПРИЛОЖЕНИЕ 2 34](#_Toc185494635)

[ПРИЛОЖЕНИЕ 3 35](#_Toc185494636)

[ПРИЛОЖЕНИЕ 5 39](#_Toc185494637)

[ПРИЛОЖЕНИЕ 6 40](#_Toc185494638)

[ПРИЛОЖЕНИЕ 7 43](#_Toc185494639)

[ПРИЛОЖЕНИЕ 8 46](#_Toc185494640)

[ПРИЛОЖЕНИЕ 9 49](#_Toc185494641)

[ПРИЛОЖЕНИЕ 10 51](#_Toc185494642)

ВВЕДЕНИЕ

Данный проект представляет собой разработку системы классов для языковых курсов, иными словами платформу, упрощающую и автоматизирующую образовательный процесс для всех участников образовательной деятельности в информационной системе «Языковые курсы»: преподавателей, учеников, а также администраторов.

На данный момент изучение иностранных языков стало более актуальной темой среди всех групп возрастной категории, вместе с этим открываются разные частные языковые школы, предлагающие большой спектр выбора курсов по изучению языковых курсов. Однако управление учебной деятельностью может быть трудоемким при использовании обычных таблиц или бумажного учета.

Актуальность темы обусловлена необходимостью автоматизации учебной деятельности в языковых школах. Автоматизация позволит упростить процессы управления курсами, учета прогресса студентов и взаимодействия между участниками учебного процесса, что позволит повысить качество обучения.

Цель проекта – разработать систему классов для приложения «Языковые курсы», позволяющую автоматизировать управление учебной деятельностью.

Задачи проекта:

* Провести анализ предметной области
* Изучить существующие программные решения
* Определить основные функциональные требования к программе
* Спроектировать архитектуру приложения
* Разработать систему классов
* Выполнить тестирование, отладку разработанной программы

Объектом работы в данном проекте является учебная деятельность в языковых курсах / образовательные организации по изучению иностранных языков

Предмет работы – основные сущности и их взаимосвязи, реализуемые в системе:

* Пользователи (студенты, преподаватели, администраторы)
* Курсы и занятия

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Информационное обеспечение задачи

В данном курсовом проекте предметной областью для решения поставленных задач выступают образовательные организации по изучению иностранных языков.

Языковые курсы – это образовательные программы, направленные на изучение иностранных языков. Обычно они проводятся в небольших группах до 10–15 человек под руководством преподавателя. [20]

Основные понятия, связанные с предметной областью

* Ученик, студент, обучающийся – человек, изучающий язык
* Преподаватель – человек, ведущий обучение, обладающий необходимыми знаниями
* Курс – программа обучения
* Уровень – степень владения языком
* Группа – студенты, обучающиеся на одном курсе
* Занятие – отдельная сессия обучения
* Расписание – график занятий, указывающий время и день недели

Деятельность организаций, предоставляющих услуги по изучению иностранных языков, заключается в том , чтобы помочь студенту освоить новый язык или улучшить имеющиеся знания с использованием различных методов обучения.

Внутренняя структура организации включает в себя (рисунок 1.1):

1. Преподавательский состав: группа квалифицированных преподавателей.

2. Администрация: сотрудники, отвечающие за организацию учебного процесса (учет студентов и преподавателей, управление учебной деятельностью)



Рисунок 1.1 Внутренняя структура

Учебный процесс в рамках языковых курсов включает в себя:

* Тестирование для определения текущего уровня владения языком
* Формирование ученического профиля: выбор курса, оплата обучения, запись в группу
* Прохождение учебных занятий с преподавателем
* Управление расписанием: редактирование, создание расписания для каждой группы
* Управление студентами: регистрация, учет, контроль посещаемости, оценка знаний, выдача сертификатов
* Планирование, разработка программы обучения
* Управление преподавателями: подбор, контроль работы, оплата труда

Для лучшего понимания взаимодействия между участниками (студентами, преподавателями и администраторами) в рамках языковых курсов, ниже приведена диаграмма прецедентов (рисунок 1.2). Диаграмма показывает основные сценарии поведения участников.

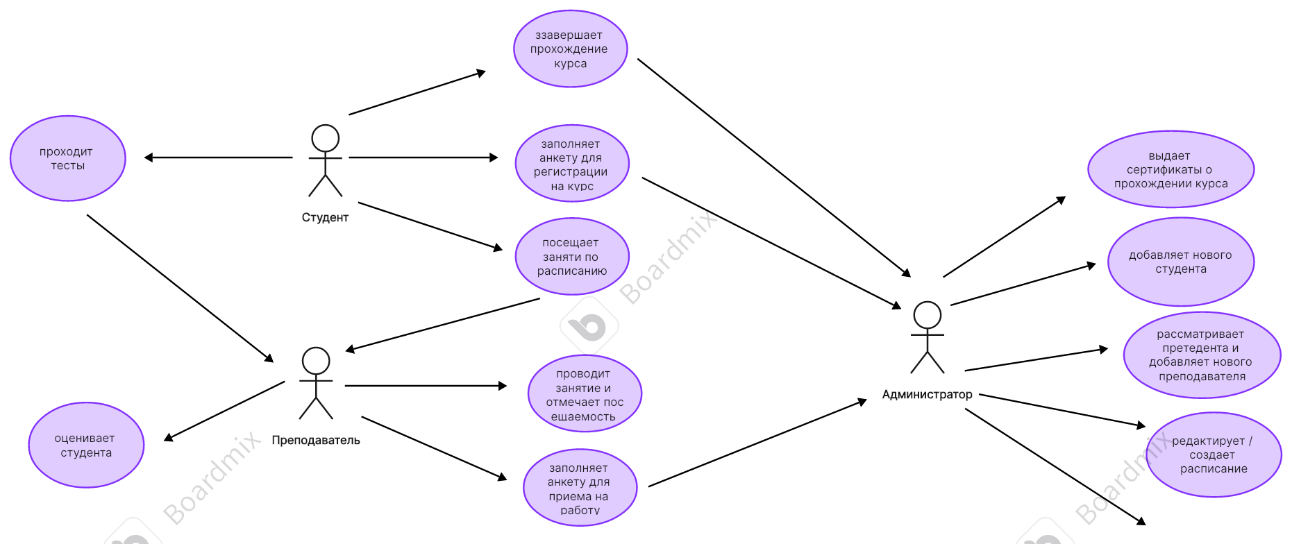


Рисунок 1.2 Диаграмма прецедентов

Таким образом, существующая проблема в данной предметной области заключается в недостаточной автоматизации процессов управления учебной деятельностью. Многие учреждения используют устаревшие методы (например: бумажные записи или таблицы), что может быть недостаточно эффективным и приводить к ошибкам, потере информации, в особенности при большом потоке студентов.

1.2 Обзор и анализ существующих программных решений

Для анализа существующих систем управления учебным процессом будут рассмотрены три системы:

1. «Мой класс» - CRM, позволяющая автоматизировать следующие процессы [14]:

* расписание занятий
* уроки, выдачу заданий
* отслеживание посещаемости и успеваемости
* абонементы и оплаты от учеников
* расчёт зарплаты педагогам
* задачи для сотрудников, напоминания
* покупку товаров для школы, склад
* работу с данными клиентов
* оповещения учеников об оплате, занятиях, пропусках. [14]

Функциональные возможности программы, которые можно реализовать в разрабатываемой программе:

* Регистрация и учет студентов: возможность регистрировать новых учеников (рисунок 1.3), просматривать данные студента

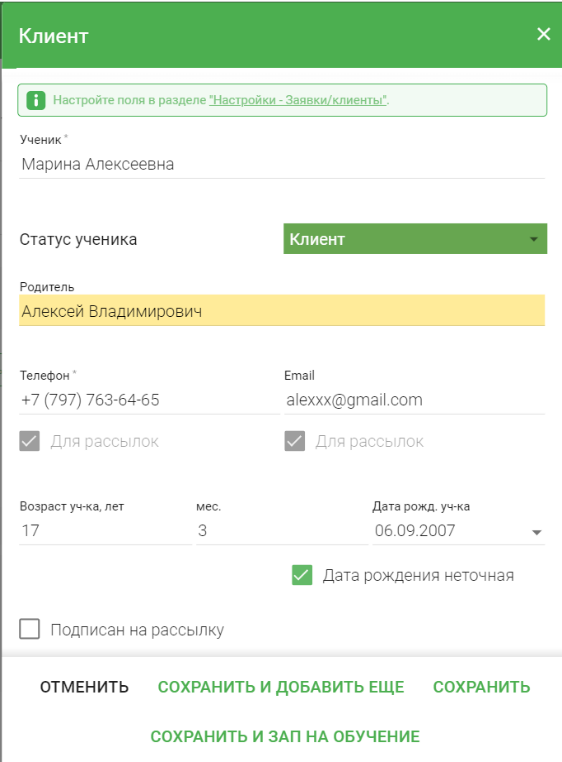


Рисунок 1.3 Окно регистрации нового студента

* Группы и распределение: студентов можно записать на курс, определив группу (рисунок 1.4)

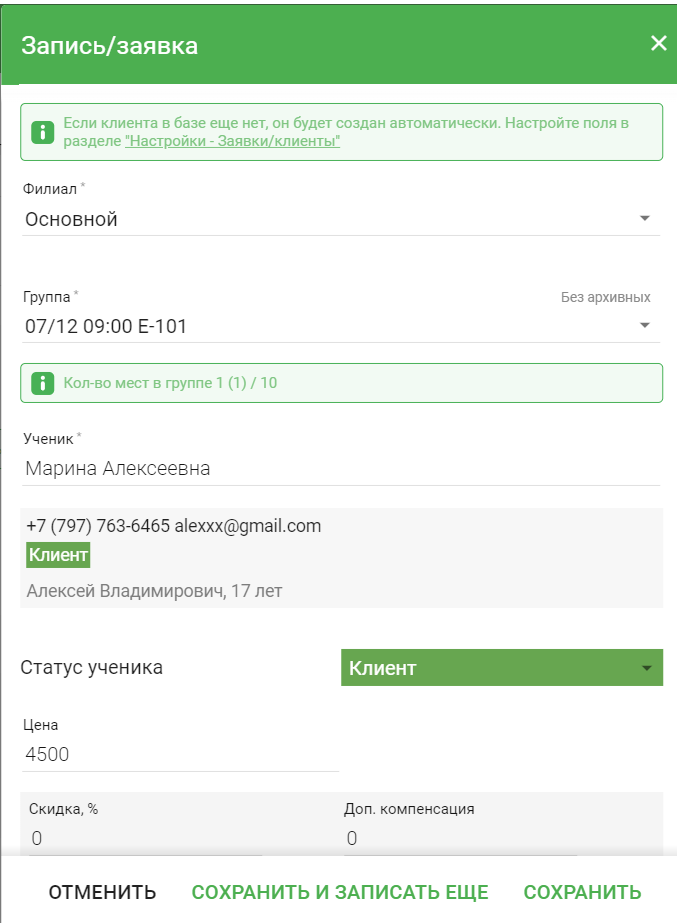


Рисунок 1.4 Запись студента на курс

* Контроль посещаемости и выставление оценок: возможность отметить посещаемость занятий и выставить оценку за занятие
* Расписание занятий: создание и управление расписанием занятий для каждой группы (рисунок 1.6)

