Практическая работа №3. Алгоритмы на графах

Дана матрица смежности неориентированного взвешенного графа (таблица 3.1, 0 означает отсутствие ребра).

1. Необходимо построить минимальное остовное дерево.
2. Запустите функцию, реализующую алгоритм поиска в глубину, для перечисления всех вершин в минимальном остовном дереве. Результат должен быть представлен с помощью одного из контейнеров STL.
3. Напишите функцию для поиска минимального пути (в смысле суммарного веса пройденных рёбер) между i-м и всеми остальными пунктами, куда можно построить маршрут. Результат должен быть представлен с помощью одного из контейнеров STL.
4. Реализовать функцию подсчета степени (количества инцидентных ребер) вершин в полученном дереве (обход дерева сделать на основе поиска в ширину). Реализовать функцию подсчета средней степени по всему дереву.

Таблица 3.1

Вариант	Матрица смежности

1.	{
	{ 0, 9, 5, 4, 8, 5, 1, 6, 1, 9, 5, 0 },
	{ 9, 0, 1, 7, 7, 8, 2, 8, 2, 0, 7, 3 },
	{ 5, 1, 0, 4, 0, 9, 4, 4, 5, 2, 3, 1 },
	{ 4, 7, 4, 0, 5, 3, 8, 2, 3, 4, 5, 9 },
	{ 8, 7, 0, 5, 0, 5, 4, 7, 7, 5, 4, 4 },
	{ 5, 8, 9, 3, 5, 0, 5, 9, 4, 8, 1, 1 },
	{ 1, 2, 4, 8, 4, 5, 0, 1, 6, 4, 2, 0 },
	{ 6, 8, 4, 2, 7, 9, 1, 0, 8, 9, 4, 4 },
	{ 1, 2, 5, 3, 7, 4, 6, 8, 0, 2, 0, 7 },
	{ 9, 0, 2, 4, 5, 8, 4, 9, 2, 0, 3, 4 },
	{ 5, 7, 3, 5, 4, 1, 2, 4, 0, 3, 0, 2 },
	{ 0, 3, 1, 9, 4, 1, 0, 4, 7, 4, 2, 0 },
	}

2.	{ { 0, 5, 3, 6, 8, 9, 7, 8, 1, 7, 0, 0, 4, 8 },

	{ 5, 0, 0, 3, 6, 9, 6, 5, 0, 8, 0, 0, 5, 6 }, { 3, 0, 0, 2, 8, 1, 3, 0, 8, 8, 5, 5, 8, 4 }, { 6, 3, 2, 0, 4, 6, 6, 4, 6, 8, 8, 6, 9, 4 }, { 8, 6, 8, 4, 0, 2, 8, 0, 9, 0, 8, 2, 0, 5 }, { 9, 9, 1, 6, 2, 0, 8, 5, 5, 9, 8, 8, 9, 8 }, { 7, 6, 3, 6, 8, 8, 0, 3, 6, 6, 8, 1, 5, 6 }, { 8, 5, 0, 4, 0, 5, 3, 0, 7, 1, 4, 7, 8, 5 }, { 1, 0, 8, 6, 9, 5, 6, 7, 0, 1, 2, 5, 2, 2 }, { 7, 8, 8, 8, 0, 9, 6, 1, 1, 0, 6, 2, 4, 8 }, { 0, 0, 5, 8, 8, 8, 8, 4, 2, 6, 0, 8, 4, 3 }, { 0, 0, 5, 6, 2, 8, 1, 7, 5, 2, 8, 0, 5, 5 }, { 4, 5, 8, 9, 0, 9, 5, 8, 2, 4, 4, 5, 0, 3 }, { 8, 6, 4, 4, 5, 8, 6, 5, 2, 8, 3, 5, 3, 0 }, }

3.	{
	{ 0, 1, 7, 0, 9, 2, 4, 9, 3, 1, 4, 7, 3 },
	{ 1, 0, 8, 6, 0, 0, 4, 8, 5, 7, 6, 7, 4 },
	{ 7, 8, 0, 9, 6, 0, 6, 1, 3, 0, 4, 4, 9 },
	{ 0, 6, 9, 0, 4, 5, 1, 1, 5, 6, 4, 9, 3 },
	{ 9, 0, 6, 4, 0, 7, 0, 0, 9, 0, 4, 7, 6 },
	{ 2, 0, 0, 5, 7, 0, 4, 5, 3, 8, 5, 1, 8 },
	{ 4, 4, 6, 1, 0, 4, 0, 3, 4, 3, 4, 8, 0 },
	{ 9, 8, 1, 1, 0, 5, 3, 0, 3, 5, 7, 5, 6 },
	{ 3, 5, 3, 5, 9, 3, 4, 3, 0, 2, 3, 0, 4 },
	{ 1, 7, 0, 6, 0, 8, 3, 5, 2, 0, 7, 9, 4 },
	{ 4, 6, 4, 4, 4, 5, 4, 7, 3, 7, 0, 9, 8 },
	{ 7, 7, 4, 9, 7, 1, 8, 5, 0, 9, 9, 0, 6 },
	{ 3, 4, 9, 3, 6, 8, 0, 6, 4, 4, 8, 6, 0 },
	}

