Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

09.03.03 Прикладная информатика

код и наименование направления подготовки

**ОТЧЕТ**

по преддипломной практике

на кафедре математического моделирования и цифрового развития бизнес систем

Выполнил студент гр. ПИ-72 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Липская Е.С/

№ группы подпись ФИО

«07»января 2022г.

Руководитель практики от СибГУТИ доцент \_\_\_\_\_ /Полетайкин А.Н/

должность подпись ФИО

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ «07» января 2022г.

оценка

Новосибирск 2022 г.

**План-график проведения производственной практики**

Тип практики: преддипломная.

Способ практики: стационарная практика.

Форма проведения практики: дискретно по периодам проведения практик.

Выдано обучающемуся Липской Екатерине Сергеевне

ФИО студента полностью

Направление: 09.03.03 Прикладная информатика

Профиль: Прикладная информатика в экономике

Курс 5, группа ПИ-72

Тема ВКР: «Спроектировать ИС планирования контактной работы в условиях дистанционного обучения»

Содержание практики

|  |  |
| --- | --- |
| Наименование видов деятельности | Дата (начало – окончание) |
| Постановка задачи на практику, формирование плана-графика проведения практики. | 01.12.2021 |
| Работа с библиотечными фондами и ЭБС СибГУТИ, сбор и анализ материалов по теме выпускной квалификационной работы. | 6.12.2021 – 08.12.2021 |
| Построение моделей и разработка базы данных | 09.012.2021 – 10.12.2021 |
| Программно-техническая реализация информационной технологии | 10.12.2021 – 17.12.2021 |
| Анализ полученных результатов, составление отчета по практике. | 19.12.2021 - 20.04.2021 |

Согласовано:

Руководитель практики от СибГУТИ доцент \_\_\_\_\_ /Полетайкин А.Н/

должность подпись ФИО

**Задание на практику:**

На основе проведенного ранее исследования предметной области по теме ВКР выполнить программно-техническую реализацию информационной системы. Выполнить и провести построение моделей необходимых для разработки системы разработать базу данных для работы приложения разработать математическое обеспечение реализовать пользовательский интерфейс приложения выполнить программную реализацию информационной системы.

**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ………………………………………………………………………5

1.Предпроектное исследование…………………………………………………6 1.1 Обзор научной литературы по тематике исследования…………………...6

2.Разработка математического обеспечения…………………………………...8

3.Программно техническая реализация информационной системы…………10

3.1 Построение моделей разрабатываемой системы………………………….10

3.2.Разработка базы данных…………………………………………………….16

4.Создание пользовательского интерфейса……………………………………18

Заключение………………………………………………………………………22

**ВВЕДЕНИЕ**

Информационные технологии стали важной частью не только повседневной жизни людей, но и деятельности различных институтов государства и разного рода учреждений. В условиях быстрого и постоянного развития информационных технологий существует потребность в совершенствовании как материальных ресурсов и используемого программного обеспечения в отдельности, так и всей информационной архитектуры в целом.

Большое число бизнес-процессов подлежит автоматизации в настоящих условиях информатизации, в том числе и процесс составления расписания, являющийся одним из основных организационных в учебных заведениях, который и будет исследован в данной работе.

Автоматизация составления расписания – задача достаточно сложная. Эта задача относится к классу задач оптимизации.

По определению оптимизация (в математике, информатике и исследовании операций) – это задача нахождения экстремума (минимума или максимума) целевой функции в некоторой области конечномерного векторного пространства, ограниченной набором линейных и/или нелинейных равенств и/или неравенств.

Теорию и методы решения задачи оптимизации изучает математическое программирование – область математики, разрабатывающая теорию, численные методы решения многомерных задач с ограничениями. [1]

Расписание – вид календаря (то есть, упорядоченность по времени), для которого указана информация о предстоящих (планируемых или потом произошедших) событиях. Оформляется обычно в виде таблицы.

Многие учебные события организованы периодичным способом, повторяясь в одно и то же время через сутки, неделю, иное целое число суток.

Подобная организация позволяет уменьшить затраты на планирование.

Оптимизация расписания относится к исследованию операций.

Расписания служат для указания того, где и что (кто) должно находиться в указанные моменты времени.

Существует множество алгоритмов оптимизации, но для решения задачи оптимизации составления расписания наиболее подходящим является генетический алгоритм.

Генетический алгоритм – это эвристический алгоритм поиска, используемый для решения задач оптимизации и моделирования путём случайного подбора, комбинирования и вариации искомых параметров с использованием механизмов, аналогичных естественному отбору в природе

Объект исследования в этой работе – расписание занятий вебинаров для организации дистанционного обучения в вузе.

Предметом исследования является многокритериальная оптимизация –расписания вебинаров в вузе

Актуальность работы. Одной из важнейших проблем качественной организации учебного процесса в высшем учебном заведении является задача формирования оптимального учебного расписания. Эта задача является трудоемкой. Автоматизация решения позволит с экономить время и значительно сократит трудозатраты. Оптимальное расписание обеспечит равномерную загрузку студенческих групп и профессорско-преподавательского состава.

Целью настоящего исследования является поиск способов оптимизации составления оптимального расписания вебинаров.

**1.Предпроектное исследование**

**1.1** **Обзор научной литературы по тематике исследования**

Развитие информационных технологий не прекращается, возможности их применения в том числе и в сфере образования расширяются. Это подталкивает к поиску более оптимальных алгоритмов для решения насущных задач.

Задачи оптимизации расписаний являются NP-трудными, и алгоритмы, применимые для их решения, реализованные на ЭВМ, зачастую требуют неприемлемо большое время работы для задач «большой размерности». Далее рассмотрим несколько источников, в которых исследователи алгоритмов раскрывают их суть, возможное применение, положительные и отрицательные стороны.

В [3] приводятся базовые сведения о специальном разделе дискретной математики – Теории расписаний. Описаны этапы становления теории, свойства и классификации задач теории расписаний, методы их решения. На примерах классических задач представлены приемы доказательства их трудоемкости и алгоритмы решения.

В [4] рассмотрены задачи условной оптимизации, для которых целевая функция и функции ограничений являются операторами, заданными правилами/алгоритмами их вычисления, т.е. их аналитическая структура не может быть использована для организации поиска оптимального решения задачи. Эта особенность делает проблематичным применение оптимальных и эвристических методов, использующих априорно известные свойства о целевой функции и функциях ограничений для организации поиска оптимального решения.

