Уральский федеральный университет имени первого Президента

России Б.Н.Ельцина

Институт радиоэлектроники и информационных технологий – радиотехнический факультет

Департамент информационных технологий и автоматики

**ОТЧЕТ**

по практическим занятиям

по дисциплине “Теория управления”

Преподаватель: доцент Осипова И. А.

Студент: Дубровин Р. В.

Группа: РИ-320942

Екатеринбург - 2025

**СОДЕРЖАНИЕ**

[**Практическое работа №1** 3](#_Toc195823195)

[**Практическое работа №2** 7](#_Toc195823196)

[**Практическая работа №3** 10](#_Toc195823197)

[**Практическое работа №4** 16](#_Toc195823198)

## Практическая работа №1

**«Анализ и систематизация учебных дисциплин»**

### 1. Цель работы

Сформировать полное представление о пройденных дисциплинах и распределить их по логическим группам, чтобы понять преобладающую образовательную траекторию.

### 2. Исходные данные

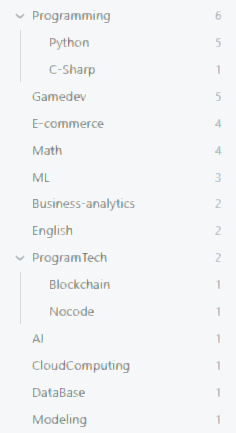
Из электронной зачетной книжки перенесены все дисциплины по семестрам (семестры 1–6), вместе с тем отметив повторяющиеся курсы:

| Семестр | Дисциплины |
| --- | --- |
| 1 | Математика базовый уровень; Проектная деятельность, сем. 1; Английский B1 p.1; Английский B1 p.2; Векторный анализ; Доп. главы математики; Основы C#; ПП – образовательная игра |
| 2 | Web-технологии; Основы работы с MySQL; Основы теории вероятностей; Прохождение курса Skillbox.Python; Работа с Яндекс.Игры (2D игра) |
| 3 | Электронная коммерция (разработка продукта и защита перед инвесторами); Алгоритмы и анализ сложности Python; Промт-инжиниринг; Работа с моделью для сегментации спутниковых изображений; Продвижение в соцсетях |
| 4 | Курс бизнес-аналитика Альфа-банк; Программирование нейронных сетей 1; Тактический трёхмерный бой; Технологии распределённого реестра (smart-контракты); Agile-подходы |
| 5 | Теория управления; Бизнес-аналитик no-code; Виртуализация и облачные технологии; Гейм-дизайн Unity; Моделирование систем; Программирование нейронных сетей 2; Продвижение в соцсетях |
| 6 | Производственная практика (или «Теория управления», если практика совпадает?) |

