



КУРСОВОЙ ПРОЕКТ

По дисциплине: МДК 01.01 Разработка программных модулей

Тема: Разработка системы классов для приложения «Танцевальный коллектив»

Специальность 09.02.07 «Информационные системы и программирование»

**Выполнил студент(ка) группы
301ИС-22**

**В.В.
Васильева**

Руководитель

**Л.Б.
Гуситинер**

Москва 2024



УТВЕРЖДАЮ

Зам. директора КМПО

_____ **С.Ф. Гасанов**

«_____» _____ **2024 г.**

ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ

По дисциплине: МДК 01.01 Разработка программных модулей

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

Студентка группы 301ИС-22 Васильева Виктория

**ТЕМА: «Разработка системы классов для приложения «Танцевальный
коллектив»**

Дата выдачи задания «_____» _____ 2024 г.

Срок сдачи проекта «_____» _____ 2024 г.

Москва 2024

Содержание

Введение.....	4
Глава 1. Теоретическая часть.....	5
1.1. Описание предметной области.....	5
1.2. Описание существующих разработок	5
Глава 2. Проектная часть.....	9
2.1. Диаграмма прецедентов.....	9
2.2. Выбор инструментов	10
2.3. Проектирование сценария	10
2.4. Диаграмма классов	12
2.5. Описание главного модуля.....	12
2.6. Описания спецификаций к модулям	15
2.7. Описание модулей	19
2.8. Описание тестовых наборов модулей	21
Глава 3. Эксплуатационная часть.....	24
3.1. Руководство оператора	24
Заключение	28
Список литературы и интернет-источников	29
Приложение 1. Код главного модуля main.c	30
Приложение 2. Код модуля func.c	32
Приложение 3. Код заголовочного файла func.h	38
Приложение 4. Код заголовочного файла date.h.....	39
Приложение 5. Код заголовочного файла perfomance.h	40
Приложение 6. Код заголовочного файла Choreographer.h	41
Приложение 7. Код заголовочного файла Perfomance.h	42

Введение

Цель данного курсового проекта — разработка программного обеспечения для автоматизации работы танцевального коллектива, которое будет поддерживать управление процессами, связанными с организацией и деятельностью танцевального коллектива. Эта система призвана повысить эффективность взаимодействия между участниками коллектива, облегчить управление выступлениями, хореографами и танцорами. Ключевыми преимуществами данной программы являются высокая скорость работы и минимальные требования к аппаратным ресурсам.

Проект состоит из нескольких этапов. На первом этапе будет рассмотрено текущее состояние предметной области танцевальных коллективов, а также исследованы существующие аналогичные решения, которые могут служить основой для разработки.

Во второй части будет представлена разработка основных классов системы, включая описание функциональности каждого класса, а также примеры кода, иллюстрирующие реализацию ключевых функций программы, таких как управление участниками, отслеживание выступлений и управление финансами.

В заключение, в третьей части будет представлено руководство для операторов системы, включающее инструкции по использованию программы и её настройке, чтобы обеспечить наилучший опыт взаимодействия с разработанным программным обеспечением.

Глава 1. Теоретическая часть

1.1. Описание предметной области

Танцевальные коллективы представляют собой важный аспект культурной жизни, предлагая разнообразные формы искусства и способствуя развитию творческих способностей участников. Управление танцевальным коллективом включает в себя несколько ключевых аспектов:

- Проведение выступлений, требующее координации между танцорами, хореографами и организаторами мероприятий. Важно отслеживать даты выступлений, места проведения и состав участников.
- Организация репетиций, так как для успешного выступления необходимо тщательно планировать расписание и места проведения тренировок. Это включает в себя выбор времени и составление графиков.
- Управление финансовыми аспектами, включая сбор взносов от участников, учет расходов на костюмы и участие в мероприятиях. Важно обеспечить прозрачность и доступность информации о финансовом состоянии коллектива.

Современные технологии предоставляют возможности для автоматизации процессов управления танцевальным коллективом. Приложения, разработанные для этой цели, могут включать функции для планирования репетиций, отслеживания выступлений, ведения финансового учета и взаимодействия между участниками, что значительно упрощает жизнь как руководству, так и танцорам.

1.2. Описание существующих разработок

На данный момент на рынке представлено несколько типов программных продуктов, предназначенных для автоматизации работы танцевальных коллективов. Эти решения отличаются по функционалу, масштабам применения и техническим требованиям. Рассмотрим основные из них.

1. Программное обеспечение для управления танцевальными коллективами

Многие танцевальные школы и коллективы используют специализированные программные продукты для оптимизации своих процессов. Такие системы обеспечивают:

- Автоматизированное планирование репетиций и выступлений.
- Учёт участия участников в мероприятиях и репетициях.
- Ведение отчётности по финансам и сбору взносов.
- Интеграцию с другими платформами, например для продаж билетов или онлайн-регистрации.

Примеры решений:

- **Система для управления расписанием (ListOk CRM):** Эта платформы позволяют организатором легко планировать и изменять график репетиций и выступлений, уведомляя участников об изменениях (Рисунок 1 – Интерфейс платформы ListOk CRM).

