Задание

1. Повторить теоретический материал, решить задачи для самостоятельного решения.

2. Фотоотчет присылать на электронную почту

С уважением, Хвастова Светлана Ивановна

!!! Если возникнут вопросы обращаться по телефону 0721389311.

Электронная почта: xvsviv@rambler.ru

Практическая работа№13. Поиск путей (маршрутов) с минимальным числом дуг (ребер).

Цель: изучить алгоритм Дейкстры для поиска маршрутов с минимальным числом дуг

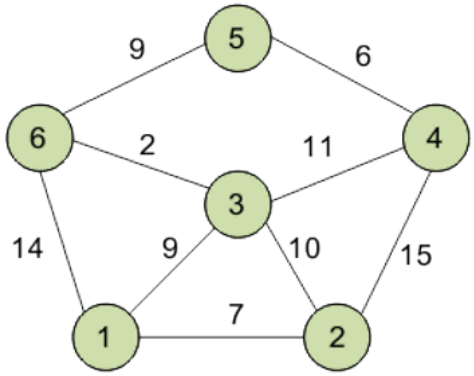
Алгоритм Дейкстры нахождения кратчайшего пути

Рассмотрим пример нахождение кратчайшего пути. Дана сеть автомобильных дорог, соединяющих области города. Некоторые дороги односторонние. Найти кратчайшие пути от центра города до каждого города области.

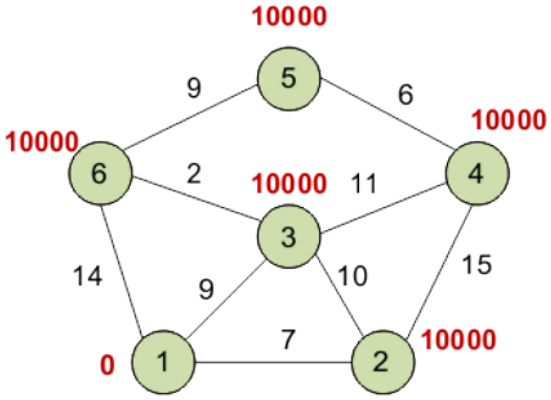
Для решения указанной задачи можно использовать алгоритм Дейкстры - алгоритм на графах, изобретённый нидерландским ученым Э. Дейкстрой в 1959 году. Находит кратчайшее расстояние от одной из вершин графа до всех остальных. Работает только для графов без рёбер отрицательного веса.

Пусть требуется найти кратчайшие расстояния от 1-й вершины до всех остальных.

Кружками обозначены вершины, линиями – пути между ними (ребра графа). В кружках обозначены номера вершин, над ребрами обозначен их вес – длина пути. Рядом с каждой вершиной красным обозначена метка – длина кратчайшего пути в эту вершину из вершины 1.

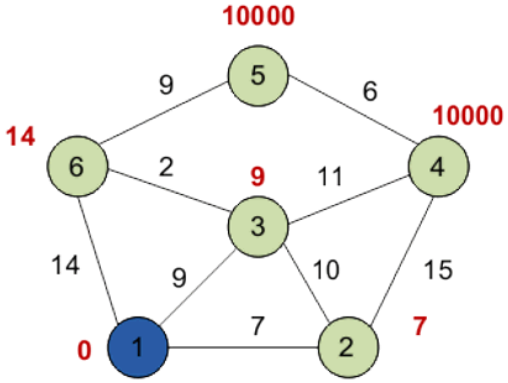


Инициализация. Метка самой вершины 1 полагается равной 0, метки остальных вершин – недостижимо большое число (в идеале - бесконечность). Это отражает то, что расстояния от вершины 1 до других вершин пока неизвестны. Все вершины графа помечаются как не посещённые.



Первый шаг. Минимальную метку имеет вершина 1. Её соседями являются вершины 2, 3 и 6. Обходим соседей вершины по очереди.

Первый сосед вершины 1 – вершина 2, потому что длина пути до неё минимальна. Длина пути в неё через вершину 1 равна сумме кратчайшего расстояния до вершины 1, значению её метки, и длины ребра, идущего из 1-й в 2-ю, то есть 0 + 7 = 7. Это меньше текущей метки вершины 2 (10000), поэтому новая метка 2-й вершины равна 7.