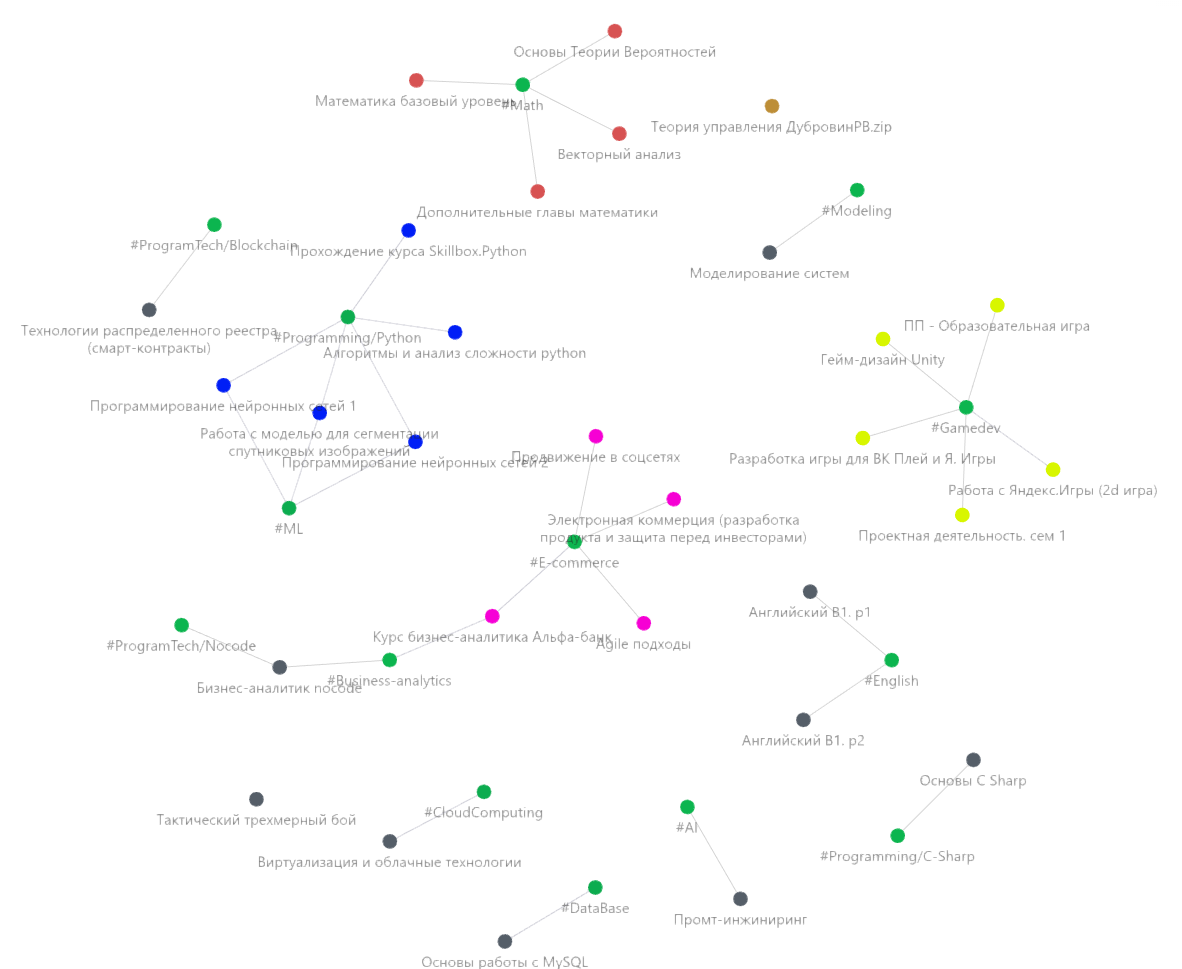
### 3. Методика

1. **Сбор дисциплин**  
    Все названия дисциплин скопированы из своей электронной зачетки в Obsidian.
2. **Группировка по темам**  
    На основе содержания курса и практических задач выделены следующие группы:
   * + - **Программирование**
         * Python (Программирование)
         * C# (Программирование)
       - **Разработка игр**
       - **Электронная коммерция**
       - **Математика**
       - **Машинное обучение**
       - **Бизнес-аналитика**
       - **Английский**
       - **Программные технологии**
         * Блокчейн (Программные технологии)
         * No-code (Программные технологии)
       - **Искусственный интеллект**
       - **Облачные вычисления**
       - **Базы данных**
       - **Моделирование**
3. **Разметка в Obsidian**

Для каждой дисциплины создана заметка с метаданными: #Math, #ProgrammingPython, #AI, #Database, #E-comercce, #GameDev, #Tech.



В теле заметки перечислены ключевые темы и ключевые проекты/задачи, которые выполнялись по курсу.



1. **Анализ количества**

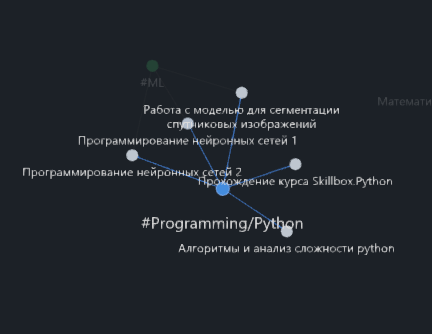
Подсчитано число дисциплин в каждой группе.

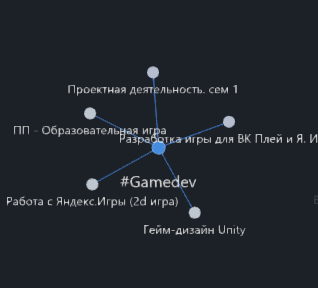
### 4. Результаты

| Группа | Кол-во дисциплин |
| --- | --- |
| Математика | 4 |
| Программирование на Python | 5 |
| Машинное обучение | 3 |
| Бизнес-аналитика | 2 |
| Электронная коммерция | 4 |
| Разработка игр | 5 |
| Виртуализация & блокчейн | 2 |

### 5. Выводы

* **Наибольшее представительство** показали группы **Программирование на Python** и ****Разработка игр**,** по 5 дисциплин каждая.





* В учебе я сочетал интерес к *Программированию на Python* и *Разработке игр*, что подтверждается равным представлением этих направлений (по 5 курсов каждый).
* *Математика* (4 курса) и *Электронная коммерция* (4 курса) дополняют профиль.
* Наименее представлены *Виртуализация* и *Компьютерные технологии* (по 1 курсу), что указывает на слабую специализацию в этих областях на текущем этапе.

**Практическая работа выполнена**, получена структурированная карта учебных курсов и определены приоритетные направления обучения.

