**Отчёт по лабораторным работам**

**Вариант 1**

**Лабораторная работа 6.**

**АЛГОРИТМЫ НА ГРАФАХ**

**Студент:**

**Адамович Карианна Павловна, 2-3**

**Преподаватель:**

**Бурмакова Анастасия Владимировна**

**г.Минск, 2025**

**ЦЕЛЬ РАБОТЫ:** Освоить сущность и программную реализацию: а) способов представления графов; б) алгоритмов поиска в ширину и глубину; в) алгоритма топологической сортировки графов. Разобрать алгоритм Прима и алгоритм Крускала.

**ЗАДАНИЕ ДЛЯ ВЫПОЛНЕНИЯ:**



***Задание 1.*** Ориентированный граф **G** взять в соответствии с вариантом. Представить его в отчете в виде матрицы смежности, матрицы инцидентности, списка смежных вершин.

Матрица смежности:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** |
| **0** | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| **1** | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| **2** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **3** | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| **4** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **5** | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| **6** | 0 | 0 | 0 | 0 | 1 | 1 | 0 |

Матрица инцидентности:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Вершины \ Рёбра** |  |  |  |  |  |  |  |  |
| 0 | -1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | -1 | -1 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 3 | 0 | 1 | 0 | -1 | -1 | 0 | 0 | 0 |
| 4 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 5 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | -1 |
| 6 | 0 | 0 | 0 | 0 | 1 | -1 | -1 | 0 |

**Список смежных вершин**

Список смежных вершин показывает, какие вершины соединены дугами для каждой вершины графа.

Вершина 0: [1]

Вершина 1: [3,4]

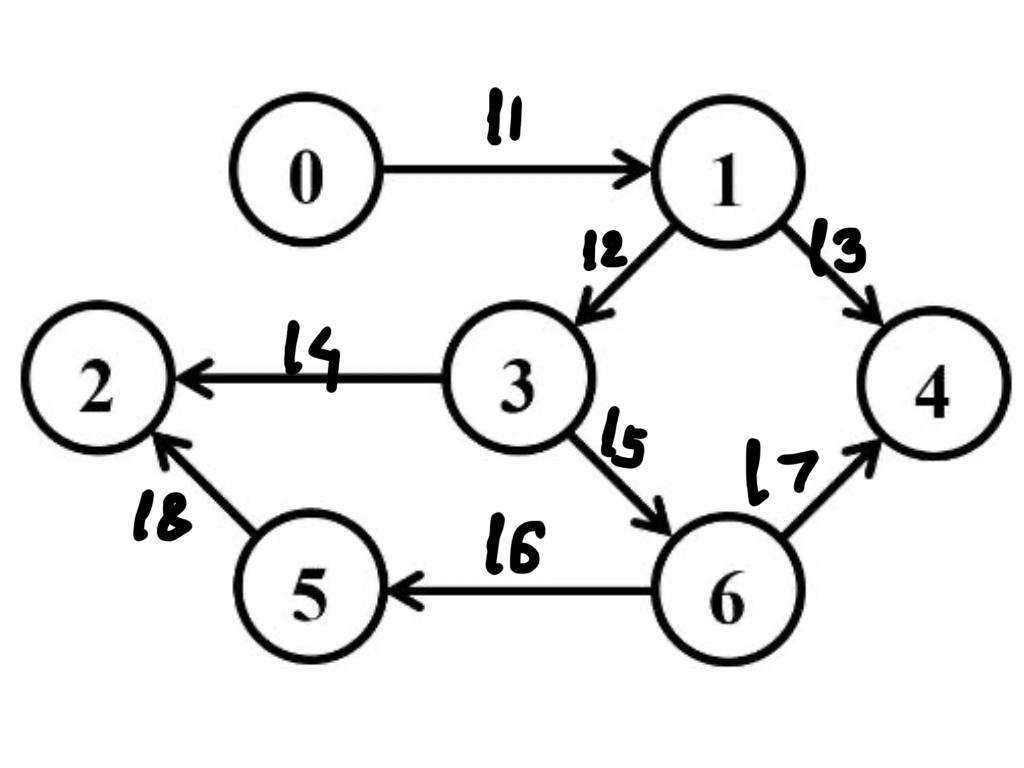
Вершина 2: -

Вершина 3: [2,6]

Вершина 4: -

Вершина 5: [2]

Вершина 6: [4,5]



***Задание 2.*** Осуществить алгоритмы поиска в ширину и глубину, а также алгоритма топологической сортировки аналогично примерам, рассмотренным на лекциях. Оформить отчет, включив в него **каждый** шаг выполнения алгоритмов.

**1**. **Поиск в ширину (BFS)**

Алгоритм обхода в ширину использует очередь для обхода вершин графа.

Старт с вершины 0:

Очередь: 0.

Посещаем вершину 1. Очередь становится: 1.

Обработка вершины 1:

Очередь: 1.