4.	{ { 0, 8, 2, 0, 5, 1, 7, 3, 5, 9, 3, 7 }, { 8, 0, 7, 5, 7, 1, 9, 1, 1, 6, 6, 9 }, { 2, 7, 0, 9, 3, 5, 1, 9, 1, 0, 8, 0 }, { 0, 5, 9, 0, 8, 8, 4, 0, 3, 5, 7, 8 }, { 5, 7, 3, 8, 0, 1, 7, 3, 0, 6, 8, 9 }, { 1, 1, 5, 8, 1, 0, 7, 0, 0, 8, 6, 9 }, { 7, 9, 1, 4, 7, 7, 0, 0, 7, 2, 5, 8 },

	{ 3, 1, 9, 0, 3, 0, 0, 0, 1, 8, 8, 1 }, { 5, 1, 1, 3, 0, 0, 7, 1, 0, 8, 6, 9 }, { 9, 6, 0, 5, 6, 8, 2, 8, 8, 0, 2, 7 }, { 3, 6, 8, 7, 8, 6, 5, 8, 6, 2, 0, 4 }, { 7, 9, 0, 8, 9, 9, 8, 1, 9, 7, 4, 0 }, }

5.	{
	{ 0, 1, 3, 0, 1, 3, 6, 6, 7, 1 },
	{ 1, 0, 1, 8, 6, 0, 0, 0, 8, 8 },
	{ 3, 1, 0, 6, 6, 4, 5, 7, 6, 2 },
	{ 0, 8, 6, 0, 1, 3, 0, 3, 4, 3 },
	{ 1, 6, 6, 1, 0, 4, 6, 8, 5, 7 },
	{ 3, 0, 4, 3, 4, 0, 1, 3, 6, 1 },
	{ 6, 0, 5, 0, 6, 1, 0, 4, 7, 1 },
	{ 6, 0, 7, 3, 8, 3, 4, 0, 1, 8 },
	{ 7, 8, 6, 4, 5, 6, 7, 1, 0, 1 },
	{ 1, 8, 2, 3, 7, 1, 1, 8, 1, 0 },
	}

6.	{
	{ 0, 5, 6, 6, 6, 4, 5, 0, 0, 8, 8, 4, 4 },
	{ 5, 0, 8, 0, 3, 8, 4, 8, 6, 6, 6, 3, 4 },
	{ 6, 8, 0, 2, 0, 0, 0, 9, 3, 5, 3, 8, 1 },
	{ 6, 0, 2, 0, 2, 4, 7, 7, 7, 9, 5, 5, 5 },
	{ 6, 3, 0, 2, 0, 1, 5, 5, 4, 4, 1, 4, 2 },
	{ 4, 8, 0, 4, 1, 0, 8, 1, 5, 4, 5, 8, 6 },
	{ 5, 4, 0, 7, 5, 8, 0, 6, 9, 0, 1, 2, 0 },
	{ 0, 8, 9, 7, 5, 1, 6, 0, 4, 6, 7, 3, 3 },
	{ 0, 6, 3, 7, 4, 5, 9, 4, 0, 4, 4, 1, 1 },
	{ 8, 6, 5, 9, 4, 4, 0, 6, 4, 0, 9, 2, 7 },
	{ 8, 6, 3, 5, 1, 5, 1, 7, 4, 9, 0, 6, 9 },
	{ 4, 3, 8, 5, 4, 8, 2, 3, 1, 2, 6, 0, 3 },
	{ 4, 4, 1, 5, 2, 6, 0, 3, 1, 7, 9, 3, 0 },
	}

7.	{ { 0, 6, 5, 6, 7, 5, 8, 8, 2 }, { 6, 0, 2, 5, 1, 4, 4, 3, 2 }, { 5, 2, 0, 0, 6, 7, 5, 4, 2 },

	{ 6, 5, 0, 0, 2, 4, 1, 7, 4 }, { 7, 1, 6, 2, 0, 8, 0, 9, 5 }, { 5, 4, 7, 4, 8, 0, 9, 8, 0 }, { 8, 4, 5, 1, 0, 9, 0, 7, 5 }, { 8, 3, 4, 7, 9, 8, 7, 0, 7 }, { 2, 2, 2, 4, 5, 0, 5, 7, 0 }, }