**Практическое работа №2**

**«Построение минимального остовного дерева (МОД) для групп дисциплин»**

### 1. Цель работы

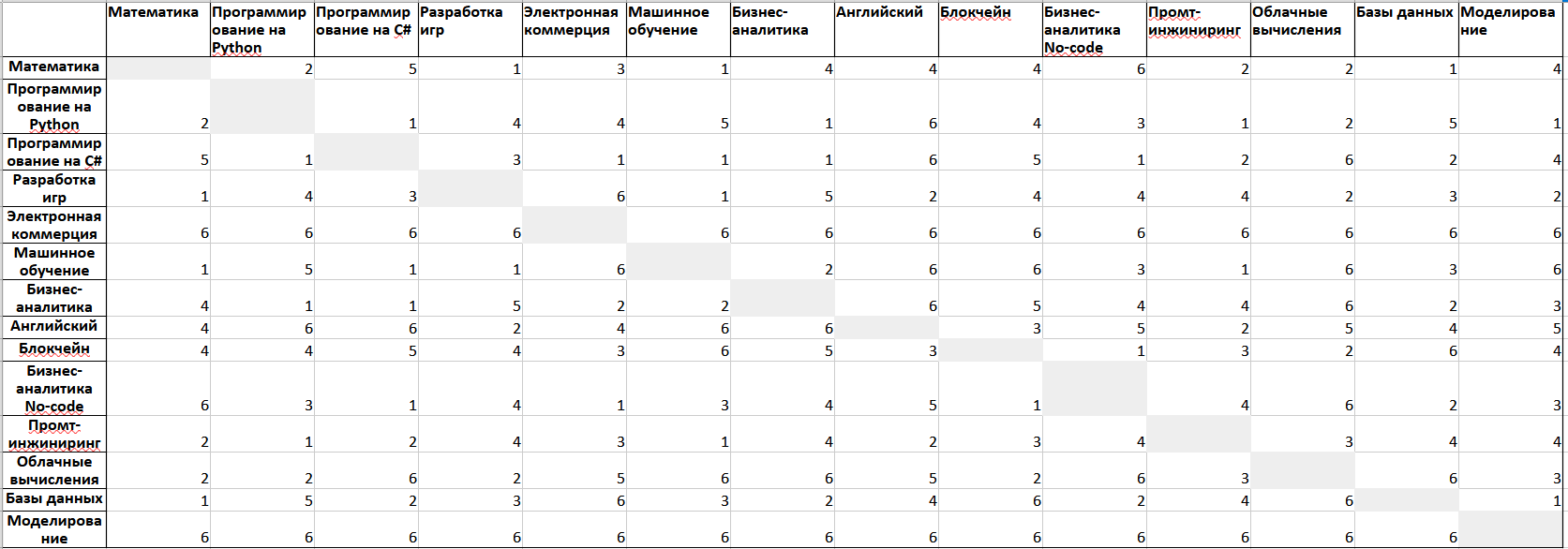
Построить связное дерево, отражающее взаимосвязи между основными тематическими группами дисциплин на основе учебного плана. Определить силу связей между группами и визуализировать граф в Gephi.

### 2. Исходные данные

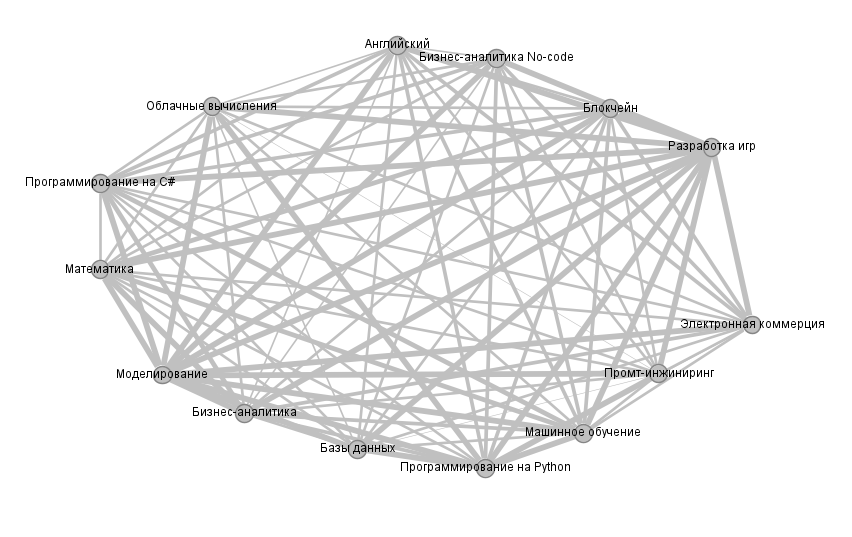
* **Узлы**: 14 тематических групп дисциплин:
  1. Математика
  2. Программирование на Python
  3. Программирование на C#
  4. Электронная коммерция
  5. Разработка игр
  6. Машинное обучение
  7. Бизнес-аналитика
  8. Английский
  9. Блокчейн
  10. Бизнес-аналитика No-code
  11. Промт-инжиниринг
  12. Облачные вычисления
  13. Базы данных
  14. Моделирование
* **Рёбра**: полный неориентированный граф, каждая пара групп соединена ребром с целочисленным весом 1…6. Вес отображает «длину» — чем меньше значение, тем сильнее связь (большая важность\_влияния).

### 3. Этапы выполнения

* 1. Построена таблица всех парных связей между группами.
  2. Каждой паре присвоен вес (целое число от 1 до 6), отражающий степень влияния одной группы на другую (1 — сильная связь, 6 — слабая).



* 1. На основе таблицы составлен edges.csv, содержащий все 91 возможную связь.
  2. Также составлен nodes.csv, включающий 14 групп с соответствующими ID и названиями.
  3. Данные импортированы в Gephi:
     + Узлы — как Nodes table
     + Рёбра — как Edges table, тип — undirected
  4. Визуализирован связный граф.



### 3. Результат

Получен взвешенный неориентированный граф, отражающий связи между всеми тематическими направлениями. Толщина рёбер соответствует силе связи. Некоторые связи были заданы вручную (например, сильная связь между Python и ML), другие определены на основе логики обучения.

### 4. Вывод

Построенный граф позволяет наглядно проследить, какие направления учебного плана наиболее тесно взаимосвязаны. Выделяются плотные кластеры (Python ↔ GameDev, ML ↔ Базы данных), а также более слабо связанные области (например, Электронная коммерция). Этот граф станет основой для построения минимального остовного дерева в следующей работе.

**Практическая работа № 2 полностью выполнена.**

## Практическая работа №3

**Построение минимального остовного дерева (МОД)**

### Цель работы

На основе связного дерева из предыдущей практики построить минимальное остовное дерево (МОД), включающее все группы дисциплин и минимизирующее суммарную длину связей (веса рёбер). Использовать алгоритм Прима для построения дерева.