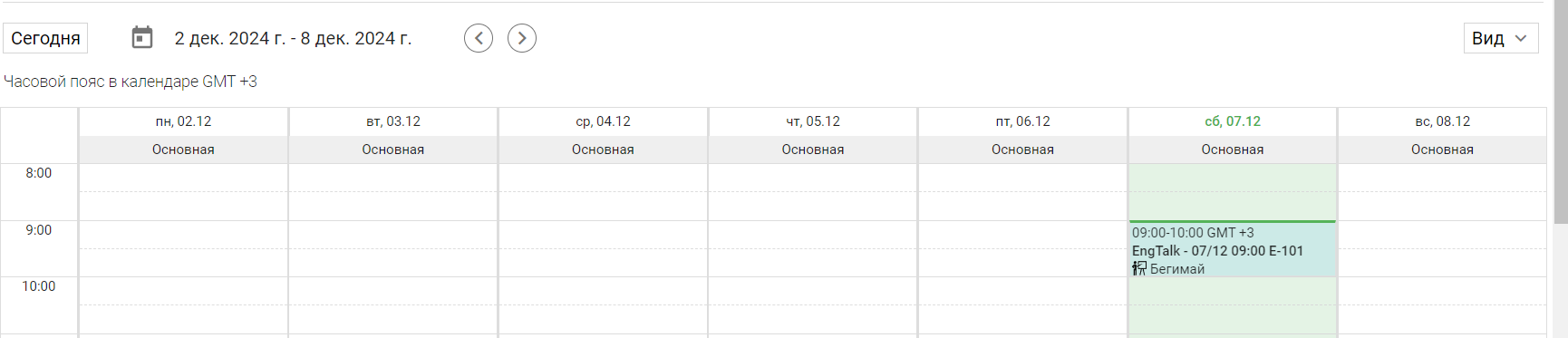


Рисунок 1.6 Просмотр созданного расписания занятий

1. Big Ben CRM — это система управления для школ иностранных языков и преподавателей, которая автоматизирует все процессы в учебном центре. [15]

Функциональные возможности программы, которые можно реализовать в разрабатываемой программе:

* Учет посещаемости студентов (рисунок 1.7)

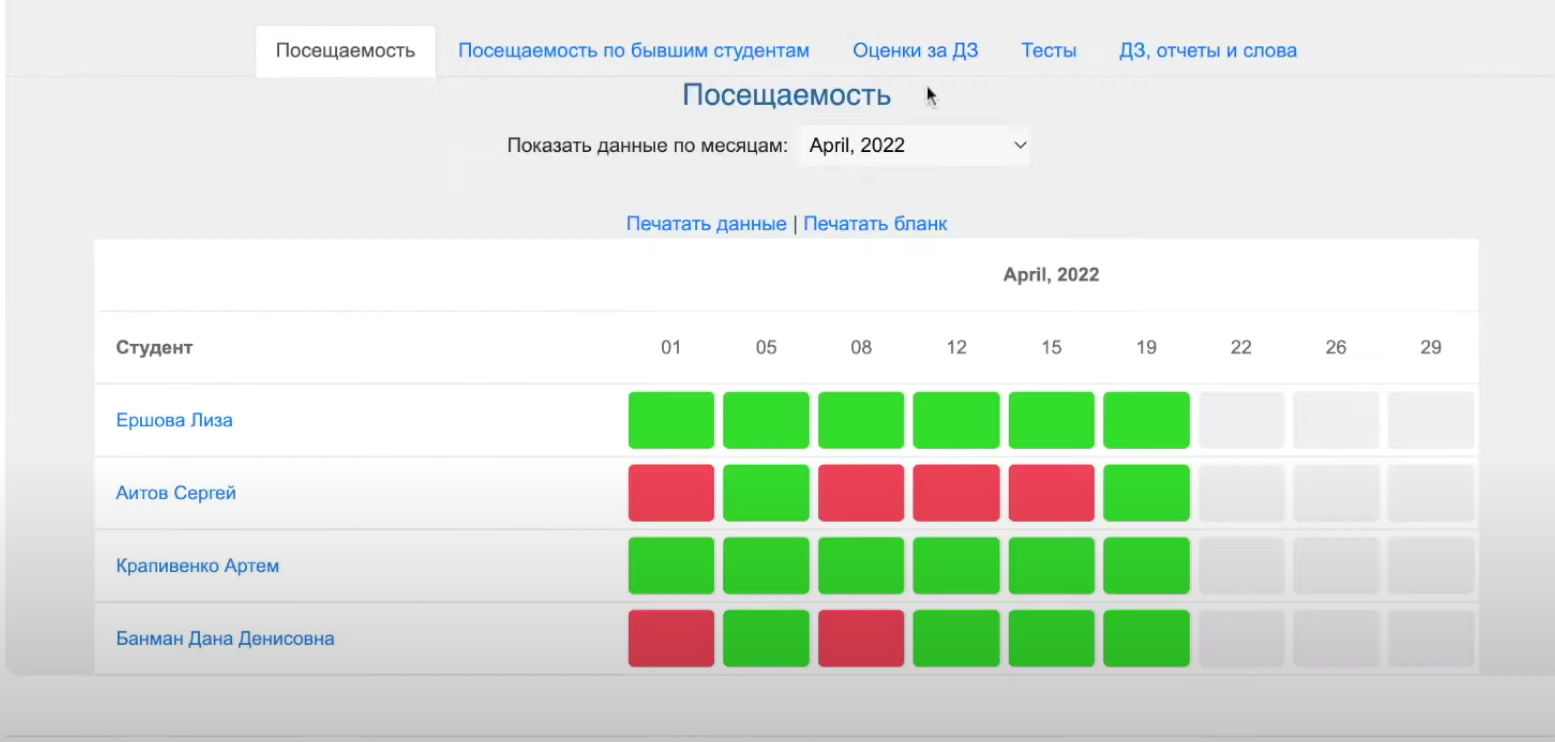


Рисунок 1.7 Окно просмотра посещаемости

* Возможность проводить онлайн тесты и анализировать результаты студентов (рисунок 1.8)

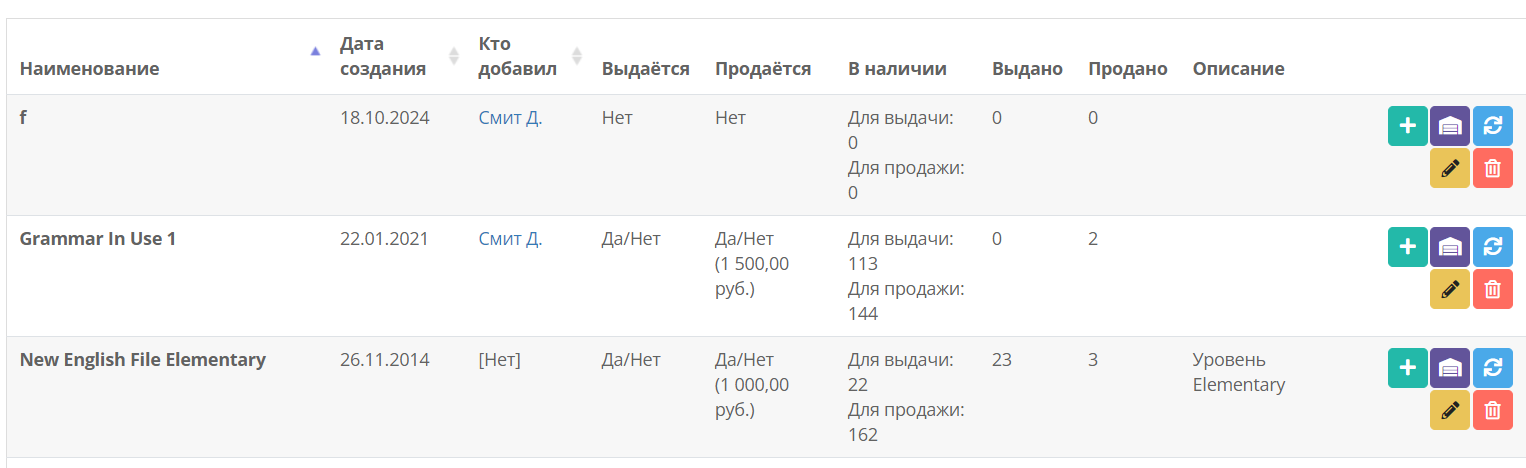


Рисунок 1.8 Окно просмотра результатов студентов по тесту

3. Hollihop - специализированная CRM-система для профессионального управления учебным центром любого типа. [16]

Функциональные возможности программы, которые можно реализовать в разрабатываемой программе:

* Наличие онлайн-библиотеки - хранение материалов (рис. 1.11)

Рисунок 1.11 Окно просмотра доступных книг

* Возможность создавать отчеты и статистику (рис. 1.12)

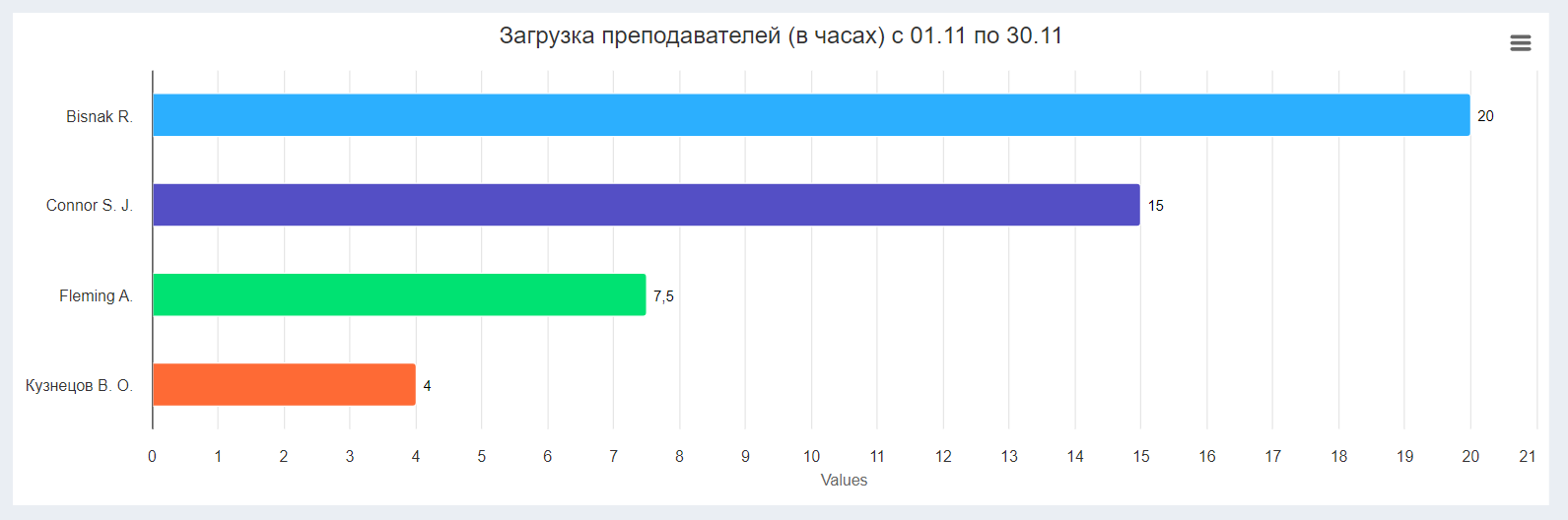


Рисунок 1.12 Пример таблицы о нагрузке преподавателей

* Размещение домашних заданий (рис. 1.13)