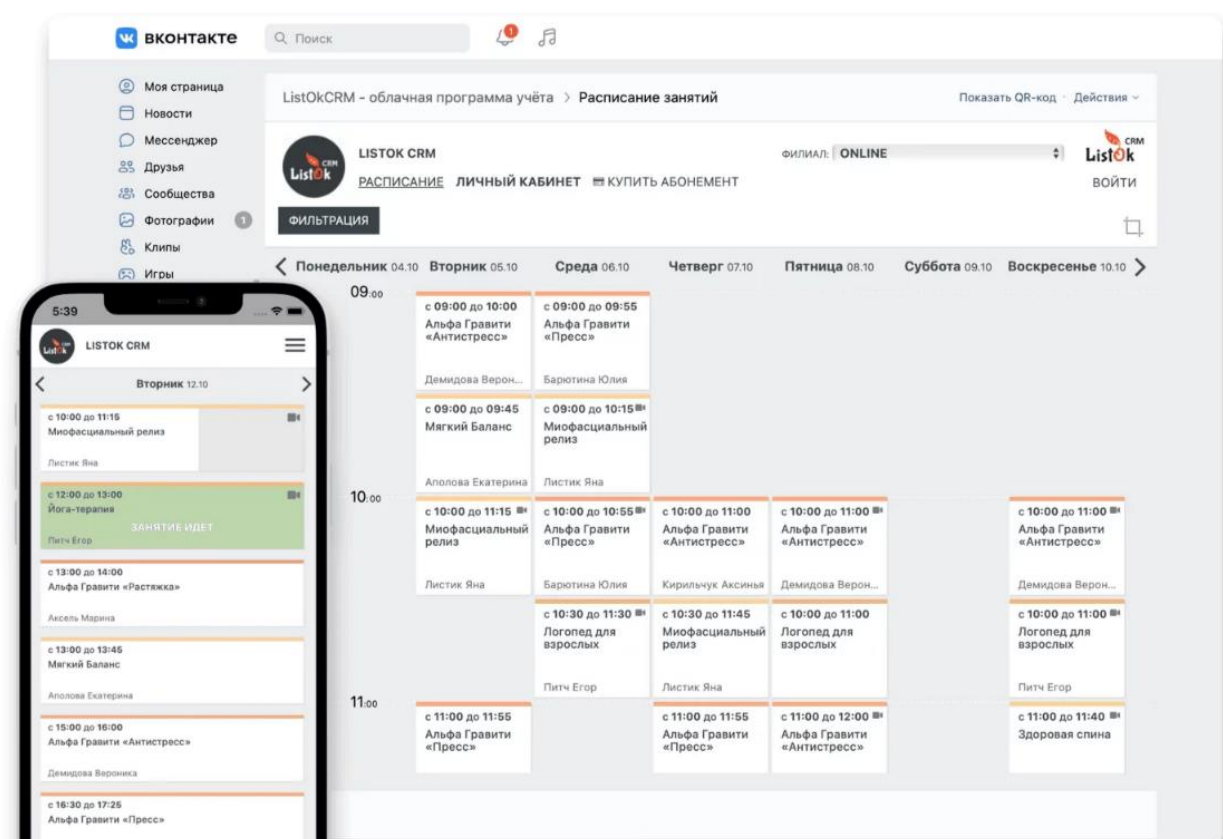


Рисунок 1 – Интерфейс платформы ListOk CRM

- **Приложение для учета участников (AppEvent):** Такая программа помогает отслеживать, кто присутствует на репетициях, фиксировать результаты выступлений, а также вести онлайн-записи в хореографический коллектив (Рисунок 2 – Интерфейс платформы AppEvent).

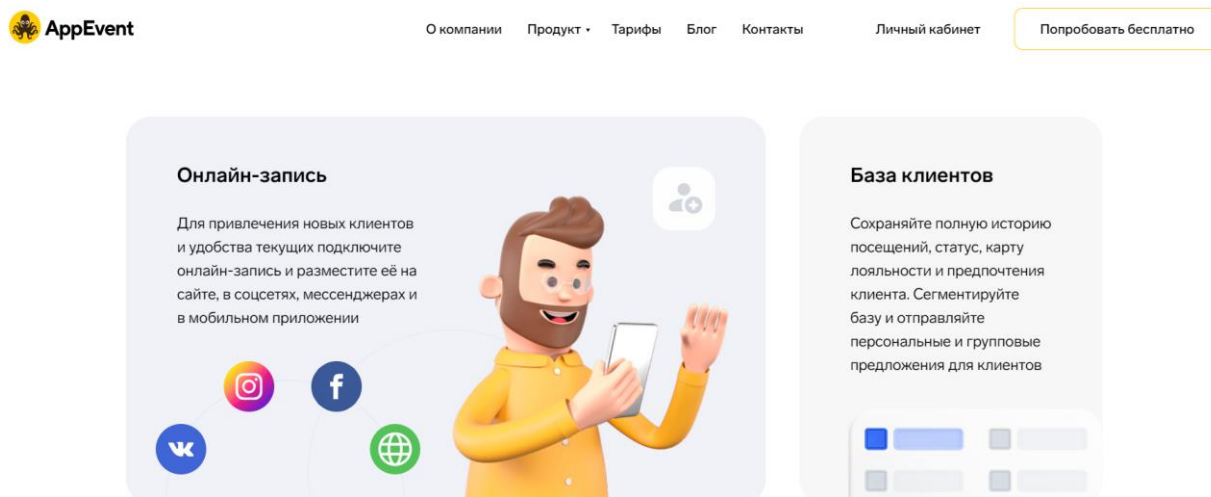


Рисунок 2 – Интерфейс платформы AppEvent

Преимущества:

- Повышение эффективности работы за счет автоматизации рутинных задач.
- Упрощение коммуникации между участниками и руководством.
- Возможность более детального анализа данных о деятельности коллектива.

Недостатки:

- Необходимость обучения сотрудников для работы с новыми системами.
- Потребность в технической поддержке для решения возможных проблем с программным обеспечением.

2. Онлайн-платформы для танцевальных коллективов

С развитием интернета и технологий в области организации мероприятий появились онлайн-платформы, которые позволяют

танцевальным коллективам эффективно управлять своей деятельностью без физического присутствия. Эти системы предоставляют:

- Автоматическое обновление расписания репетиций и выступлений в зависимости от наличия участников.
- Поддержку различных форматов мероприятий (групповые репетиции, мастер классы, выступления).
- Интеграцию с системами онлайн-коммуникации и платежами для продажи билетов.

Примеры:

- **Онлайн-сервисы для продажи билетов на выступления:**

Платформа **Nethouse** предоставляет танцевальным коллективам возможность управлять продажей билетов на свои мероприятия, а также организовывать расписание и взаимодействовать с участниками (Рисунок 3 – Интерфейс платформы NetHouse).