Опишем кратко суть алгоритма. Алгоритм позволяет сформировать оптимальную особь (расписание) путем скрещивания различных особей, их мутации и селекции. При этом каждая особь должна удовлетворять всем строгим ограничениям, которые задаются до начала работы алгоритма. А оптимальная особь должна наиболее хорошо удовлетворять критериям оптимальности, которые также задаются заранее.

Рассмотрены наиболее широко используемые алгоритмы, опирающиеся на метод проб и ошибок: генетические и эволюционные алгоритмы, алгоритмы имитации отжига, муравьиные алгоритмы, алгоритмы случайного поиска (ненаправленного, направленного, направленного с самообучением).

В [5] рассматриваются классические NP-трудные задачи теории расписаний для одного и нескольких приборов с критерием минимизации максимального временного смещения (Lmax) и быстродействия (Сmax). Предлагается качественно новая схема нахождения приближенного решения. Вводится понятие метрики (расстояния) между примерами R. Идея предлагаемого подхода состоит в построении по исходному примеру задачи другого примера, для которого удается найти оптимальное или приближенное решение с минимальным расстоянием до исходного примера во введенной метрике.

Для решения данной задачи могут быть использованы как эвристические алгоритмы, так и точные методы решения, например, метод раскраски графа. Но использование точных методов решения повлечет за собой сильный рост временных затрат, при увеличении объема входной информации. В связи с этим, было принято решение использовать *эвристический метод*, а именно – *генетический алгоритм.*

**2.Разработка математического обеспечения**

Рассмотрим целевую функцию, в которой отражены интересы преподавателей и студентов:

где *r* – расписание,  – весовые коэффициенты, указывающие на приоритеты студентов и преподавателей как субъектов учебного процесса,  и – приоритеты требований студентов и преподавателей соответственно, – требования групп студентов, – преподаватели, – группы преподавателей, – предпочтения преподавателей, – приоритеты таких предпочтений, – количество требований студентов, – количество групп преподавателей, определяемое их должностями, научными степенями и учеными званиями, – количество преподавателей, – количество преподавателей в *i*-й группе, – область ограничений, – множество учебных дисциплин, групп студентов и преподавателей соответственно

В подавляющем большинстве случаев найти оптимальное решение за приемлемое время не удается, к тому же оптимальное расписание априори не известно. Тогда ограничиваются приемлемым решением, которое является допустимым и определяется по некоторому внешнему критерию.

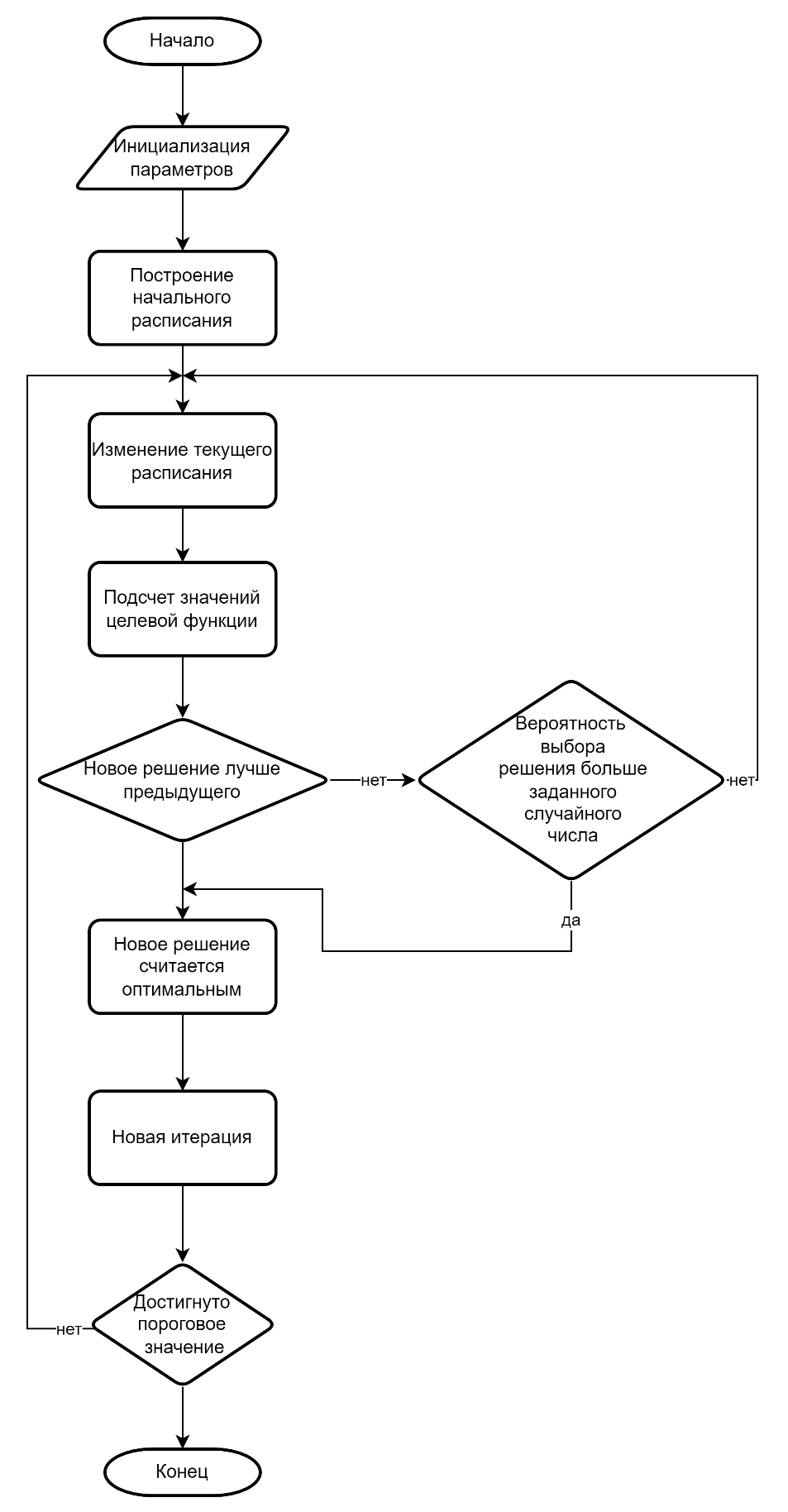


Рисунок 2.1 - Блок схема генетического алгоритма поиска оптимального расписания

Генетический алгоритм – это в первую очередь эволюционный алгоритм, другими словами, основная фишка алгоритма – скрещивание (комбинирование Идея алгоритма взята у природы. Путем перебора и отбора получается более менее оптимальная комбинация.