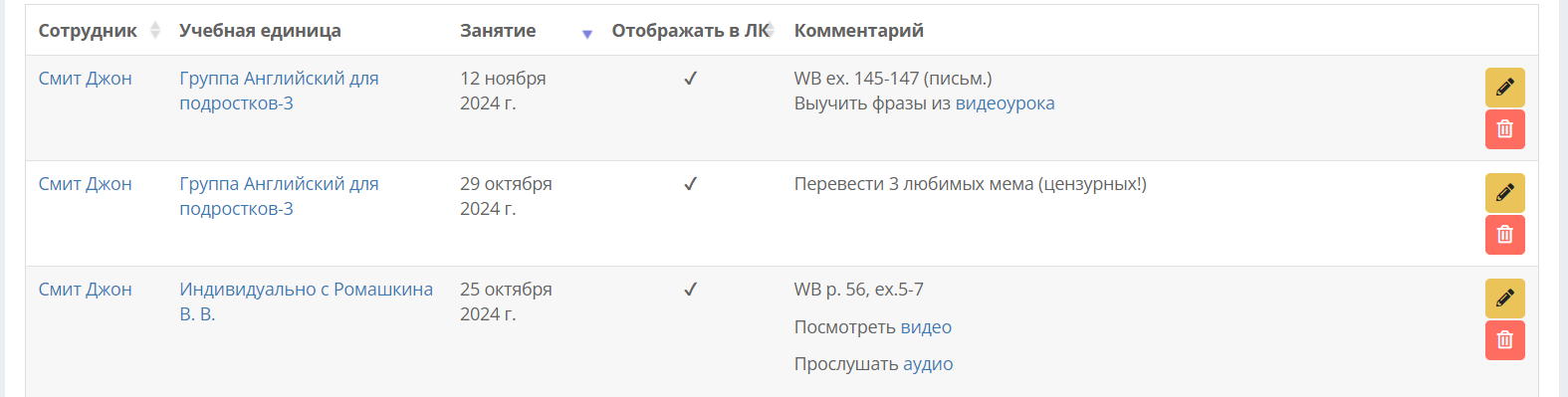


Рисунок 1.13 Окно просмотра прикрепленных заданий

1.3 Постановка задачи. Структура входной и выходной информации

Разрабатываемое программное обеспечение (ПО) должно обеспечить и выполнять следующие функции:

* Управление студентами и преподавателями – регистрация, хранение личной информации, возможность изменять данные
* Автоматизированное создание сертификатов об окончании курсов
* Управление расписанием – возможность создавать, редактировать, просматривать расписание
* Авторизация в систему с учетом роли пользователя (студент, администратор, преподаватель) с различными правами доступа

Организация входных и выходных данных:

1. Вход в систему

* Входные данные: логин и пароль
* Выходные данные: личный кабинет пользователя

2. Печать сертификата об окончании обучения:

* Входные данные: данные о студенте (какой курс окончил, уровень)
* Выходные данные: документ – сертификат, подтверждающий завершение обучения

3. Вывод информации по группе, студенту или преподавателе

* Входные данные: номер группы, фамилия имя студента или преподавателя
* Выходные данные: вывод информации по введенным данным

Выбор модели жизненного цикла

Для разрабатываемого программного обеспечения (ПО) для автоматизации управления учебной деятельности на языковых курс подходит итеративная модель жизненного цикла, которая позволяет разбить процесс разработки на повторяющиеся циклы, в каждом из которых продукт улучшается, в зависимости от отзывов заказчика.

Итеративная модель жизненного цикла имеет ряд преимуществ:

* Гибкость в изменениях. Заказчики могут вносить изменения на каждом этапе разработки
* Возможность получать обратный отзыв от заказчика на каждом этапе
* Выявление проблем и недостатков системы на раннем этапе, что позволяет избежать ошибок
* Постепенное улучшение и добавление новых функциональных возможностей.

Данную модель рекомендуется использовать в следующих случаях:

* Возможны изменения в требовании к ПО
* Продукт создается с целью на постоянное улучшение и добавление новых функций
* Необходимо активное участие пользователей во время процесса разработки

Таким образом, итеративная модель разработки программного продукта соответствует жизненному циклу разрабатываемой программы.

В ходе анализа систем были протестированы и рассмотрены существующие программные решения для выявления функциональных особенностей, которые можно реализовать в разрабатываемой программе. Также были сформулированы требования к системе и определена модель жизненного цикла для проекта.

2. ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА СИСТЕМЫ КЛАССОВ

2.1 Построение модели системы

Рассмотрим проектирование диаграммы классов для языковых курсов. Система предназначена для работы с тремя основными ролями: студентами, преподавателями и менеджерами. Каждая роль имеет свои функции, которые реализуются через классы и методы.

Для начала необходимо определить основные сущности, которые будут использоваться в системе. В рамках проекта выделяются следующие сущности:

1. Студенты:

* Хранят информацию о студентах, включая логин, группу, ФИО, контактные данные и дату рождения.
* Имеют доступ к просмотру расписания.

1. Преподаватели:

* Хранят информацию о преподавателях, включая логин, ФИО, контактные данные и список групп, в которых они преподают.
* Имеют доступ к просмотру расписания.

1. Группы:

* Хранят информацию о группах, включая ID группы, преподавателя, уровень и курс.
* Связаны со студентами и преподавателями.

1. Расписание:

* Хранят информацию о занятиях, включая группу, день недели, время начала, курс и преподавателя.

1. Менеджеры:

* Управляют всей системой, включая регистрацию пользователей, редактирование расписания и вывод сертификатов.

Во избежание использования повторных атрибутов в классах, представляющих студентов и преподавателей необходимо использовать родительский класс, который будет иметь такие атрибуты, как: логин, фамилия, имя, отчество, логин, телефон и дата рождения. Классы Students и Teachers, следовательно, наследуются от класса Users.

Для визуализации структуры системы была спроектирована диаграмма классов. Основные классы и их связи представлены на диаграмме (приложение 1)

Далее необходимо спроектировать сценарий работы системы для моделирования взаимодействия пользователя с системой. Для примера на рисунке 2.2 изображена схема взаимодействия пользователя (в данном случае только преподавателя или студента) с системой.

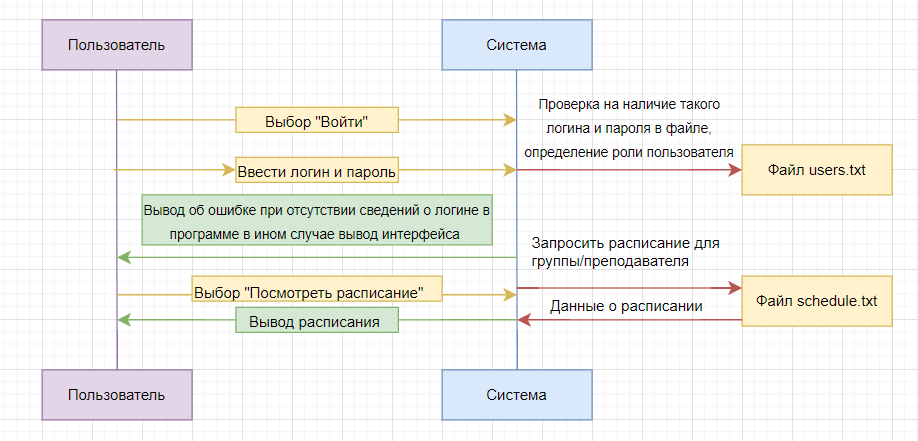


Рисунок 2.2 Схема взаимодействия преподавателя/студента с системой

В рамках построения модели также был спроектирован примерный дизайн программы, включающий основные экраны для взаимодействия с системой. Примеры окон представлены в приложении 9.

2.2 Описание главного модуля

В модуль main импортируются все входящие модули и реализуется авторизация и регистрация.

Для авторизации создана функция log\_in() при которой открывается файл «users.txt», содержащий логин, пароль и роль пользователя (менеджер, преподаватель, студент). Введенные логин и пароль сверяются с данными из файла и в зависимости от роли пользователя, вызывается функция: для студента – student\_interface, для преподавателя – teacher\_interface, для менеджера – manager\_interface.

Для регистрации создана функция sign\_up(), которая позволяет пользователю логин, который создается менеджером при регистрации пользователя и выдается ему заранее. При регистрации в поле пароля менеджер вводит знак «\_». При наличии логина и знака нижнего подчеркивания, означающего отсутствие пароля, пользователю дается возможность придумать пароль, который автоматически заносится в файл. При наличии пароля в файле выходит сообщение о том, что пользователь уже зарегистрирован, в ином случае, что такого пользователя нет в системе.

Листинг функций главного модуля приведены в приложении 4.

2.3 Описание спецификаций к модулям

В качестве хранения данных будут использоваться текстовые файлы в формате CSV (значения, разделенные запятыми). Необходимо создать пять файлов: users.txt – хранит данные о пользователей для авторизации и регистрации (логин, пароль, роль), students.txt, teachers.txt и groups.txt - хранят данные о студентах, преподавателей и группах соответственно, schedule.txt – хранит весь график занятий для групп.

Для каждого класса создается файл (модуль), содержащий все его методы. Каждый класс реализует методы для работы с соответствующими файлами, включая чтение, запись и обработку данных.

2.4 Описание модулей

Модуль students.py расширяет базовый класс User и добавляет атрибут group (номер группы). Он реализует методы для работы с расписанием и интерфейсом студента. Пояснения к методам класса Students приведены в таблице 2.3. Код функций в приложении 2.

Таблица 2.3 Пояснения к методам класса Students

|  |  |
| --- | --- |
| Класс Students | |
| Метод класса(функции) | Пояснение |
| init () | Инициализация атрибутов студента: user\_id, group, surname, name, last\_name, e\_mail, phone, birth\_date |
| str () | Возвращает строковое представление данных студента |
| get\_student\_group() | Возвращает группу студента по его логину из файла students.txt |
| get\_schedule\_for\_student() | Выводит расписание для группы (номер группы определяется функцией get\_student\_group()) |
| student\_interface() | Интерфейс для студента: позволяет просматривать расписание или выйти |

Модуль schedule.py содержит методы для вывода расписания в различных форматах. Пояснения к методам класса Schedule приведены в таблице 2.1

Таблица 2.1 Методы класса Schedule

|  |  |
| --- | --- |
| Метод класса(функции) | Пояснение |
| Класс Schedule | |
| init () | group, day\_of\_week, start\_time, course, teacher |
| str () | Возвращает строковое представление данных группы |
| without\_group() | Возвращает строковое представление расписания без указания группы. Используется для вывода расписания для студентов |
| without\_teacher() | Возвращает строковое представление расписания без указания преподавателя. Используется для вывода расписания для преподавателя |

Модуль teachers.py реализует методы для работы с расписанием и интерфейсом преподавателя. Пояснения к методам класса Teachers приведены в таблицe 2.4. Код функций в приложении 3.

