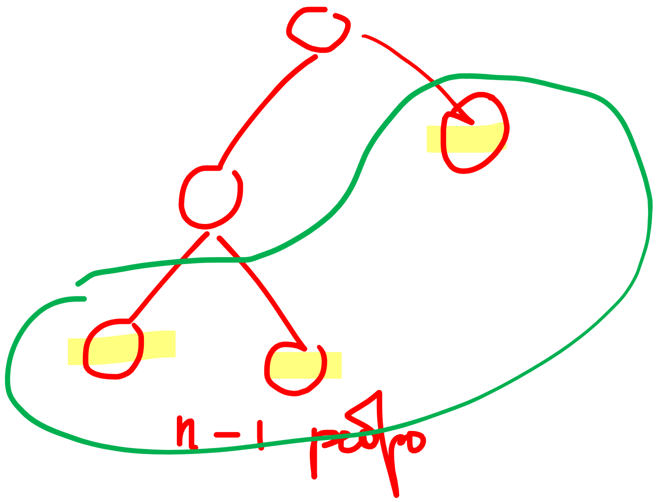
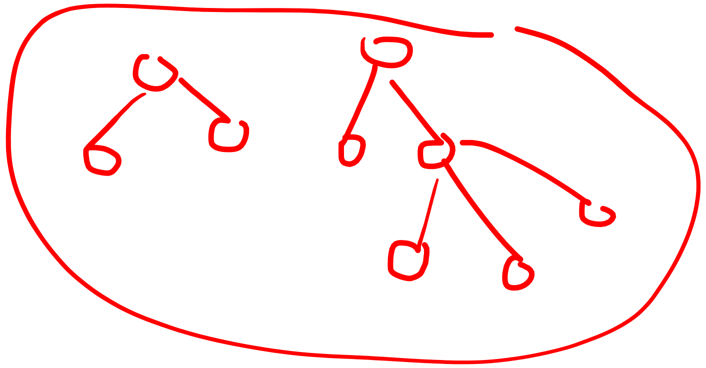
Остов минимального веса. Алгоритм Прима. Алгоритм Краскала. Система непересекающихся множеств.

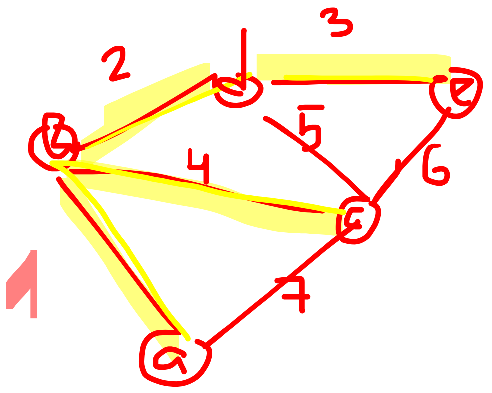
Определения

- **Дерево**
- **Лес**
- **Лист** ст. 1
- **Крона**
- **Каркас (остов)**



n − 1 ребро

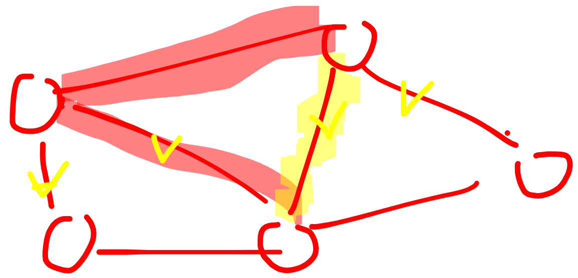




Алгоритм Прима

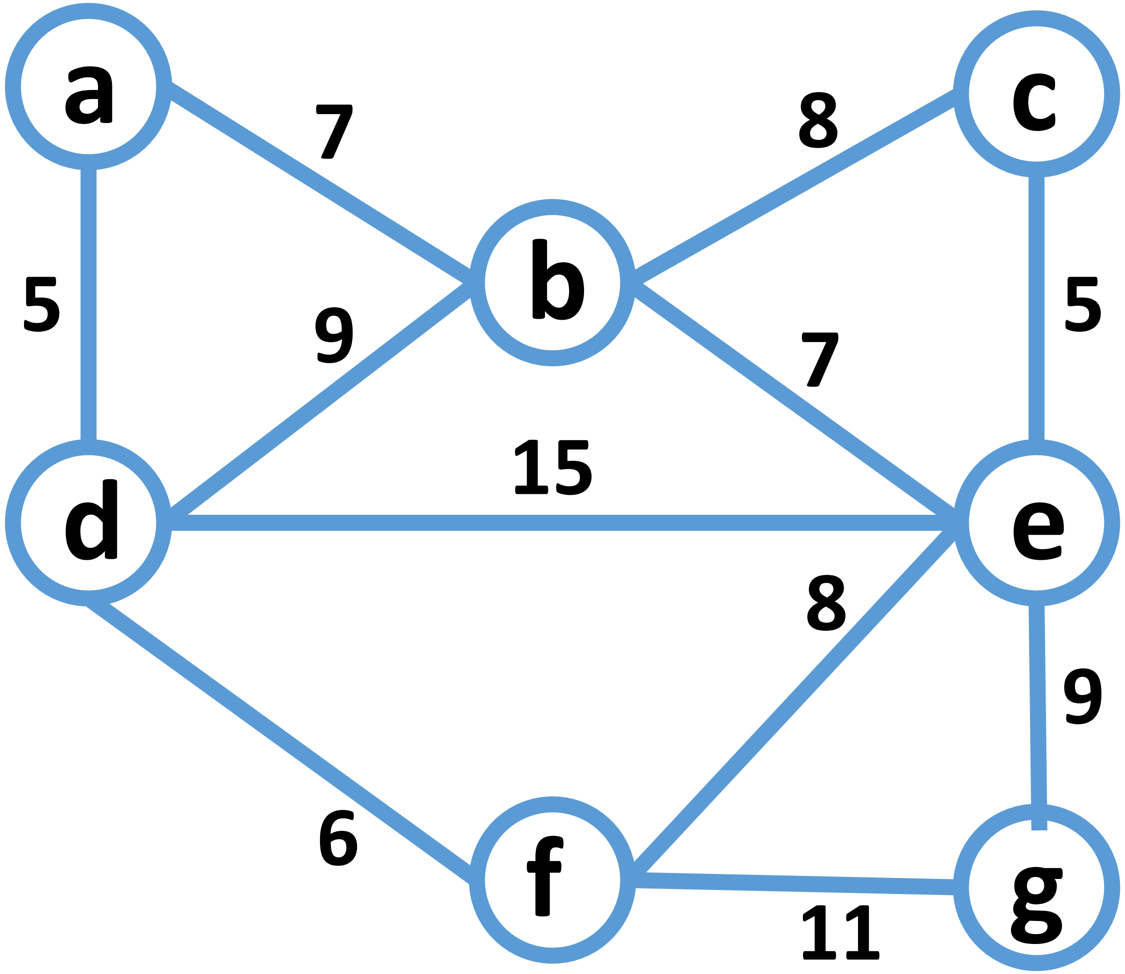
Алгоритм Прима — алгоритм построения минимального остовного дерева взвешенного связного неориентированного графа. Алгоритм впервые был открыт в 1930 году чешским математиком Войцехом Ярником, позже переоткрыт Робертом Примом в 1957 году, и, независимо от них, Э. Дейкстрой в 1959 году.

Алгоритм Прима

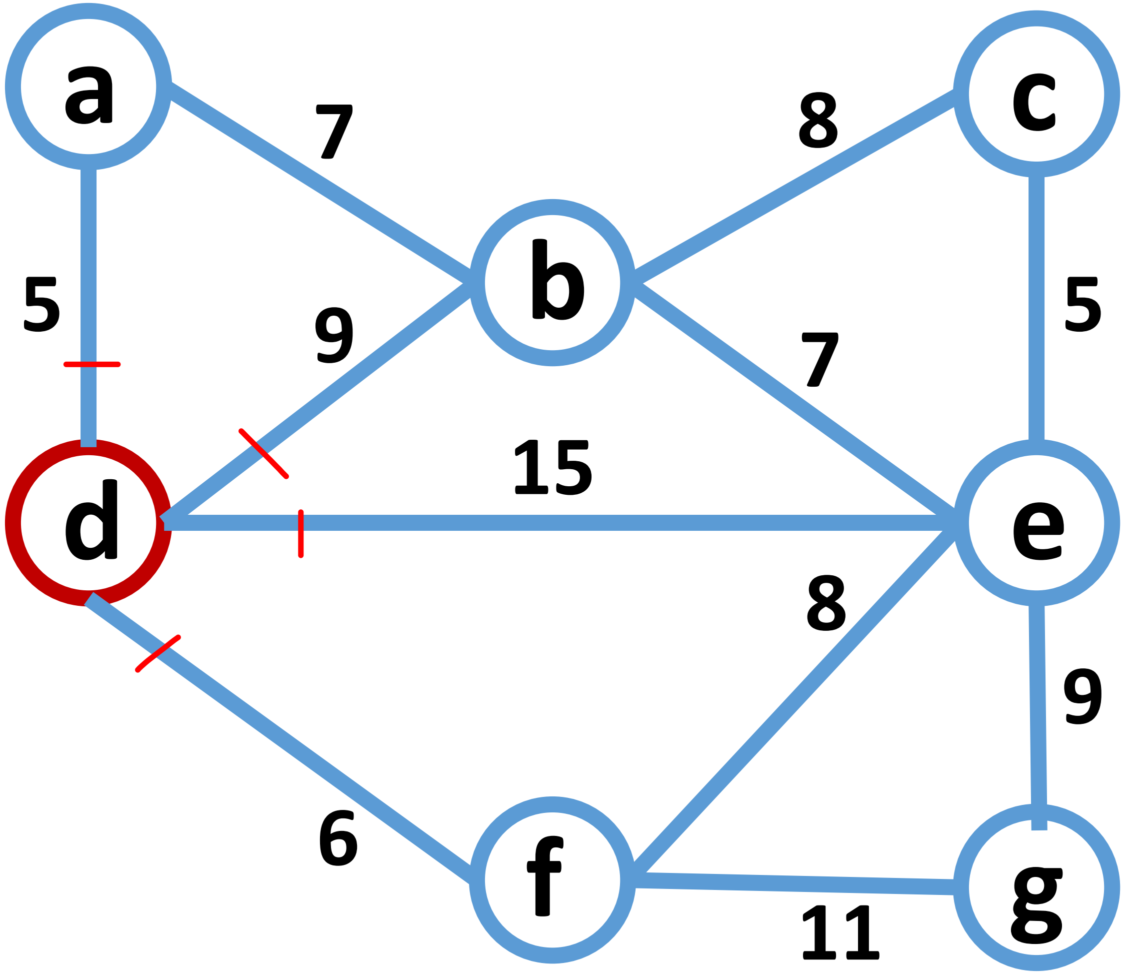


- На вход алгоритма подаётся связный неориентированный граф. Для каждого ребра задаётся его стоимость.
- Сначала берётся произвольная вершина и находится ребро, инцидентное данной вершине и обладающее наименьшей стоимостью. Найденное ребро и соединяемые им две вершины образуют дерево. Затем, рассматриваются рёбра графа, один конец которых — уже принадлежащая дереву вершина, а другой — нет; из этих рёбер выбирается ребро наименьшей стоимости. Выбираемое на каждом шаге ребро присоединяется к дереву. Рост дерева происходит до тех пор, пока не будут исчерпаны все вершины исходного графа. n-1
- Результатом работы алгоритма является остовное дерево минимальной стоимости.