Из вершины 1 переходим в вершины 3 и 4. Очередь: 3, 4.

Обработка вершины 3:

Очередь: 3, 4.

Из вершины 3 переходим в вершины 2 и 6. Очередь: 4, 2, 6.

Обработка вершины 4:

Очередь: 4, 2, 6.

Вершина 4 не имеет исходящих рёбер.

Обработка вершины 2:

Очередь: 2, 6.

Вершина 2 не имеет исходящих рёбер.

Обработка вершины 6:

Очередь: 6.

Из вершины 6 переходим в вершины 4 и 5. Очередь: 5.

Обработка вершины 5:

Очередь: 5.

Из вершины 5 переходим в вершину 2. Очередь становится пустой.

Итог: BFS посещает вершины в следующем порядке: 0 → 1 → 3 → 4 → 2 → 6 → 5.



1. **Поиск в глубину (DFS)**

Алгоритм обхода в глубину использует стек для последовательного спуска по графу.

Шаги для графа (начинаем с вершины 0):

Старт с вершины 0:

Переходим в вершину 1.

Из вершины 1:

Углубляемся в вершину 3.

Из вершины 3:

Переходим в вершину 2. Так как у вершины 2 нет выходов, возвращаемся к вершине 3.

Из вершины 3 переходим в вершину 6.

Из вершины 6:

Углубляемся в вершину 4. Так как у вершины 4 нет выходов, возвращаемся к вершине 6.

Затем переходим к вершине 5. Из вершины 5 переходим снова в вершину 2.

Завершаем обход, возвращаясь к вершине 1, а затем к вершине 0.

Итог: DFS обходит вершины в порядке: 2 → 4 → 5 → 6 → 3 → 1 → 0.

**3.Топологическая сортировка**

Топологическая сортировка используется для ориентированных ацикличных графов (DAG). Мы обходим граф в глубину и записываем вершины в обратном порядке завершения.

Шаги выполнения (с использованием DFS):

Обход начат с вершины 0. Добавили 0.

Перешли к 1. Добавили 1.

Перешли к 3. Добавили 3.

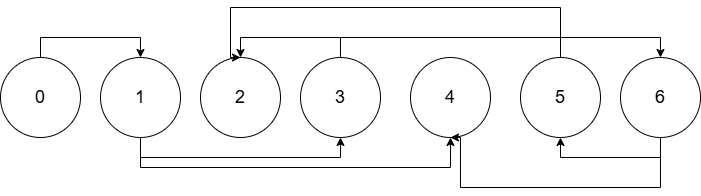
Перешли к 2. Добавили 2.

Вернулись к 3. Перешли к 6. Добавили 6.

Перешли к 5. Добавили 5.

Обход завершён. Вершины в порядке завершения: 5 -> 6 -> 3 -> 2 -> 1 -> 0.

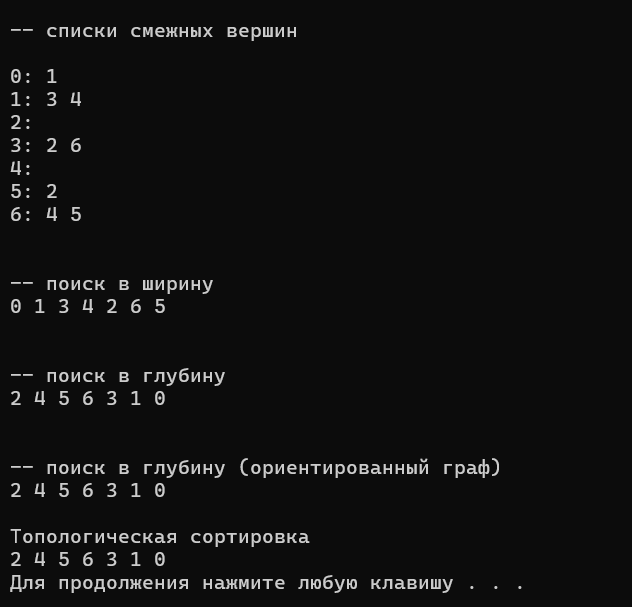
Результат топологической сортировки: 2 → 4 → 5 → 6 → 3 → 1 → 0



***Задание 3.*** Осуществить программную реализацию алгоритмов на C++. Разработать структуры **AMatrix** и **АList**  для представления ориентированного графа матричным и списковым способом. Разработать функции преобразования из одного способа представления в другой. Разработать функцию **BFS** обхода вершин графа, используя метод поиска в ширину. Продемонстрировать работу функции. Копии экрана вставить в отчет.

***Задание 4.*** Разработать функцию **DFS**  обхода вершин графа, используя метод поиска глубину. Продемонстрировать работу функции. Копии экрана вставить в отчет.

***Задание 5.*** Доработайте функцию **DFS**,для выполнения топологической сортировки графа. Продемонстрировать работу функции. Копии экрана вставить в отчет.