Таблица 2.4 Пояснения к методам класса Teachers

|  |  |
| --- | --- |
| Класс Teachers | |
| Метод класса(функции) | Пояснение |
| init () | Инициализация атрибутов преподавателя: teacher\_id, surname, name, last\_name, e\_mail, phone |
| str () | Возвращает строковое представление данных преподавателя |
| get\_teacher\_surname() | Возвращает фамилию преподавателя по его логину из файла teachers.txt |
| get\_schedule\_for\_teacher() | Выводит расписание для преподавателя по его фамилии (используется предыдущая функция) |
| teacher\_interface() | Интерфейс для преподавателя: позволяет просматривать расписание или выйти |

Модуль groups.py содержит методы для работы с информацией о группах. Структура класса приведена в таблице 2.1

Таблица 2.1 Методы класса groups.py

|  |  |
| --- | --- |
| Метод класса(функции) | Пояснение |
| Класс Groups | |
| init () | Инициализация атрибутов группы: group\_id, teacher, level, course |
| str () | Возвращает строковое представление данных группы |

Модуль user.py является базовым классом для всех пользователей системы. Он содержит общую информацию о пользователе, такую как логин, ФИО, email и телефон. Методы класса Users приведены в таблице 2.2.

Таблица 2.2 Методы класса Users

|  |  |
| --- | --- |
| Метод класса(функции) | Пояснение |
| Класс Users | |
| init () | Инициализация атрибутов пользователя: user\_id, surname, name, last\_name, e\_mail, birth\_date |
| str () | Возвращает строковое представление данных о пользователе |

Модуль managers.py отвечает за административные задачи, такие как регистрация пользователей, редактирование расписания, выдача сертификатов и управление группами. Методы класса Managers (таблица 2.4). Код функций данного модуля приведен в приложении 4.

Таблица 2.4 Пояснения к методам класса Managers

|  |  |  |
| --- | --- | --- |
| Класс Managers | | |
| Метод класса(функции) | | Пояснение |
| print\_file() | Открывает текстовый документ с анкетой для заполнения новыми студентами/преподавателями при подборе персонала | |
| register\_student() | Регистрирует нового студента и сохраняет данные в файл students.txt. Также добавляет нового пользователя в файл users.txt с указанием логина и роли | |
| register\_teacher() | Регистрирует нового преподавателя и сохраняет данные в файл teachers.txt. Также добавляет нового пользователя в файл users.txt с указанием логина и роли | |
| get\_schedule\_for\_group() | Вывод расписания для определенной группы | |
| get\_info\_student() | Вывод данных о студенте по его ФИ | |
| get\_info\_teacher() | Вывод информации о преподавателе по его ФИ | |
| get\_info() | Интерфейс для вывода информации с выбором (о преподавателе/студенте) | |
| print\_certificate() | Заносит данные в шаблонный документ docx с указанием ФИО, курса, языка, даты для создания сертификата об окончании обучения | |
| new\_lesson() | Добавление нового занятия в расписание | |
| delete\_lesson() | Удаление занятия из расписания | |
| edit\_lesson() | Редактирует занятие по определённому элементу(время, группа, преподаватель, день недели) | |
| edit\_schedule() | Интерфейс для редактирования расписания. Вызывает соответствующую функцию в соответствии с выбором пользователя | |
| groups\_info() | Выводит информации о группе | |
| add\_group() | Добавление новой группы в файл groups.txt | |

Для создания документа docx по шаблону импортируется класс DocxTemplate модуля docxtpl. Функция print\_certificate() является методом класса Managers и позволяет создать документ .docx для печати сертификата студенту, окончившему курс. Данная функция принимает шесть параметров, вводимых с клавиатуры пользователем:

* student\_name – ФИО студента
* language – язык, который был изучен
* level – уровень владения языком по окончанию курса
* course – какой курс был пройден
* organization – название организации
* date – дата выдачи сертификата

Далее введенные данные заполняются в словарь, где ключи словаря соответствуют вводимым переменным, которые будут заменены в шаблоне. Затем создаётся новый документ с именем и фамилией студента. Шаблон сертификата приведен в приложении 3.

Листинг функции print\_certificate():

from docxtpl import DocxTemplate

def print\_certificate(self, student\_name, language, level, course, organization, date):

try:

doc = DocxTemplate("certificate\_templ.docx")

context = {

'student\_name': student\_name,

'language': language,

'level': level,

'course': course,

'organization': organization,

'date': date}

doc.render(context)

file = f"certificate\_{student\_name.replace(' ', '\_')}.docx"

doc.save(file)

except Exception as e:

print(f"Ошибка {e}")

Для выгрузки данных в excel импортируется класс Workbook() модуля openpxyl. Метод students\_to\_excel()класса Managers позволяет выгрузить актуальные данные о всех студентах на текущую дату в программу Microsoft Excel. Данная функция считывает все данные из текстового файла students.txt, создает объект класса Workbook и сохраняет эти данные в документе «new\_excel.xlsx».

Листинг функции students\_to\_excel():

from openpxyl import Workbook

from openpyxl.styles import Font, Alignment, Border, Side

from datetime import date

def students\_to\_excel(self):

try:

with open('students.txt', 'r', encoding='utf-8') as file:

lines = file.readlines()

wb = Workbook()

ws = wb.active

today = date.today()

ws['A1'] = f"Количество студентов на {today}"

ws['A1'].font = Font(bold=True, size=14)

ws['A1'].alignment = Alignment(horizontal='center')

ws.merge\_cells('A1:H1')

headers = ["Логин", "Группа", "Фамилия", "Имя", "Отчество", "E-mail", "Телефон", "Дата рождения"]

ws.append(headers)

for line in lines:

st = line.strip().split(',')

ws.append(st)

wb.save('new\_excel.xlsx')

print(f"Данные экспортированы в файл new\_excel.xlsx")

os.startfile('new\_excel.xlsx')

except Exception as e:

print(f"Ошибка: {e}")

2.5 Расчет сложности алгоритма

Сложность алгоритма – это показатель, характеризующий зависимость времени выполнения алгоритма от объема входных данных. [21]

Конструкторы классов (\_\_init\_\_) выполняют простые операции присваивания, следовательно выполняется константное количество операций – сложность алгоритма O(1)

Методы \_\_str\_\_ возвращают строковое представление объектов и также имеют сложность O(1), т. к. строка формируется из уже известных атрибутов.

Расчет сложности алгоритма для классов Students и Teachers:

get\_student\_group() – проходит по всем строкам в файле students.txt и проводит поиск логина в файле, в случае, если логин не будет найден, будет прочитано n строк – сложность алгоритма O(n) (сложность зависит от количества строк в файле)

В функциях get\_schedule\_for\_student(), get\_teacher\_name(), get\_schedule\_for\_teacher() также проводится линейный поиск в файле, сложность O(n).

Расчет сложности алгоритма для класса Managers:

Методы register\_student, register\_teacher, new\_lesson, add\_group имеют сложность O(1) т. к. принимают определенное количество данных и выполняют одну операцию – запись данных в файлы.

Методы get\_schedule\_for\_group, get\_info\_student, get\_info\_teacher, delete\_lesson, edit\_lesson, groups\_info выполняют линейный поиск в файлах – сложность алгоритма – O(n)

В функциях главного модуля также проводится линейный поиск в файлах, соответственно, время выполнения зависит от количества данных в файлах (O(n)).

Таким образом, в ходе проектирования и разработки системы была реализована система классов. Были созданы файлы для хранения данных и методы работы с файлами: чтение, редактирование, запись данных.

3. ТЕСТИРОВАНИЕ И ОТЛАДКА СИСТЕМЫ КЛАССОВ

3.1 Описание тестовых наборов модулей

Для тестирования программы необходимо создать план тестирования с указанием сценариев. Таблица с планом тестирования приведена в приложении 6. Данный план охватывает тестирование всех функциональных возможностей разработанной программы.

3.2 Описание применения средств отладки

Для тестирования разработанной программы будет использоваться ручное тестирование программы. В файлы users, students, teachers, schedule, groups были внесены тестовые данные (рисунок 3.1)

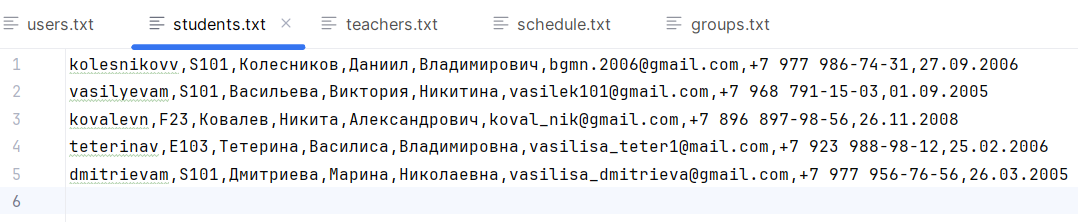


Рисунок 3.1 Содержание файла students.txt (тестовые данные)

3.2.1 Тестирование класса Students

Для начала создается экземпляр класса, затем тестируются методы данного класса (рис. 3.2)

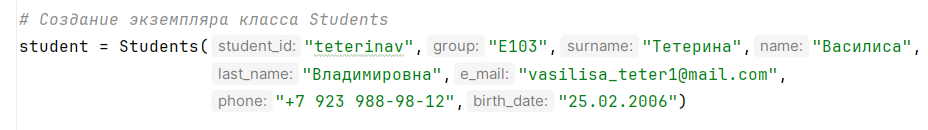


Рисунок 3.2 Создание экземпляра класса

Далее необходимо протестировать методы класса и сравнить результаты (рис. 3.3, рис. 3.4)

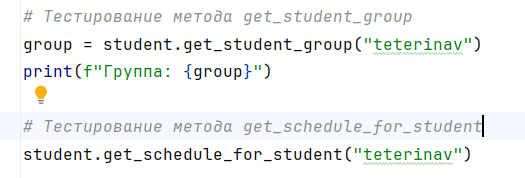


Рисунок 3.3 Тестирование методов класса

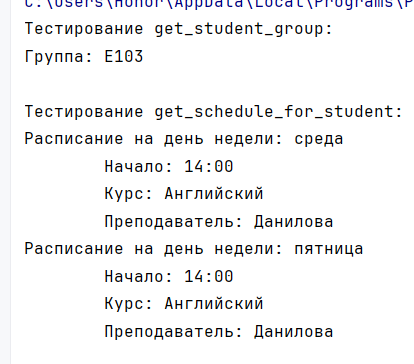


Рисунок 3.4 Результаты тестирования

3.2.2 Тестирование класса Teachers

Для начала создается экземпляр класса, затем тестируются методы данного класса (рис. 3.5)

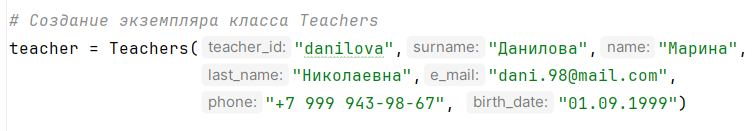


Рисунок 3.5 Создание экземпляра Teachers

Далее необходимо протестировать методы класса и сравнить результаты (рис. 3.6, рис. 3.7)

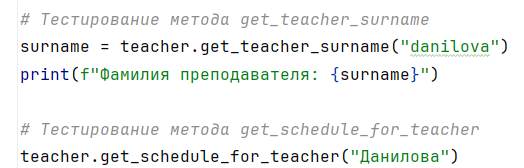


Рисунок 3.6 Тестирование методов класса

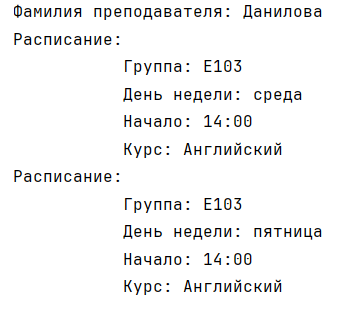


Рисунок 3.7 Результаты тестирования

3.2.3 Тестирование класса Managers

Для начала необходимо создать экземпляр класса, затем протестировать методы данного класса. (рис. 3.8)