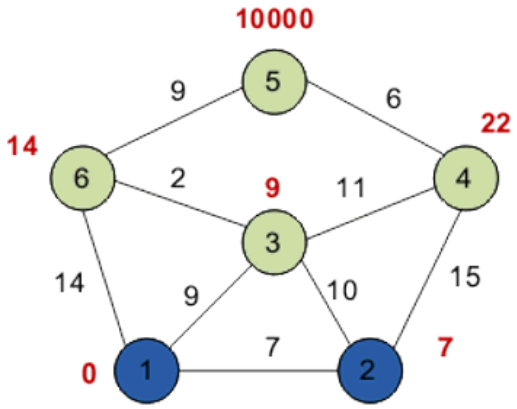
Аналогично находим длины пути для всех других соседей (вершины 3 и 6).

Все соседи вершины 1 проверены. Текущее минимальное расстояние до вершины 1 считается окончательным и пересмотру не подлежит. Вершина 1 отмечается как посещенная.

Второй шаг. Шаг 1 алгоритма повторяется. Снова находим «ближайшую» из непосещенных вершин. Это вершина 2 с меткой 7.

Снова пытаемся уменьшить метки соседей выбранной вершины, пытаясь пройти в них через 2-ю вершину. Соседями вершины 2 являются вершины 1, 3 и 4.

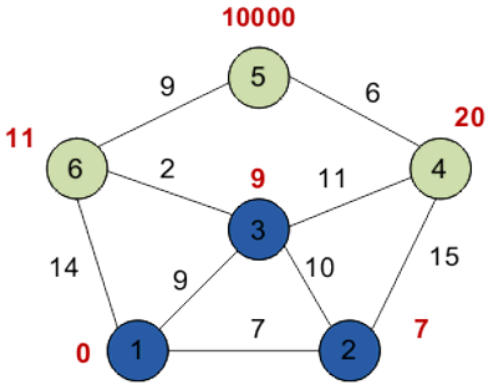
Вершина 1 уже посещена. Следующий сосед вершины 2 — вершина 3, так как имеет минимальную метку из вершин, отмеченных как не посещённые. Если идти в неё через 2, то длина такого пути будет равна 17 (7 + 10 = 17). Но текущая метка третьей вершины равна 9, а 9 < 17, поэтому метка не меняется.



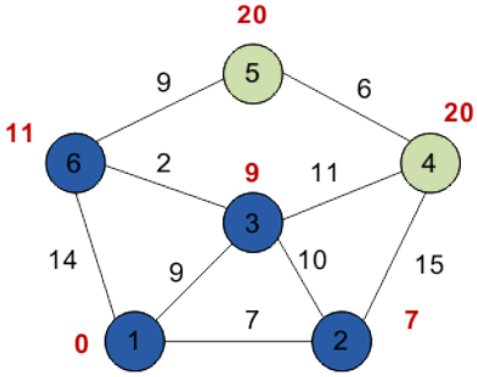
Ещё один сосед вершины 2 — вершина 4. Если идти в неё через 2-ю, то длина такого пути будет равна 22 (7 + 15 = 22). Поскольку 22<10000, устанавливаем метку вершины 4 равной 22.

Все соседи вершины 2 просмотрены, помечаем её как посещенную.

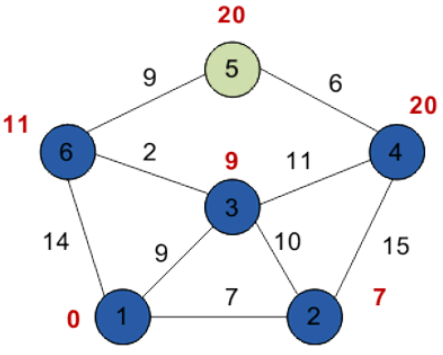
Третий шаг. Повторяем шаг алгоритма, выбрав вершину 3. После её «обработки» получим следующие результаты.



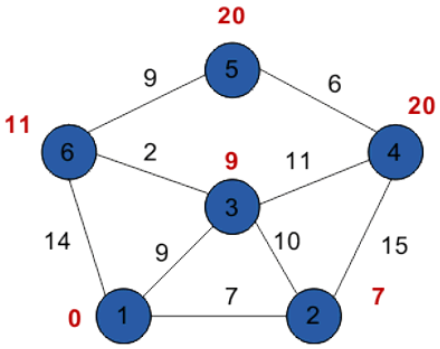
Четвертый шаг



Пятый шаг



Шестой шаг



Таким образом, кратчайшим путем из вершины 1 в вершину 5 будет путь через вершины 1 - 3 - 6 - 5, поскольку таким путем мы набираем минимальный вес, равный 20.