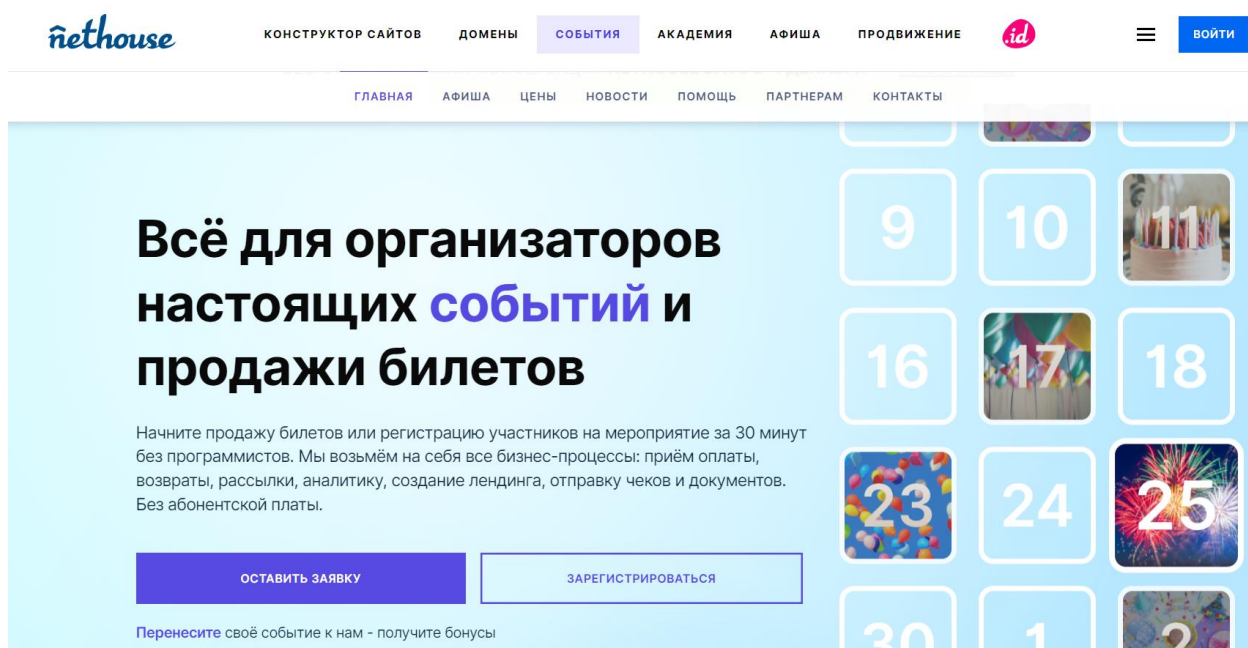


Рисунок 3 – Интерфейс платформы NetHouse

Преимущества:

- Доступность 24/7.
- Широкий выбор инструментов для управления билетами, продажами и мероприятиями.

- Минимальные затраты на администрирование и организацию мероприятий.

Недостатки:

- Зависимость от доступности интернета.
- Возможные риски безопасности, связанные с защитой данных пользователей и финансовых транзакций.

Глава 2. Проектная часть

2.1 Диаграмма прецедентов

Для определения вариантов использования к проекту была построена диаграмма прецедентов (Рисунок 4 - Диаграмма прецедентов для проекта).



Рисунок 4 - Диаграмма прецедентов для проекта

2.2 Выбор инструментов

Я выбрала язык программирования C для своего курсового проекта по нескольким причинам:

1. **Производительность:** C предоставляет низкоуровневый доступ к памяти и ресурсам, что позволяет создавать высокопроизводительные программы.

2. **Широкая применимость:** C является основой для многих современных языков программирования, таких как C++, C#, и Java. Знание C поможет мне легче освоить другие языки.

Для разработки проекта я выбрала среду CLion по нескольким причинам:

1. **Современный интерфейс:** CLion предлагает удобный и интуитивно понятный интерфейс, который облегчает процесс программирования.

2. **Поддержка C и C++:** CLion имеет отличную интеграцию с языками C и C++, что упрощает написание кода и его отладку.

3. **Инструменты рефакторинга:** Среда предоставляет мощные инструменты для рефакторинга кода, что позволяет улучшать его структуру без потери работоспособности.

В итоге, сочетание языка C и среды разработки CLion является идеальным выбором для достижения высоких результатов в моем курсовом проекте.

2.3 Проектирование сценария

Проектирование сценария для танцевального коллектива представляет собой описание последовательности действий, которые должны быть выполнены участниками или организаторами для достижения

определенной цели, например, подготовки и проведения выступления. Это важный этап в процессе планирования, который помогает понять, как коллектив будет взаимодействовать друг с другом и какую работу необходимо выполнить для успешного представления.

Для танцевального коллектива можно рассмотреть несколько основных сценариев (Рисунок 5 - Сценарий использования) , таких как: вывод информации о хореографах, вывод информации о танцорах, организация выступления. На основе этих сценариев можно разрабатывать шаги и взаимодействие между участниками коллектива, что способствует более слаженной и эффективной работе команды.

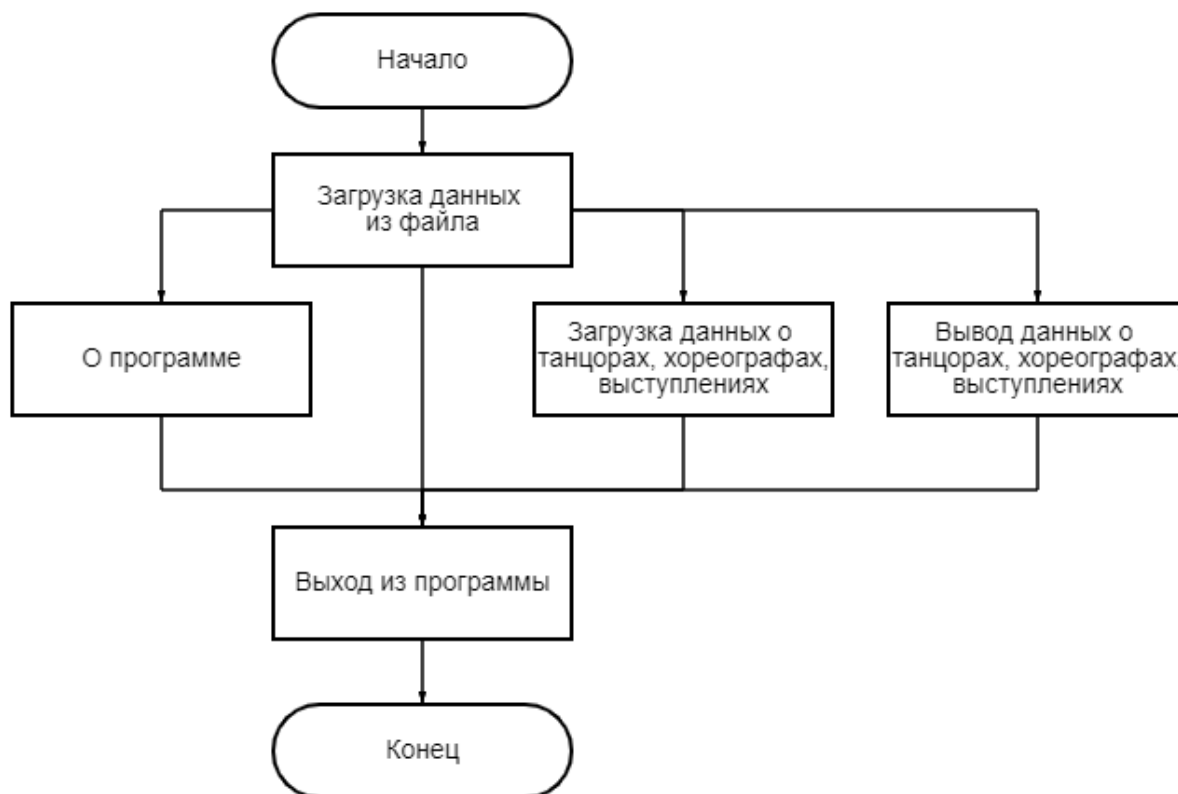


Рисунок 5 - Сценарий использования.