Алгоритм делится на три этапа:

1. Скрещивание
2. Селекция (отбор)
3. Формирования нового поколения

Если результат не устраивает, эти шаги повторяются до тех пор, пока результат не начнет удовлетворять или произойдет одно из ниже перечисленных условий:

1. Количество поколений (циклов) достигнет заранее выбранного максимума;
2. Исчерпано время на мутацию.

**3.Программно техническая реализация информационной системы**

Конечная цель разрабатываемой системы – снижение трудозатрат, сокращение затрачиваемого времени на оптимальное планирование вебинаров. Данный функционал будет реализован с помощью генетического алгоритма поиска оптимального решения.

**3.1Построение моделей разрабатываемой информационной системы**

Рассмотрим функциональную модель разрабатываемой информационной системы (рисунок 3.1)

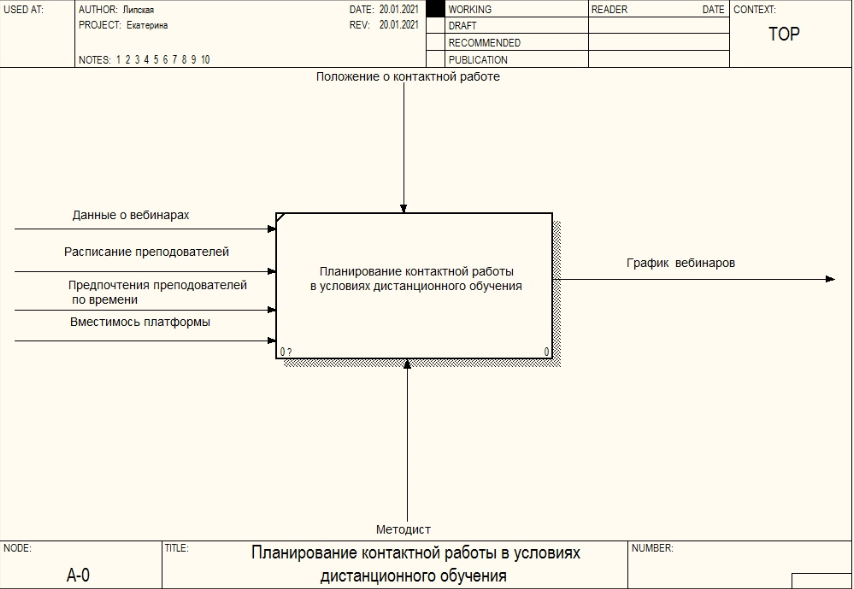


Рисунок 3.1- Модель «черный ящик» в нотации IDEF0

Как видим входными данными у нас являются данные о вебинарах –это список дисциплин, по которым необходимо планировать вебинары, а также расписание и предпочтение преподавателей и вместимость платформы. На выходе получаем график проведения вебинаров. При этом регламентирующим документом будет являться положение о контактной работе, а действующим лицом – методист.

Проведем декомпозицию нашего процесса (рис 3.1-3.3).

Для получения оптимального графика вебинаров необходимо определить длительность каждого из них, затем определить оптимальное время для проведения и затем уже составить сам график вебинаров

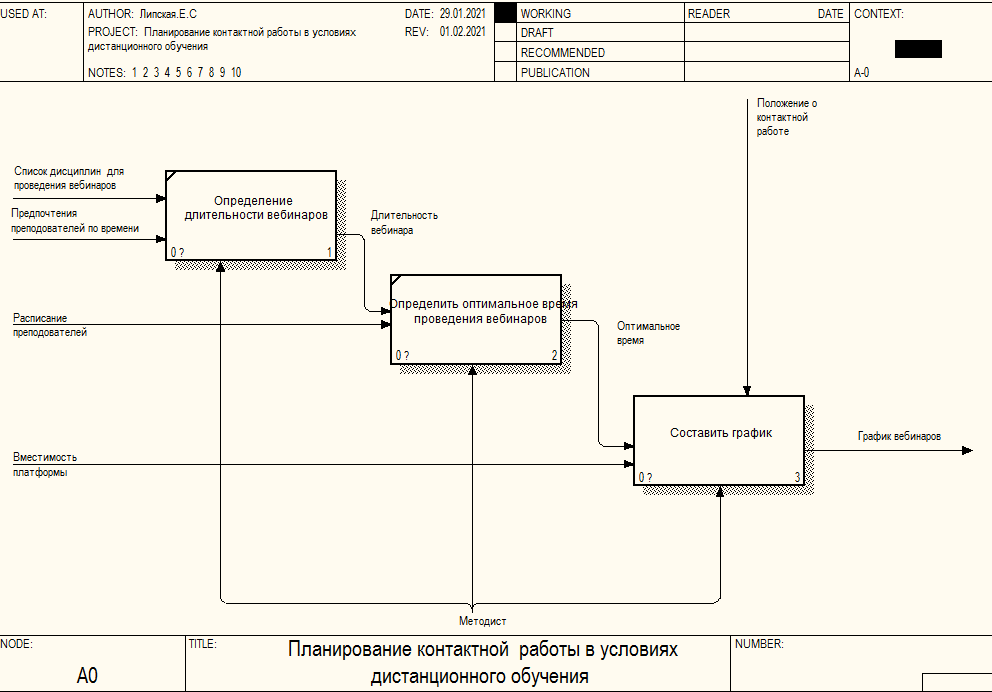


Рисунок 3.2 - Декомпозиция диаграммы А0

При этом процесс составления графика включает в себя распределение вебинаров по пространственно-временна сетке с учетом всех имеющихся ограничений и уже только после этого формируется график.