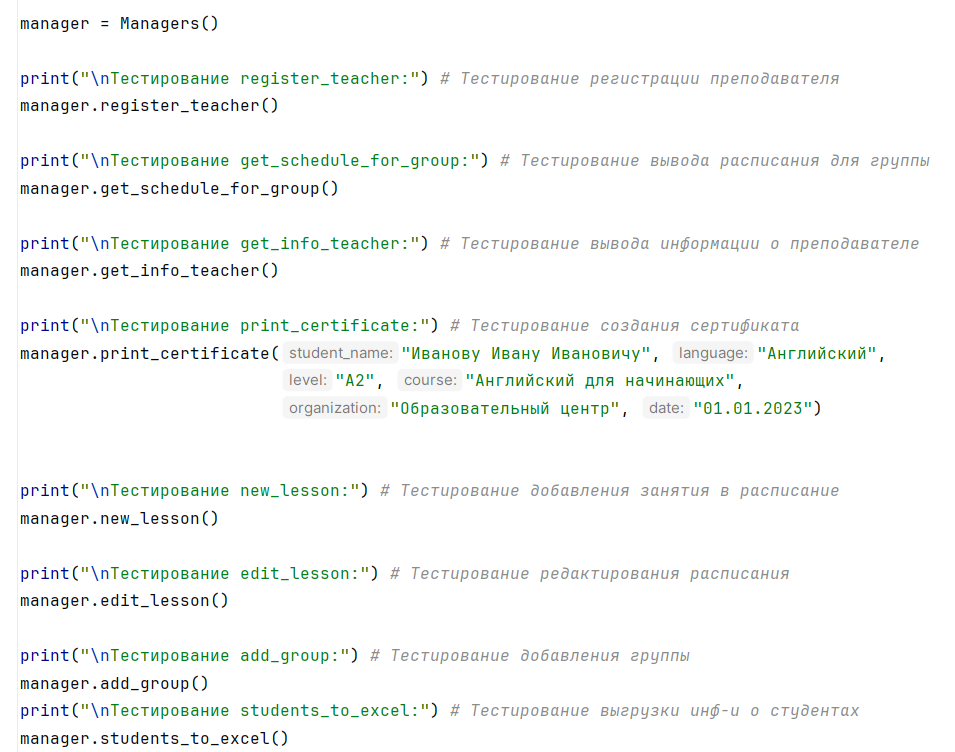


Рисунок 3.8 Тестирование основных методов класса Managers

Результаты тестирования приведены в приложении 7

3.2.4 Тестирование основного модуля

Результаты тестирования модуля main приведены в приложении 8

В ходе тестирования были проверены функции таких модулей программы как: students.py, teachers.py, managers.py, а также функции основного модуля программы. Были выявлены и исправлены ошибки, выявленные в процессе тестирования. Результаты тестирования показали, что система работает в соответствии с требованиями и выполняет свои функции.

ЗАКЛЮЧЕНИЕ

В ходе разработки программы был проведен анализ предметной области, разработка программы с реализацией системы классов и тестирование, согласно поставленной цели и задачам.

Данный программный продукт имеет следующие функции:

* Регистрация студентов и преподавателей
* Хранение данных о пользователях
* Печать сертификата об окончания курса для студентов
* Экспорт списка студентов в excel
* Управление группами и расписанием (создание, удаление)

Разработанная программа предназначена для образовательных организаций для автоматизации основных процессов в управлении учебной деятельностью, что позволит повысить эффективность деятельности на языковых курсах, снизить нагрузку для сотрудников, вести учет студентов в более удобном формате.

Данные проект также имеет перспективы на дальнейшее улучшение. Рассматривается добавление таких функций как:

* Возможность выставлять баллы студентам за их успеваемость и вести их учет
* Отслеживать посещение студентов
* Возможность выполнять интерактивные онлайн-тесты в программе для подкрепления изученной темы, результаты прохождения которых будут доступны преподавателю
* Использование базы данных для безопасного и надежного хранения данных
* Использование графического интерфейса для повышения удобства использования программой

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Стандарты и законодательные материалы

1. ГОСТ 7.32-2017 Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления.

2. ГОСТ 7.0.100-2018 Система стандартов по информации, библиотечному и издательскому делу. Библиографическая запись. Библиографическое описание. Общие требования и правила составления.

3. ГОСТ 7.9-95 Система стандартов по информации, библиотечному и издательскому делу. Реферат и аннотация. Общие требования.

4. ГОСТ 7.11-2004 (ИСО 832:1994) Система стандартов по информации, библиотечному и издательскому делу. Библиографическая запись. Сокращение слов и словосочетаний на иностранных европейских языках.

5. ГОСТ Р 7.0.12-2011 Национальный стандарт Российской Федерации. Система стандартов по информации, библиотечному и издательскому делу. Библиографическая запись сокращение слов и словосочетаний на русском языке. Общие требования и правила.

6. ГОСТ 7.79-2000 (ИСО 9-95) Система стандартов по информации, библиотечному и издательскому делу. Правила транслитерации кирилловского письма латинским алфавитом.

7. ГОСТ 7.80-2000 Система стандартов по информации, библиотечному и издательскому делу. Библиографическая запись. Заголовок. Общие требования и правила составления.

8. ГОСТ Р ИСО/МЭК 12207 Информационная технология. Процессы жизненного цикла программных средств.

9. Федеральный закон от 27 июля 2006 года № 149-ФЗ "Об информации, информационных технологиях и о защите информации".

10. Федеральный закон от 27 июля 2006 года № 152-ФЗ "О персональных данных".

11. Федеральный закон от 29 июля 2017 года № 187-ФЗ "О безопасности критической Учебники и учебные пособия:

12. Объектно-ориентированный Python, 4-е изд.: учебник / Стивен Лотт, Дасти Филлипс, Санкт-Петербург: «Питер», 2024

Интернет-источники:

13. Обучающий курс на stepik «Добрый, добрый Python с Сергеем Балакиревым»: <https://stepik.org/course/100707/syllabus>

14. CRM «Мой класс» - Режим доступа: <https://moyklass.com/crm>

15. CRM «BigBen» - Режим доступа: [https://bigbencrm.ru/](https://bigbencrm.ru/%20)

16. HOLLIHOP - Режим доступа: <https://holyhope.ru/>

17. <https://pythonist.ru/avtomatizacziya-otchetov-v-excel-s-pomoshhyu-python/>

18. <https://docs-python.ru/packages/modul-openpyxl/>

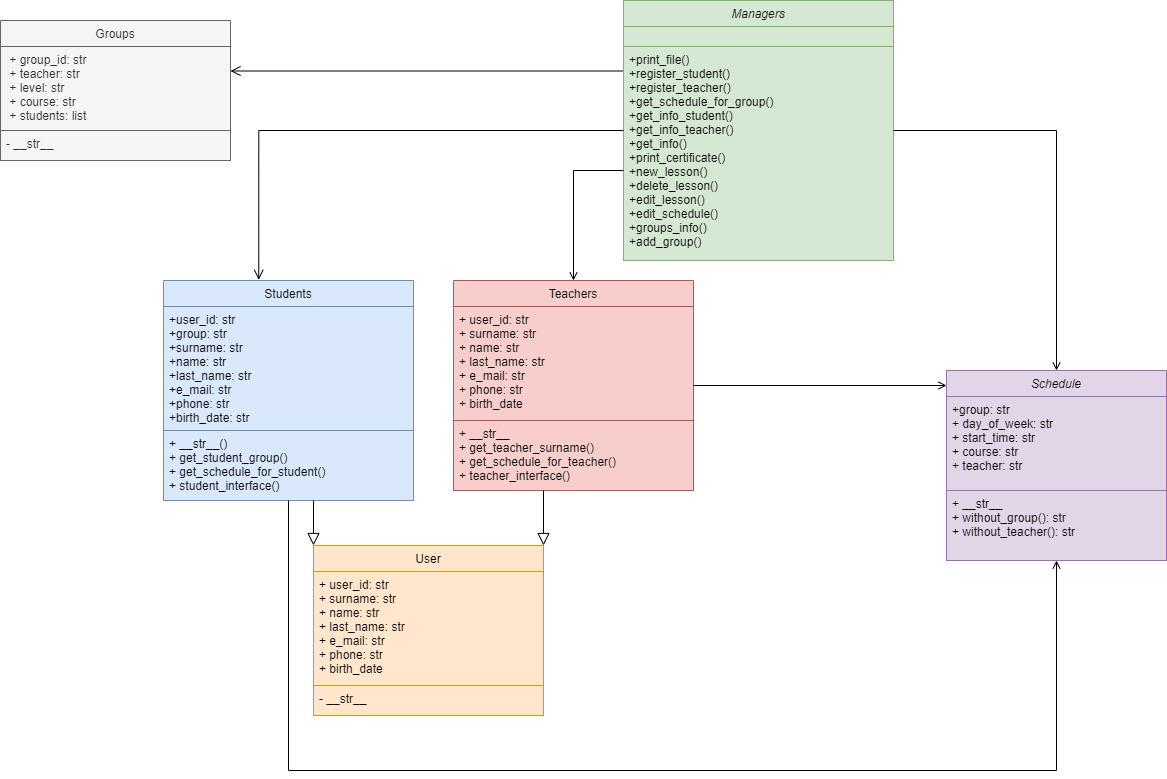
19. «Языковые школы: востребованность и актуальность обучения», авторы: Лазаревич Светлана Валерьевна, Дюдякова Светлана Владимировна

20. Как за две недели продвинуться в изучении иностранного языка. Зачем нужны языковые курсы: StudyInFocus, 2022 г.: <https://studyinfocus.ru/yazykovye-kursy/>

21. https://gitverse.ru/blog/articles/development/325-slozhnost-algoritmov-glubokij-razbor-olog-