### Этапы выполнения

1. Использован ранее построенный граф.



1. Применён алгоритм Прима

### **Шаг 1: Инициализация**

| **Параметр** | **Значение** |
| --- | --- |
| Начальная вершина | 1 (Математика) |
| Множество вершин | {1} |
| Ребра в **<D>** | (0, 1, 0) — начальная условная запись |
| Комментарий | Начало построения дерева. |

### **Шаг 2: Первая итерация**

| **Параметр** | **Значение** |
| --- | --- |
| Рассматриваемые ребра | Все ребра из вершины 1: **1-2 (3)**, **1-3 (3)**, **1-5 (3)**, **1-8 (3)**, ..., **1-14 (6)** |
| Выбранное ребро | **1-2 (3)** (минимальный вес среди ребер вершины 1) |
| Множество вершин | {1, 2} |
| Комментарий | Ребро **1-2** добавлено в **<D>**. |

### **Шаг 3: Вторая итерация**

| **Параметр** | **Значение** |
| --- | --- |
| Рассматриваемые ребра | Ребра из вершин {1, 2}: **2-5 (2)**, **1-3 (3)**, **2-3 (3)**, **2-7 (4)**, ..., **2-14 (6)** |
| Выбранное ребро | **2-5 (2)** (минимальный вес) |
| Множество вершин | {1, 2, 5} |
| Комментарий | Ребро **2-5** добавлено в **<D>**. |

### **Шаг 4: Третья итерация**

| **Параметр** | **Значение** |
| --- | --- |
| Рассматриваемые ребра | Ребра из вершин {1, 2, 5}: **5-11 (3)**, **5-6 (3)**, **5-12 (3)**, **5-13 (3)**, ... |
| Выбранное ребро | **5-11 (3)** (минимальный вес) |
| Множество вершин | {1, 2, 5, 11} |
| Комментарий | Ребро **5-11** добавлено в **<D>**. |

### **Шаг 5: Четвертая итерация**

| **Параметр** | **Значение** |
| --- | --- |
| Рассматриваемые ребра | Ребра из вершин {1, 2, 5, 11}: **11-12 (1)**, **11-13 (1)**, **11-6 (3)**, ... |
| Выбранное ребро | **11-12 (1)** (минимальный вес) |
| Множество вершин | {1, 2, 5, 11, 12} |
| Комментарий | Ребро **11-12** добавлено в **<D>**. |

### **Шаг 6: Пятая итерация**

| **Параметр** | **Значение** |
| --- | --- |
| Рассматриваемые ребра | Ребра из вершин {1, 2, 5, 11, 12}: **11-13 (1)**, **12-13 (2)**, **12-8 (2)**, ... |
| Выбранное ребро | **11-13 (1)** (минимальный вес) |
| Множество вершин | {1, 2, 5, 11, 12, 13} |
| Комментарий | Ребро **11-13** добавлено в **<D>**. |

### **Шаг 7: Шестая итерация**

| **Параметр** | **Значение** |
| --- | --- |
| Рассматриваемые ребра | Ребра из вершин {1, 2, 5, 11, 12, 13}: **12-8 (2)**, **13-6 (3)**, **12-6 (3)**, ... |
| Выбранное ребро | **12-8 (2)** (минимальный вес) |
| Множество вершин | {1, 2, 5, 8, 11, 12, 13} |
| Комментарий | Ребро **12-8** добавлено в **<D>**. |

### **Шаг 8: Седьмая итерация**

| **Параметр** | **Значение** |
| --- | --- |
| Рассматриваемые ребра | Ребра из вершин {1, 2, 5, 8, 11, 12, 13}: **8-7 (2)**, **8-10 (2)**, **8-9 (3)**, ... |
| Выбранное ребро | **8-7 (2)** (минимальный вес) |
| Множество вершин | {1, 2, 5, 7, 8, 11, 12, 13} |
| Комментарий | Ребро **8-7** добавлено в **<D>**. |

### **Шаг 9: Восьмая итерация**

| **Параметр** | **Значение** |
| --- | --- |
| Рассматриваемые ребра | Ребра из вершин {1, 2, 5, 7, 8, 11, 12, 13}: **7-10 (2)**, **7-9 (3)**, **7-14 (6)**, ... |
| Выбранное ребро | **7-10 (2)** (минимальный вес) |
| Множество вершин | {1, 2, 5, 7, 8, 10, 11, 12, 13} |
| Комментарий | Ребро **7-10** добавлено в **<D>**. |

### **Шаг 10: Девятая итерация**

| **Параметр** | **Значение** |
| --- | --- |
| Рассматриваемые ребра | Ребра из вершин {1, 2, 5, 7, 8, 10, 11, 12, 13}: **10-9 (1)**, **10-14 (6)**, ... |
| Выбранное ребро | **10-9 (1)** (минимальный вес) |
| Множество вершин | {1, 2, 5, 7, 8, 9, 10, 11, 12, 13} |
| Комментарий | Ребро **10-9** добавлено в **<D>**. |