2.4 Диаграмма классов

В диаграмме классов (Рисунок 6 – Диаграмма классов) представлены четыре класса Date (для даты), Dancer (для танцоров), Choreographer (для хореографов) и Performances (для выступлений).

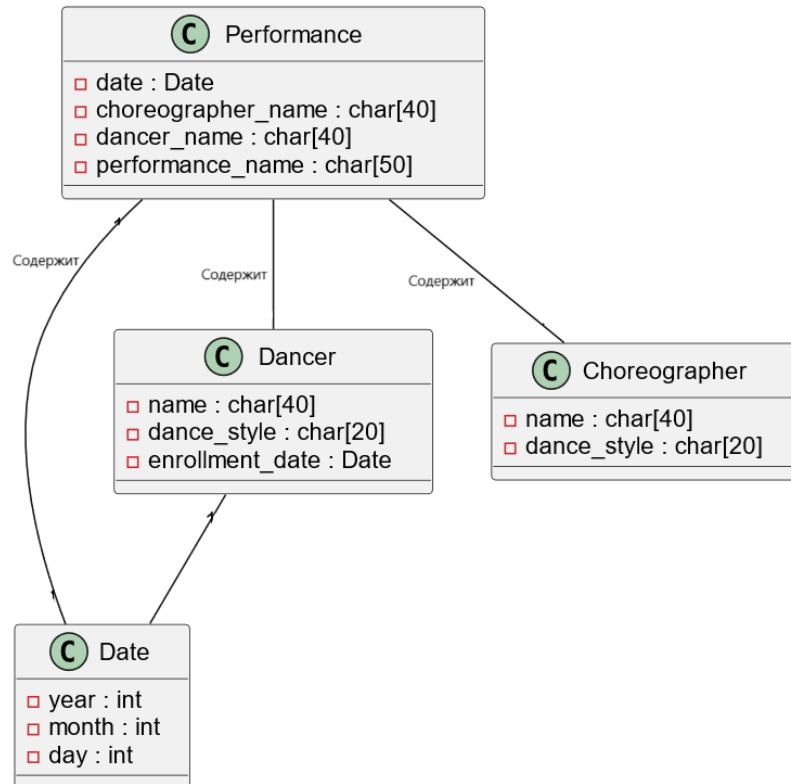


Рисунок 6 - Диаграмма классов

2.5 Описание главного модуля

Главный модуль представляет собой файл main.c.

К главному модулю подключаются остальные модули, содержащие в себе классы и функции.

Листинг 1. Код главного модуля

```
#include <stdio.h>    // Подключение стандартной библиотеки ввода-вывода
#include <locale.h>    // Подключение библиотеки для работы с локалью
#include <string.h>    // Подключение библиотеки для работы со строками
#include "func.h"
```

```

#define AR_LEN 1000    // Определение константы для максимального количества
записей

#define TRUE 1         // Определение константы для логического значения "истина"

// Основная функция программы
int main() {

    setlocale(LC_ALL, "C"); // Установка локали для корректного отображения текста

    Dancer dancers[AR_LEN]; // Массив для хранения данных о танцорах
    Choreographer choreographers[AR_LEN]; // Массив для хранения данных о хореографах
    Performance performances[AR_LEN]; // Массив для хранения данных о выступлениях

    int dancer_count = 0; // Количество танцоров
    int choreographer_count = 0; // Количество хореографов
    int performance_count = 0; // Количество выступлений

    while (TRUE) {
        int item = -1;
        show_menu(); // Отображение меню
        scanf("%d", &item); // Чтение выбранного пункта меню

        if (item == 0) {
            printf("Спасибо за использование нашей разработки\n");
            break;
        } else if (item == 1) {
            printf("Студентка Васильева Виктория Владимировна группа 301ИС-22\n");
            printf("Тема курсового проекта: Автоматизация танцевального коллектива\n");
        } else if (item == 2) {
            dancer_count = load_dancers("dancers.txt", dancers, AR_LEN);
            choreographer_count = load_choreographers("choreographers.txt", choreographers,
AR_LEN);
            performance_count = load_performances("performances.txt", performances, AR_LEN);
            printf("Данные успешно загружены из файлов\n");
        } else if (item == 3) {

```

```

        write_performances_to_file("output.txt", performances, performance_count);
    } else if (item == 4) {
        execute_query(performances, performance_count);
    } else {
        printf("Ошибка. Пункты от 0 до 4.\n");
    }

    getchar(); // Для захвата символа новой строки
}

return 0;
}

```

Ниже представлена схема главного модуля программы.

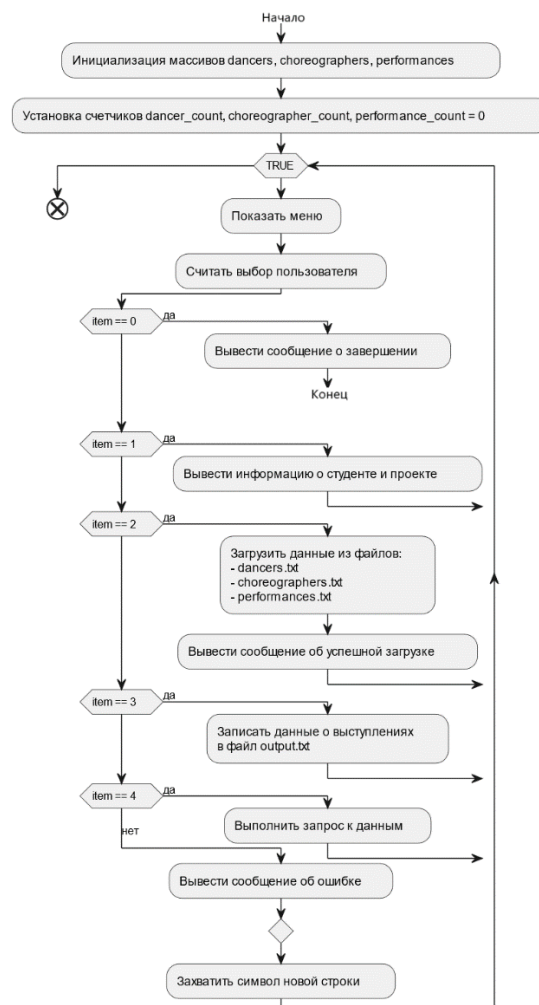


Рисунок 7 - Блок-схема главного модуля.

2.6 Описания спецификаций к модулям

Спецификация к модулю func (Листинг 2).