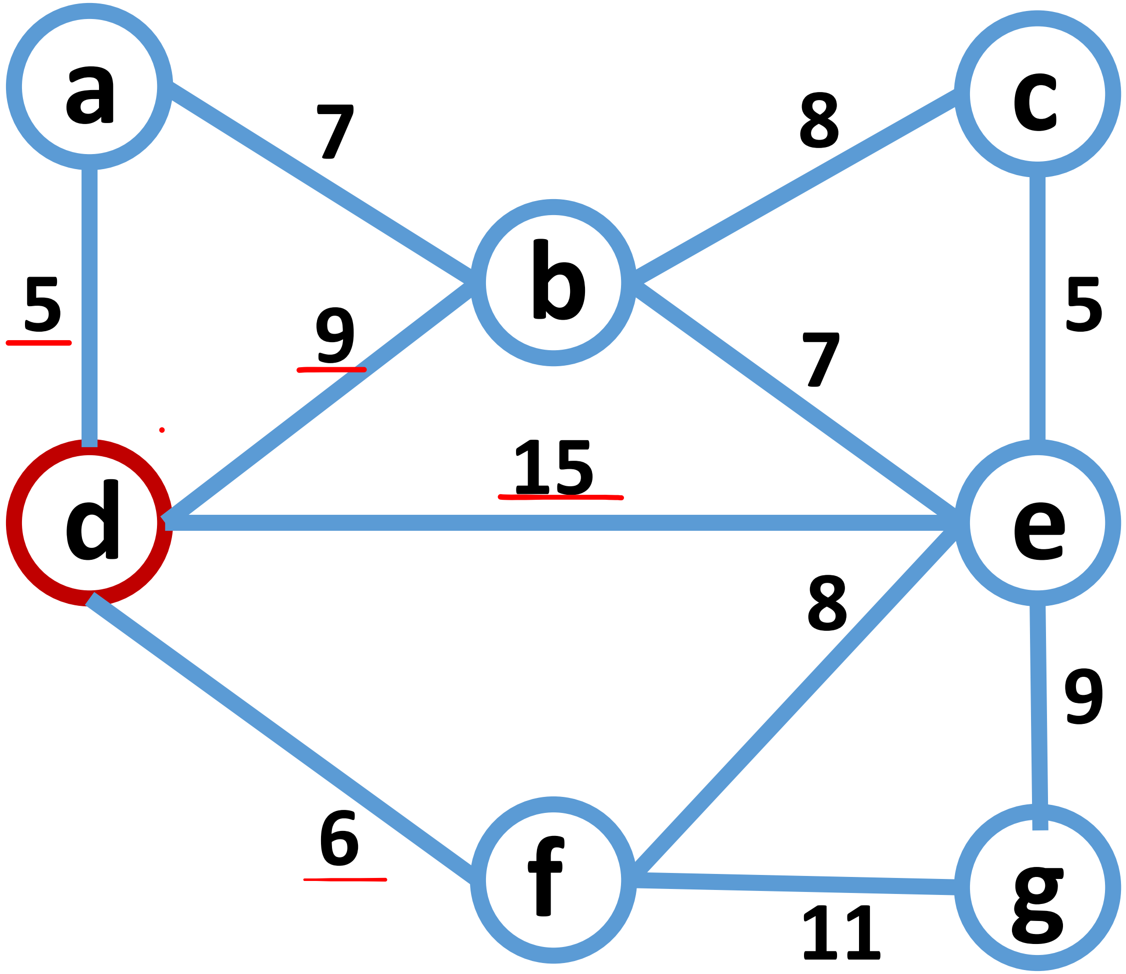
http://www.e-maxx-ru.1gb.ru/algo/mst_prim



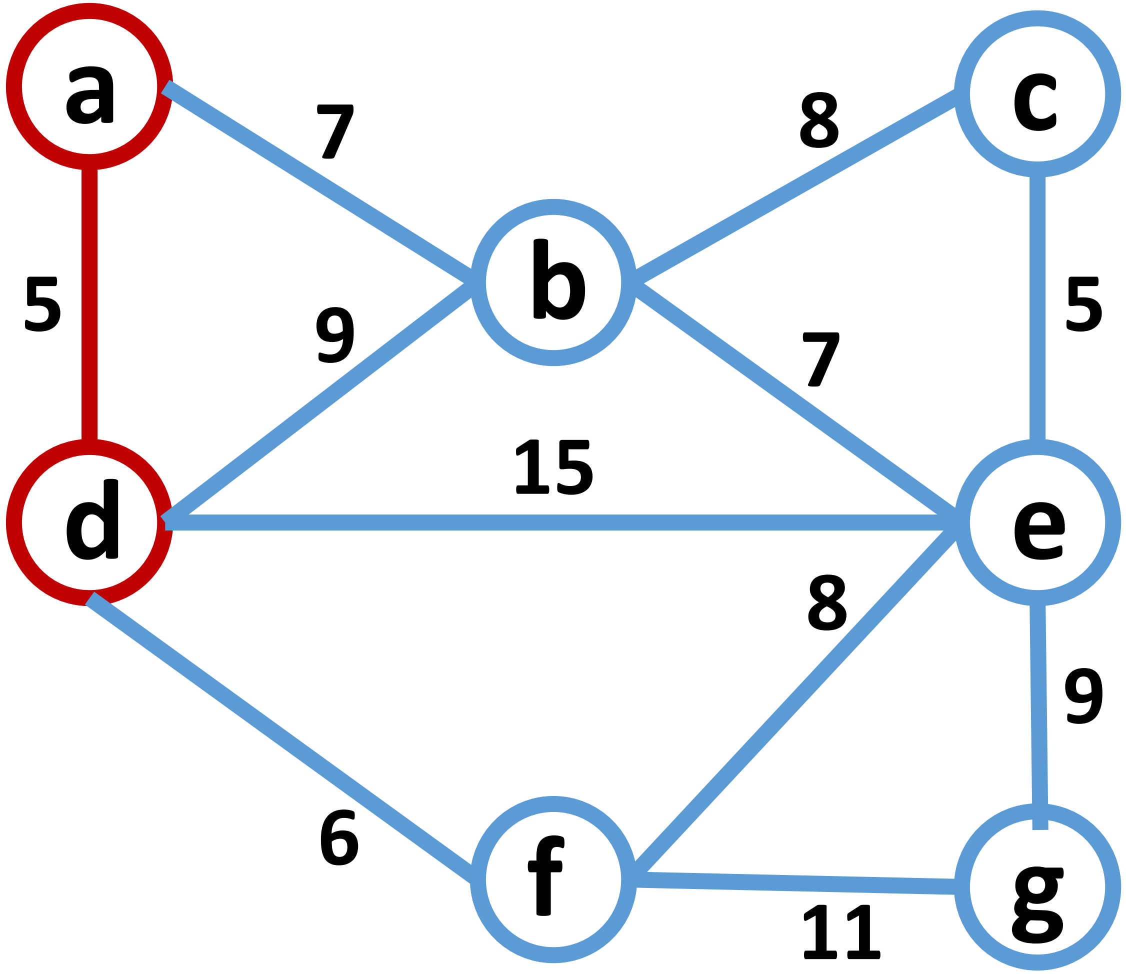
Выбранные вершины U	Ребро (u, v)	Невыбранные вершины V\U
{}		{a,b,c,d,e,f,g}



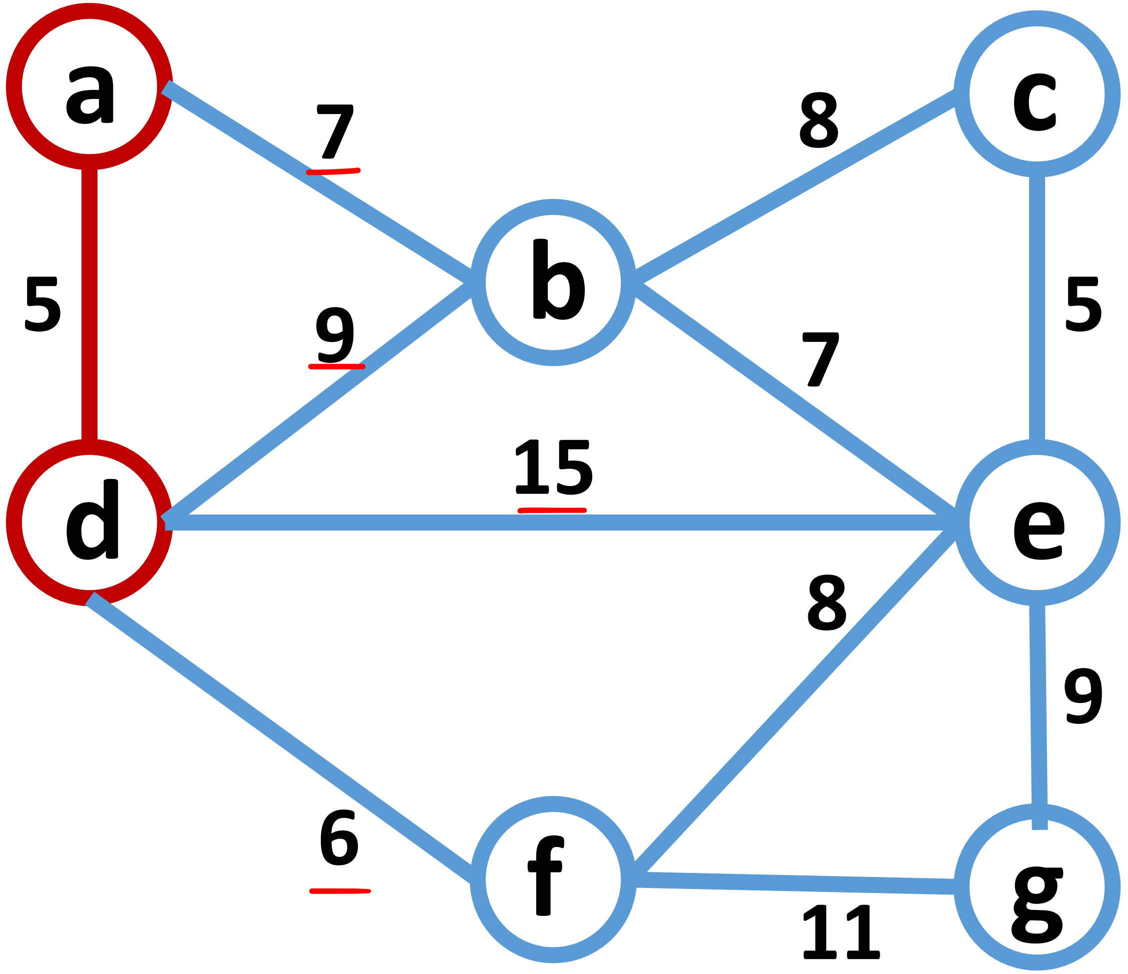
Выбранные вершины U	Ребро (u, v)	Невыбранные вершины V\U
{}		{a,b,c,d,e,f,g}
{d}		



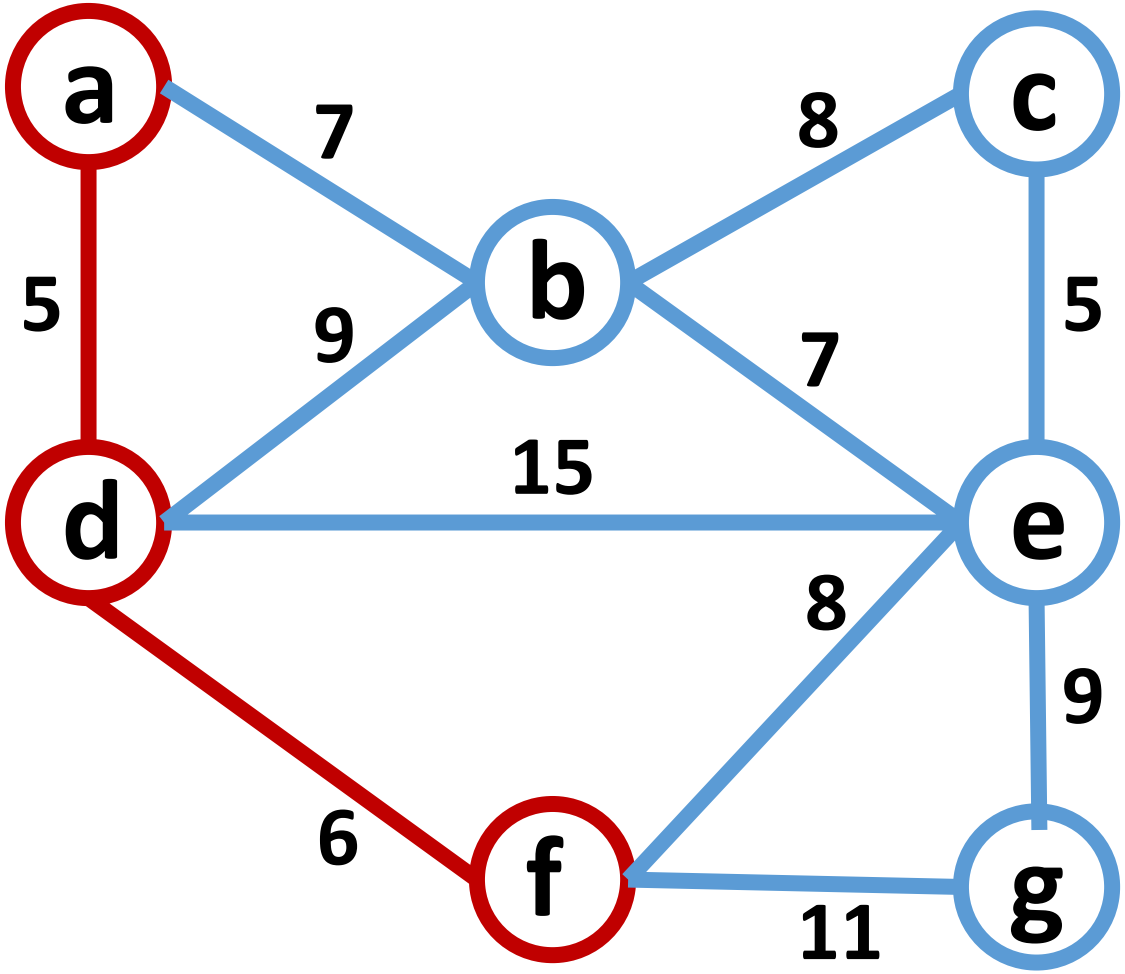
Выбранные вершины U	Ребро (u, v)	Невыбранные вершины V\U
{}		{a,b,c,d,e,f,g}
{d}	5, 6, 9, 15	{a,b,c,e,f,g}