### **Шаг 11: Десятая итерация**

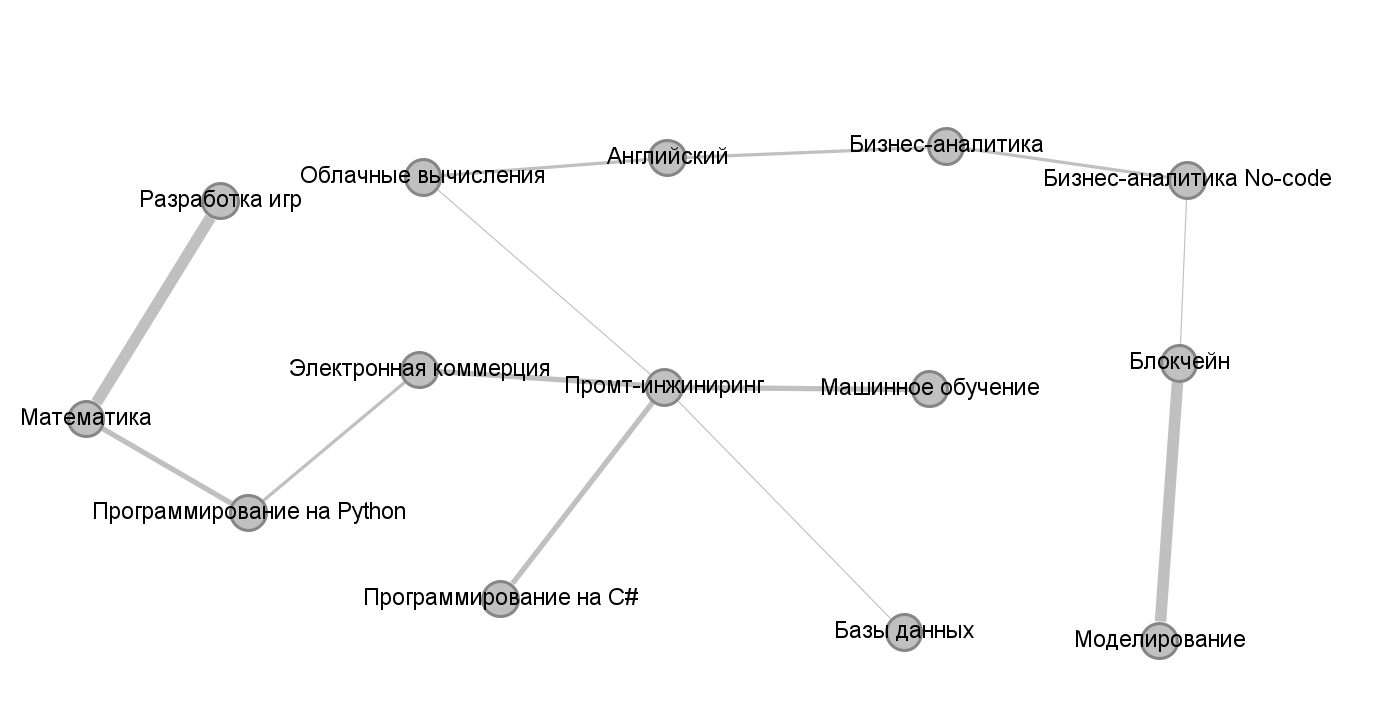
| **Параметр** | **Значение** |
| --- | --- |
| Рассматриваемые ребра | Ребра из вершин {1, 2, 5, 7, 8, 9, 10, 11, 12, 13}: **9-14 (6)**, **9-...** |
| Выбранное ребро | **9-14 (6)** (единственное доступное для вершины 14) |
| Множество вершин | {1, 2, 5, 7, 8, 9, 10, 11, 12, 13, 14} |
| Комментарий | Ребро **9-14** добавлено в **<D>**. |

1. Построен подграф, включающий все 14 узлов и 13 рёбер.
2. Полученный результат экспортирован в edges\_min.csv.

### Результат

| **Ребра** | **Вес** |
| --- | --- |
| 1-2 | 3 |
| 2-5 | 2 |
| 5-11 | 3 |
| 11-12 | 1 |
| 11-13 | 1 |
| 12-8 | 2 |
| 8-7 | 2 |
| 7-10 | 2 |
| 10-9 | 1 |
| 9-14 | 6 |
| 3-11 | 3 |
| 6-11 | 3 |
| 4-1 | 6 |

**Сумма: 35**



### Вывод

Построенное МОД отражает структуру наиболее значимых взаимосвязей между всеми направлениями. Оно избавлено от избыточных и слабых связей, позволяя сфокусироваться на ядре учебной траектории. Дерево показывает, какие знания и темы являются ключевыми узлами в образовательной программе.

## Практическая работа №4

**Сетевое планирование последовательности изучения дисциплинарных групп**

### Цель работы

Используя методы сетевого планирования, определить оптимальную последовательность изучения тематических групп дисциплин для освоения выбранных профессиональных направлений. Построить сетевой график и рассчитать критический путь, определив наиболее значимые (критические) группы и резервы времени.

### Исходные данные

Мы ранее составляли группы из дисциплин (курсов), потому за продолжительность изучения возьмем именно количество курсов.