8.	{
	{ 0, 7, 8, 9, 1, 6, 3, 2, 0, 8, 2, 3, 0 },
	{ 7, 0, 7, 6, 0, 6, 7, 1, 4, 1, 1, 1, 1 },
	{ 8, 7, 0, 4, 0, 8, 0, 1, 0, 7, 7, 7, 6 },
	{ 9, 6, 4, 0, 8, 1, 2, 4, 5, 2, 2, 9, 8 },
	{ 1, 0, 0, 8, 0, 5, 7, 6, 7, 3, 4, 9, 0 },
	{ 6, 6, 8, 1, 5, 0, 7, 2, 1, 8, 9, 2, 9 },
	{ 3, 7, 0, 2, 7, 7, 0, 9, 5, 9, 6, 4, 9 },
	{ 2, 1, 1, 4, 6, 2, 9, 0, 4, 3, 2, 6, 9 },
	{ 0, 4, 0, 5, 7, 1, 5, 4, 0, 7, 1, 3, 6 },
	{ 8, 1, 7, 2, 3, 8, 9, 3, 7, 0, 9, 8, 3 },
	{ 2, 1, 7, 2, 4, 9, 6, 2, 1, 9, 0, 5, 6 },
	{ 3, 1, 7, 9, 9, 2, 4, 6, 3, 8, 5, 0, 9 },
	{ 0, 1, 6, 8, 0, 9, 9, 9, 6, 3, 6, 9, 0 },
	}

9.	{ { 0, 9, 7, 1, 5, 4, 5, 3, 8, 1, 0, 7, 4, 0, 8 }, { 9, 0, 7, 3, 2, 7, 0, 9, 8, 5, 0, 6, 4, 1, 3 }, { 7, 7, 0, 2, 2, 2, 2, 3, 9, 5, 1, 5, 0, 4, 4 }, { 1, 3, 2, 0, 4, 4, 1, 0, 6, 9, 7, 2, 3, 6, 2 }, { 5, 2, 2, 4, 0, 4, 4, 8, 4, 2, 4, 5, 7, 6, 9 }, { 4, 7, 2, 4, 4, 0, 9, 0, 3, 1, 6, 4, 8, 8, 8 }, { 5, 0, 2, 1, 4, 9, 0, 2, 2, 4, 5, 4, 2, 6, 1 }, { 3, 9, 3, 0, 8, 0, 2, 0, 4, 1, 9, 9, 5, 5, 7 }, { 8, 8, 9, 6, 4, 3, 2, 4, 0, 0, 7, 3, 7, 4, 1 }, { 1, 5, 5, 9, 2, 1, 4, 1, 0, 0, 7, 6, 1, 2, 9 }, { 0, 0, 1, 7, 4, 6, 5, 9, 7, 7, 0, 9, 6, 7, 8 }, { 7, 6, 5, 2, 5, 4, 4, 9, 3, 6, 9, 0, 6, 2, 2 }, { 4, 4, 0, 3, 7, 8, 2, 5, 7, 1, 6, 6, 0, 2, 7 }, { 0, 1, 4, 6, 6, 8, 6, 5, 4, 2, 7, 2, 2, 0, 7 },

	{ 8, 3, 4, 2, 9, 8, 1, 7, 1, 9, 8, 2, 7, 7, 0 }, }

10.	{
	{ 0, 9, 0, 2, 7, 4, 7, 8, 4 },
	{ 9, 0, 5, 5, 0, 7, 8, 1, 5 },
	{ 0, 5, 0, 6, 3, 9, 3, 3, 0 },
	{ 2, 5, 6, 0, 3, 4, 3, 9, 5 },
	{ 7, 0, 3, 3, 0, 0, 9, 0, 4 },
	{ 4, 7, 9, 4, 0, 0, 5, 9, 6 },
	{ 7, 8, 3, 3, 9, 5, 0, 9, 8 },
	{ 8, 1, 3, 9, 0, 9, 9, 0, 1 },
	{ 4, 5, 0, 5, 4, 6, 8, 1, 0 },
	}

11.	{
	{ 0, 8, 0, 1, 4, 1, 4, 9, 3, 9 },
	{ 8, 0, 5, 1, 4, 7, 9, 0, 2, 0 },
	{ 0, 5, 0, 9, 6, 0, 5, 6, 0, 6 },
	{ 1, 1, 9, 0, 0, 8, 5, 1, 1, 0 },
	{ 4, 4, 6, 0, 0, 3, 9, 0, 6, 5 },
	{ 1, 7, 0, 8, 3, 0, 1, 1, 4, 7 },
	{ 4, 9, 5, 5, 9, 1, 0, 2, 1, 8 },
	{ 9, 0, 6, 1, 0, 1, 2, 0, 7, 8 },
	{ 3, 2, 0, 1, 6, 4, 1, 7, 0, 7 },
	{ 9, 0, 6, 0, 5, 7, 8, 8, 7, 0 },
	}