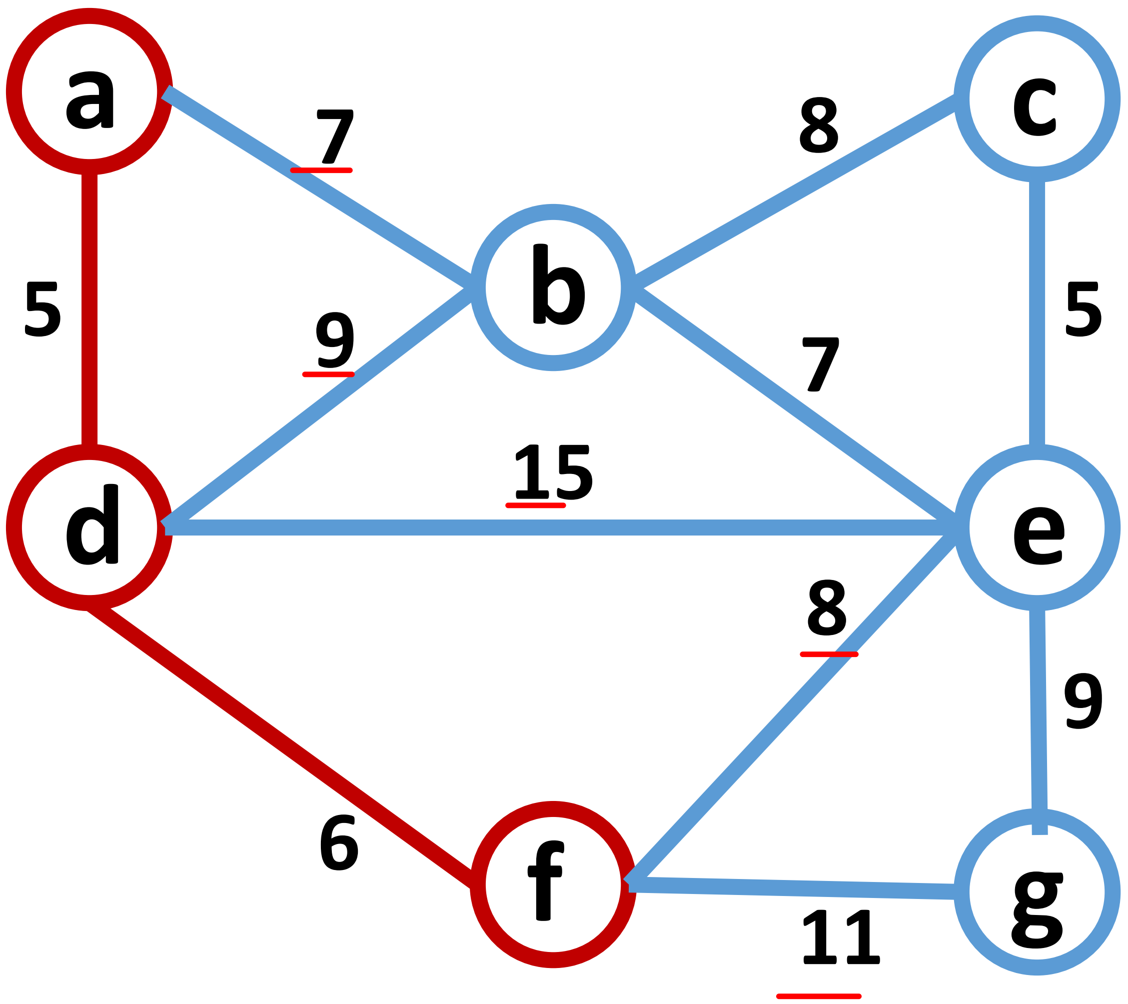
Выбранные вершины U	Ребро (u, v)	Невыбранные вершины V\U
{}		{a,b,c,d,e,f,g}
{d}	5, 6, 9, 15	{a,b,c,e,f,g}
{a,d}		



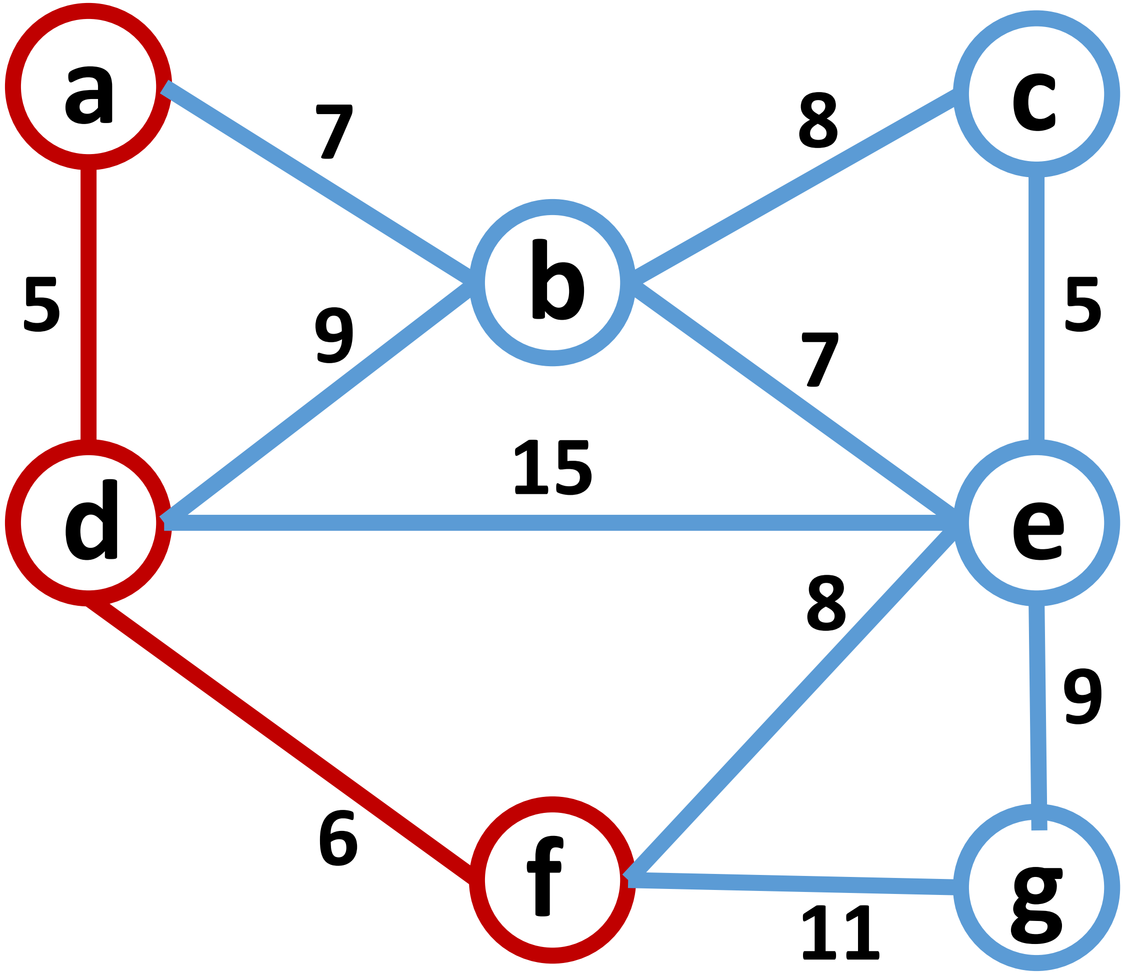
Выбранные вершины U	Ребро (u, v)	Невыбранные вершины V\U
{}		{a,b,c,d,e,f,g}
{d}	5, 6, 9, 15	{a,b,c,e,f,g}
{a,d}	7, 9, 6, 15	{b,c,e,f,g}



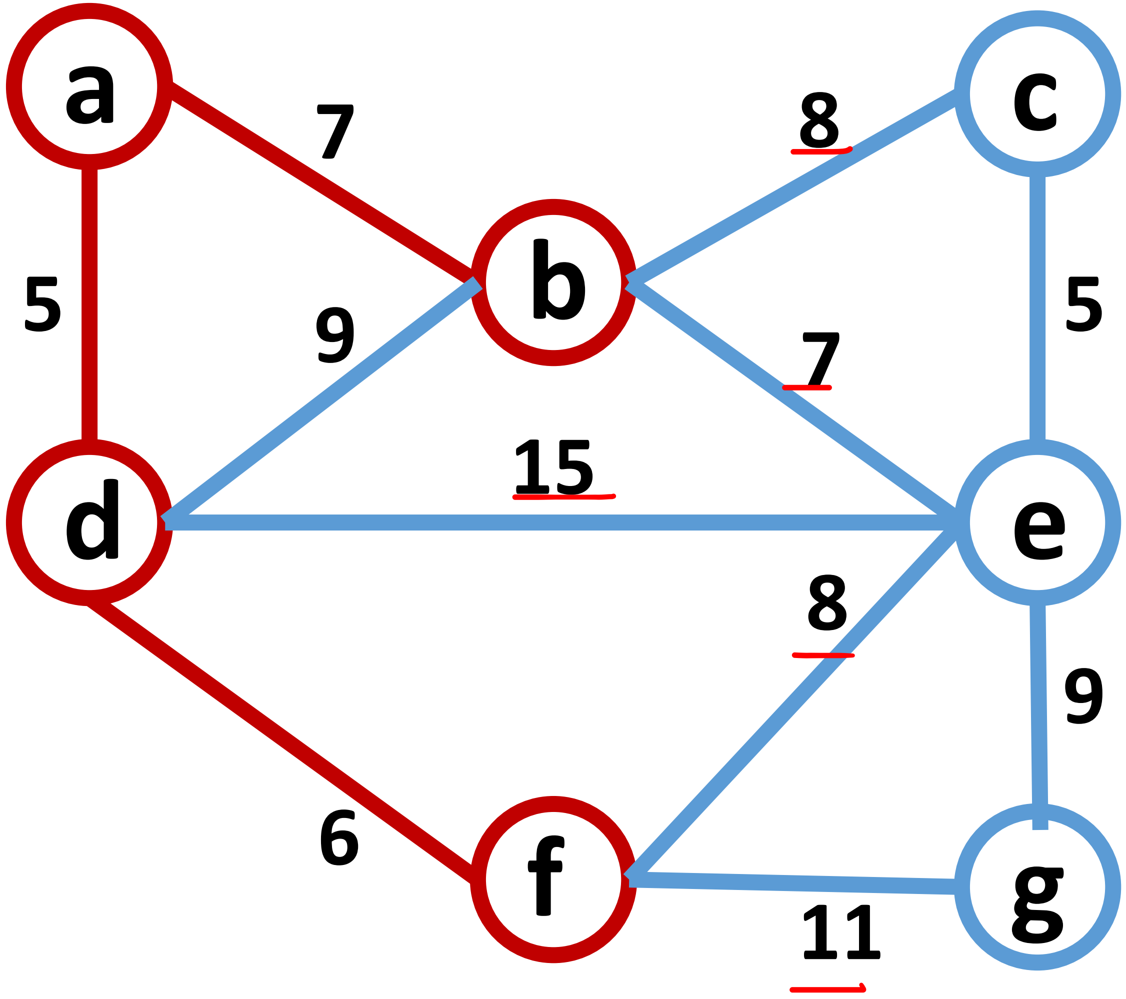
Выбранные вершины U	Ребро (u, v)	Невыбранные вершины V\U
{}		{a,b,c,d,e,f,g}
{d}	5, 6, 9, 15	{a,b,c,e,f,g}
{a,d}	7, 9, 6, 15	{b,c,e,f,g}



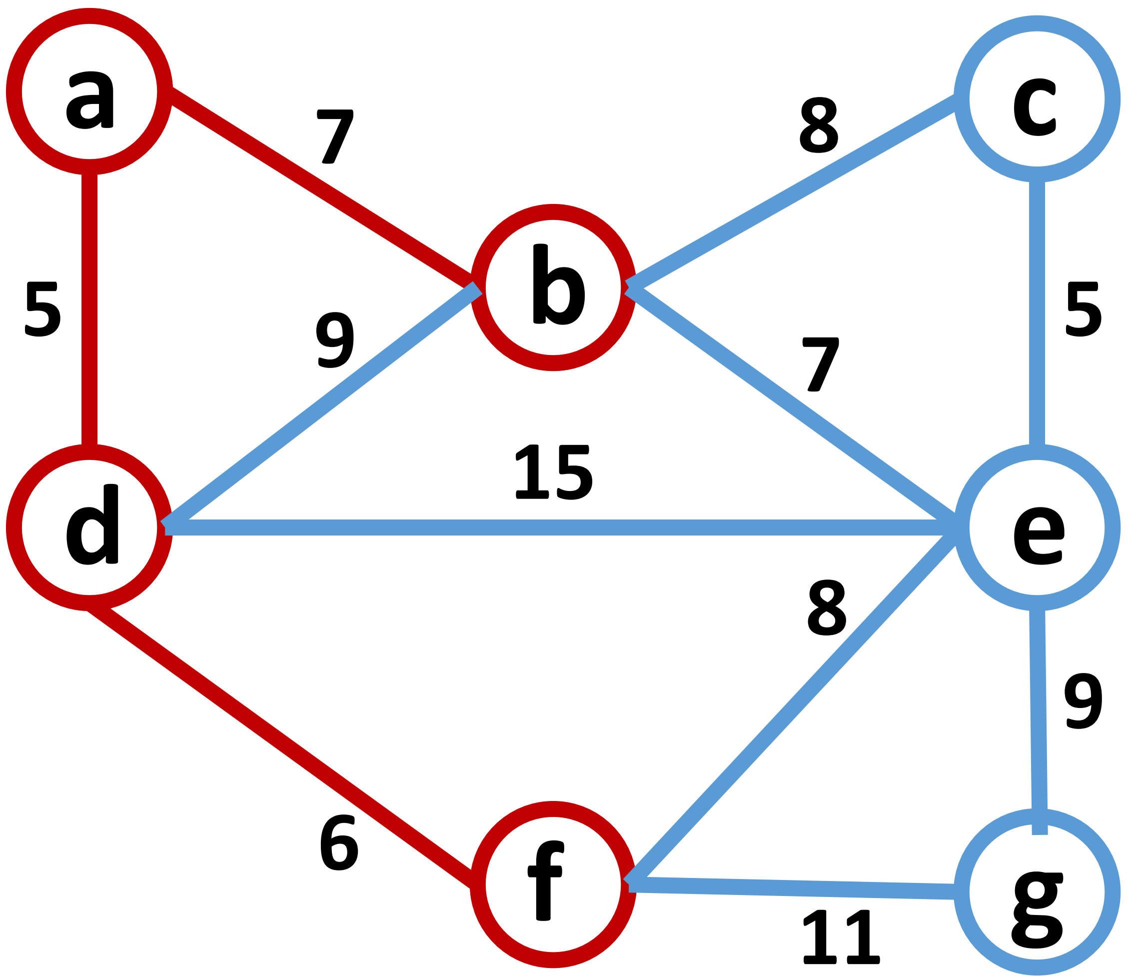
Выбранные вершины U	Ребро (u, v)	Невыбранные вершины V\U
{}		{a,b,c,d,e,f,g}
{d}	5, 6, 9, 15	{a,b,c,e,f,g}
{a,d}	7, 9, 6, 15	{b,c,e,f,g}
{a,d,f}		