РИЛОЖЕНИЕ 1

**Диаграмма классов**

****

ПРИЛОЖЕНИЕ 2

**Основные методы класса Students**

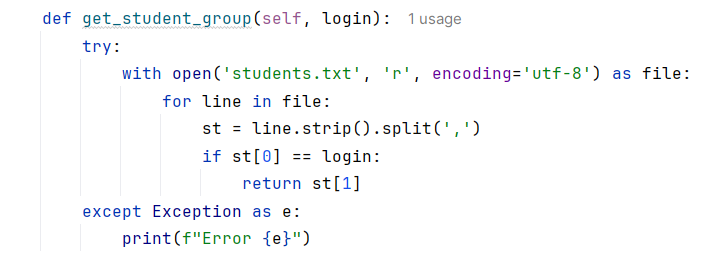


Рисунок 1 Функция для определения группы студента

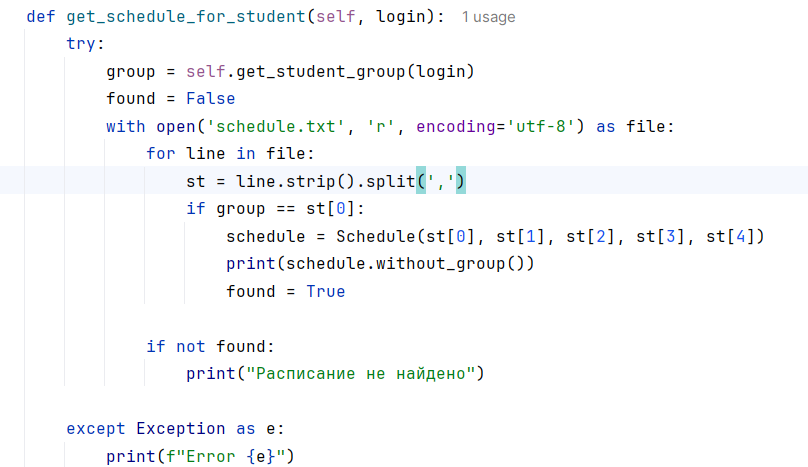


Рисунок 2. Функция для вывода расписания

ПРИЛОЖЕНИЕ 3

**Основные методы класса Teachers**

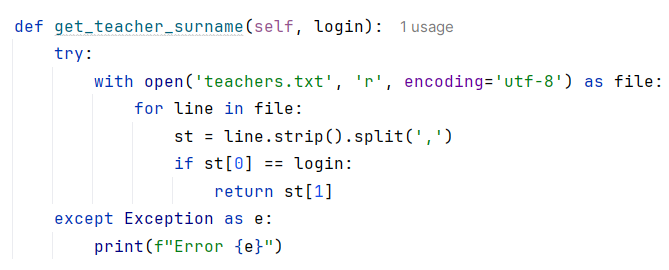


Рисунок 3. Функция для определения фамилии преподавателя

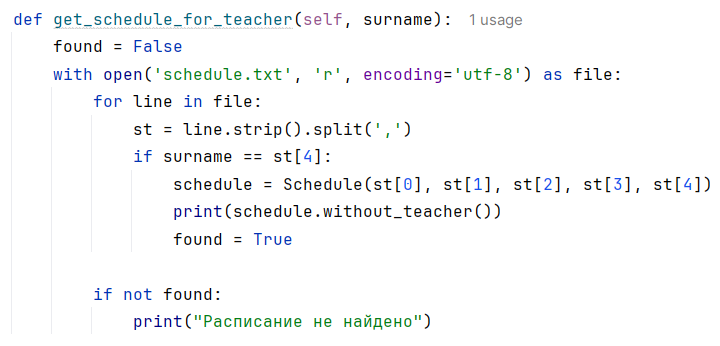


Рисунок 4. Функция для вывода расписания

**ПРИЛОЖЕНИЕ 4**

**Основные методы класса Managers**

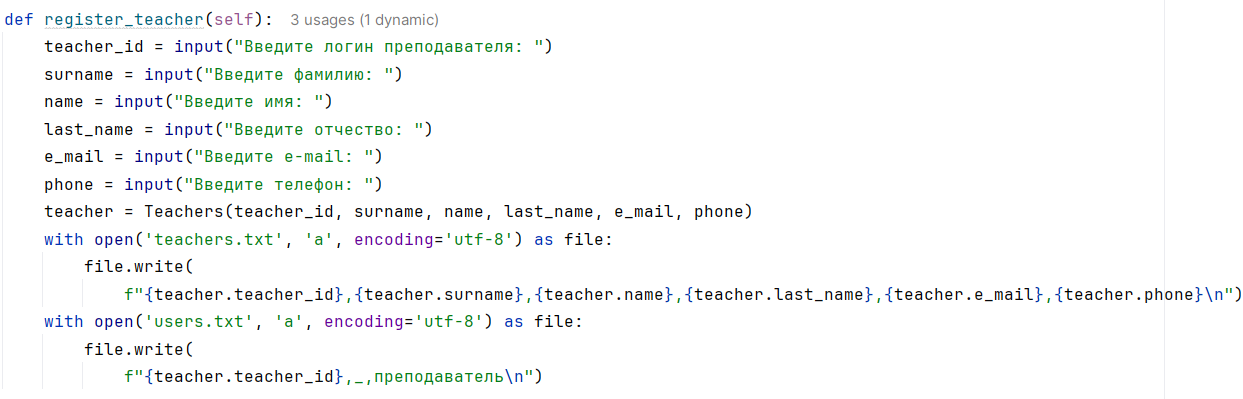


Рисунок 5. Функция регистрации преподавателя (аналогична для студента)

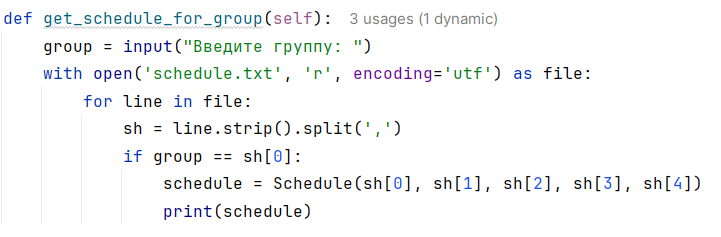


Рисунок 6. Функция вывода расписания для группы

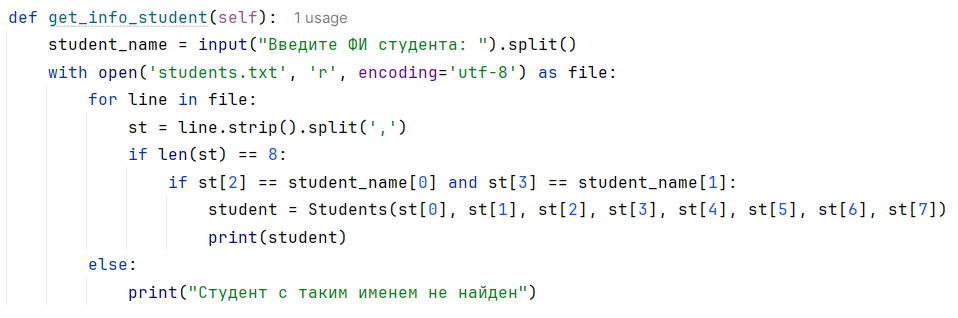


Рисунок 7. Функция вывода информации о студенте (аналогична для преподавателя)