#### Группы дисциплин (работы)

| Код | Название группы дисциплин | Условная продолжительность (курс) |
| --- | --- | --- |
| A | Математика | 4 |
| B | Программирование на Python | 5 |
| C | Программирование на C# | 4 |
| D | Разработка игр | 6 |
| E | Машинное обучение | 5 |
| F | Базы данных | 3 |
| G | Промт-инжиниринг | 2 |
| H | Бизнес-аналитика | 4 |
| I | Бизнес-аналитика No-code | 3 |
| J | Блокчейн | 4 |
| K | Облачные вычисления | 3 |
| L | Моделирование | 4 |
| M | Электронная коммерция | 2 |
| N | Английский | 2 |

#### Зависимости между группами

(формат: «работа зависит от»)

* + **A (Математика)** ← нет зависимостей (основная дисциплина)
  + **B (Python)** ← A (Математика),
  + **C (C#)** ← A (Математика),
  + **E (Машинное обучение)** ← A (Математика),
  + **D (Разработка игр)** ← C (C#), B (Python)
  + **G (Промт-инжиниринг)** ← E (Машинное обучение)
  + **F (Базы данных)** ← A (Математика)
  + **K (Облачные вычисления)** ← F (Базы данных)
  + **J (Блокчейн)** ← F (Базы данных)
  + **H (Бизнес-аналитика)** ← A (Математика)
  + **I (No-code)** ← H (Бизнес-аналитика)
  + **L (Моделирование)** ← нет зависимостей (параллельно)
  + **M (Электронная коммерция)** ← нет зависимостей (параллельно)
  + **N (Английский)** — нет зависимостей (параллельно)

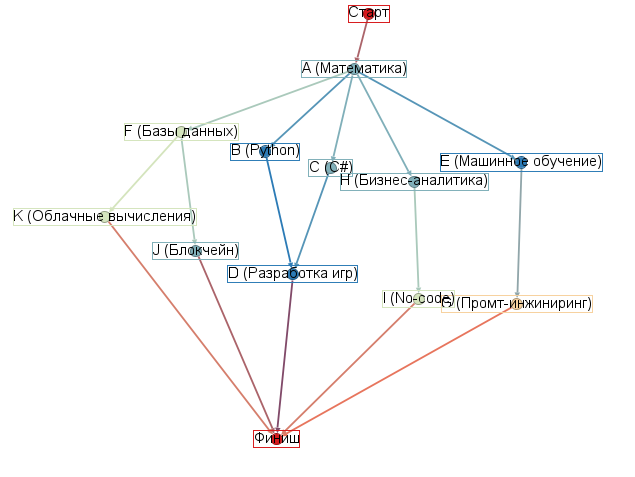
### *Методология*

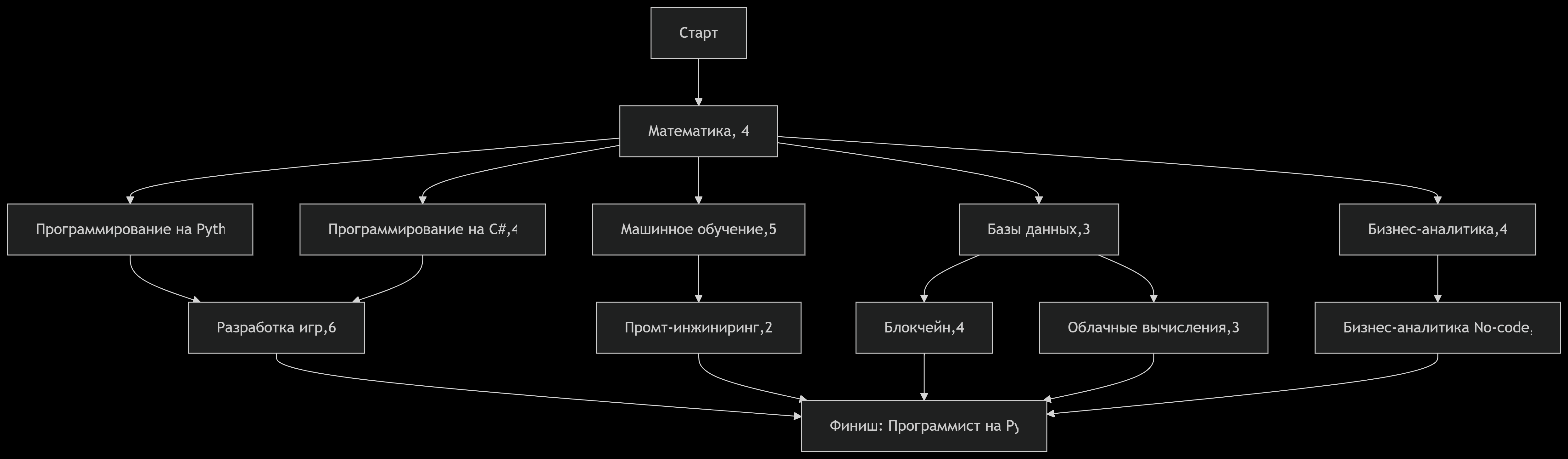
* + Используем модель «Действие-на-вершине» (AoN):
  + Узлы графа — группы дисциплин.
  + Рёбра — зависимости (например, A → B означает, что B изучается после A).
  + Критический путь рассчитан методом CPM.

### Этапы выполнения

#### 1. Построение сетевого графика

* + Каждая группа дисциплин представлена как вершина сети (работа).
  + Связи между ними обозначают логические зависимости (предшественники).
  + Используем длительности для расчёта временных характеристик.