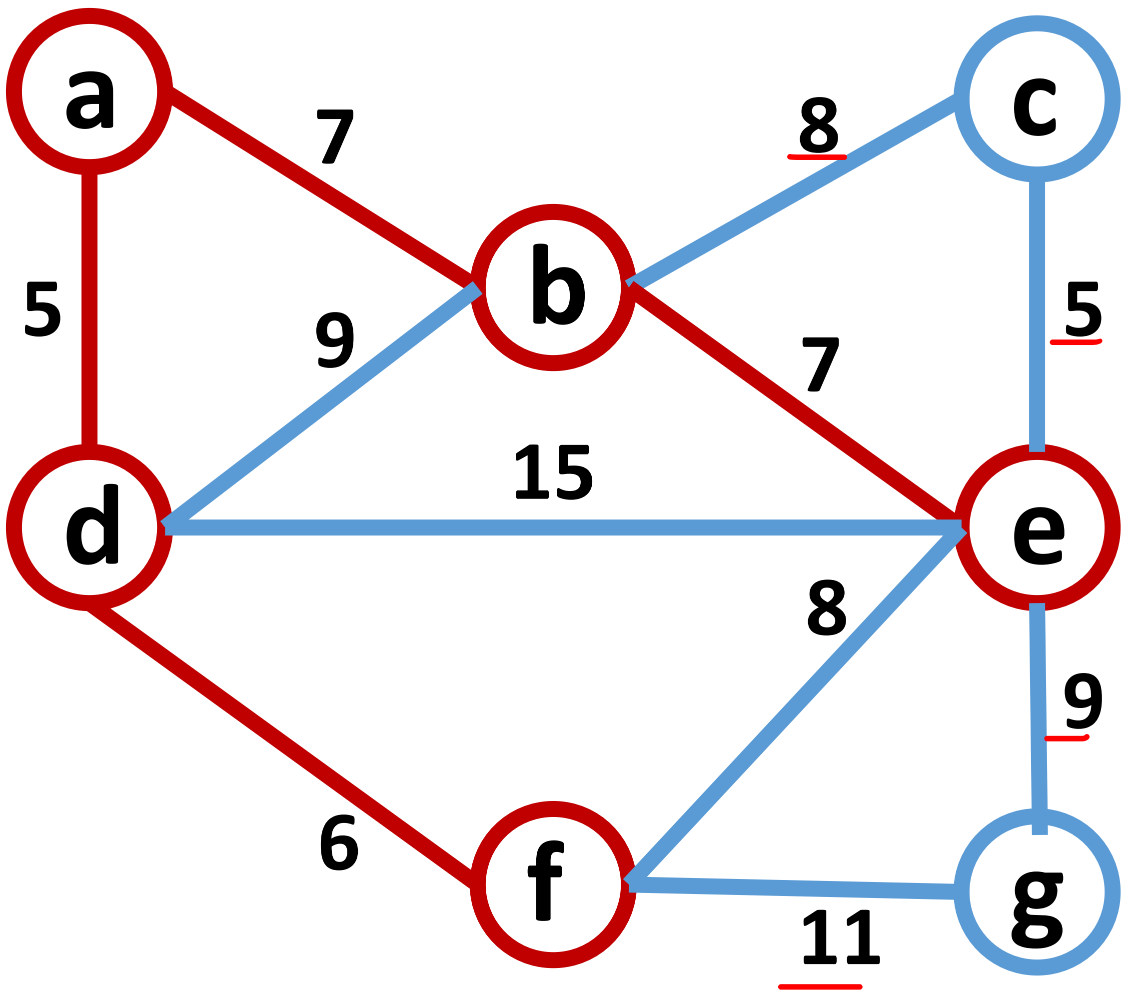
Выбранные вершины U	Ребро (u, v)	Невыбранные вершины V\U
{}		{a,b,c,d,e,f,g}
{d}	5, 6, 9, 15	{a,b,c,e,f,g}
{a,d}	7, 9, 6, 15	{b,c,e,f,g}
{a,d,f}	7, 9, 15, 8, 11	{b,c,e,g}



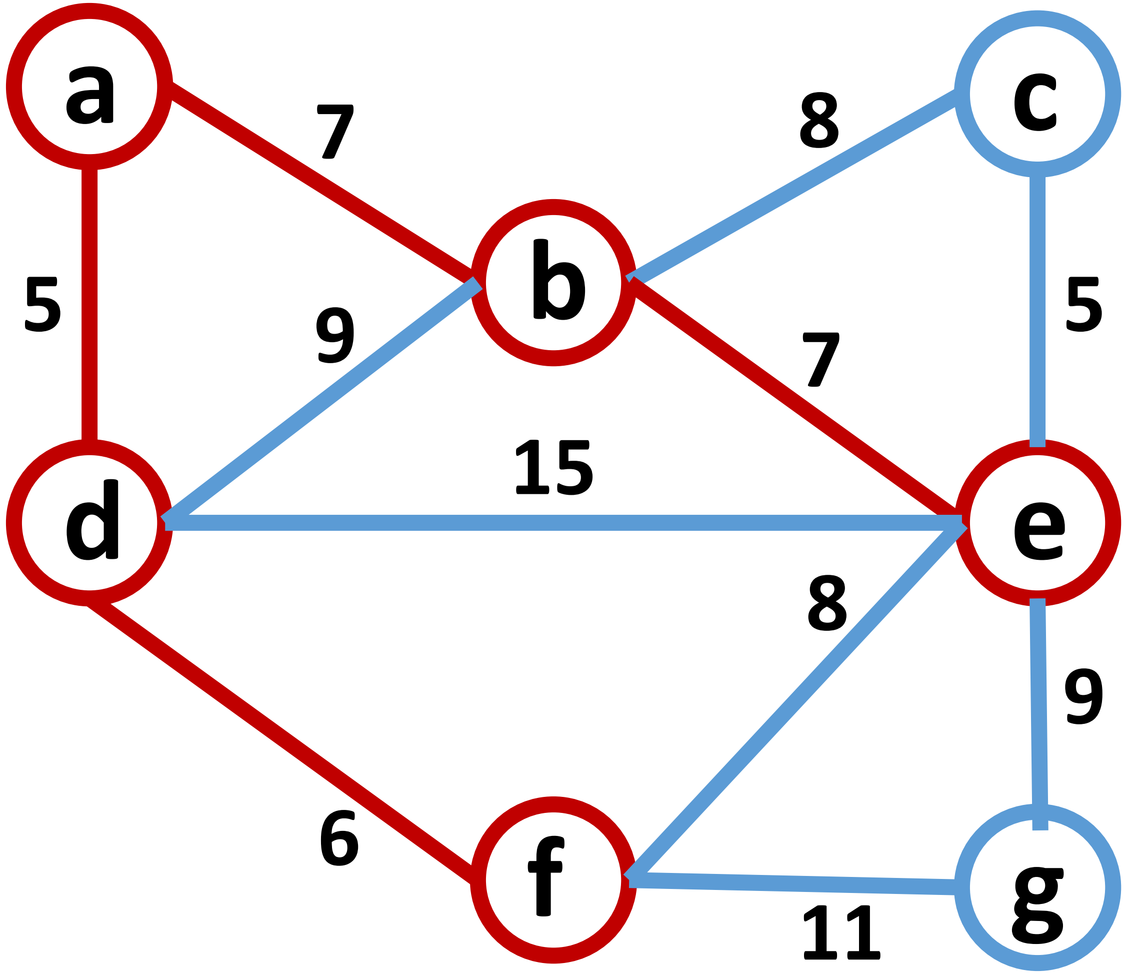
Выбранные вершины U	Ребро (u, v)	Невыбранные вершины V\U
{}		{a,b,c,d,e,f,g}
{d}	5, 6, 9, 15	{a,b,c,e,f,g}
{a,d}	7, 9, 6, 15	{b,c,e,f,g}
{a,d,f}	7, 9, 15, 8, 11	{b,c,e,g}
{a,b,d,f}		



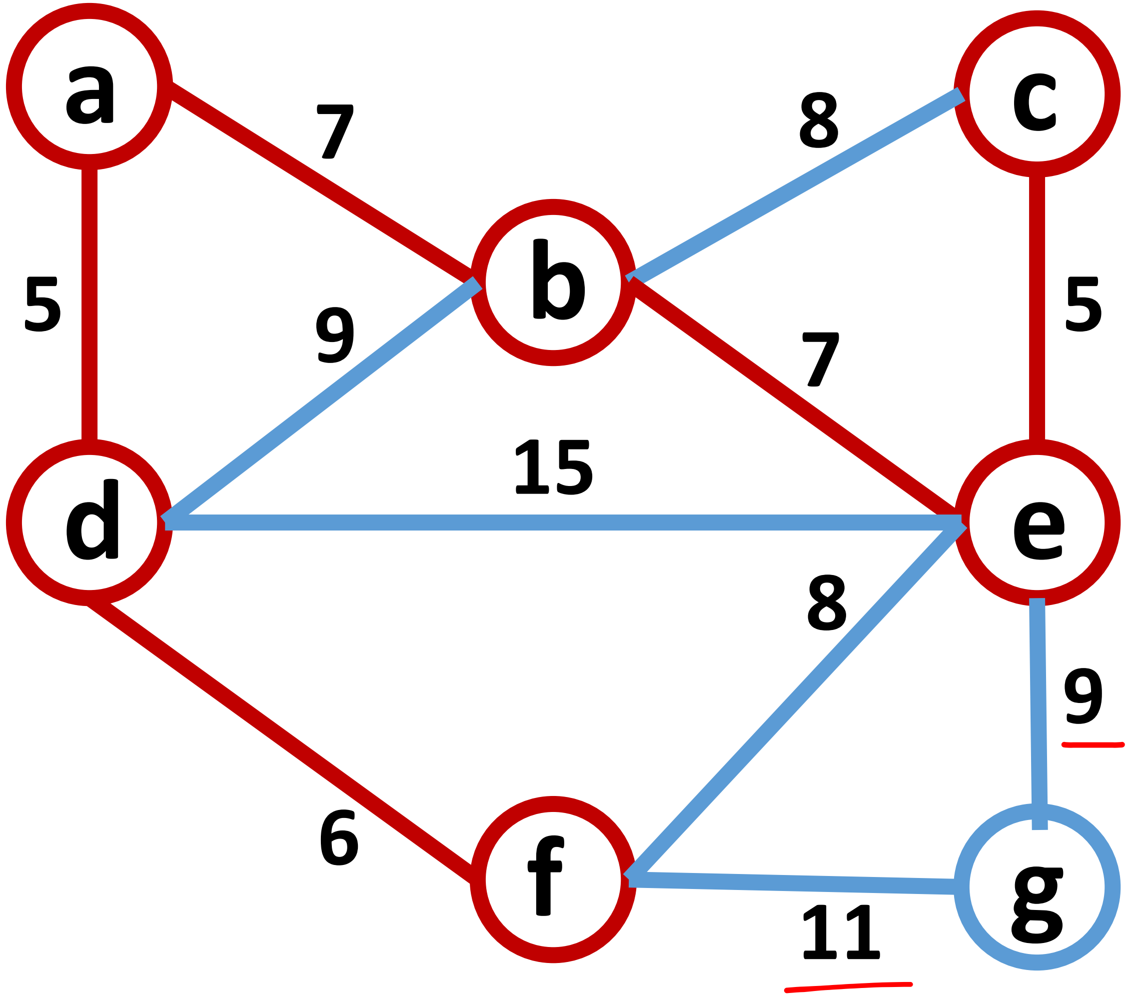
Выбранные вершины U	Ребро (u, v)	Невыбранные вершины V\U
{}		{a,b,c,d,e,f,g}
{d}	5, 6, 9, 15	{a,b,c,e,f,g}
{a,d}	7, 9, 6, 15	{b,c,e,f,g}
{a,d,f}	7, 9, 15, 8, 11	{b,c,e,g}
{a,b,d,f}	8, 7, 15, 8, 11	{c,e,g}