12.	{ { 0, 1, 4, 2, 5, 6, 4, 0, 2, 1, 1 }, { 1, 0, 7, 6, 5, 2, 9, 5, 9, 8, 3 }, { 4, 7, 0, 7, 0, 9, 0, 8, 2, 0, 8 }, { 2, 6, 7, 0, 9, 7, 2, 8, 0, 8, 1 }, { 5, 5, 0, 9, 0, 7, 7, 5, 0, 1, 8 }, { 6, 2, 9, 7, 7, 0, 1, 8, 5, 9, 0 }, { 4, 9, 0, 2, 7, 1, 0, 6, 4, 9, 4 }, { 0, 5, 8, 8, 5, 8, 6, 0, 9, 6, 6 }, { 2, 9, 2, 0, 0, 5, 4, 9, 0, 8, 8 }, { 1, 8, 0, 8, 1, 9, 9, 6, 8, 0, 7 }, { 1, 3, 8, 1, 8, 0, 4, 6, 8, 7, 0 },

	}

13.	{
	{ 0, 9, 9, 7, 6, 9, 9, 5, 3 },
	{ 9, 0, 3, 9, 0, 7, 8, 9, 5 },
	{ 9, 3, 0, 8, 1, 7, 1, 2, 4 },
	{ 7, 9, 8, 0, 9, 0, 4, 2, 2 },
	{ 6, 0, 1, 9, 0, 3, 1, 9, 1 },
	{ 9, 7, 7, 0, 3, 0, 8, 0, 3 },
	{ 9, 8, 1, 4, 1, 8, 0, 7, 7 },
	{ 5, 9, 2, 2, 9, 0, 7, 0, 4 },
	{ 3, 5, 4, 2, 1, 3, 7, 4, 0 },
	}

14.	{
	{ 0, 6, 2, 1, 9, 1, 8, 1, 4, 8, 6, 1, 3 },
	{ 6, 0, 2, 5, 1, 9, 9, 8, 1, 7, 9, 1, 1 },
	{ 2, 2, 0, 4, 2, 2, 5, 3, 4, 6, 0, 3, 0 },
	{ 1, 5, 4, 0, 1, 2, 4, 9, 4, 8, 8, 0, 9 },
	{ 9, 1, 2, 1, 0, 3, 5, 4, 4, 4, 5, 4, 8 },
	{ 1, 9, 2, 2, 3, 0, 2, 5, 1, 6, 9, 5, 8 },
	{ 8, 9, 5, 4, 5, 2, 0, 7, 9, 3, 5, 9, 6 },
	{ 1, 8, 3, 9, 4, 5, 7, 0, 5, 2, 0, 9, 2 },
	{ 4, 1, 4, 4, 4, 1, 9, 5, 0, 6, 9, 2, 9 },
	{ 8, 7, 6, 8, 4, 6, 3, 2, 6, 0, 9, 5, 4 },
	{ 6, 9, 0, 8, 5, 9, 5, 0, 9, 9, 0, 5, 1 },
	{ 1, 1, 3, 0, 4, 5, 9, 9, 2, 5, 5, 0, 0 },
	{ 3, 1, 0, 9, 8, 8, 6, 2, 9, 4, 1, 0, 0 },
	}

15.	{ { 0, 6, 1, 9, 4, 4, 2, 3, 5 }, { 6, 0, 2, 2, 4, 0, 5, 5, 0 }, { 1, 2, 0, 1, 6, 9, 4, 6, 3 }, { 9, 2, 1, 0, 1, 9, 9, 4, 3 }, { 4, 4, 6, 1, 0, 2, 8, 3, 1 }, { 4, 0, 9, 9, 2, 0, 9, 1, 2 }, { 2, 5, 4, 9, 8, 9, 0, 8, 8 }, { 3, 5, 6, 4, 3, 1, 8, 0, 9 }, { 5, 0, 3, 3, 1, 2, 8, 9, 0 },