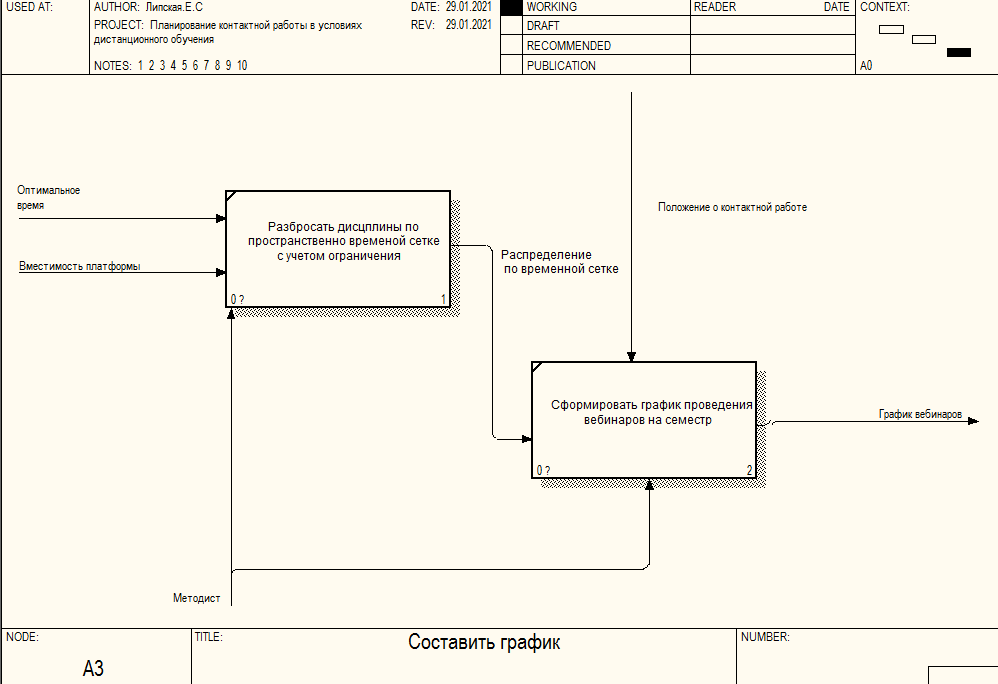


Рисунок 3.3 - Декомпозиция диаграммы А3

Взаимодействие пользователя с приложением отражено на диаграмме вариантов использования на рисунке 3.4