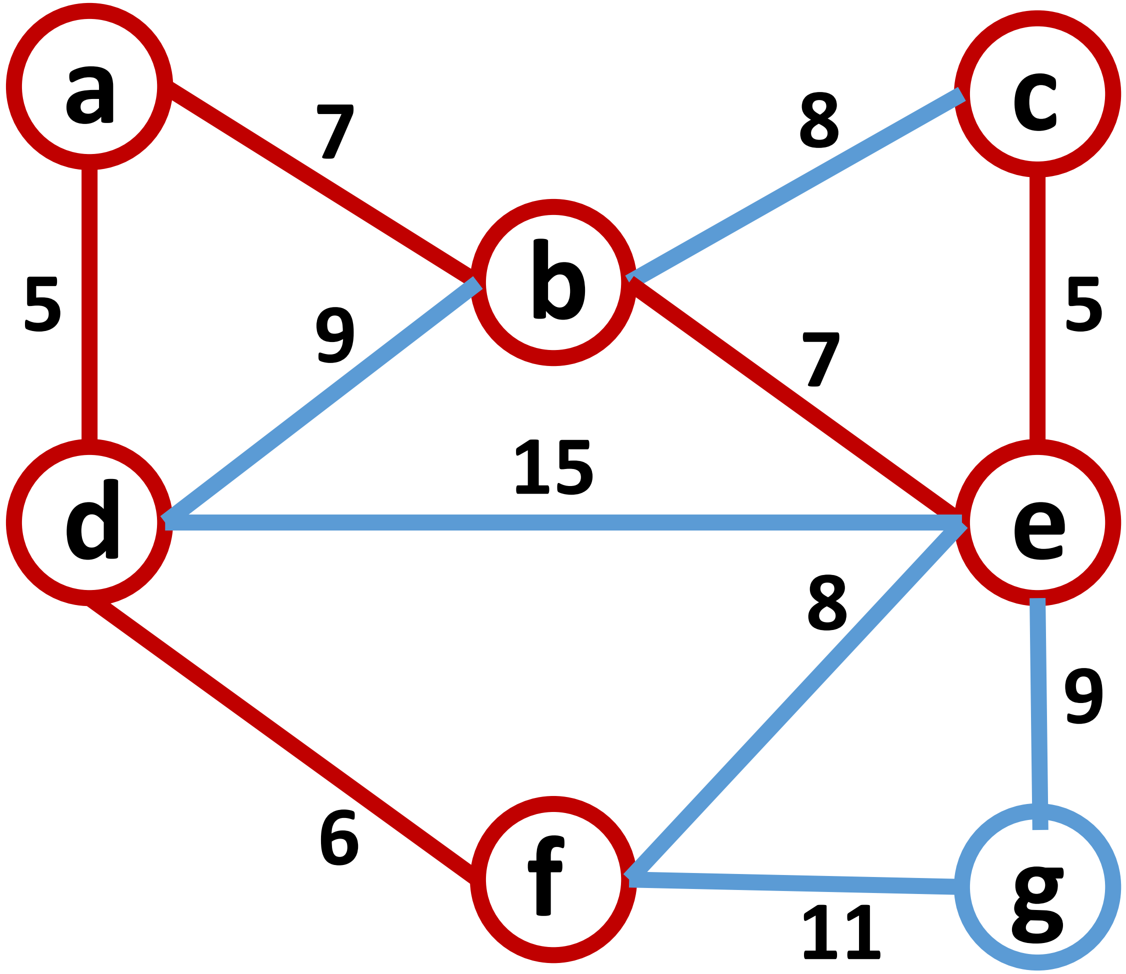
Выбранные вершины U	Ребро (u, v)	Невыбранные вершины V\U
{}		{a,b,c,d,e,f,g}
{d}	5, 6, 9, 15	{a,b,c,e,f,g}
{a,d}	7, 9, 6, 15	{b,c,e,f,g}
{a,d,f}	7, 9, 15, 8, 11	{b,c,e,g}
{a,b,d,f}	8, 7, 15, 8, 11	{c,e,g}
{a,b,d,e,f}		



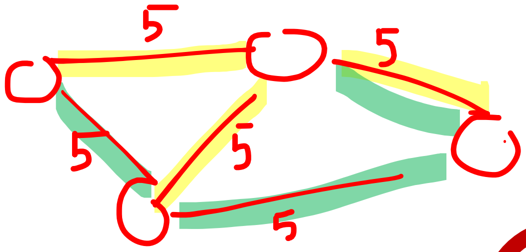
Выбранные вершины U	Ребро (u, v)	Невыбранные вершины V\U
{}		{a,b,c,d,e,f,g}
{d}	5, 6, 9, 15	{a,b,c,e,f,g}
{a,d}	7, 9, 6, 15	{b,c,e,f,g}
{a,d,f}	7, 9, 15, 8, 11	{b,c,e,g}
{a,b,d,f}	8, 7, 15, 8, 11	{c,e,g}
{a,b,d,e,f}	8, 5, 9, 11	{c,g}

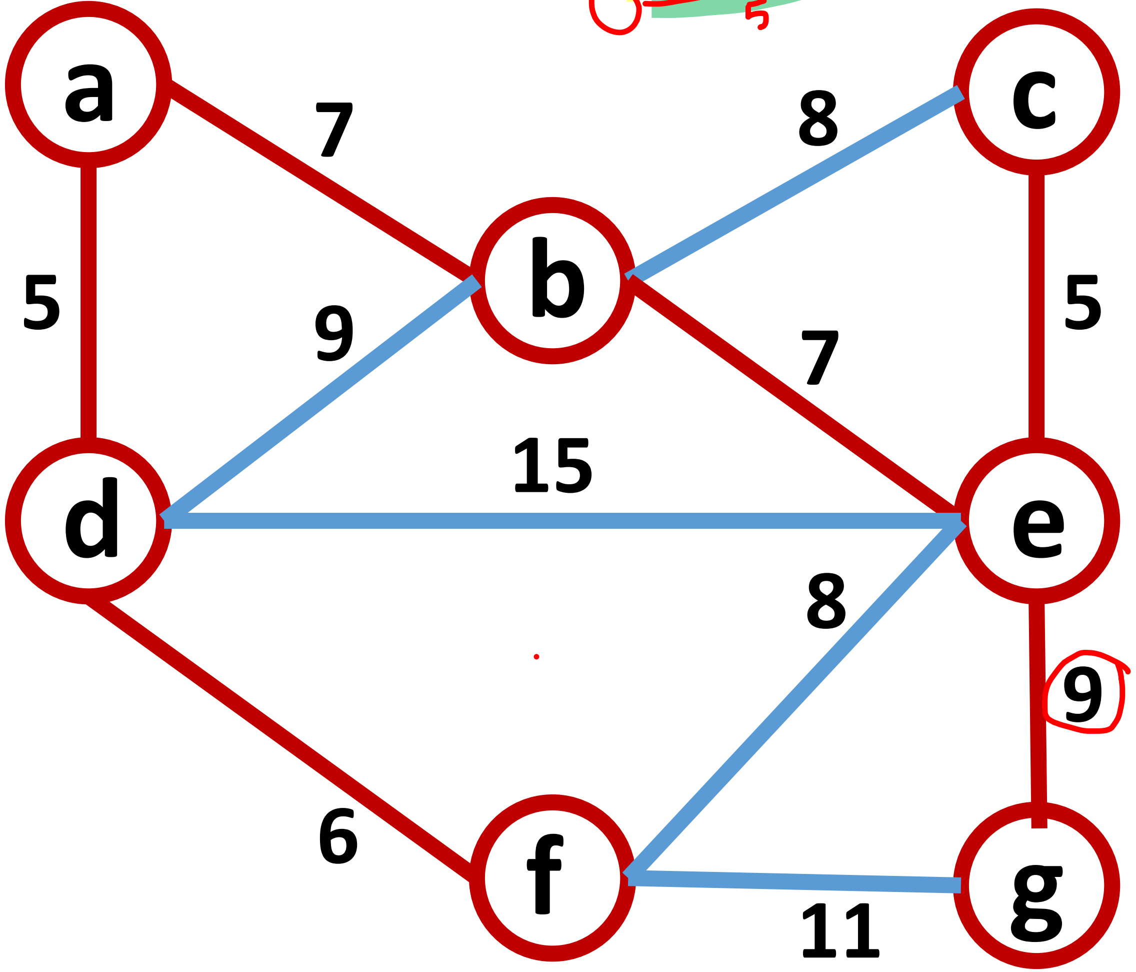


Выбранные вершины U	Ребро (u, v)	Невыбранные вершины V\U
{}		{a,b,c,d,e,f,g}
{d}	5, 6, 9, 15	{a,b,c,e,f,g}
{a,d}	7, 9, 6, 15	{b,c,e,f,g}
{a,d,f}	7, 9, 15, 8, 11	{b,c,e,g}
{a,b,d,f}	8, 7, 15, 8, 11	{c,e,g}
{a,b,d,e,f}	8, 5, 9, 11	{c,g}
{a,b,c,d,e,f}		

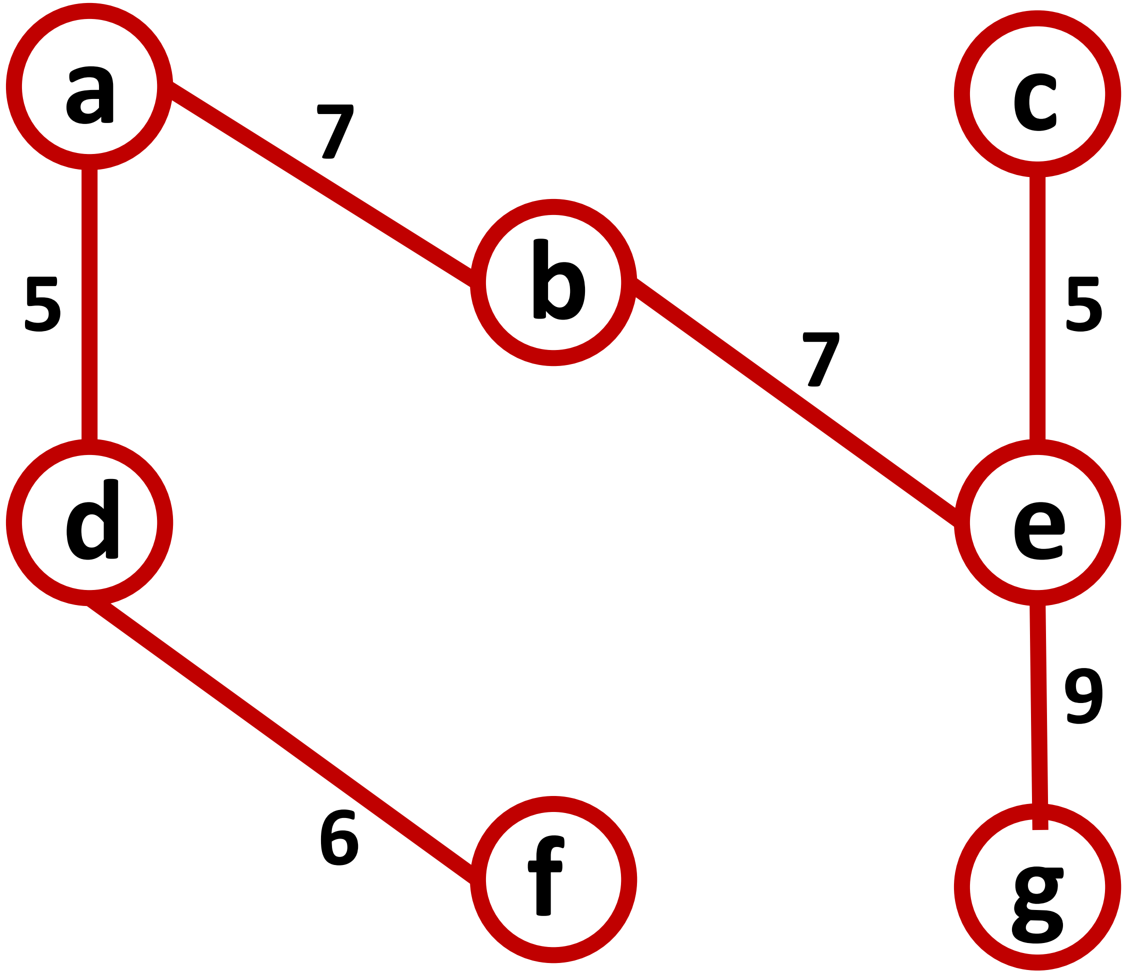


Выбранные вершины U	Ребро (u, v)	Невыбранные вершины V\U
{}		{a,b,c,d,e,f,g}
{d}	5, 6, 9, 15	{a,b,c,e,f,g}
{a,d}	7, 9, 6, 15	{b,c,e,f,g}
{a,d,f}	7, 9, 15, 8, 11	{b,c,e,g}
{a,b,d,f}	8, 7, 15, 8, 11	{c,e,g}
{a,b,d,e,f}	8, 5, 9, 11	{c,g}
{a,b,c,d,e,f}	9, 11	{g}





Выбранные вершины U	Ребро (u, v)	Невыбранные вершины V\U
{}		{a,b,c,d,e,f,g}
{d}	5, 6, 9, 15	{a,b,c,e,f,g}
{a,d}	7, 9, 6, 15	{b,c,e,f,g}
{a,d,f}	7, 9, 15, 8, 11	{b,c,e,g}
{a,b,d,f}	8, 7, 15, 8, 11	{c,e,g}
{a,b,d,e,f}	8, 5, 9, 11	{c,g}
{a,b,c,d,e,f}	9, 11	{g}
{a,b,c,d,e,f,g}		{}



Выбранные вершины U	Ребро (u, v)	Невыбранные вершины V\U
{}		{a,b,c,d,e,f,g}
{d}	5, 6, 9, 15	{a,b,c,e,f,g}
{a,d}	7, 9, 6, 15	{b,c,e,f,g}
{a,d,f}	7, 9, 15, 8, 11	{b,c,e,g}
{a,b,d,f}	8, 7, 15, 8, 11	{c,e,g}
{a,b,d,e,f}	8, 5, 9, 11	{c,g}
{a,b,c,d,e,f}	9, 11	{g}
{a,b,c,d,e,f,g}		{}

Вес дерева: 39

Алгоритм Крускала (Краскала)

Алгоритм Краскала - алгоритм построения минимального остовного дерева взвешенного связного неориентированного графа.
Алгоритм описан Джозефом Краскалом в 1956 году, этот алгоритм почти не отличается от алгоритма Борувки предложенный Отакаром Борувкой в 1926 году.