	}

16.	{
	{ 0, 5, 3, 1, 0, 3, 4, 3, 6, 0, 4 },
	{ 5, 0, 6, 7, 9, 4, 3, 3, 6, 9, 6 },
	{ 3, 6, 0, 4, 5, 1, 9, 5, 3, 1, 8 },
	{ 1, 7, 4, 0, 5, 5, 2, 4, 2, 5, 8 },
	{ 0, 9, 5, 5, 0, 3, 8, 2, 6, 4, 3 },
	{ 3, 4, 1, 5, 3, 0, 3, 2, 2, 2, 8 },
	{ 4, 3, 9, 2, 8, 3, 0, 7, 6, 7, 6 },
	{ 3, 3, 5, 4, 2, 2, 7, 0, 4, 3, 4 },
	{ 6, 6, 3, 2, 6, 2, 6, 4, 0, 7, 3 },
	{ 0, 9, 1, 5, 4, 2, 7, 3, 7, 0, 9 },
	{ 4, 6, 8, 8, 3, 8, 6, 4, 3, 9, 0 },
	}

17.	{
	{ 0, 9, 7, 9, 6, 9, 5, 5, 6, 3, 6 },
	{ 9, 0, 2, 3, 7, 6, 5, 6, 7, 7, 0 },
	{ 7, 2, 0, 5, 0, 0, 6, 8, 0, 5, 6 },
	{ 9, 3, 5, 0, 6, 2, 5, 1, 1, 2, 2 },
	{ 6, 7, 0, 6, 0, 0, 1, 0, 5, 8, 3 },
	{ 9, 6, 0, 2, 0, 0, 4, 2, 8, 3, 0 },
	{ 5, 5, 6, 5, 1, 4, 0, 5, 9, 7, 4 },
	{ 5, 6, 8, 1, 0, 2, 5, 0, 9, 2, 6 },
	{ 6, 7, 0, 1, 5, 8, 9, 9, 0, 1, 0 },
	{ 3, 7, 5, 2, 8, 3, 7, 2, 1, 0, 4 },
	{ 6, 0, 6, 2, 3, 0, 4, 6, 0, 4, 0 },
	}

18.	{ { 0, 0, 0, 7, 6, 0, 1, 4, 4, 5, 6, 6, 7 }, { 0, 0, 6, 4, 4, 1, 2, 3, 2, 0, 1, 4, 2 }, { 0, 6, 0, 8, 8, 4, 3, 6, 5, 3, 6, 6, 5 }, { 7, 4, 8, 0, 5, 4, 5, 8, 0, 9, 3, 6, 8 }, { 6, 4, 8, 5, 0, 1, 4, 2, 7, 7, 7, 2, 0 }, { 0, 1, 4, 4, 1, 0, 7, 4, 4, 2, 4, 2, 6 }, { 1, 2, 3, 5, 4, 7, 0, 7, 1, 2, 2, 9, 8 },

	{ 4, 3, 6, 8, 2, 4, 7, 0, 8, 4, 2, 3, 2 }, { 4, 2, 5, 0, 7, 4, 1, 8, 0, 2, 5, 8, 1 }, { 5, 0, 3, 9, 7, 2, 2, 4, 2, 0, 5, 9, 7 }, { 6, 1, 6, 3, 7, 4, 2, 2, 5, 5, 0, 9, 6 }, { 6, 4, 6, 6, 2, 2, 9, 3, 8, 9, 9, 0, 5 }, { 7, 2, 5, 8, 0, 6, 8, 2, 1, 7, 6, 5, 0 }, }

19.	{
	{ 0, 4, 0, 7, 1, 8, 9, 7, 6, 8, 3 },
	{ 4, 0, 5, 3, 5, 3, 2, 3, 9, 6, 2 },
	{ 0, 5, 0, 4, 9, 9, 9, 6, 2, 7, 2 },
	{ 7, 3, 4, 0, 5, 7, 8, 4, 1, 8, 1 },
	{ 1, 5, 9, 5, 0, 4, 8, 2, 3, 4, 2 },
	{ 8, 3, 9, 7, 4, 0, 6, 1, 5, 4, 6 },
	{ 9, 2, 9, 8, 8, 6, 0, 0, 7, 7, 6 },
	{ 7, 3, 6, 4, 2, 1, 0, 0, 9, 2, 7 },
	{ 6, 9, 2, 1, 3, 5, 7, 9, 0, 1, 1 },
	{ 8, 6, 7, 8, 4, 4, 7, 2, 1, 0, 5 },
	{ 3, 2, 2, 1, 2, 6, 6, 7, 1, 5, 0 },
	}