Листинг 2. Код класса func

```
#include <stdio.h>    // Подключение стандартной библиотеки ввода-вывода
#include "Dancer.h"
#include "Performance.h"
#include "Choreographer.h"

#define AR_LEN 1000   // Определение константы для максимального количества записей
#define TRUE 1        // Определение константы для логического значения "истина"

// Функция для преобразования данных о танцоре в строку
void rec_to_s_dancer(char s[100], Dancer dancer) {
    sprintf(s, "Имя: %s, Стилль танца: %s, Дата зачисления: %02d.%02d.%04d",
            dancer.name, dancer.dance_style,
            dancer.enrollment_date.day,
            dancer.enrollment_date.month,
            dancer.enrollment_date.year);
}

// Функция для преобразования данных о хореографе в строку
void rec_to_s_choreographer(char s[100], Choreographer choreographer) {
    sprintf(s, "Имя: %s, Стилль танца: %s", choreographer.name, choreographer.dance_style);
}

// Функция для преобразования данных о выступлении в строку
void rec_to_s_performance(char s[100], Performance performance) {
    sprintf(s, "Дата: %02d.%02d.%04d, Танцор: %s, Название выступления: %s",
            performance.date.day, performance.date.month, performance.date.year,
            performance.dancer_name, performance.performance_name);
}

// Функция для отображения меню
void show_menu() {
    printf(
        "0. Выход\n"
        "1. О программе\n"
```

```

    "2. Загрузка данных из файла\n"
    "3. Вывод данных из файлов\n"
    "4. Выполнение запроса\n");
}

// Функция для загрузки данных о танцорах из файла
int load_dancers(char *fname, Dancer dancers[], int limit) {
    FILE *in = fopen(fname, "r"); // Открытие файла для чтения
    if (!in) {
        printf("Ошибка открытия файла!\n");
        return 0;
    }

    int count = 0;
    Dancer rec;

    // Чтение данных из файла
    while (count < limit && fscanf(in, "%s %s %d.%d.%d",
        rec.name,
        rec.dance_style,
        &rec.enrollment_date.day,
        &rec.enrollment_date.month,
        &rec.enrollment_date.year) == 5) {
        dancers[count] = rec;
        count++;
    }

    fclose(in); // Закрытие файла
    return count;
}

// Функция для загрузки данных о хореографах из файла
int load_choreographers(char *fname, Choreographer choreographers[], int limit) {
    FILE *in = fopen(fname, "r"); // Открытие файла для чтения
    if (!in) {
        printf("Ошибка открытия файла!\n");
        return 0;
    }

```



```

int count = 0;
Choreographer rec;

// Чтение данных из файла
while (count < limit && fscanf(in, "%s %s", rec.name, rec.dance_style) == 2) {
    choreographers[count] = rec;
    count++;
}

fclose(in); // Закрытие файла
return count;
}

// Функция для загрузки данных о выступлениях из файла
int load_performances(char *fname, Performance performances[], int limit) {
    FILE *in = fopen(fname, "r"); // Открытие файла для чтения
    if (!in) {
        printf("Ошибка открытия файла!\n");
        return 0;
    }

    int count = 0;
    Performance rec;

    // Чтение данных из файла
    while (count < limit && fscanf(in, "%d.%d.%d %s %s %s",
        &rec.date.day,
        &rec.date.month,
        &rec.date.year,
        rec.choreographer_name,
        rec.dancer_name,
        rec.performance_name) == 6) {
        performances[count] = rec;
        count++;
    }

    fclose(in); // Закрытие файла

```

```

    return count;
}

// Функция для вывода данных в файл
void write_performances_to_file(char *fname, Performance performances[], int limit) {
    FILE *out = fopen(fname, "w"); // Открытие файла для записи
    if (!out) {
        printf("Ошибка открытия файла для записи!\n");
        return;
    }

    for (int i = 0; i < limit; i++) {
        fprintf(out, "Дата: %02d.%02d.%04d, Танцор: %s, Название выступления: %s\n",
            performances[i].date.day, performances[i].date.month, performances[i].date.year,
            performances[i].dancer_name, performances[i].performance_name);
    }

    fclose(out); // Закрытие файла
    printf("Данные успешно записаны в файл %s\n", fname);
}

// Функция для выполнения запроса
void execute_query(Performance performances[], int limit) {
    printf("Введите дату выступления (в формате ДД.ММ.ГГГГ): ");
    int day, month, year;
    scanf("%d.%d.%d", &day, &month, &year);

    int found = 0;
    for (int i = 0; i < limit; i++) {
        if (performances[i].date.day == day &&
            performances[i].date.month == month &&
            performances[i].date.year == year) {
            printf("Дата: %02d.%02d.%04d, Танцор: %s, Название выступления: %s\n",
                performances[i].date.day, performances[i].date.month, performances[i].date.year,
                performances[i].dancer_name, performances[i].performance_name);
            found = 1;
        }
    }
}

```

```

    if (!found) {
        printf("Выступлений на указанную дату не найдено.\n");
    }
}

```

Спецификация к модулю func (Листинг 3).

Листинг 3. Код класса func

```

#ifndef FUNC_H
#define FUNC_H

#include "Dancer.h"
#include "Performance.h"
#include "Choreographer.h"

void rec_to_s_dancer(char s[100], Dancer dancer);
void rec_to_s_choreographer(char s[100], Choreographer choreographer);
void rec_to_s_performance(char s[100], Performance performance);
void show_menu();
int load_dancers(char *fname, Dancer dancers[], int limit);
int load_choreographers(char *fname, Choreographer choreographers[], int limit);
int load_performances(char *fname, Performance performances[], int limit);
void write_performances_to_file(char *fname, Performance performances[], int limit);
void execute_query(Performance performances[], int limit);

#endif //FUNC_H

```

2.7 Описание модулей

В данной главе описаны используемые модули.

1. Главный модуль

Главный модуль представляет собой файл main.c. К главному модулю подключаются остальные модули, содержащие в себе классы и функции.

2. Модуль функций

В модуле функций находятся функции, используемые в проекте:

- `print()` - вывод на экран
- `FILE *in = fopen(fname, "r");` - открытие данных и файла
- `FILE *out = fopen(fname, "w");` - создание файла и вывод данных
- `getchar()` – Пауза для меню
- `setlocale(LC_ALL, "C");` - Подключение Русского языка.

3. Модуль `Date.h`

`Date.h` содержит определение структуры `Date`, которая используется для хранения и работы с датами. Этот модуль представляет собой заголовочный файл, который определяет структуру и предотвращает повторное включение файла в проект. В него входят:

- Год
- Месяц
- День

4. Модуль `Choreographer.h`

Модуль `Choreographer.h` содержит определение структуры `Choreographer`, которая используется для хранения информации о хореографах. В него входят:

- Имя хореографа
- Стил танца

5. Модуль `Dancer.h`

`Dancer.h` содержит определение структуры `Dancer`, которая используется для хранения информации о танцорах. В него входят:

- Имя танцора
- Стил танца
- Дата зачисления

6. Модуль `Performance.h`

`Performance.h` содержит определение структуры `Performance`, которая используется для хранения информации о выступлениях танцевального коллектива. В него входят:

- Дата выступления
- Имя хореографа
- Имя танцора
- Название выступления

7. Модуль func.h

Класс func предназначен для работы с функциями чтобы сама программа работала без нареканий.