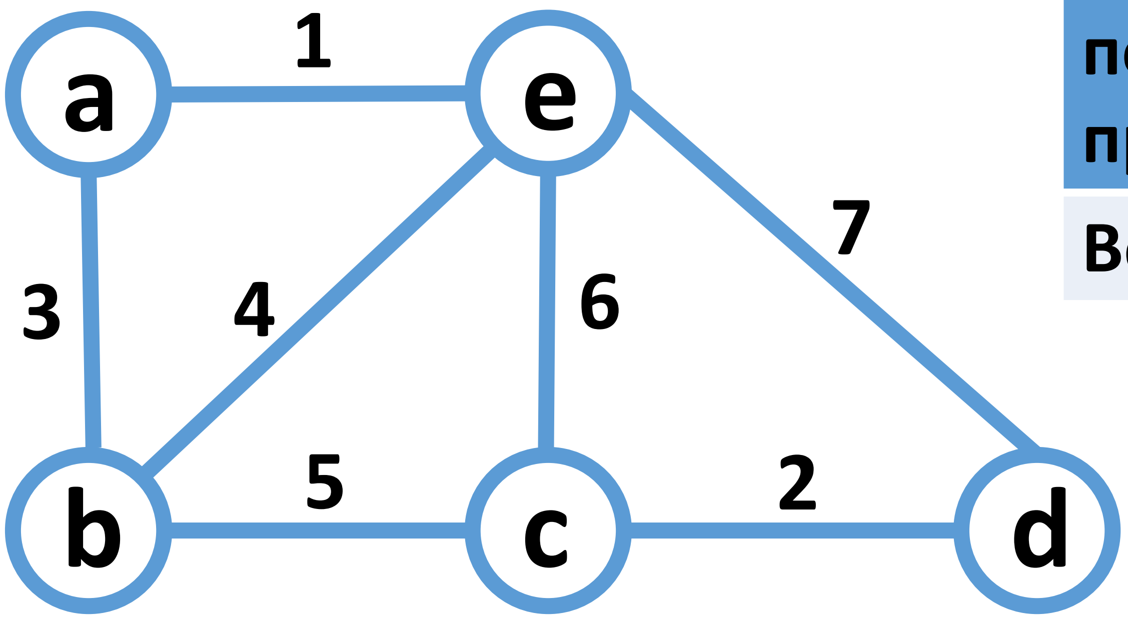
При эффективной реализации можно считать, что асимптотика работы алгоритма Краскала определяется асимптотикой сортировки ребер (O(E log E))

https://ru.wikipedia.org/wiki/Алгоритм_Краскала

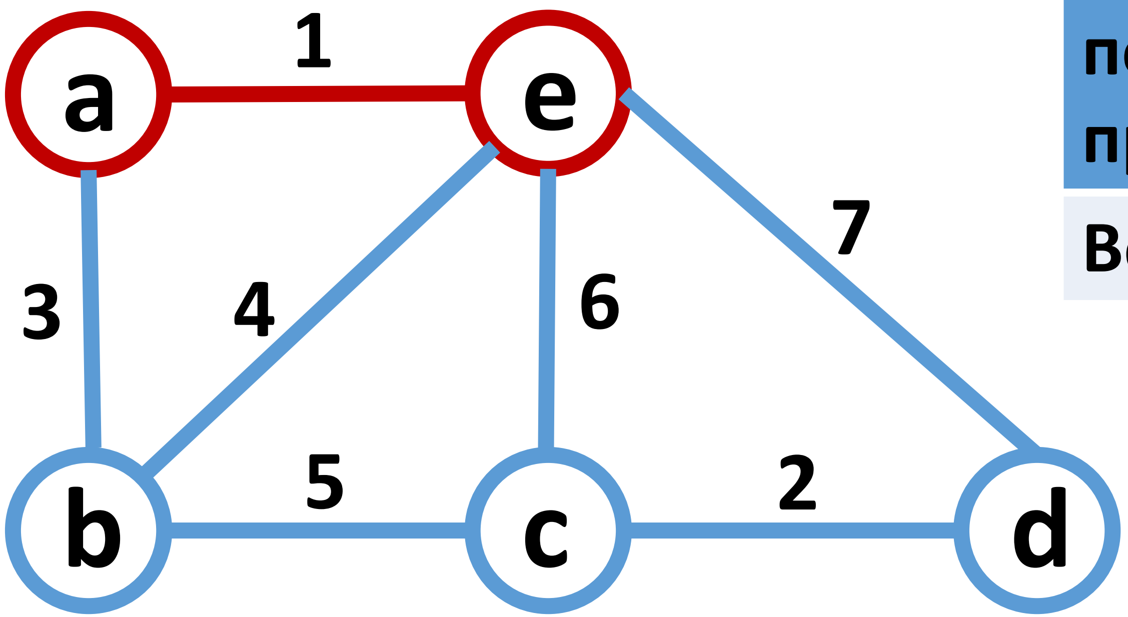
Алгоритм Крускала (Краскала)

- Вначале текущее множество рёбер устанавливается пустым.
- Затем, пока это возможно, проводится следующая операция: из всех рёбер, добавление которых к уже имеющемуся множеству не вызовет появление в нём цикла, выбирается ребро минимального веса и добавляется к уже имеющемуся множеству. Когда таких рёбер больше нет, алгоритм завершён.
- Подграф данного графа, содержащий все его вершины и найденное множество рёбер, является его остовным деревом минимального веса.

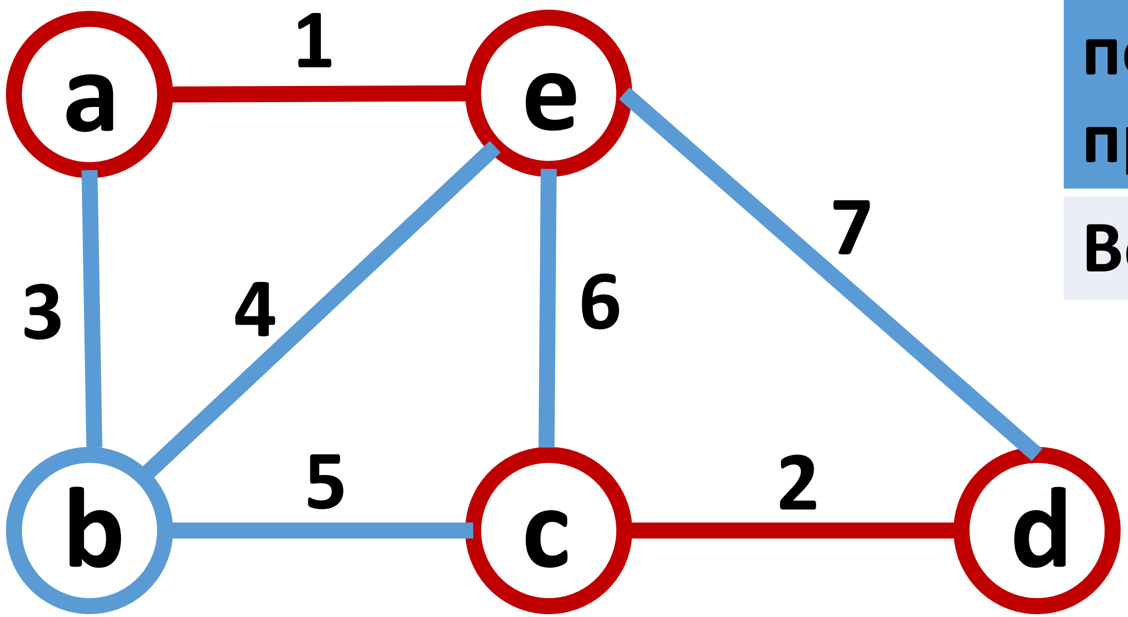
https://ru.wikipedia.org/wiki/Алгоритм_Краскала



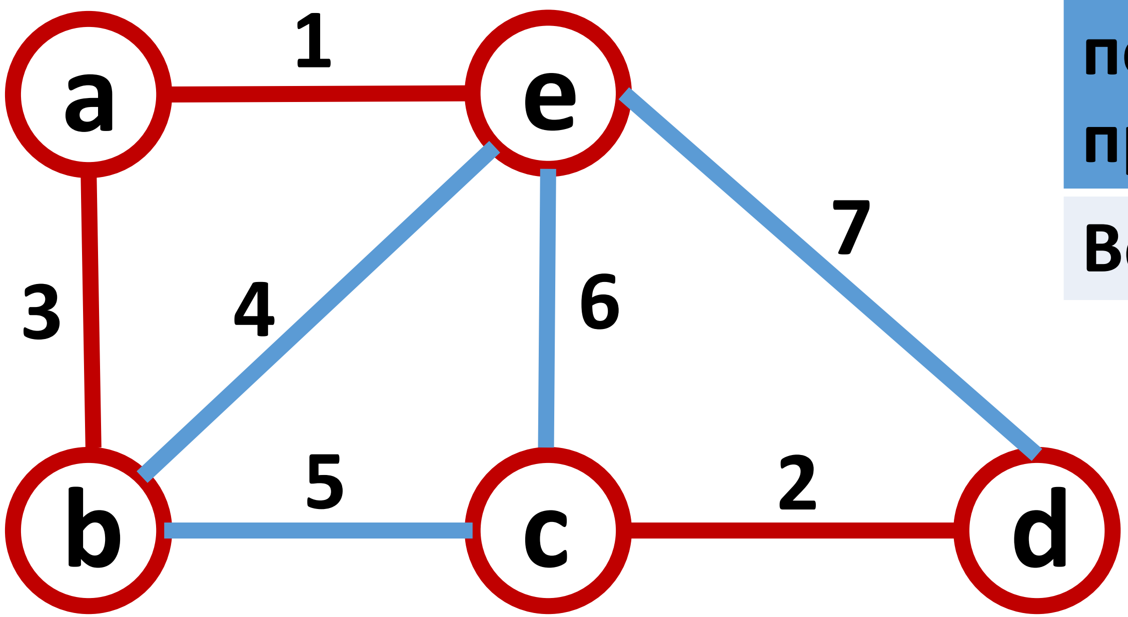
Рёбра (в порядке их просмотра)	ae	cd	ab	be	bc	ec	ed
Веса рёбер	1	2	3	4	5	6	7



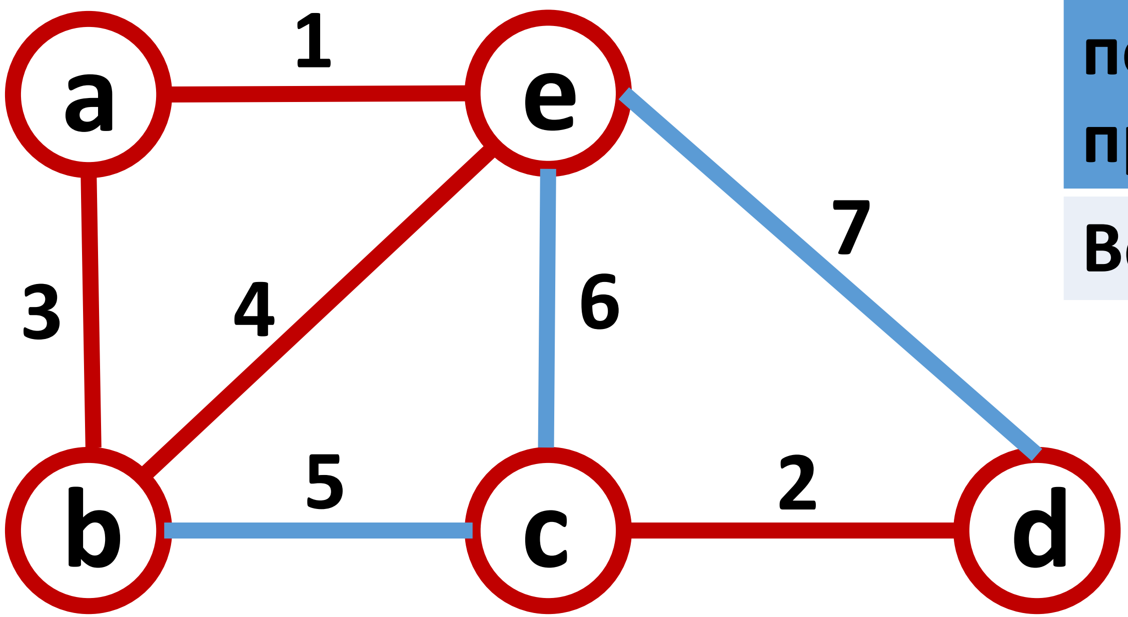
Рёбра (в порядке их просмотра)	ae	cd	ab	be	bc	ec	ed
Веса рёбер	1	2	3	4	5	6	7