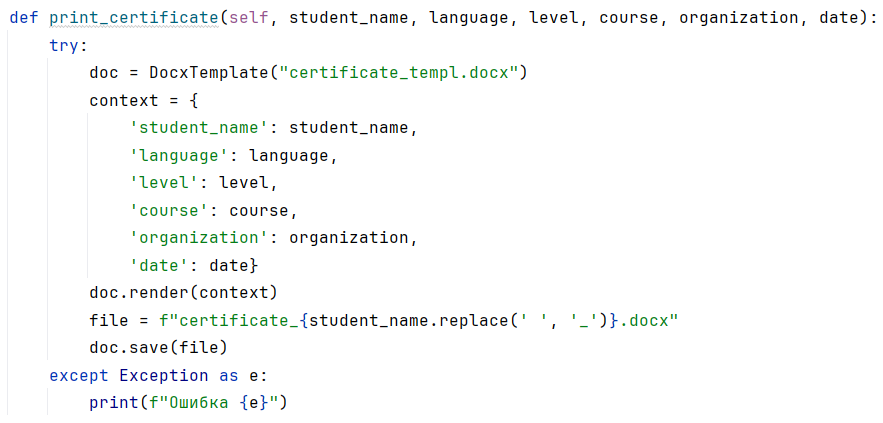


Рисунок 8. Функция создания документа по шаблону

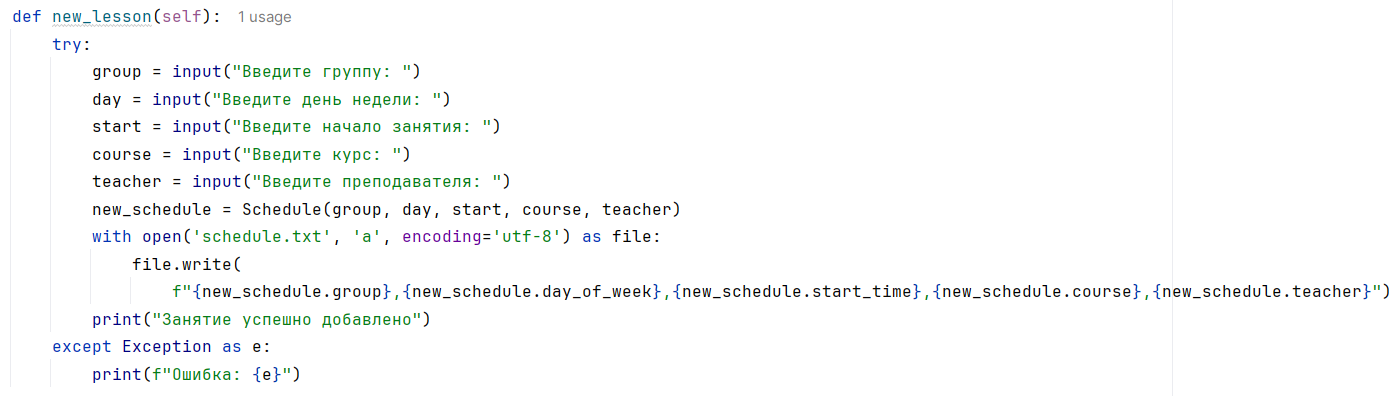


Рисунок 9. Функция добавления нового занятия в расписания

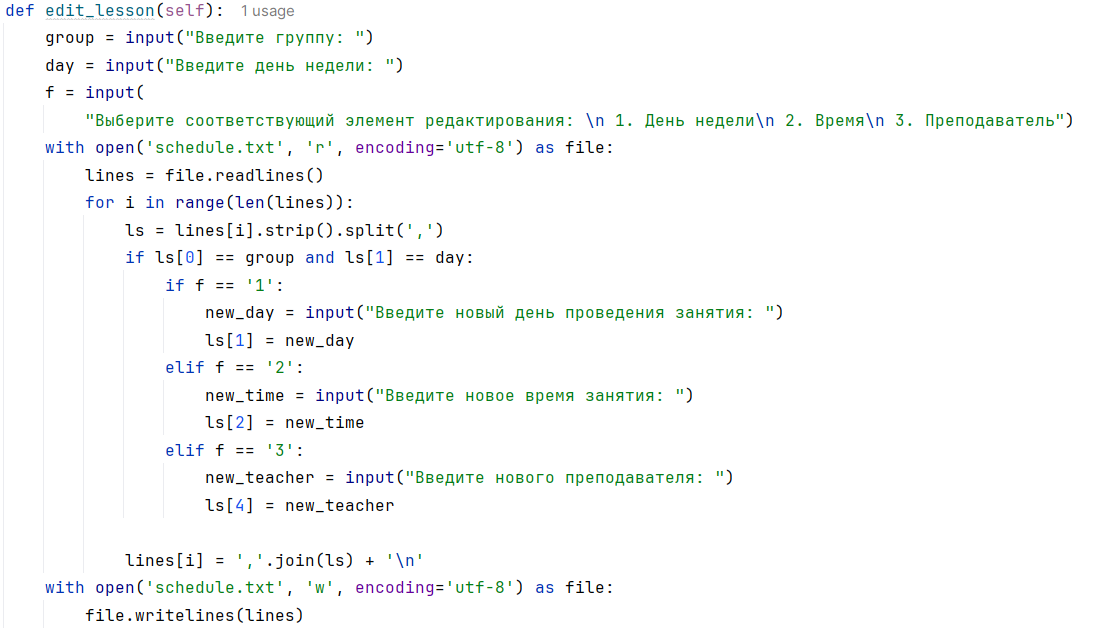


Рисунок 10. Функция редактирования занятия в расписании

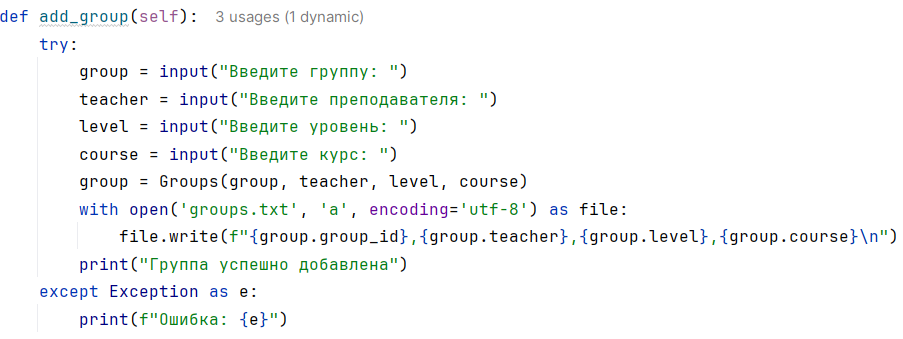


Рисунок 11. Функция добавления новой группы

ПРИЛОЖЕНИЕ 5

**Пример сертификата**

****

ПРИЛОЖЕНИЕ 6

**План тестирования**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID теста | Описание теста(тип) | Предусловия | Шаги для воспроизведения | Ожидаемый результат | Фактический результат |
| 1. | Авторизация в системе (позитивный) | 1.Пользователь находится на странице входа в личный кабинет  2. Пользователь ранее был зарегистрирован в системе | 1. Выбрать действие «Войти»  2. Ввести логин и пароль  3. Нажать на Enter | Пользователю открывается доступ и выводится сообщение «Вы успешно авторизовались» и выводятся действия, которые он может совершить | Пользователю открывается доступ и выводится сообщение «Вы успешно авторизовались» и выводятся действия, которые он может совершить |
| 2. | Регистрация (негативный) | 1. Пользователь находится на странице регистрации  2. Пользователь был ранее зарегистрирован в системе | 1. Выбрать действие «Зарегистрироваться»  2. Ввести логин  3. Ввести пароль в поле «Придумайте пароль»  4. Нажать на Enter | Пользователю выводится сообщение «Ошибка. Вы уже зарегистрированы» | Пользователю выводится сообщение «Ошибка. Вы уже зарегистрированы» |
| 3. | Вывод информации о преподавателе  (позитивный) | 1. Менеджер вводит данные о пользователе: фамилию и имя  2. Данные об этом пользователе имеются в системе | 1. Войти в систему как менеджер.  2. Выбрать действие "Личные данные преподавателя/ученика".  3. Выбрать действие "Личные данные преподавателя".  4. Ввести ФИ преподавателя.  5. Нажать на Enter | Пользователю выводятся все данные о пользователе | Пользователю выводятся все данные о пользователе |
| 4. | Вывод информации о студенте  (негативный) | 1. Менеджер вводит данные о пользователе: фамилию и имя  2. Данных об этом пользователе нет в системе | 1. Войти в систему как менеджер.  2. Выбрать действие "Личные данные преподавателя/ученика".  3. Выбрать действие "Личные данные студента".  4. Ввести ФИ преподавателя.  5. Нажать на Enter | Пользователю выводится сообщение «Такого студента не существует» | Пользователю выводится сообщение «Такого студента не существует» |
| 5. | Просмотр расписания (студенту)  (позитивный) | 1. Пользователь выбирает действие «Просмотр расписания»  2. Данные о расписании есть в файле данных | 1. Войти в систему как студент  2. Выбрать действие «Посмотреть расписание»  3. Нажать на Enter | Пользователю выводится расписание согласно его группе | Пользователю выводится расписание согласно его группе |
| 6. | Просмотр расписания (преподавателю)  (позитивный) | 1. Пользователь выбирает действие «Просмотр расписания»  2. Данные о расписании есть в файле данных | 1. Войти в систему как преподаватель  2. Выбрать действие «Посмотреть расписание»  3. Нажать на Enter | Пользователю выводится расписание согласно его фамилии | Пользователю выводится расписание согласно его фамилии |
| 7. | Вывод данных о группе (негативный) | 1. Менеджер выбирает действие «Информация о группе»  2. Данных о группе, вводимых пользователем нет | 1. Выбрать действие «Информация о группе»  3. Ввести номер группы  4. Нажать на Enter | Менеджеру выводится сообщение «Информации о данной группе нет в системе» | Менеджеру выводится сообщение «Информации о данной группе нет в системе» |
| 8. | Редактирование расписания для группы (позитивный) | 1. Менеджер вводит номер группы для редактирования расписания  2. Для данной группы нет расписания | 1. Войти в систему как менеджер (администратор)  2. Выбрать действие «Редактировать расписание»  3. Выбрать действие «Изменение элемента расписания»  4. Ввести номер группы  5. Нажать на Enter | Пользователю выходит сообщение «Расписания для данной группы не существует» | Пользователю выходит сообщение «Расписания для данной группы не существует» |
| 9. | Создание сертификата для студента | 1. Менеджер вводит данные о студенте, окончившем курс, не вписав данные в поле «Дата выдачи: » | 1. Войти в систему как менеджер (администратор)  2. Выбрать действие «Распечатать сертификат»  3. Ввести все необходимые данные, пропустив поле ввода даты  4. Нажать на Enter | Вывод сообщения «Ошибка. Введите все данные» | Вывод сообщения «Ошибка. Введите все данные» |
| 10. | Добавление нового занятия в расписание (позитивный) | 1. Менеджер вносит данные для добавления нового занятия в расписание | 1. Войти в систему как менеджер (администратор)  2. Выбрать действие «Редактировать расписание»  3. Выбрать действие «Добавление записи в расписание»  3. Ввести все необходимые данные  3. Нажать на Enter | В файл schedule добавляются новые данные.  Пользователю выводится сообщение «Занятие успешно добавлено» | В файл schedule добавляются новые данные.  Пользователю выводится сообщение «Занятие успешно добавлено» |
| 11. | Удаление занятия из расписания (негативный) | 1. Менеджер вводит группу и день недели занятия  2. Данного занятия нет в системе | 1. Войти в систему как менеджер (администратор)  2. Выбрать действие «Редактировать расписание»  3. Выбрать действие «Удаление записи в расписание»  4. Ввести все необходимые данные, в поле «Введите группу» прописать группу, которой нет  5. Нажать на Enter | Пользователю выводится ошибка «Не удалось удалить расписание. Данного занятия нет в расписании» | Пользователю выводится ошибка «Не удалось удалить расписание. Данного занятия нет в расписании» |
| 12. | Просмотр всех занятий у группы (позитивный) | 1. Менеджер хочет вывести информацию о всех существующих занятиях | 1. Войти в систему как менеджер (администратор)  2. Выбрать действие «Отслеживание расписания»  3. Ввести номер группы  4. Нажать на Enter | Пользователю выводятся все занятия группы | Пользователю выводятся все занятия группы |
| 13. | Выгрузка данных о студентах в Excel (позитивный) | 1. Менеджер хочет выгрузить все данные о студентах, учащихся в организации в excel | 1. Войти в систему как менеджер (администратор)  2. Выбрать действие «Выгрузка в excel»  3. Нажать на Enter | Пользователю автоматически открывается файл .xlxs со всеми данными | Пользователю автоматически открывается файл .xlxs со всеми данными |
| 14. | Регистрация нового преподавателя в системе (позитивный) | 1. Менеджер добавляет нового сотрудника (преподавателя) в систему | 1. Войти в систему как менеджер (администратор)  2. Выбрать действие «Регистрация нового преподавателя»  3. Ввести все данные о преподавателе (логин, ФИО, контакты)  4. Нажать на Enter | Логин сохраняется в файле users (пароль по умолчанию пустой для новых пользователей)  Данные о преподавателе заносятся в файл teachers. | Логин сохраняется в файле users (пароль по умолчанию пустой для новых пользователей)  Данные о преподавателе заносятся в файл teachers. |
| 15. | Регистрация нового студента в системе (негативный) | 1. Менеджер добавляет нового студента в систему | 1. Войти в систему как менеджер (администратор)  2. Выбрать действие «Регистрация нового студента»  3. Оставить все поля пустыми  4. Нажать на Enter | Пользователю выводится сообщение «Ошибка. Введите все необходимые данные» | Пользователю выводится сообщение «Ошибка. Введите все необходимые данные» |