20.	{
	{ 0, 5, 2, 7, 4, 8, 8, 8, 0, 6, 8, 4 },
	{ 5, 0, 7, 2, 0, 8, 9, 6, 4, 2, 5, 4 },
	{ 2, 7, 0, 1, 3, 3, 8, 3, 2, 6, 3, 6 },
	{ 7, 2, 1, 0, 6, 8, 0, 6, 3, 9, 4, 3 },
	{ 4, 0, 3, 6, 0, 9, 2, 3, 5, 9, 6, 8 },
	{ 8, 8, 3, 8, 9, 0, 9, 5, 4, 6, 9, 1 },
	{ 8, 9, 8, 0, 2, 9, 0, 0, 0, 5, 5, 8 },
	{ 8, 6, 3, 6, 3, 5, 0, 0, 0, 7, 6, 3 },
	{ 0, 4, 2, 3, 5, 4, 0, 0, 0, 6, 5, 4 },
	{ 6, 2, 6, 9, 9, 6, 5, 7, 6, 0, 8, 1 },
	{ 8, 5, 3, 4, 6, 9, 5, 6, 5, 8, 0, 9 },
	{ 4, 4, 6, 3, 8, 1, 8, 3, 4, 1, 9, 0 },
	}

21.	{ { 0, 6, 7, 6, 2, 9, 4, 6, 4, 7, 1 }, { 6, 0, 1, 1, 7, 7, 4, 7, 4, 8, 3 },

	{ 7, 1, 0, 4, 5, 5, 7, 2, 3, 9, 0 }, { 6, 1, 4, 0, 4, 6, 6, 8, 5, 3, 6 }, { 2, 7, 5, 4, 0, 9, 5, 0, 6, 9, 7 }, { 9, 7, 5, 6, 9, 0, 9, 2, 0, 8, 1 }, { 4, 4, 7, 6, 5, 9, 0, 4, 5, 8, 5 }, { 6, 7, 2, 8, 0, 2, 4, 0, 0, 4, 0 }, { 4, 4, 3, 5, 6, 0, 5, 0, 0, 7, 1 }, { 7, 8, 9, 3, 9, 8, 8, 4, 7, 0, 4 }, { 1, 3, 0, 6, 7, 1, 5, 0, 1, 4, 0 }, }

22.	{
	{ 0, 7, 7, 9, 0, 4, 7, 6, 4, 4, 1, 4 },
	{ 7, 0, 7, 5, 5, 2, 1, 1, 8, 5, 9, 0 },
	{ 7, 7, 0, 8, 9, 0, 9, 8, 9, 7, 4, 1 },
	{ 9, 5, 8, 0, 7, 3, 9, 6, 5, 5, 5, 2 },
	{ 0, 5, 9, 7, 0, 9, 9, 3, 5, 9, 0, 2 },
	{ 4, 2, 0, 3, 9, 0, 3, 2, 3, 2, 9, 3 },
	{ 7, 1, 9, 9, 9, 3, 0, 2, 7, 2, 4, 7 },
	{ 6, 1, 8, 6, 3, 2, 2, 0, 3, 3, 6, 9 },
	{ 4, 8, 9, 5, 5, 3, 7, 3, 0, 5, 6, 2 },
	{ 4, 5, 7, 5, 9, 2, 2, 3, 5, 0, 6, 4 },
	{ 1, 9, 4, 5, 0, 9, 4, 6, 6, 6, 0, 7 },
	{ 4, 0, 1, 2, 2, 3, 7, 9, 2, 4, 7, 0 },
	}

23.	{
	{ 0, 8, 2, 8, 9, 3, 7, 1, 5, 6 },
	{ 8, 0, 2, 9, 4, 7, 7, 1, 3, 5 },
	{ 2, 2, 0, 9, 5, 2, 9, 5, 9, 9 },
	{ 8, 9, 9, 0, 0, 9, 7, 7, 2, 8 },
	{ 9, 4, 5, 0, 0, 3, 2, 0, 3, 1 },
	{ 3, 7, 2, 9, 3, 0, 3, 0, 5, 9 },
	{ 7, 7, 9, 7, 2, 3, 0, 8, 7, 0 },
	{ 1, 1, 5, 7, 0, 0, 8, 0, 2, 6 },
	{ 5, 3, 9, 2, 3, 5, 7, 2, 0, 7 },
	{ 6, 5, 9, 8, 1, 9, 0, 6, 7, 0 },
	}

24.	{

	{ 0, 7, 7, 7, 0, 2, 4, 4, 3, 2, 3 }, { 7, 0, 0, 7, 1, 3, 0, 5, 0, 4, 6 }, { 7, 0, 0, 4, 6, 7, 0, 3, 9, 4, 1 }, { 7, 7, 4, 0, 4, 8, 4, 8, 5, 3, 5 }, { 0, 1, 6, 4, 0, 5, 5, 2, 2, 9, 6 }, { 2, 3, 7, 8, 5, 0, 7, 1, 5, 9, 5 }, { 4, 0, 0, 4, 5, 7, 0, 8, 6, 5, 2 }, { 4, 5, 3, 8, 2, 1, 8, 0, 2, 2, 8 }, { 3, 0, 9, 5, 2, 5, 6, 2, 0, 9, 4 }, { 2, 4, 4, 3, 9, 9, 5, 2, 9, 0, 2 }, { 3, 6, 1, 5, 6, 5, 2, 8, 4, 2, 0 }, }