***Задание 6.*** По графу, соответствующему варианту составить минимальное остовное дерево по алгоритму Прима. Шаги построения отразить в отчете.

Веса ребер принять:

W:

W(e0,1)=8; W(e1,0)=5;

W(e0,2)=1; W(e2,0)=3;

W(e0,3)=2; W(e3,0)=8;

W(e1,3)=11; W(e3,1)=4;

W(e1,4)=5; W(e4,1)=3;

W(e2,3)=7; W(e3,2)=9;

W(e2,5)=11; W(e5,2)=10;

W(e4,3)=4; W(e3,4)=1;

W(e4,6)=10; W(e6,4)=2;

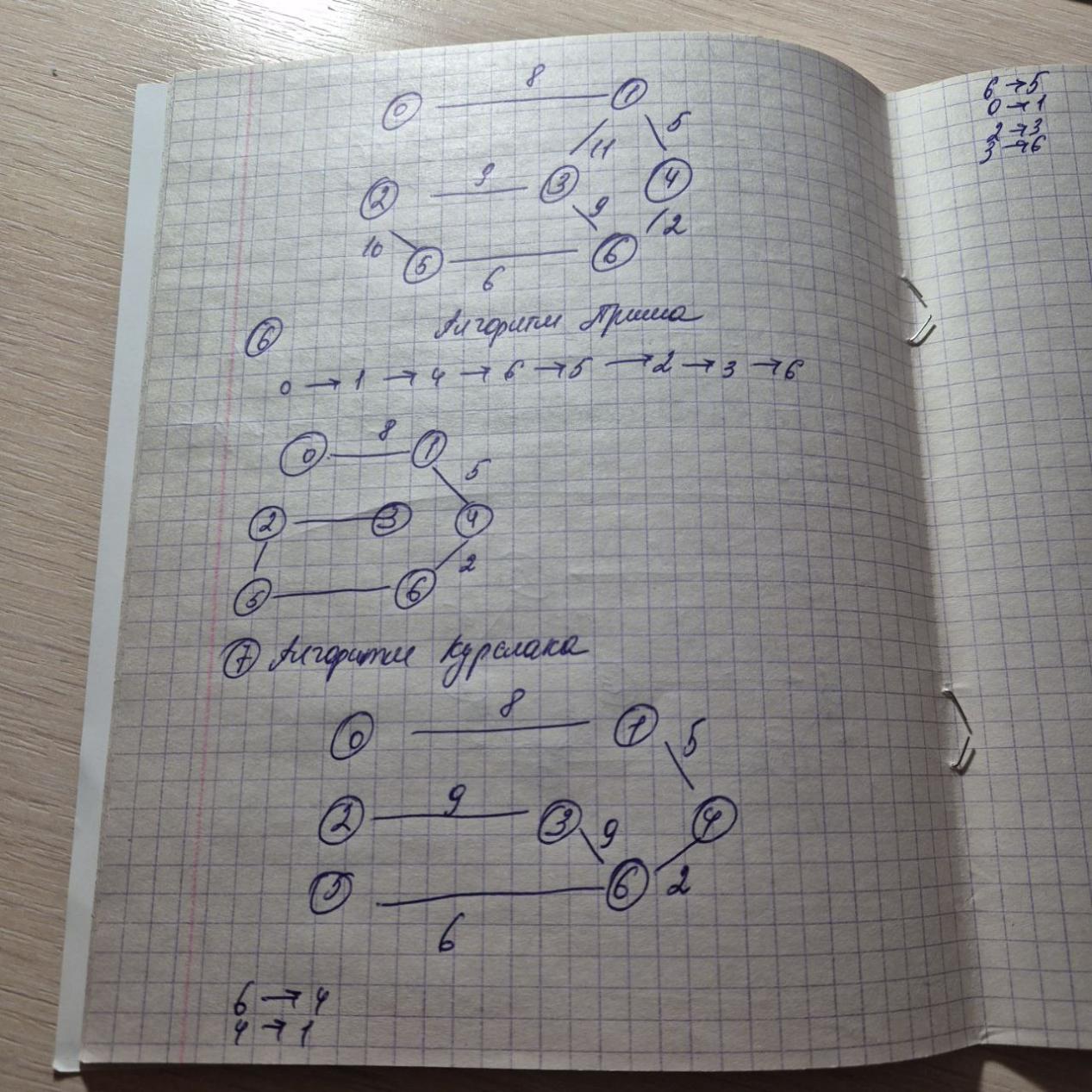
W(e5,6)=2; W(e6,5)=6;

W(e5,3)=3; W(e3,5)=6;

W(e6,3)=7; W(e3,6)=9;



**Алгоритм Прима**

****

***Задание 7.*** По графу, соответствующему варианту составить минимальное остовное дерево по алгоритму Крускала. Шаги построения отразить в отчете.