Реализация алгоритма Дейкстры

Для хранения весов графа используется квадратная матрица. В заголовках строк и столбцов находятся вершины графа. А веса дуг графа размещаются во внутренних ячейках таблицы. Граф не содержит петель, поэтому на главной диагонали матрицы содержатся нулевые значения.

	1	2	3	4	5	6
1	0	7	9	0	0	14
2	7	0	10	15	0	0
3	9	10	0	11	0	2
4	0	15	11	0	6	0
5	0	0	0	6	0	9
6	14	0	2	0	9	0

Задание для самостоятельной работы:

Взвешенный граф задан с помощью матрицы смежности, построить граф, и найти минимальный путь от вершины 1 до вершины 5 с помощью алгоритма Дейкстры.

Вариант 1

	1	**2**	**3**	**4**	**5**
1	0	1	7	4	7
2	1	0	5	5	3
3	7	5	0	4	2
4	4	5	4	0	6
5	7	3	2	6	0

Вариант 2

	1	**2**	**3**	**4**	**5**
1	0	10	7	12	7
2	10	0	14	5	3
3	7	14	0	8	6
4	12	5	8	0	6
5	7	3	6	6	0

Вариант 3

	1	**2**	**3**	**4**	**5**
1	0	1	7	4	7
2	1	0	2	5	3
3	7	2	0	4	2
4	4	5	4	0	6
5	7	3	2	6	0

Вариант 4

	1	**2**	**3**	**4**	**5**
1	0	1	5	2	7
2	1	0	4	5	3
3	5	4	0	4	2
4	2	5	4	0	3
5	7	3	2	3	0

Вариант 5

	1	**2**	**3**	**4**	**5**
1	0	7	6	9	7
2	7	0	5	5	3
3	6	5	0	4	2
4	9	5	4	0	5
5	7	3	2	5	0

Вариант 6

	1	**2**	**3**	**4**	**5**
1	0	7	6	11	7
2	7	0	1	8	3
3	6	1	0	4	2
4	11	8	4	0	6
5	7	3	2	6	0

Вариант 7

	1	**2**	**3**	**4**	**5**
1	0	3	6	9	7
2	3	0	4	5	3
3	6	4	0	5	7
4	9	5	5	0	6
5	7	3	7	6	0

Вариант 8

	1	**2**	**3**	**4**	**5**
1	0	10	16	12	7
2	10	0	14	5	3
3	16	14	0	4	12
4	12	5	4	0	6
5	7	3	12	6	0

Вариант 9

	1	**2**	**3**	**4**	**5**
1	0	3	6	2	7
2	3	0	4	5	3
3	6	4	0	4	2
4	2	5	4	0	2
5	7	3	2	2	0

Вариант 10

	1	**2**	**3**	**4**	**5**
1	0	3	6	2	7
2	3	0	4	5	3
3	6	4	0	4	2
4	2	5	4	0	2
5	7	3	2	2	0

Вариант 11

	1	**2**	**3**	**4**	**5**
1	0	10	6	12	7
2	10	0	14	5	3
3	6	14	0	10	2
4	12	5	10	0	6
5	7	3	2	6	0

Вариант 12

	1	**2**	**3**	**4**	**5**
1	0	10	6	12	7
2	10	0	14	5	3
3	6	14	0	10	2
4	12	5	10	0	6
5	7	3	2	6	0

Вариант 13

	1	**2**	**3**	**4**	**5**
1	0	10	16	12	7
2	10	0	9	5	3
3	16	9	0	4	12
4	12	5	4	0	8
5	7	3	12	8	0

Вариант 14

	1	**2**	**3**	**4**	**5**
1	0	8	6	12	7
2	8	0	4	7	9
3	6	4	0	4	2
4	12	7	4	0	6
5	7	9	2	6	0

Вариант 15

	1	**2**	**3**	**4**	**5**
1	0	4	6	12	7
2	4	0	10	5	3
3	6	10	0	12	2
4	12	5	12	0	6
5	7	3	2	6	0