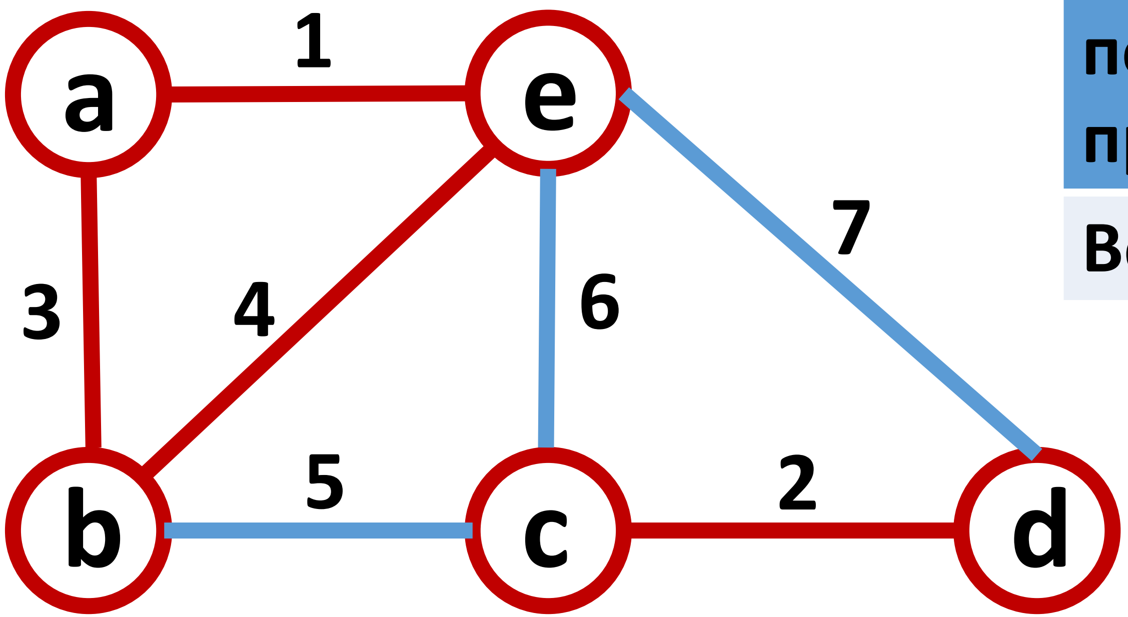
Рёбра (в порядке их просмотра)	ae	cd	ab	be	bc	ec	ed
Веса рёбер	1	2	3	4	5	6	7



Рёбра (в порядке их просмотра)	ae	cd	ab	be	bc	ec	ed
Веса рёбер	1	2	3	4	5	6	7

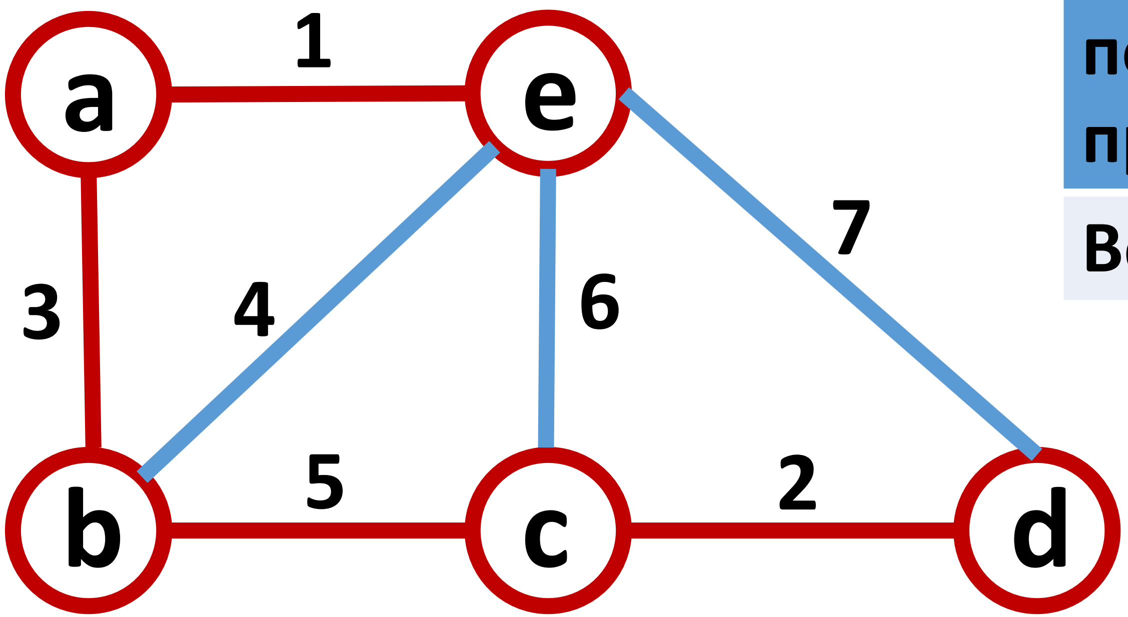


Рёбра (в порядке их просмотра)	ae	cd	ab	be	bc	ec	ed
Веса рёбер	1	2	3	4	5	6	7

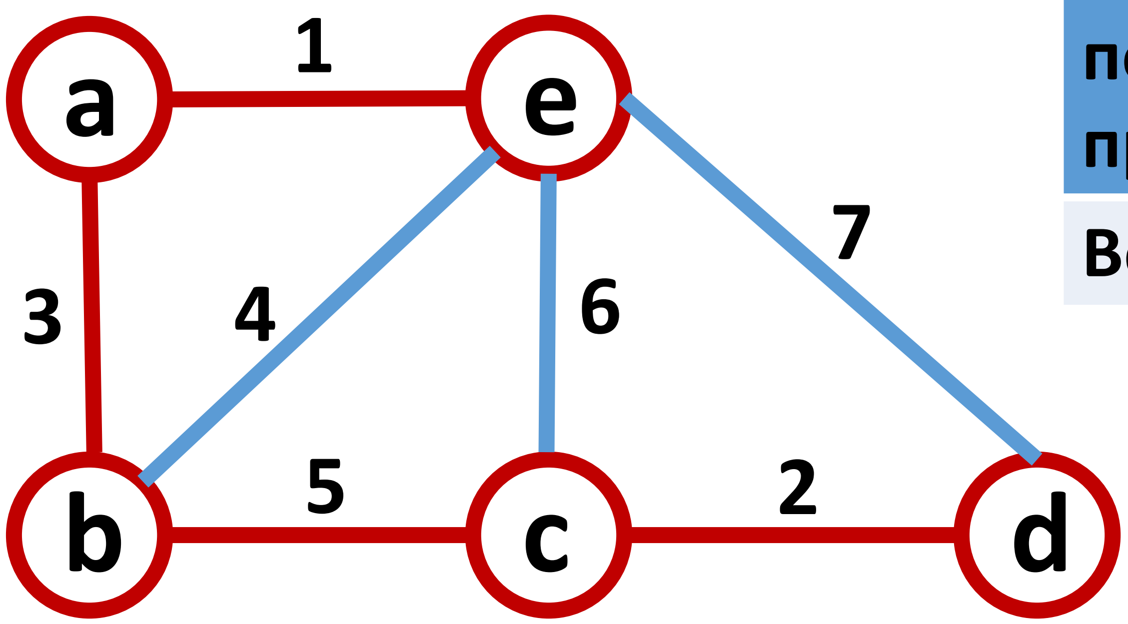


Рёбра (в порядке их просмотра)	ae	cd	ab	be	bc	ec	ed
Веса рёбер	1	2	3	4	5	6	7

Нельзя! Образовался цикл! (вершины b, e уже из одного дерева)

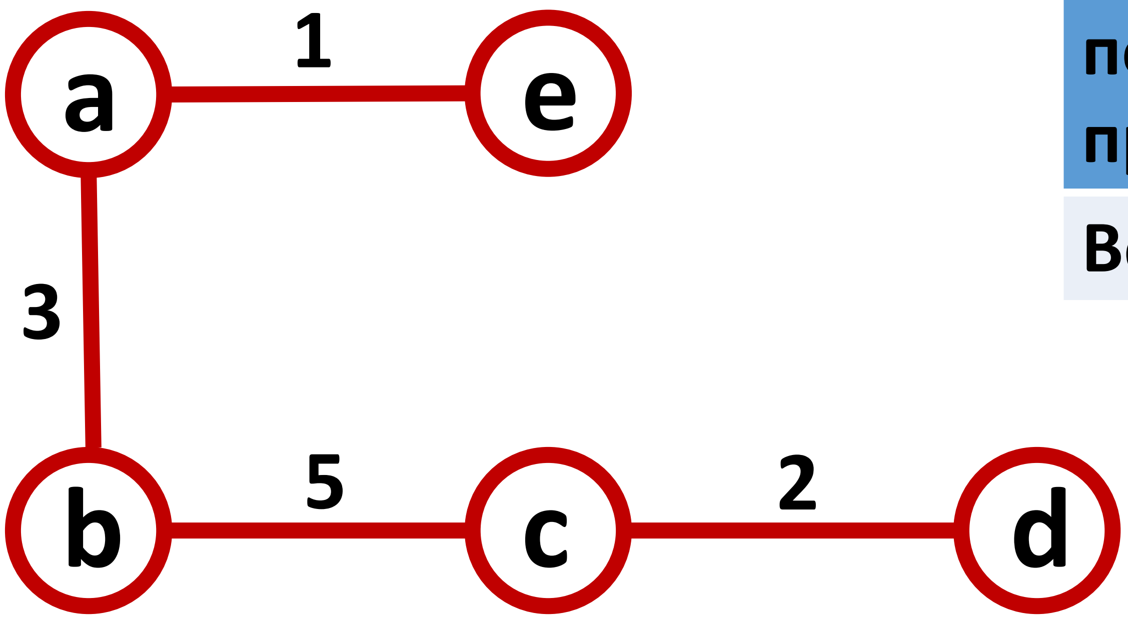


Рёбра (в порядке их просмотра)	ae	cd	ab	be	bc	ec	ed
Веса рёбер	1	2	3	4	5	6	7



Рёбра (в порядке их просмотра)	ae	cd	ab	be	bc	ec	ed
Веса рёбер	1	2	3	4	5	6	7

Использовали уже все вершины. Оставшиеся ребра be, ec, ed соединяют вершины из одного дерева.



Рёбра (в порядке их просмотра)	ae	cd	ab	be	bc	ec	ed
Веса рёбер	1	2	3	4	5	6	7

Вес дерева: 11

Система непересекающихся множеств (СНМ)

Эта структура данных предоставляет следующие возможности. Изначально имеется несколько элементов, каждый из которых находится в отдельном (своём собственном) множестве. За одну операцию можно **объединить два каких-либо множества**, а также можно **запросить, в каком множестве** сейчас находится указанный элемент. Также, в классическом варианте, вводится ещё одна операция — создание нового элемента, который помещается в отдельное множество.