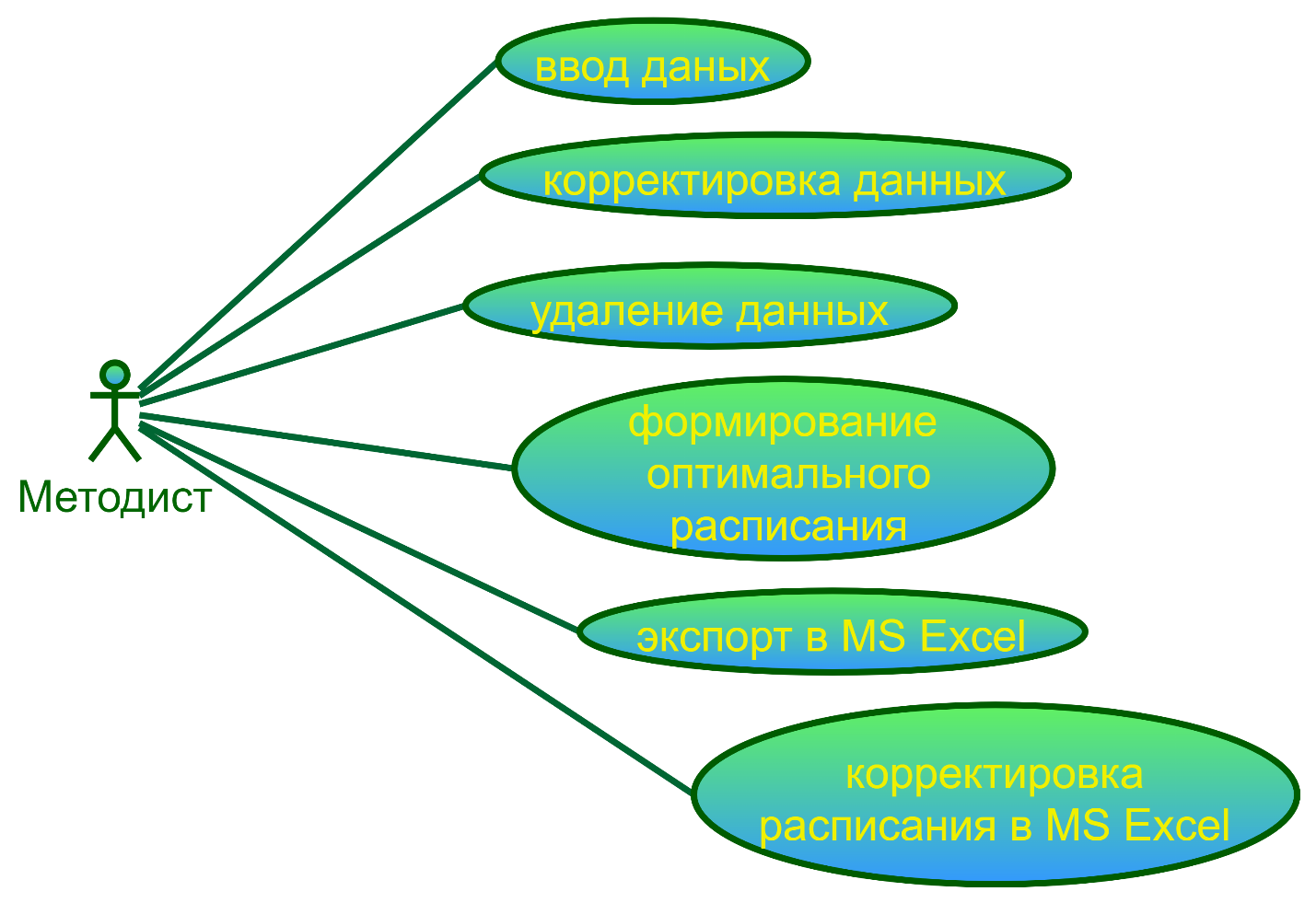


Рисунок 3.4 - Диаграмма вариантов использования приложения

Структура разрабатываемой информационной системы показана на рисунке 3.5 ниже

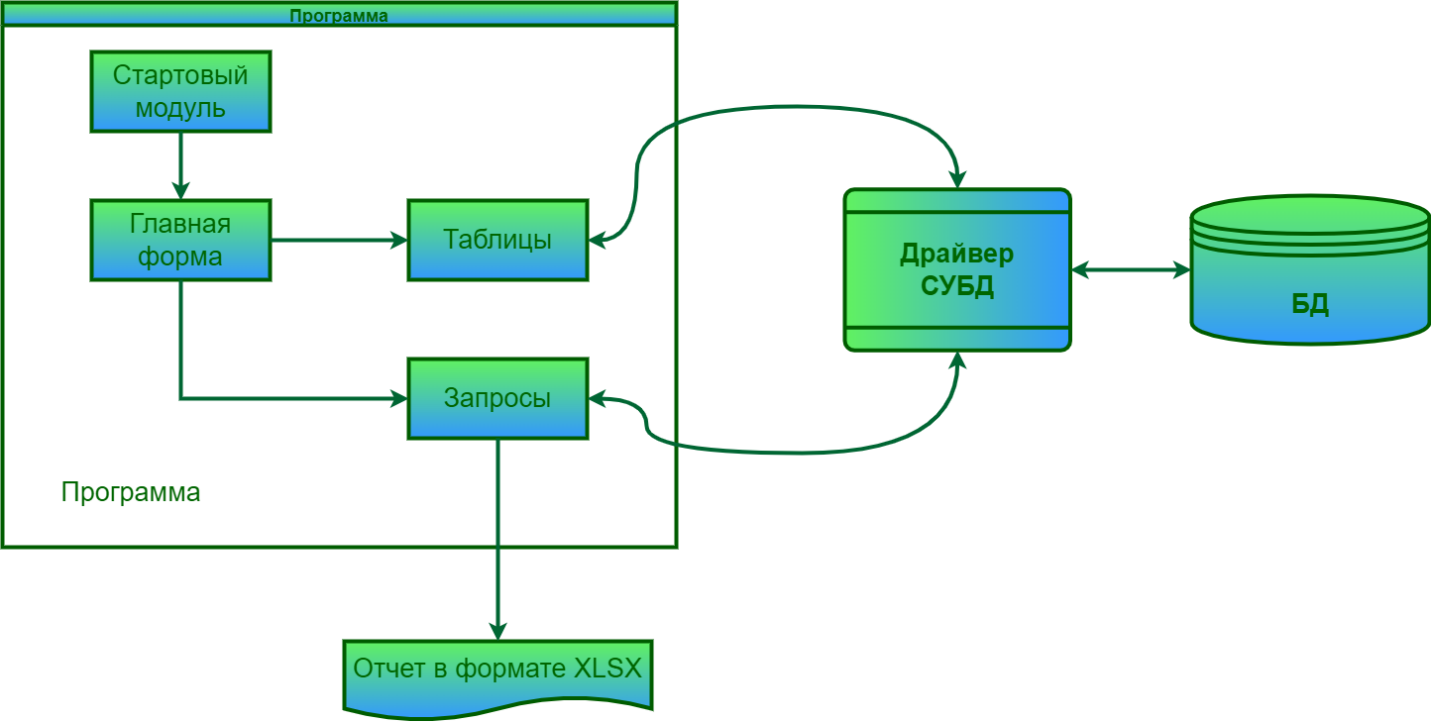


Рисунок 3.5 - Структура приложения

В стартовом модуле инициализируется приложение. На главной форме расположено меню для выбора действия:

* ввода данных;
* построения оптимального плана.

Таблицы – сущности базы данных, в которых хранятся исходные данные для построения оптимального расписания

База данных – внешнее хранилище, взаимодействие с которым возможно через драйвер СУБД. В нашем случае используется встраиваемая СУБД MS SQL Server.

Результат оптимизации нужно сохранить в табличном виде во внешнем файле с возможностью его корректировать. Для этого используется формат файла MS Excel.

В основе работы каждой подсистемы лежит событийная модель построения графического интерфейса пользователя, использование графических компонентов – грида, вкладок, кнопок, использование готовых библиотек для взаимодействия с MS Excel.

На рисунке 3.6 ниже представлена диаграмма классов

В настоящем решении связей между классами нет, так как не используется наследование классов, часть классов по своей сути – экранные формы для ввода данных. Часть классов не имеют методов, их роль – в хранении данных определенной структуры. А класс Solver содержат в себе методы для поиска оптимального решения и оперирует данными, которые извлекает уже из самой базы данных. Извлекаемые данные попадают в массивы классов-структур. Работать с массивами в памяти проще для поиска оптимального решения.

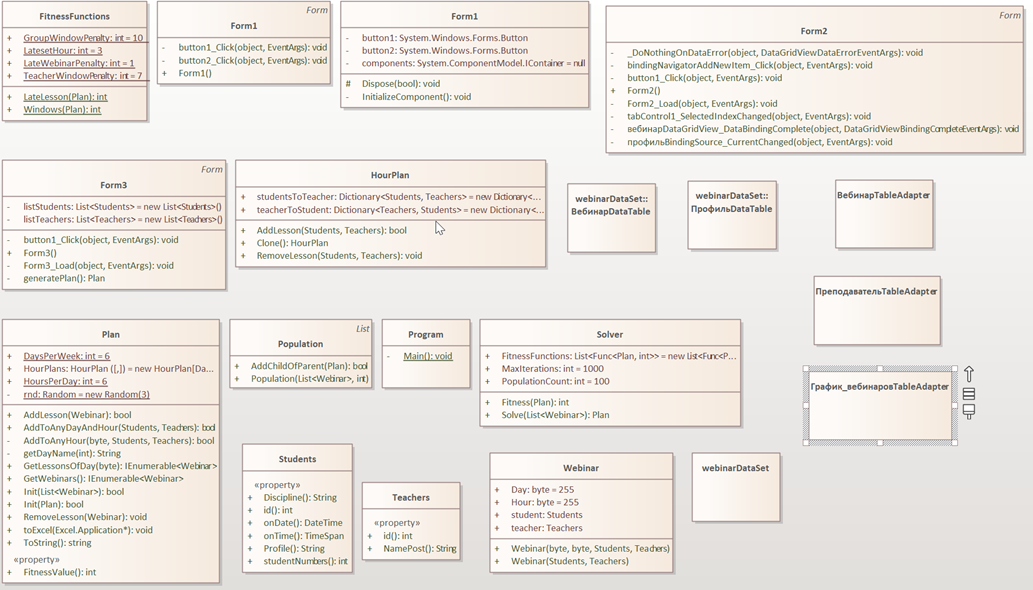


Рисунок 3.6 - Диаграмма классов

Диаграмма классов является ключевым элементом в объектно-ориентированном моделировании. На диаграмме классы представлены в рамках, содержащих три компонента:

В верхней части написано имя класса. Имя класса выравнивается по центру и пишется полужирным шрифтом. Имена классов начинаются с заглавной буквы. Если класс абстрактный — то его имя пишется полужирным курсивом.