25.	{
	{ 0, 1, 4, 4, 8, 5, 6, 7, 5, 2, 4, 2 },
	{ 1, 0, 6, 9, 1, 2, 3, 1, 2, 8, 9, 5 },
	{ 4, 6, 0, 7, 4, 8, 9, 6, 2, 6, 7, 6 },
	{ 4, 9, 7, 0, 2, 0, 0, 8, 8, 8, 7, 8 },
	{ 8, 1, 4, 2, 0, 3, 9, 2, 7, 7, 3, 1 },
	{ 5, 2, 8, 0, 3, 0, 0, 6, 4, 4, 5, 3 },
	{ 6, 3, 9, 0, 9, 0, 0, 2, 2, 9, 2, 3 },
	{ 7, 1, 6, 8, 2, 6, 2, 0, 7, 4, 2, 6 },
	{ 5, 2, 2, 8, 7, 4, 2, 7, 0, 3, 4, 6 },
	{ 2, 8, 6, 8, 7, 4, 9, 4, 3, 0, 3, 4 },
	{ 4, 9, 7, 7, 3, 5, 2, 2, 4, 3, 0, 7 },
	{ 2, 5, 6, 8, 1, 3, 3, 6, 6, 4, 7, 0 },
	}

26.	{ { 0, 3, 8, 7, 6, 6, 0, 2, 0, 0 }, { 3, 0, 9, 6, 3, 9, 9, 5, 1, 4 }, { 8, 9, 0, 4, 2, 4, 9, 8, 8, 0 }, { 7, 6, 4, 0, 5, 8, 5, 0, 3, 7 }, { 6, 3, 2, 5, 0, 2, 8, 8, 9, 4 }, { 6, 9, 4, 8, 2, 0, 6, 9, 7, 6 }, { 0, 9, 9, 5, 8, 6, 0, 1, 8, 4 }, { 2, 5, 8, 0, 8, 9, 1, 0, 6, 7 }, { 0, 1, 8, 3, 9, 7, 8, 6, 0, 6 }, { 0, 4, 0, 7, 4, 6, 4, 7, 6, 0 },

	}

27.	{
	{ 0, 9, 4, 3, 8, 7, 8, 2, 1 },
	{ 9, 0, 8, 2, 1, 0, 7, 9, 5 },
	{ 4, 8, 0, 8, 4, 4, 7, 5, 5 },
	{ 3, 2, 8, 0, 3, 4, 6, 2, 6 },
	{ 8, 1, 4, 3, 0, 5, 3, 2, 2 },
	{ 7, 0, 4, 4, 5, 0, 5, 2, 6 },
	{ 8, 7, 7, 6, 3, 5, 0, 0, 0 },
	{ 2, 9, 5, 2, 2, 2, 0, 0, 6 },
	{ 1, 5, 5, 6, 2, 6, 0, 6, 0 },
	}

28.	{
	{ 0, 7, 6, 4, 7, 0, 9, 2, 7 },
	{ 7, 0, 7, 6, 8, 5, 7, 4, 6 },
	{ 6, 7, 0, 6, 2, 1, 8, 6, 0 },
	{ 4, 6, 6, 0, 5, 8, 4, 7, 1 },
	{ 7, 8, 2, 5, 0, 0, 0, 2, 5 },
	{ 0, 5, 1, 8, 0, 0, 2, 2, 3 },
	{ 9, 7, 8, 4, 0, 2, 0, 6, 1 },
	{ 2, 4, 6, 7, 2, 2, 6, 0, 3 },
	{ 7, 6, 0, 1, 5, 3, 1, 3, 0 },
	}

29.	{
	{ 0, 9, 9, 3, 9, 6, 2, 9, 1, 5, 7 },
	{ 9, 0, 4, 3, 1, 3, 3, 3, 2, 6, 0 },
	{ 9, 4, 0, 4, 6, 1, 7, 5, 6, 7, 6 },
	{ 3, 3, 4, 0, 7, 0, 6, 6, 9, 5, 9 },
	{ 9, 1, 6, 7, 0, 8, 2, 3, 7, 3, 8 },
	{ 6, 3, 1, 0, 8, 0, 6, 9, 3, 7, 2 },
	{ 2, 3, 7, 6, 2, 6, 0, 8, 3, 6, 6 },
	{ 9, 3, 5, 6, 3, 9, 8, 0, 3, 0, 3 },
	{ 1, 2, 6, 9, 7, 3, 3, 3, 0, 4, 0 },
	{ 5, 6, 7, 5, 3, 7, 6, 0, 4, 0, 8 },
	{ 7, 0, 6, 9, 8, 2, 6, 3, 0, 8, 0 },
	}