ПРИЛОЖЕНИЕ 7

**Результаты тестирования модуля Managers**

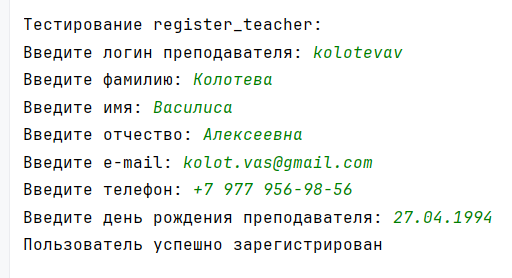


Рисунок 1. Тестирование регистрации преподавателя

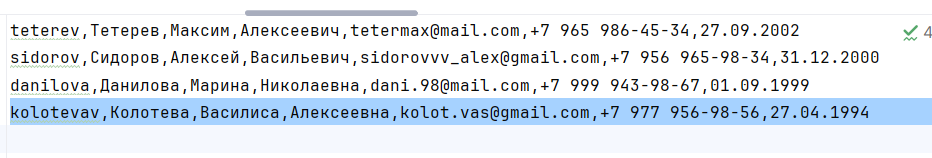


Рисунок 2. Результат тестирования регистрации. Файл teachers

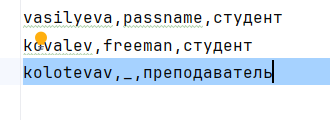


Рисунок 2. Результат тестирования регистрации. Файл users

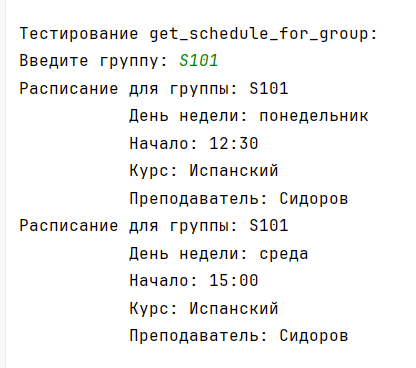


Рисунок 3. Результат тестирования вывода расписания для группы

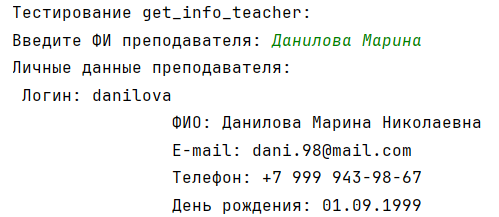


Рисунок 4. Результат тестирования вывода информации о преподавателе

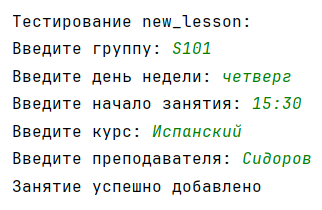


Рисунок 5. Тестирование добавления нового занятия

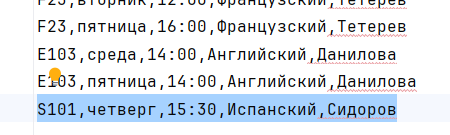


Рисунок 5. Результат тестирования добавления нового занятия

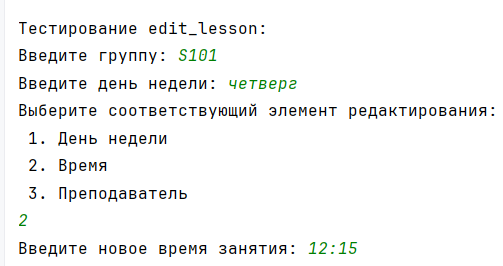


Рисунок 6. Тестирование редактирования расписания

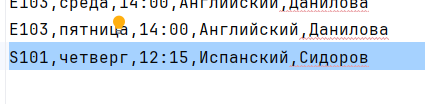


Рисунок 6. Результат тестирования редактирования расписания

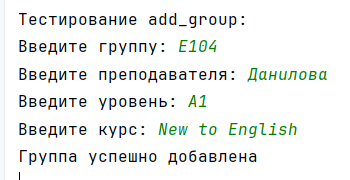


Рисунок 7. Тестирование добавления новой группы

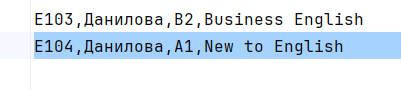


Рисунок 8. Результат тестирования добавления новой группы

ПРИЛОЖЕНИЕ 8

**Тестирование основного модуля**

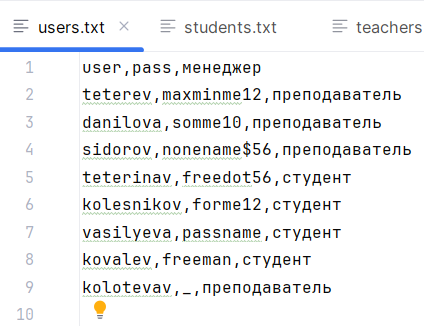


Рисунок 9. Исходные тестовые данные в файле users

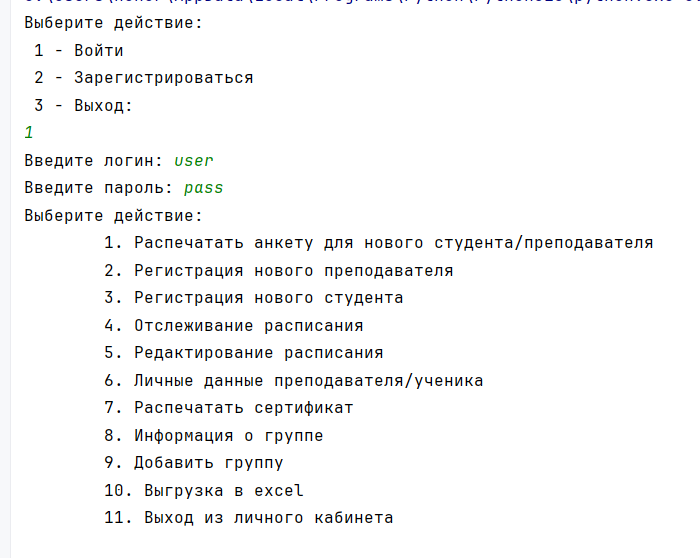


Рисунок 10. Тестирование авторизации менеджера

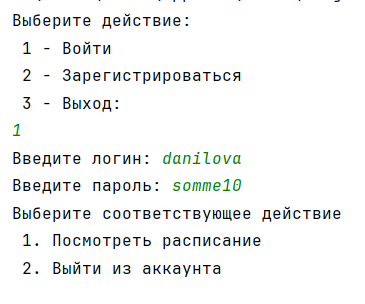


Рисунок 11. Тестирование авторизации преподавателя

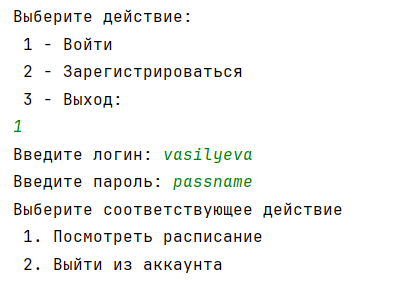


Рисунок 12. Тестирование авторизации студента

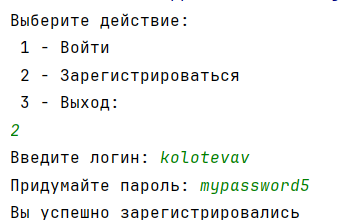


Рисунок 13. Тестирование регистрации

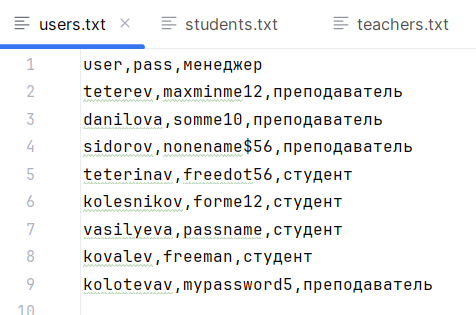


Рисунок 14. Данные в файле users после регистрации

ПРИЛОЖЕНИЕ 9

**Блок-схемы**

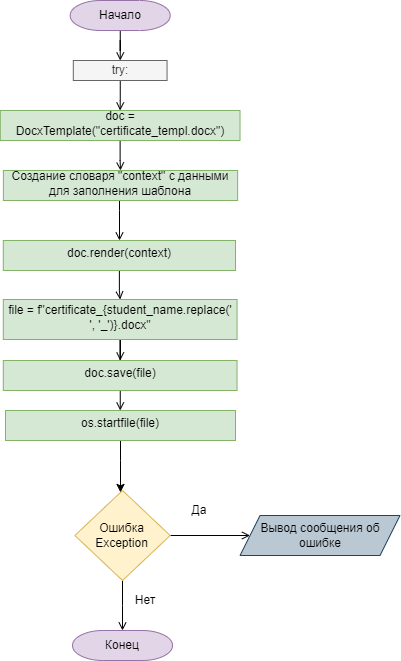


Рисунок 15. Блок – схема функции print\_certificate

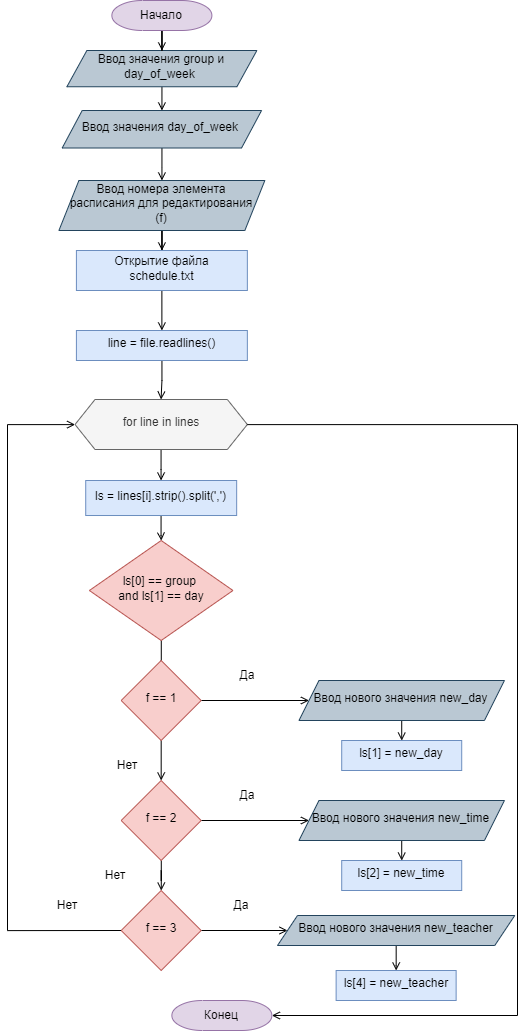


Рисунок 16. Блок-схема к функции edit\_lesson()

ПРИЛОЖЕНИЕ 10

**Проектирование дизайна системы**

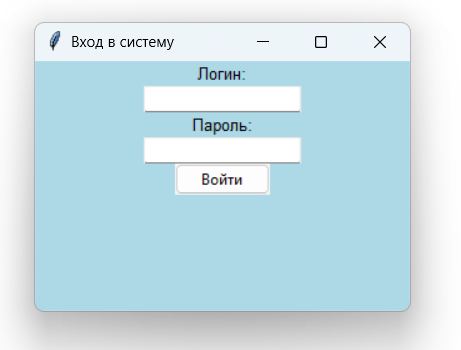


Рисунок 17. Окно авторизации

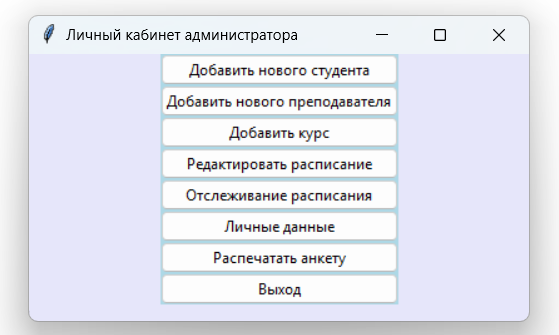


Рисунок 18. Меню выбора действий менеджера

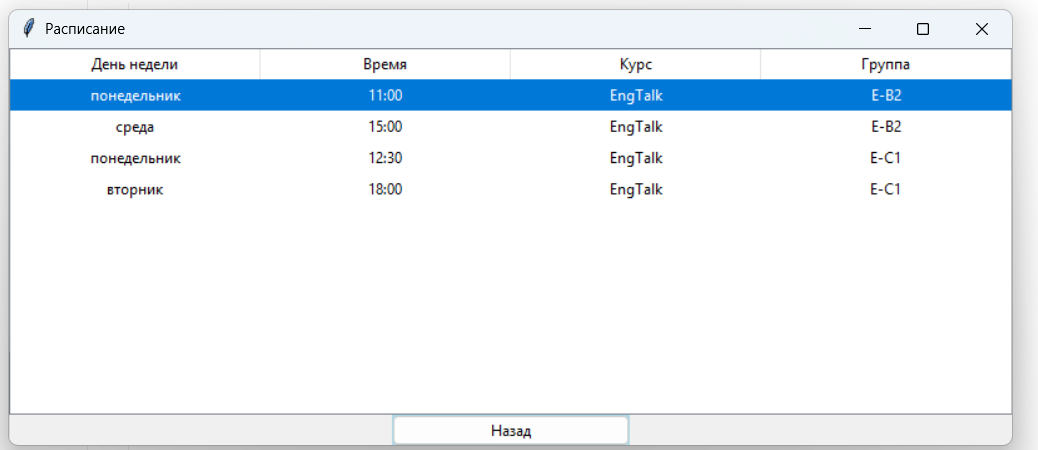


Рисунок 19. Окно просмотра расписания (для преподавателя)

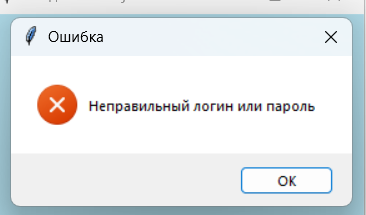


Рисунок 20. Окно вывода сообщения об ошибке при авторизации