Посередине располагаются поля (атрибуты) класса. Они выровнены по левому краю.

Нижняя часть содержит методы класса. Они также выровнены по левому краю.

Для задания видимости членов класса (то есть – любым атрибутам или методам), эти обозначения должны быть размещены перед именем участника:

+ Публичный (Public) – виден внутри и за пределами класса;

- Приватный (Private) – виден только внутри класса;

# Защищённый (Protected) – виден внутри класса и классах-наследниках;

Диаграмма классов наглядно отражает структуру всего приложения. Роль каждого класса четко разграничена.

В базовом классе можно реализовать общий функционал, который используется во всех классах-наследниках.

Таким образом, можно сделать выводы, что ООП достаточно удобно и практично при разработке больших и сложных проектов, так как позволяет избегать дублирование кода; к тому же программы, разработанные в стиле ООП проще в сопровождении и добавлении нового функционала в программу

**3.2 Разработка базы данных**

Для разрабатываемой информационной системы потребуется база данных, предназначенная для хранения информации.

Разработку логической модели следует начать с выделения сущностей и связей между ними. Предполагается, что таких сущностей будет 7, их названия и атрибуты приведены в таблице 3.2.1

Таблица 3.2.1 – Сущности логической модели данных

|  |  |  |  |
| --- | --- | --- | --- |
| № | Сущность | Атрибуты | Описание |
| 1 | Преподаватель | **Код\_преподователя**  ФИО | Данные о преподавателях |
| 2 | Расписание  преподавателя | **Код\_расписание\_преподавателя**  Занятое время | Данные о расписании  преподавателей |
| 3 | Предпочтения  преподователя | **Код \_предпочтения преподователя**  Время начало  Время конец | Данные о предпочтениях преподавателей |
| 4 | Дисциплина | **Код \_дисциплина**  Наименование | Данные о дисциплинах |
| 5 | Студент | Код\_студент  Количество | Данные о студентах |
| 6 | Вебинар | **Код\_вебинар**  Длительность | Данные о вебинаре |
| 7 | График вебинаров | **Код\_график вебинаров**  Время  Дата | Данные о графике вебинаров |

Общий вид логической модели базы данных показан на рисунке 3.2.1

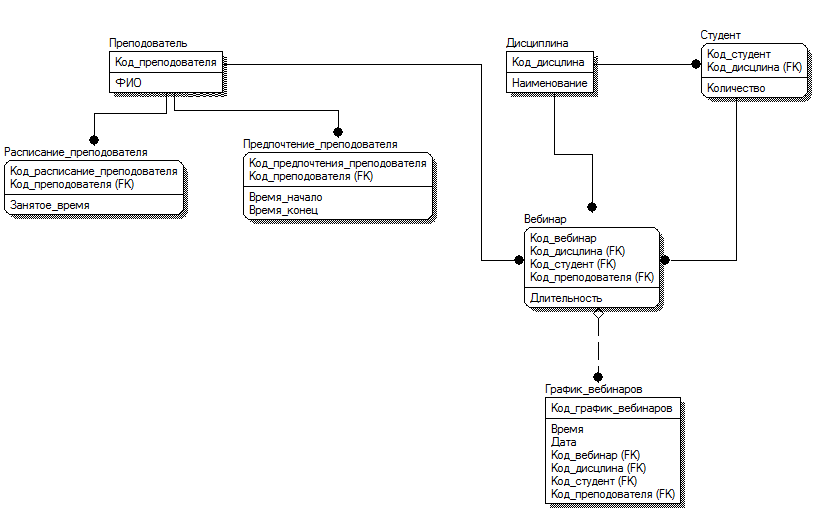


Рисунок 3.2.1 - Логическая ER-диаграмма базы данных

На основе логической модели данных строится физическая модель, общий вид которой представлен на рисунке 3.2.2

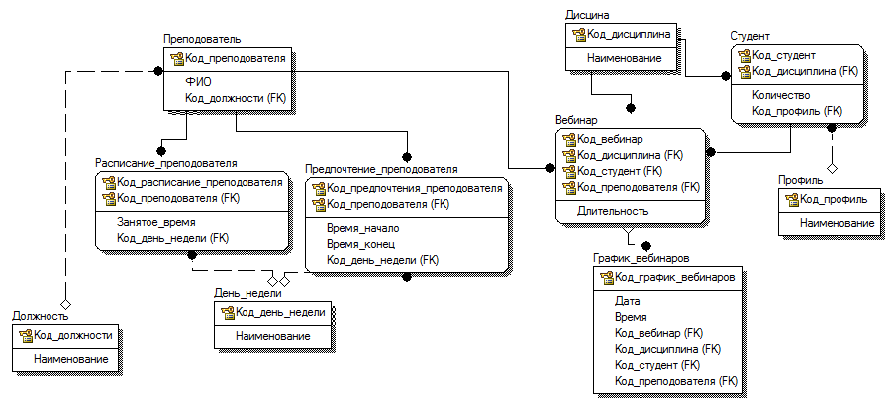


Рисунок 3.2.2 - Физическая ER-диаграмма базы данных

Описание физической модели представлено в таблице 3.2.1

Таблица 3.2.1 – Сущности физической модели данных

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Сущность** | **Атрибуты** | **Описание** |
| 1 | Должность | **Код\_должность**  Наименование | Данные о должностях |
| 2 | Преподаватель | **Код\_предаватель**  ФИО | Данные о преподавателях |
| 3 | Расписание преподавателя | **Код\_расписания\_преподавателя**  Занятое время | Данные о расписании преподавателей |
| 4 | Предпочтения преподавателя | **Код\_предпочтения\_преподавателя**  Время начало  Время конец | Данные о предпочтениях преподавателей |
| 5 | День недели | **Код\_день\_недели**  Наименование | Данных о днях недели |
| 6 | Дисциплина | **Код\_дисциплина**  Наименование | Данные о дисциплинах |
| 7 | Студент | **Код\_студент**  Количество | Данные о студентах |
| 8 | Профиль | **Код\_профиль**  Код профиля  Наименование | Данные о профилях |
| 9 | Вебинар | **Код\_вебинар**  Количество заявок  Длительность | Данные о вебинарах |
| 10 | График вебинаров | **Код\_график\_вебинаров**  Время  Дата | Данные о графике вебинаров |

В таблице 3.2.1 представлено краткое описание разработанных таблиц

|  |  |  |
| --- | --- | --- |
| **№** | Имя таблицы | Описание |
| 1 | Должность | Наименование должнности |
| 2 | Преподаватель | Наименование преподавателей с указанием ФИО и должности |
| 3 | Расписание преподавателя | Занятое время преподавателей. |
| 4 | Предпочтения преподавателя | Данные о предпочтениях преподавателей по дням недели и времени проведения вебинаров |
| 5 | День недели | Наименование дней недели |
| 6 | Дисциплина | Наименование дисциплины |
| 7 | Студент | Список количества студентов, обучающихся по заданному профилю указанным дисциплинам |
| 8 | Профиль | Наименование профилей студентов. |
| 9 | Вебинар | Список вебинаров содержит код преподавателя и длительность |
| 10 | График вебинаров | График вебинаров содержит код дисциплины, число заявок, дату и время проведения. По коду преподавателя определяется наименование вебинара, так как вебинары закреплены за преподавателями. |

**4.Создание пользовательского интерфейса**

При запуске разработанного приложения пользователь попадает на главное окно приложения.