30.	{

	{ 0, 9, 9, 8, 5, 9, 5, 1, 0, 5, 3, 6 }, { 9, 0, 7, 4, 7, 5, 6, 6, 5, 8, 4, 3 }, { 9, 7, 0, 2, 7, 7, 6, 5, 8, 0, 8, 1 }, { 8, 4, 2, 0, 0, 9, 0, 9, 4, 0, 4, 8 }, { 5, 7, 7, 0, 0, 0, 1, 1, 8, 9, 7, 5 }, { 9, 5, 7, 9, 0, 0, 6, 5, 3, 2, 3, 7 }, { 5, 6, 6, 0, 1, 6, 0, 5, 7, 5, 4, 4 }, { 1, 6, 5, 9, 1, 5, 5, 0, 2, 2, 6, 2 }, { 0, 5, 8, 4, 8, 3, 7, 2, 0, 3, 8, 1 }, { 5, 8, 0, 0, 9, 2, 5, 2, 3, 0, 4, 7 }, { 3, 4, 8, 4, 7, 3, 4, 6, 8, 4, 0, 6 }, { 6, 3, 1, 8, 5, 7, 4, 2, 1, 7, 6, 0 }, }

Код 3.1. Обход в глубину и ширину графа, заданного с помощью матрицы смежности

```
#include <iostream>

using namespace std;
int main()
 {

	// матрица смежности

	vector<vector<int> > mat =

	{
		{0, 1, 2, 0, 0, 0, 0},
		{1, 0, 2, 0, 0, 0, 0},
		{2, 2, 0, 4, 0, 0, 1},
		{0, 0, 4, 0, 1, 2, 2},
		{0, 0, 0, 1, 0, 1, 0},
		{0, 0, 0, 2, 1, 0, 0},
		{0, 0, 1, 2, 0, 0, 0}
	};

	vector <int> used(7, 0);
	//0 – вершина не посещена при поиске, 1 – помещена в структуру данных для вершин,
	//но не обработана, 2 – обработана, смежные вершины помещены в структуру данных
	//DFS – поиск в глубину
	stack<int> Stack;
	int iter = 0;

	Stack.push(0); // помещаем в очередь первую вершину
	while (!Stack.empty())
```

```
	{ // пока стек не пуст
		int node = Stack.top(); // извлекаем вершину
		Stack.pop();

		std::cout << "\nDFS at vertex " << node<< endl;

		if (used[node] == 2) continue;
		used[node] = 2; // отмечаем ее как посещенную
		iter++;
		for (int j = 0; j < 7; j++)
		{ // проверяем для нее все смежные вершины
			if (mat[node][j] > 0 && used[j] != 2)
			{ // если вершина смежная и не обнаружена
				Stack.push(j); // добавляем ее в стек
				used[j] = 1; // отмечаем вершину как обнаруженную
			}
		}
		std::cout << node << endl; // выводим номер вершины
	}
	std::cout << "\nVisited vertices";
	for (int i = 0; i < 7; i++) std::cout << used[i] << " ";

	for (int i = 0; i < 7; i++)
		used[i] = 0;
	queue<int> Queue;

	//BFS – поиск в ширину
	Queue.push(0);              //в качестве начальной вершины используем 0.
	used[0] = 2;
	vector <int> dist(7, 10000); //расстояния до вершин от 0-й в числе ребер
	dist[0] = 0;
	 iter = 0;
	while (!Queue.empty())
	{
		int node = Queue.front();  //извлекаем из очереди текущую вершину
		Queue.pop();

		//Здесь должна быть обработка текущей вершины.
		std::cout << "\nBFS at vertex " << node<< endl;
		if (used[node] == 2) continue;
			used[node] = 2;
		iter++;
		for (int j = 0; j < 7; j++)
		{ // проверяем для нее все смежные вершины
			if (mat[node][j] > 0 && used[j] != 2)
			{ // если вершина смежная и не обнаружена
				Queue.push(j); // добавляем ее в очередь
				used[j] = 1; // отмечаем вершину как обнаруженную
				if (dist[j] > dist[node] + 1)
					dist[j] = dist[node] + 1;
			}
		}
	}
	std::cout << "\nVisited vertices";
```

```
        for (int i = 0; i < 7; i++) std::cout << used[i] << " ";
        std::cout << "\nDistances";
        for (int i = 0; i < 7; i++) std::cout << dist[i] << " ";


        char c; cin >> c;
        return 0;
}
```