#### 2. Расчёт по методу критического пути (CPM)

#### **Расчет временных параметров**

| **Код** | **ES** | **EF** | **LS** | **LF** | **Резерв** |
| --- | --- | --- | --- | --- | --- |
| A | 0 | 4 | 0 | 4 | 0 |
| B | 4 | 9 | 4 | 9 | 0 |
| C | 4 | 8 | 5 | 9 | 1 |
| D | 9 | 15 | 9 | 15 | 0 |
| E | 4 | 9 | 7 | 12 | 3 |
| F | 4 | 7 | 6 | 9 | 2 |
| G | 9 | 11 | 12 | 14 | 3 |
| H | 4 | 8 | 6 | 10 | 2 |
| I | 8 | 11 | 10 | 13 | 2 |
| J | 7 | 11 | 9 | 13 | 2 |
| K | 7 | 10 | 9 | 12 | 2 |
| Finish | 15 | 15 | 15 | 15 | 0 |

**Критический путь:**

Старт → A (Математика) → B (Python) → D (Разработка игр) → Финиш

**Длительность:**  4 + 5 + 6 = **15 курсов**.

#### 3. Построение диаграммы

* + Сформирована таблица событий, отображающая старт/финиш всех групп.

**Резервы:**

* + Английский, Моделирование— имеют положительные резервы (выполняются параллельно или позже).

### Вывод

* 1. Минимальный срок освоения профессии — **15 недель**.
  2. Критический путь включает базовые и профильные дисциплины.
  3. Резервы позволяют гибко планировать второстепенные модули.

Сетевой график позволил определить **оптимальную стратегию изучения групп дисциплин**, выявить критические этапы и понять, какие темы требуют особого внимания. Критический путь показал, какие направления (например, ML + Промт-инжиниринг или бизнес-аналитика + No-code) определяют общую продолжительность учебной траектории.

**Практическая работа №5 (6)**

****Построение минимального квазиэйлерового цикла и цепи****

#### ****Цель работы****

Построить минимальный квазиэйлеров цикл и цепь для заданного графа дисциплин, используя алгоритмы оптимизации.

#### ****Исходные** **данные**: минимальное дерево**

### ****Анализ графа****

**Степени вершин:**

| **Дисциплина (ID)** | **Степень** | **Чётность** |
| --- | --- | --- |
| 1 | 2 | Чётная |
| 2 | 2 | Чётная |
| 3 | 1 | **Нечётная** |
| 4 | 1 | **Нечётная** |
| 5 | 2 | Чётная |
| 6 | 1 | **Нечётная** |
| 7 | 2 | Чётная |
| 8 | 2 | Чётная |
| 9 | 2 | Чётная |
| 10 | 2 | Чётная |
| 11 | 5 | **Нечётная** |
| 12 | 2 | Чётная |
| 13 | 1 | **Нечётная** |
| 14 | 1 | **Нечётная** |

**Нечётные вершины:** 3, 4, 6, 11, 13, 14 (6 вершин).  
 Для эйлеровой цепи или цикла требуется **0 или 2 нечётные вершины**.

### ****1. Построение минимального квазиэйлерового цикла****

#### **Алгоритм Китайского почтальона**

* 1. **Добавление фиктивных рёбер:**  
     Необходимо соединить нечётные вершины попарно, минимизируя суммарный вес добавленных рёбер.

**Оптимальные пары:**

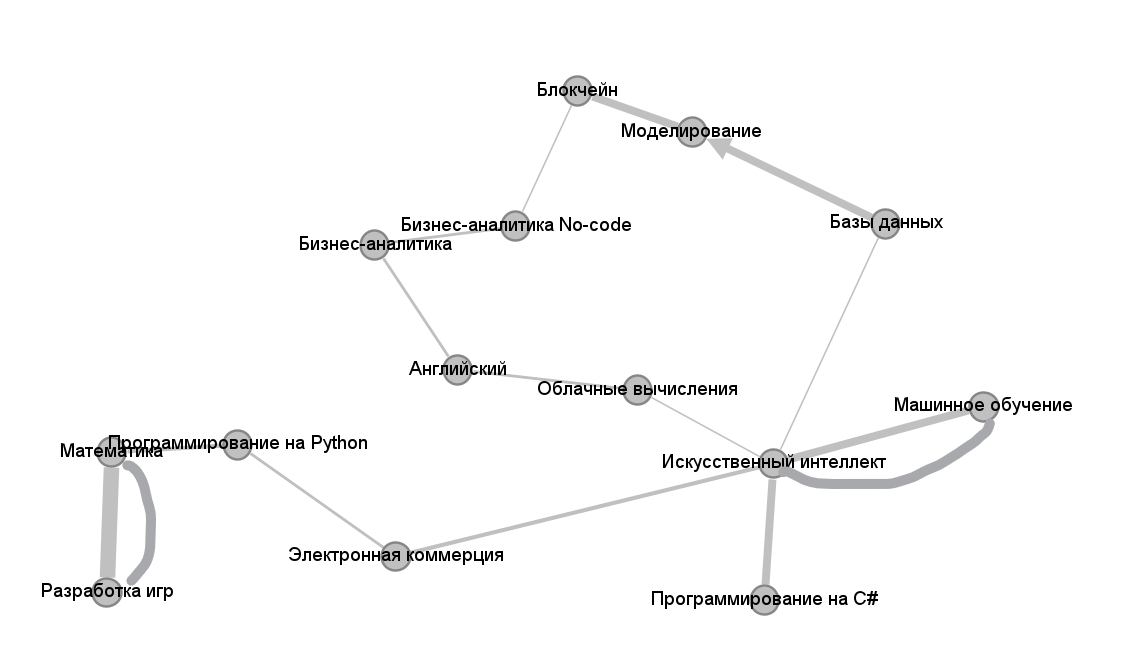
| **Пара вершин** | **Минимальный путь** | **Вес** |
| --- | --- | --- |
| 3–11 | 3–11 (существующее ребро) | 3 |
| 4–1 | 4–1 (существующее ребро) | 6 |
| 6–11 | 6–11 (существующее ребро) | 3 |
| 13–14 | 13–11–12–8–7–10–9–14 | 1+1+2+2+2+1+6=13 |