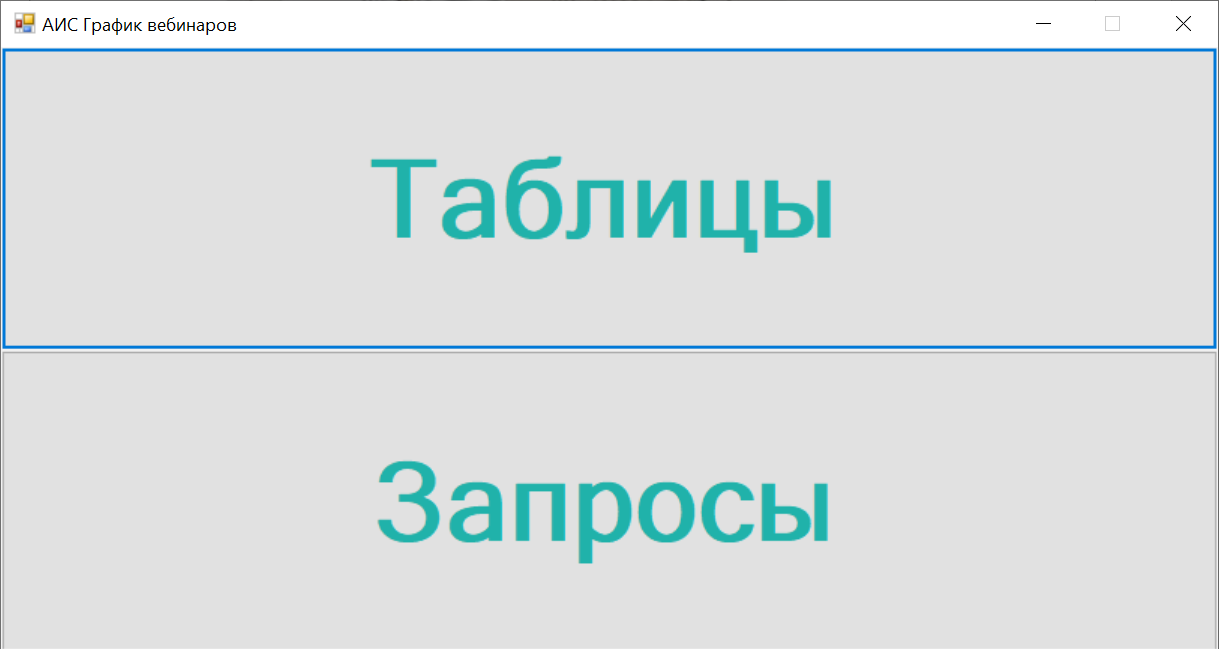


Рисунок 4.1

Для ввода и редактирования данных в БД пользователю необходимо нажать на кнопку «Таблицы», после чего откроется окно с вкладками рисунок 4.2 Ввод данных.

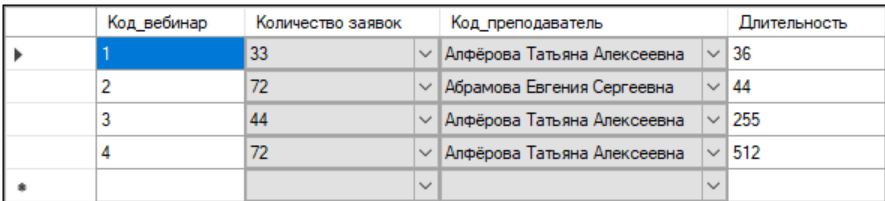


Рисунок 4.2 - Ввод данных

Далее пользователю необходимо сохранить данные На рисунке 4.5 представлено сообщение об успешном сохранении данных в базе после нажатия кнопки «Сохранить». После успешного сохранения данные в БД обновляются.

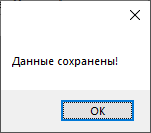


Рисунок 4.5 - Сообщение об успешном сохранении данных

«График\_вебинаров» в окне «Запросы», которое вызывается из главной формы. В данном окне формируется график вебинаров. автоматически на основе ранее введённых данных после открытия формы «Запросы».

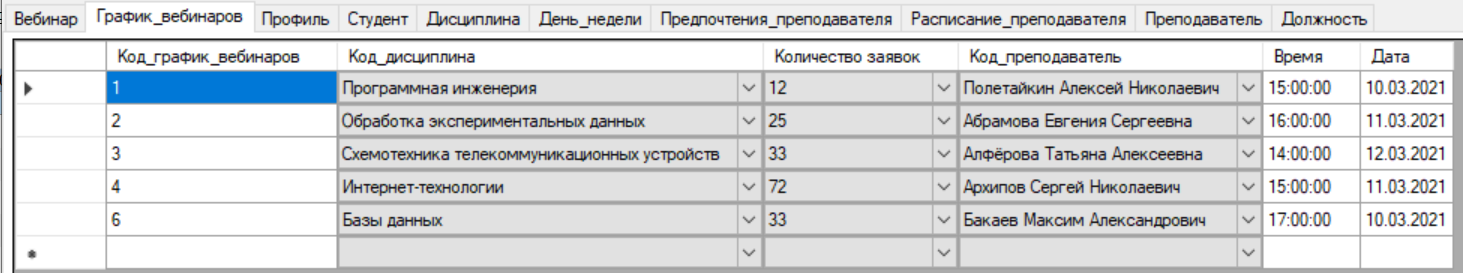


Рисунок 4.6 - Построение оптимального расписания

На рисунке 4.7 представлен Экспорт оптимального расписания в MS Excel. После нажатия кнопки «Оптимизация ® Excel» с возможность корректировки и сохранения на компьютере пользователя.