2.8 Описание тестовых наборов модулей

В этом разделе будут показаны результаты функционального тестирования.

Функциональное тестирование — это процесс проверки того, как программа выполняет свои функции. Оно направлено на проверку поведения программы в соответствии с её спецификацией или ожидаемым поведением.

1. Тестирование меню и выбора пунктов

Цель: Убедиться, что меню отображается корректно, и программа реагирует на выбор пользователя.

Шаги:

1. Запустите программу.
2. Убедитесь, что меню отображается.
3. Введите пункты меню (например, 0, 1, 2, 3, 4) и проверьте, что программа корректно реагирует на каждый пункт.

Ожидаемый результат:

- Меню отображается корректно.
- При выборе пункта 0 программа завершает работу.
- При выборе пункта 1 выводится информация о программе.
- При выборе пунктов 2, 3, 4 программа выполняет соответствующие действия.

Полученный результат:

- Меню отображается корректно.
- При выборе пункта 0 программа завершает работу.
- При выборе пункта 1 выводится информация о программе.
- При выборе пунктов 2, 3, 4 программа выполняет

соответствующие действия.

2. Тестирование загрузки данных из файлов

Цель: Убедиться, что данные о танцорах, хореографах и выступлениях корректно загружаются из файлов.

Шаги:

1. Убеждаюсь, что файлы `dancers.txt`, `choreographers.txt` и `performances.txt` существуют и содержат корректные данные.
2. Выбираю пункт меню 2.
3. Проверяю, что программа выводит сообщение о успешной загрузке данных.

Ожидаемый результат:

- Данные загружаются корректно.
- Количество загруженных записей соответствует количеству строк в файлах.

Полученный результат:

- Данные загружаются корректно.
- Количество загруженных записей соответствует количеству строк в файлах.

3. Тестирование интеграции всех функций

Цель: Убедиться, что все функции программы работают вместе корректно.

Шаги:

1. Загрузить данные (пункт меню 2).
2. Вывести данные в файл (пункт меню 3).

3. Выполнить запрос (пункт меню 4).
4. Проверяю, что все данные корректно загружаются, записываются и обрабатываются.

Ожидаемый результат:

- Все функции программы работают корректно вместе.

Полученный результат:

- Все функции программы работают корректно вместе.

Глава 3. Эксплуатационная часть

3.1 Руководство оператора

3.1.1 Назначение программы

1. Функциональное назначение программы

Основной функцией программы "Автоматизация работы танцевального коллектива" является автоматизация процессов управления данными танцевального коллектива. Оно позволяет пользователям загружать данные из файлов, отображать их и выходить из программы. Данные включают информацию о танцорах, хореографах и выступлениях.

2. Эксплуатационное назначение программы

Программа "Автоматизация танцевального коллектива" может быть использована в любом танцевальном коллективе, стремящемся повысить эффективность своей деятельности и улучшить организационные процессы.

3. Состав функций

Функции:

1. Выход
2. О программе
3. Загрузка данных из файла
4. Вывод данных из файлов
5. Выполнение запроса

4. Условия выполнения программы

Минимальный состав аппаратных средств

Минимальный состав используемых технических (аппаратных) средств:

- Операционная система: Windows 7, 8.1 или 10;
- Процессор: частота не менее 1,8 ГГц.
- ОЗУ (оперативная память): не менее 10 МБ;
- Свободное место на жестком диске: не менее 448 МБ;
- Видеоадаптер с минимальным разрешением не менее 144p.

Минимальный состав программных средств

Дополнительные программы не требуются

Требования к персоналу

Базовые навыки работы с компьютером

5. Загрузка и запуск программы

Программа состоит из 10 файлов: «main.c», «func.c», «func.h», «Date.h», «Dancer.h», «Performance.h», «Choreographer.h», «dancers.txt», «performances.txt», «choreographers.txt». В текстовых файлах находятся данные о выступлениях, танцорах и хореографах. Все эти файлы должны находиться в одном каталоге для корректной работы.

При запуске программы должен открыться терминал:

```
"C:\Users\v604\CLionProjects\untitled1 - копия\cmake-build-debug\untitled1.exe"
0. Выход
1. 0 программе
2. Загрузка данных из файла
3. Вывод данных из файлов
4. Выполнение запроса
```

Рисунок 8. Главное окно программы

6. Выполнение программы

Для выхода из программы нужно нажать клавишу '0'.

```
"C:\Users\v604\CLionProjects\untitled1 - копия\cmake-build-debug\untitled1.exe"
0. Выход
1. 0 программе
2. Загрузка данных из файла
3. Вывод данных из файлов
4. Выполнение запроса
0
Спасибо за использование моей разработки
Process finished with exit code 0
```

Рисунок 9. Окно выхода из программы

Для вывода информации о программе нужно нажать клавишу '1'.

```
1
Студентка Васильева Виктория Владимировна группа 301ИС-22
Тема курсового проекта: Автоматизация танцевального коллектива
0. Выход
1. 0 программе
2. Загрузка данных из файла
3. Вывод данных из файлов
4. Выполнение запроса
```

Рисунок 11. Окно информации «О программе»

Для загрузки информации из файлов нужно нажать клавишу '2'.

```
2
Данные успешно загружены из файлов
0. Выход
1. 0 программе
2. Загрузка данных из файла
3. Вывод данных из файлов
4. Выполнение запроса
```

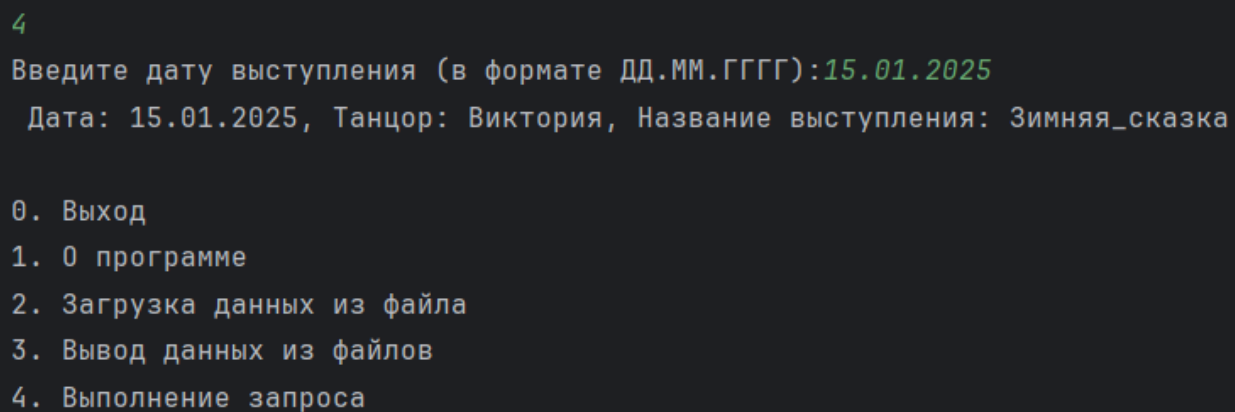
Рисунок 12. Загрузка транзакций из файла