Таким образом, базовый интерфейс данной структуры данных состоит всего из трёх операций:

- $\mathrm{make\_set}(x)$ — **добавляет** новый элемент x, помещая его в новое множество, состоящее из одного него.
- $\mathrm{union\_sets}(x, y)$ — **объединяет** два указанных множества (множество, в котором находится элемент x, и множество, в котором находится элемент y).
- $\mathrm{find\_set}(x)$ — **возвращает, в каком множестве** находится указанный элемент x.

```
MakeSet(1);
MakeSet(2);
MakeSet(3);
MakeSet(4);
MakeSet(5);
```

ind:	1	2	3	4	5
par.	4	5	5	4	5

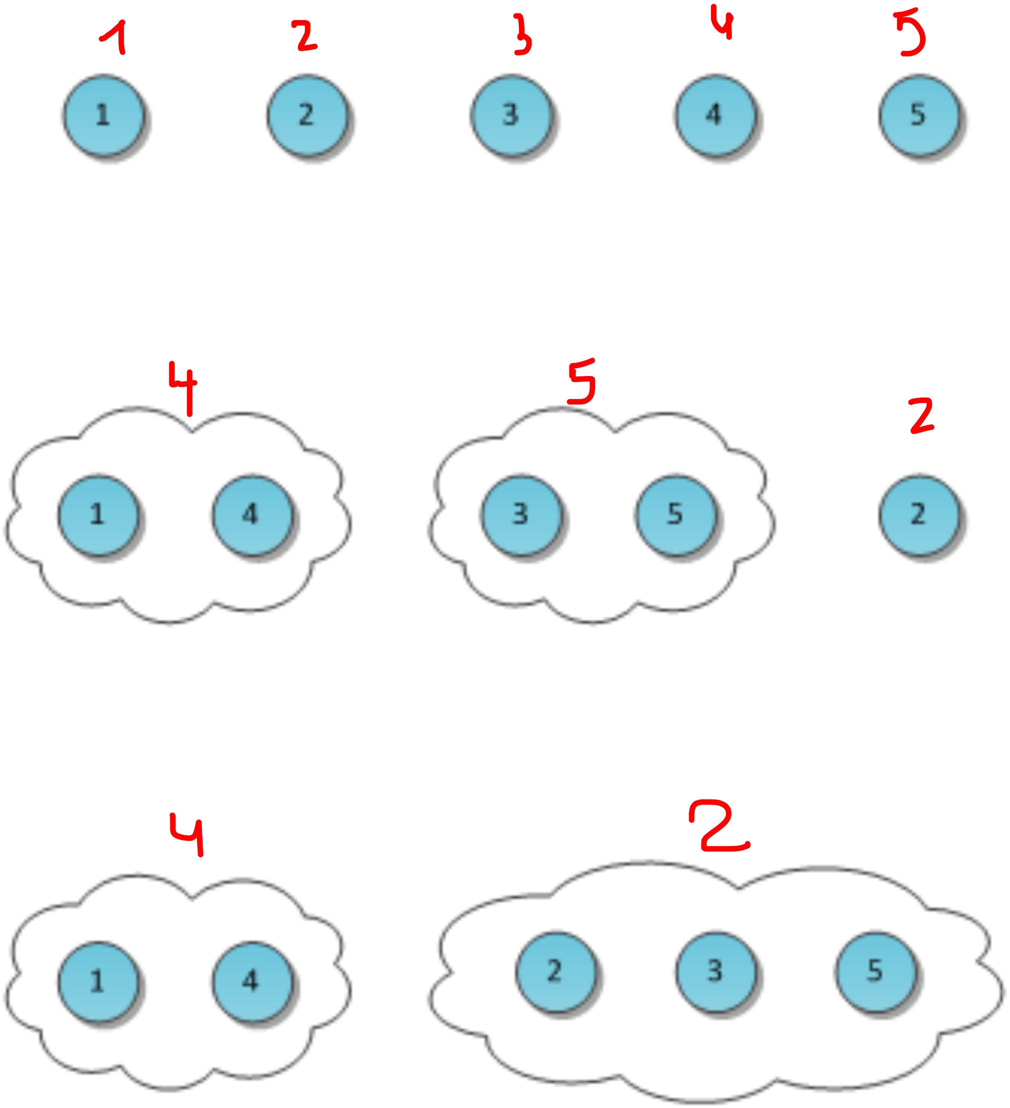
```
Find(4) = 4
```

```
Unite(1, 4);
Unite(3, 5);
```

```
Find(4) = 4
Find(1) = 4
Find(2) = 2
```

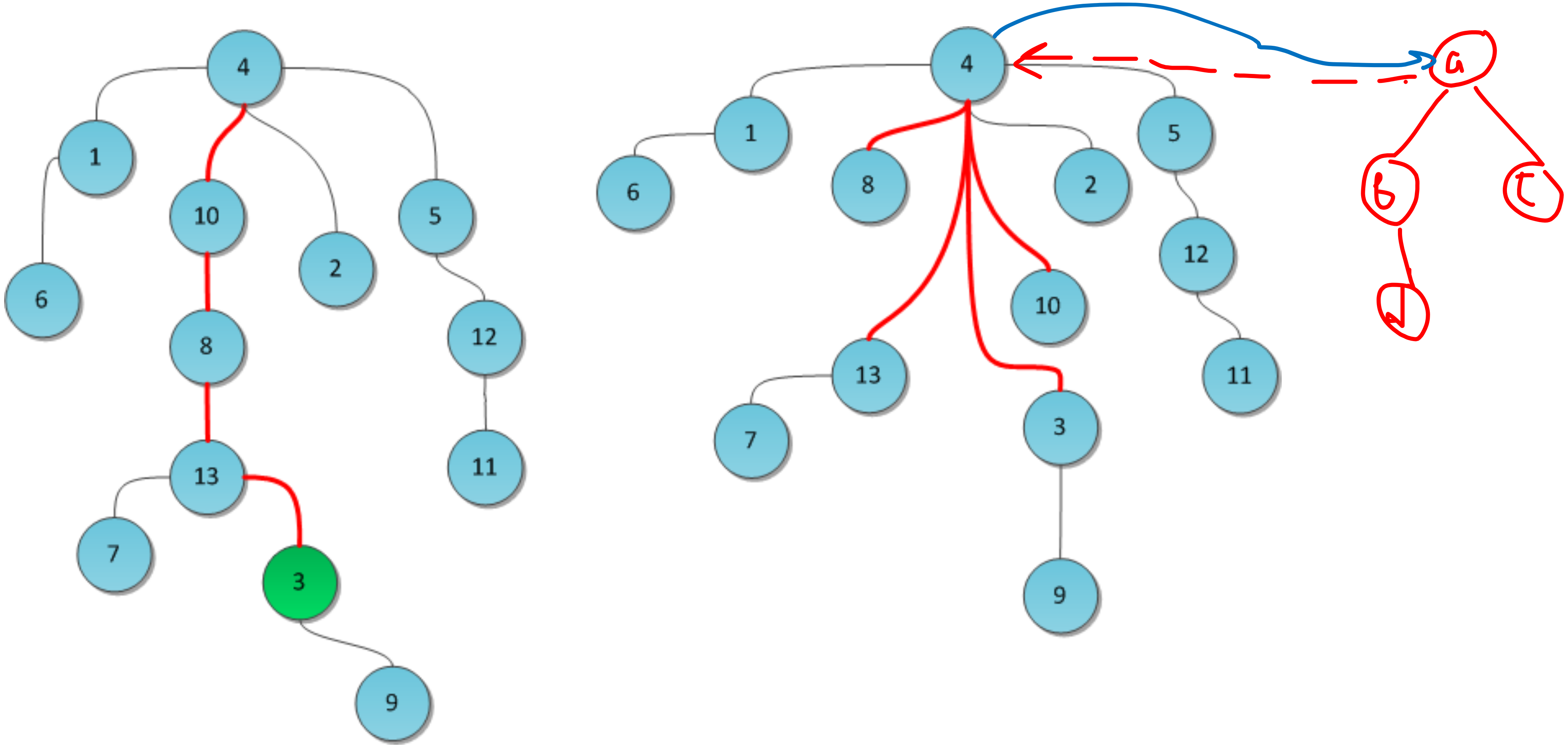
```
Unite(5, 2);
```

```
Find(5) = 2
Find(3) = 2
```

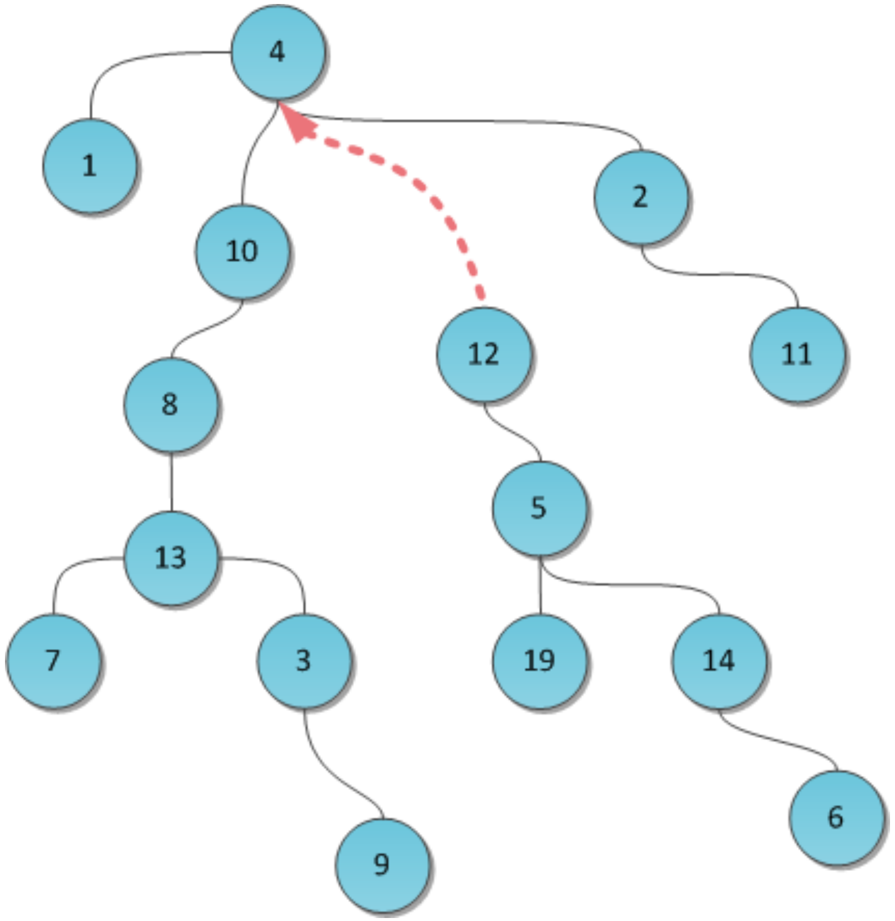


https://habr.com/ru/post/104772/

Эвристика сжатия пути



Слияние

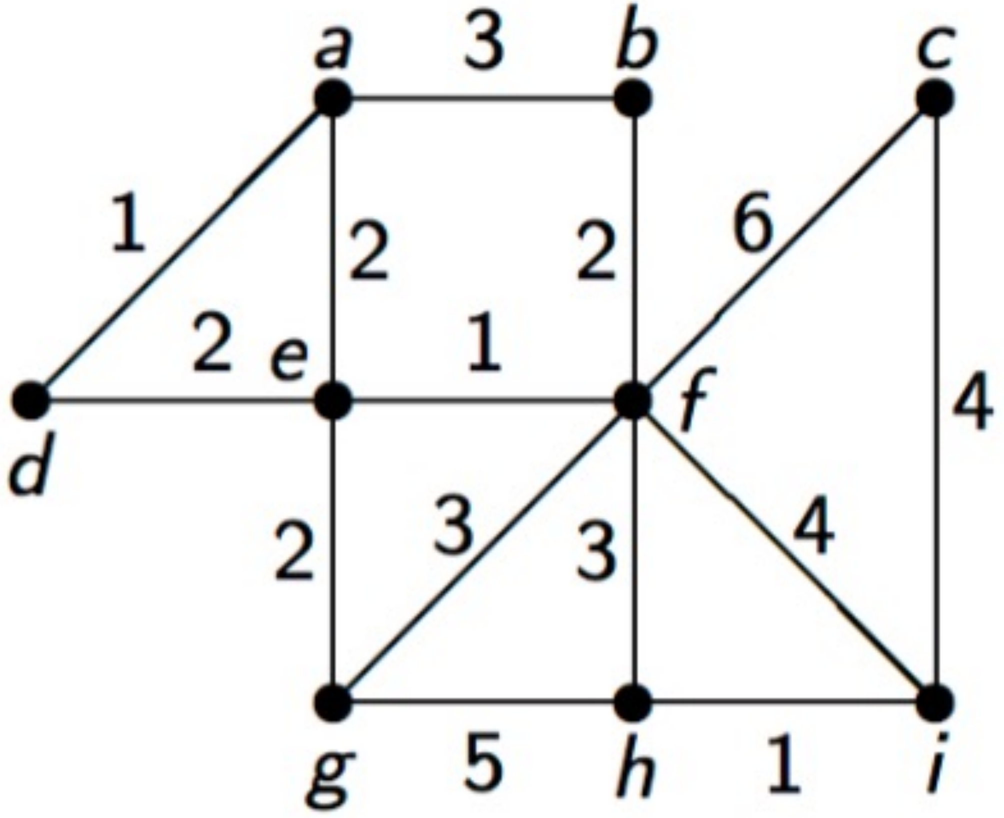


Свойства минимального остова (каркаса минимального веса)

- Минимальный остов **уникален, если веса всех рёбер различны**. В противном случае, может существовать несколько минимальных остовов (конкретные алгоритмы обычно получают один из возможных остовов).
- Минимальный остов является также и **остовом с минимальным произведением** весов рёбер.
(доказывается это легко, достаточно заменить веса всех рёбер на их логарифмы)
- Минимальный остов является также и **остовом с минимальным весом самого тяжелого ребра**.
(это утверждение следует из справедливости алгоритма Крускала)
- **Остов максимального веса** ищется аналогично остову минимального веса, достаточно поменять знаки всех рёбер на противоположные и выполнить любой из алгоритм минимального остова.

http://www.e-maxx-ru.1gb.ru/algo/mst_kruskal

Построить каркас минимального веса алгоритмом Краскала



Построить каркас минимального веса алгоритмом Прима