Извлечение исходных данных перед оптимизацией производится запросом:

SELECT a.Код\_студент

,x.Количество

,y.Наименование\_дисциплины

,x.Наимеование\_профиля

,a.Время

,a.Дата

,y.[ФИО]

,y.Код\_преподователь

,y.НаименованиеДолжности

FROM dbo.[График\_вебинаров] a

INNER JOIN (SELECT [Код\_студент], [Количество], p.Наименование [Наимеование\_профиля]

FROM dbo.Студент b

LEFT OUTER JOIN dbo.Профиль p ON p.[Код\_профиль] = b.[Код\_профиль]) x ON a.Код\_студент = x.Код\_студент

INNER JOIN (SELECT a.[Код\_преподователь]

, a.[ФИО]

, d.Наименование [НаименованиеДолжности]

, p.[Наименование] [Наименование\_дисциплины]

FROM dbo.Преподаватель a

INNER JOIN dbo.Должность d ON d.Код\_должность = a.Код\_должность

INNER JOIN dbo.Дисциплина p ON p.Код\_дисциплина = a.Код\_дисциплина) y ON y.Код\_преподователь = a.Код\_преподаватель

ORDER BY y.[ФИО];

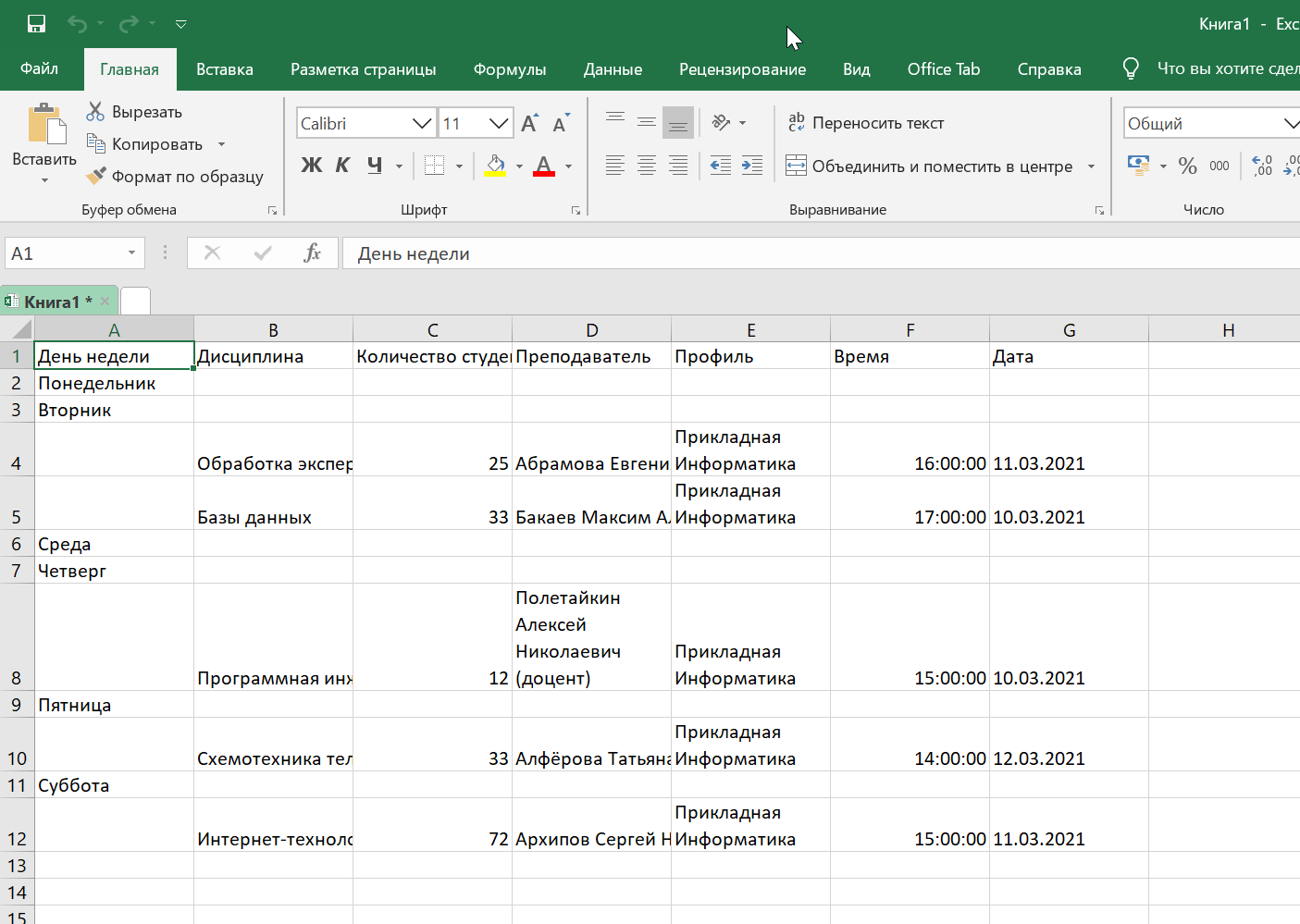


Рисунок 4.7 - Экспорт оптимального расписания в MS Excel

Полученное расписание оптимизировано в соответствии с заданными исходными данными. Достигнут эффект повышения качества учебного процесса.

**Заключение**

В разработанном проекте был реализован генетический алгоритм на языке программирования C# Изучена суть генетического алгоритма, его преимущества и недостатки в применении для составления оптимальных расписаний. Преимуществом является высокая скорость работы алгоритма. Недостатком является неидеальное по оптимальности составление алгоритма, а близкое к идеальному. Степень оптимальности зависит от конечного числа итераций алгоритма. Чем оно больше, тем выше точность, но больше и затраты по времени.

Программа позволяет пользователю вводить и редактировать исходные данные и условия оптимизации. После завершения ввода исходные данные можно сохранить в базе данных. При следующем запуске приложение сохраненные данные откроются автоматически.

Для получения оптимального расписания достаточно нажать соответствующую кнопку в интерфейсе приложения. Оптимальное расписание будет выгружено в MS Excel (наличие которого в системе обязательно). В MS Excel можно просмотреть и подправить расписание

Были получены навыки разработки графического интерфейса пользователя, изучены средства работы с локальной базой данных MS SQL.