Для записи данных в файл нужно нажать клавишу '3'.

```
3
Данные успешно записаны в файл output.txt
0. Выход
1. 0 программе
2. Загрузка данных из файла
3. Вывод данных из файлов
4. Выполнение запроса
```

Рисунок 13. Вывод данных в файл

Для выполнения запроса с данными из файлов нужно нажать клавишу '4'.



4
Введите дату выступления (в формате ДД.ММ.ГГГГ):15.01.2025
Дата: 15.01.2025, Танцор: Виктория, Название выступления: Зимняя_сказка

- 0. Выход
- 1. 0 программе
- 2. Загрузка данных из файла
- 3. Вывод данных из файлов
- 4. Выполнение запроса

Рисунок 15. Сохранение транзакций в файл

Заключение

В результате выполнения курсового проекта была написана программа для танцевального коллектива, направленная на автоматизацию процессов подготовки и проведения выступлений.

В ходе работы были проанализированы предметная область, существующие разработки посвященные данному направлению, а также получены практические навыки с работой и программированием в Clion.

Я намерена продолжить работу над данным проектом с целью улучшения организации и повышения удобства взаимодействия участников коллектива. Планы по доработке следующие:

1. Надежность и обработка ошибок:
 - Улучшить обработку неправильно отформатированных входных файлов.
 - Добавить проверку целостности данных и переполнения буфера.
2. Пользовательский интерфейс:
 - Улучшить взаимодействие с пользователем с помощью проверки ввода и более удобного интерфейса.
3. Эффективность кода:
 - Сократить дублирование кода, создав универсальные функции для загрузки данных.
4. Масштабируемость:
 - Рассмотреть возможность динамического выделения памяти для обработки больших наборов данных.
5. Безопасность:
 - Внедрить меры по предотвращению уязвимостей в системе безопасности, таких как переполнение буфера.

Список литературы и интернет-источников

1. Сайт билетного сервиса "Nethouse": (<https://events.nethouse.ru/>)
 2. Мобильное приложение для онлайн-записи в хореографический коллектив (<https://a2is.ru/catalog/crm-sistemy-vse/appevent>)
 3. Облачная программа для учета посещений в школах танцев (<https://a2is.ru/catalog/crm-dlya-upravleniya-fitness-tsentrom/listok-crm>)
 4. Керниган, Б., Ритчи, Д. Язык программирования Си. 4-е изд. — М.: Вильямс, 2014.
 5. Бесплатный курс на российской образовательной платформе открытых онлайн-курсов и уроков Stepik «Добрый, добрый C/C++ с Сергеем Балакиревым»: (<https://stepik.org/course/193691/syllabus>)
6. Хант, А. Практическое программирование на С. — М.: Вильямс, 2010.
 7. ИСО/МЭК 9126. Информационные технологии. Оценка продукции программного обеспечения. Характеристики качества и инструкции по их применению. Международная организация стандартов, Женева, 1991.
 8. Макконнелл, С. Совершенный код. Мастер-класс / С. Макконнелл. — М.: Издательско-торговый дом «Русская редакция»; СПб. : Питер, 2005.
 9. Денисова, А. Д. Основные стили современной хореографии / А. Д. Денисова. — Текст : непосредственный // Молодой ученый. — 2020. — № 3 (293). — С. 303-306. — URL: <https://moluch.ru/archive/293/66513/>

Приложение 1. Код главного модуля main.c

```
#include <stdio.h>    // Подключение стандартной библиотеки ввода-вывода
#include <locale.h>    // Подключение библиотеки для работы с локалью
#include <string.h>    // Подключение библиотеки для работы со строками
#include "func.h"

#define AR_LEN 1000   // Определение константы для максимального количества записей
#define TRUE 1        // Определение константы для логического значения "истина"

// Основная функция программы
int main() {
    setlocale(LC_ALL, "C"); // Установка локали для корректного отображения текста

    Dancer dancers[AR_LEN]; // Массив для хранения данных о танцорах
    Choreographer choreographers[AR_LEN]; // Массив для хранения данных о хореографах
    Performance performances[AR_LEN]; // Массив для хранения данных о выступлениях

    int dancer_count = 0; // Количество танцоров
    int choreographer_count = 0; // Количество хореографов
    int performance_count = 0; // Количество выступлений

    while (TRUE) {
        int item = -1;
        show_menu(); // Отображение меню
        scanf("%d", &item); // Чтение выбранного пункта меню

        if (item == 0) {
            printf("Спасибо за использование моей разработки\n");
            break;
        } else if (item == 1) {
            printf("Студентка Васильева Виктория Владимировна группа 301ИС-22\n");
            printf("Тема курсового проекта: Автоматизация танцевального коллектива\n");
        } else if (item == 2) {
            dancer_count = load_dancers("dancers.txt", dancers, AR_LEN);
            choreographer_count = load_choreographers("choreographers.txt", choreographers,
            AR_LEN);
            performance_count = load_performances("performances.txt", performances, AR_LEN);
            printf("Данные успешно загружены из файлов\n");
        } else if (item == 3) {
            write_performances_to_file("output.txt", performances, performance_count);
        } else if (item == 4) {
            execute_query(performances, performance_count);
        } else {

```

```
    printf("Ошибка. Пункты от 0 до 4.\n");  
}  
  
    getchar(); // Для захвата символа новой строки  
}  
  
return 0;  
}
```