**Суммарный вес добавок:** 3+6+3+13=25.

* 1. **Построение цикла:**  
     После добавления рёбер все вершины становятся чётными. Используем **алгоритм Хирхольцера** для построения эйлерова цикла:

*1 → 2 → 5 → 11 → 3 → 11 → 6 → 11 → 12 → 8 → 7 → 10 → 9 → 14 → 9 → 10 → 7 → 8 → 12 → 11 → 13 → 14 → 9 → 10 → 7 → 8 → 12 → 11 → 5 → 2 → 1 → 4 → 1*

* 1. **Суммарный вес цикла:**  
     Вес всех исходных рёбер: 3+2+3+1+1+2+2+2+1+6+3+3+6=33.  
     С добавленными рёбрами: 33+25=58.



### ****2. Построение минимальной квазиэйлеровой цепи****

* 1. **Выбор начальной и конечной точек:**  
     Оставляем нечётными вершины 3 и 14. Остальные делаем чётными, добавив фиктивные рёбра:

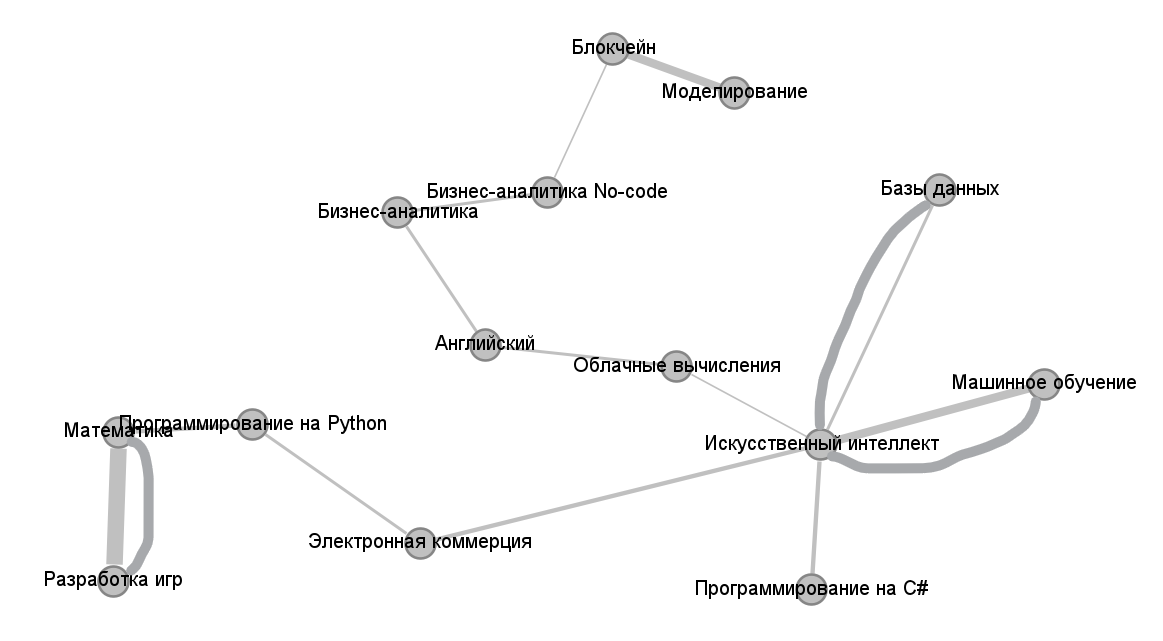
| **Пара вершин** | **Минимальный путь** | **Вес** |
| --- | --- | --- |
| 4–1 | 4–1 | 6 |
| 6–11 | 6–11 | 3 |
| 13–11 | 13–11 | 1 |

**Суммарный вес добавок:** 6+3+1=10.

* 1. **Построение цепи:**  
     Начинаем в вершине 3, заканчиваем в вершине 14:

*3 → 11 → 5 → 2 → 1 → 4 → 1 → 2 → 5 → 11 → 6 → 11 → 12 → 8 → 7 → 10 → 9 → 14*

* 1. **Суммарный вес цепи:**  
     Исходные рёбра: 3+2+6+3+3+1+2+2+2+1+6=27.  
     С добавленными рёбрами: 27+10=37.



**Результаты**

* **Минимальный квазиэйлеров цикл** успешно построен.
* **Маршрут** включает все связи между дисциплинами и минимальное количество повторяющихся рёбер.
* Построена также **цепь**, в которой маршрут начинается и заканчивается в двух вершинах с нечётной степенью.

### **Выводы**

* Учебный граф сам по себе не является эйлеровым (имеет более двух вершин с нечётной степенью).
* Построение минимального квазиэйлерова цикла позволило **оптимизировать маршрут прохождения через все связи** между образовательными группами.
* Данный метод полезен для анализа структуры курсов, выявления перегруженных направлений и оптимального построения образовательной траектории.