Приложение 2. Код модуля func.c

```
#include <stdio.h>    // Подключение стандартной библиотеки ввода-вывода

#include "Dancer.h"

#include "Performance.h"

#include "Choreographer.h"


#define AR_LEN 1000    // Определение константы для максимального количества записей

#define TRUE 1         // Определение константы для логического значения "истина"


// Функция для преобразования данных о танцоре в строку
void rec_to_s_dancer(char s[100], Dancer dancer) {
    sprintf(s, "Имя: %s, Стилль танца: %s, Дата зачисления: %02d.%02d.%04d",
            dancer.name, dancer.dance_style,
            dancer.enrollment_date.day,
            dancer.enrollment_date.month,
            dancer.enrollment_date.year);
}


// Функция для преобразования данных о хореографе в строку
void rec_to_s_choreographer(char s[100], Choreographer choreographer) {
    sprintf(s, "Имя: %s, Стилль танца: %s", choreographer.name, choreographer.dance_style);
}


// Функция для преобразования данных о выступлении в строку
void rec_to_s_performance(char s[100], Performance performance) {
    sprintf(s, "Дата: %02d.%02d.%04d, Танцор: %s, Название выступления: %s",
            performance.date.day, performance.date.month, performance.date.year,
```



```

        performance.dancer_name, performance.performance_name);
    }

// Функция для отображения меню
void show_menu() {
    printf(
        "0. Выход\n"
        "1. О программе\n"
        "2. Загрузка данных из файла\n"
        "3. Вывод данных из файлов\n"
        "4. Выполнение запроса\n");
}

// Функция для загрузки данных о танцорах из файла
int load_dancers(char *fname, Dancer dancers[], int limit) {
    FILE *in = fopen(fname, "r"); // Открытие файла для чтения
    if (!in) {
        printf("Ошибка открытия файла!\n");
        return 0;
    }

    int count = 0;
    Dancer rec;

    // Чтение данных из файла
    while (count < limit && fscanf(in, "%s %s %d.%d.%d",
        rec.name,
        rec.dance_style,

```

```

        &rec.enrollment_date.day,

        &rec.enrollment_date.month,

        &rec.enrollment_date.year) == 5) {

    dancers[count] = rec;

    count++;

}

fclose(in); // Заккрытие файла

return count;

}

// Функция для загрузки данных о хореографах из файла

int load_choreographers(char *fname, Choreographer choreographers[], int limit) {

    FILE *in = fopen(fname, "r"); // Открытие файла для чтения

    if (!in) {

        printf("Ошибка открытия файла!\n");

        return 0;

    }

    int count = 0;

    Choreographer rec;

    // Чтение данных из файла

    while (count < limit && fscanf(in, "%s %s", rec.name, rec.dance_style) == 2) {

        choreographers[count] = rec;

        count++;

    }

```

```

fclose(in); // Закрытие файла

return count;
}

// Функция для загрузки данных о выступлениях из файла
int load_performances(char *fname, Performance performances[], int limit) {
    FILE *in = fopen(fname, "r"); // Открытие файла для чтения
    if (!in) {
        printf("Ошибка открытия файла!\n");
        return 0;
    }

    int count = 0;
    Performance rec;

    // Чтение данных из файла
    while (count < limit && fscanf(in, "%d.%d.%d %s %s %s",
        &rec.date.day,
        &rec.date.month,
        &rec.date.year,
        rec.choreographer_name,
        rec.dancer_name,
        rec.performance_name) == 6) {
        performances[count] = rec;
        count++;
    }

    fclose(in); // Закрытие файла

```

```

    return count;
}

// Функция для вывода данных в файл
void write_performances_to_file(char *fname, Performance performances[], int limit) {
    FILE *out = fopen(fname, "w"); // Открытие файла для записи
    if (!out) {
        printf("Ошибка открытия файла для записи!\n");
        return;
    }

    for (int i = 0; i < limit; i++) {
        fprintf(out, "Дата: %02d.%02d.%04d, Танцор: %s, Название выступления: %s\n",
            performances[i].date.day, performances[i].date.month, performances[i].date.year,
            performances[i].dancer_name, performances[i].performance_name);
    }

    fclose(out); // Закрытие файла
    printf("Данные успешно записаны в файл %s\n", fname);
}

// Функция для выполнения запроса
void execute_query(Performance performances[], int limit) {
    printf("Введите дату выступления (в формате ДД.ММ.ГГГГ): ");
    int day, month, year;
    scanf("%d.%d.%d", &day, &month, &year);

    int found = 0;

```

```

for (int i = 0; i < limit; i++) {
    if (performances[i].date.day == day &&
        performances[i].date.month == month &&
        performances[i].date.year == year) {
        printf("Дата: %02d.%02d.%04d, Танцор: %s, Название выступления: %s\n",
            performances[i].date.day, performances[i].date.month, performances[i].date.year,
            performances[i].dancer_name, performances[i].performance_name);
        found = 1;
    }
}

if (!found) {
    printf("Выступлений на указанную дату не найдено.\n");
}
}

```

Приложение 3. Код заголовочного файла func.h

```
#ifndef FUNC_H
#define FUNC_H

#include "Dancer.h"
#include "Performance.h"
#include "Choreographer.h"

void rec_to_s_dancer(char s[100], Dancer dancer);
void rec_to_s_choreographer(char s[100], Choreographer choreographer);
void rec_to_s_performance(char s[100], Performance performance);
void show_menu();
int load_dancers(char *fname, Dancer dancers[], int limit);
int load_choreographers(char *fname, Choreographer choreographers[], int limit);
int load_performances(char *fname, Performance performances[], int limit);
void write_performances_to_file(char *fname, Performance performances[], int limit);
void execute_query(Performance performances[], int limit);

#endif //FUNC_H
```

Приложение 4. Код заголовочного файла **Dancer.h**

```
#ifndef DANCER_H

#define DANCER_H

#include "Date.h"

// Структура для хранения информации о танцоре
typedef struct _dancer {
    char name[40];    // Имя танцора
    char dance_style[20]; // Стил танца
    Date enrollment_date; // Дата зачисления
} Dancer;

#endif //DANCER_H
```

Приложение 5. Код заголовочного файла Date.h

```
#ifndef DATE_H
#define DATE_H

// Структура для хранения даты
typedef struct _date {
    int year;      // Год
    int month;     // Месяц
    int day;       // День
} Date;

#endif //DATE_H
```


Приложение 6. Код заголовочного файла Choreographer.h

```
#ifndef CHOREOGRAPHER_H
#define CHOREOGRAPHER_H

// Структура для хранения информации о хореографе
typedef struct _choreographer {
    char name[40];    // Имя хореографа
    char dance_style[20]; // Стил ь танца
} Choreographer;

#endif //CHOREOGRAPHER_H
```

Приложение 7. Код заголовочного файла **Perfomance.h**

```
#ifndef PERFORMANCE_H
#define PERFORMANCE_H

#include "Date.h"

// Структура для хранения информации о выступлении
typedef struct _performance {
    Date date;          // Дата выступления
    char choreographer_name[40]; // Имя хореографа
    char dancer_name[40]; // Имя танцора
    char performance_name[50]; // Название выступления
} Performance;

#endif //PERFORMANCE_H
```