MST – Prim's Algorithm, A Primer.

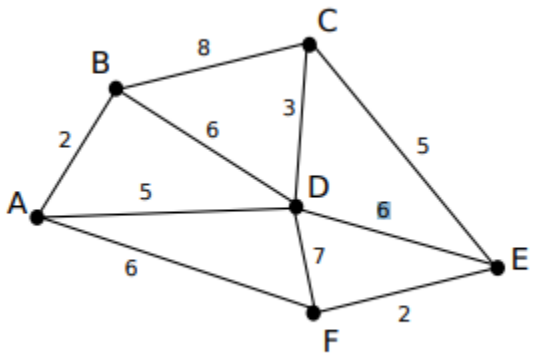
We've already looked at Kruskal's algorithm. It works by considering the edges of a graph, `G`, with `n` vertices.

```
Kruskal's algorithm:
sort the edges of G in increasing order by length
keep a subgraph S of G, initially empty
for each edge e in sorted order
    if the endpoints of e are disconnected in S
    add e to S
return S
```

Rather than build a subgraph one edge at a time, Prims's algorithm builds a tree one vertex at a time.

```
Prim's algorithm:
let T be a single vertex x
while (T has fewer than n vertices)
{
    find the smallest edge connecting T to G-T
    add it to T
}
```

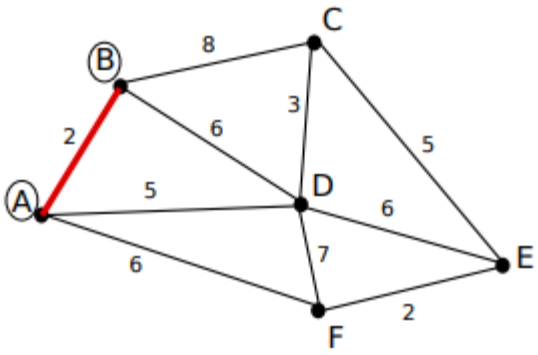
Basically, at each stage (or iteration of the loop) we consider the vertices



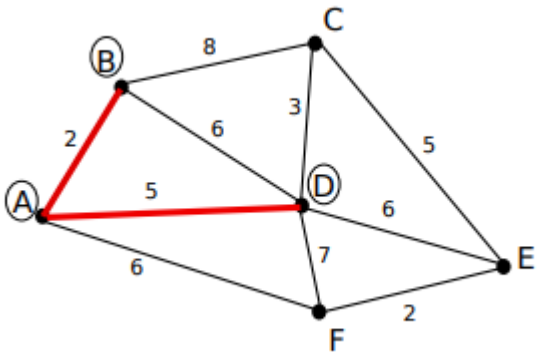
For Prim's you can select any vertex to being the algorithm. Let's say we select A.

The vertex A has 3 edge connections to the remaining vertices (B,C,D,E,F).

The algorithm chooses the edge to B (note that it's a greedy algorithm, just like Kruskal's):



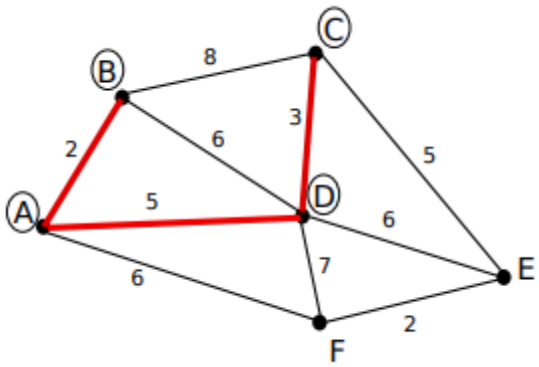
There are 4 edges connecting the set A,B, to the remaining nodes C,D,E,F (these are (B,C), (B,D), (A,D), (A,F)) Edge (A,D) is chosen:



There are five (NOT 6) edges connecting the set A,B,D to the nodes C,E,F. They are (A,F), (D,F), (D,E), (D,C), (B,C).

Of course (B,D) is not considered because both are in the set T (and, therefore, none of them are in the set G-T).

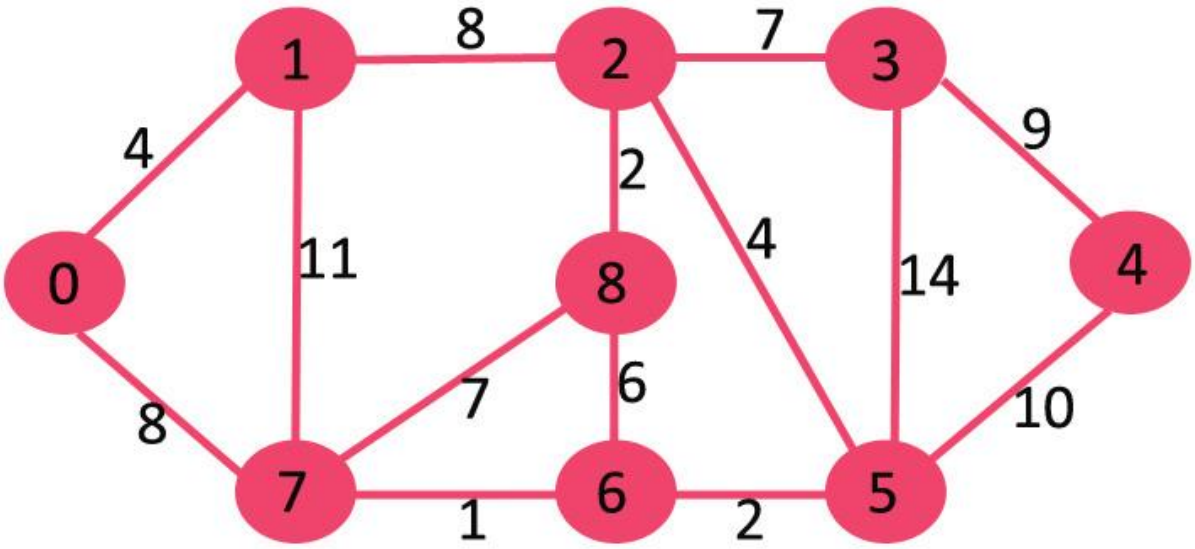
(D,C) is picked:



As an exercise, you should be able to finish this…

As a supplementary exercise, figure out the output of Kruskal's algorithm for this Graph. Is it the same as Prim's?

Now, try it on this graph (we've already examined it using kruskal's algorithm). Assume you start at vertex 0 (for ties just pick any of the tied edges).



Since tied edges mean that we arbitrarily pick one, then it should be obvious (especially after a hand-trace) that we can produce more than one Minimum Spanning Tree (early in this example we will have to choose between (0,7) or (1,2)[1] – this will affect the subsequent edges of our MST).

The important point is that any/all MSTs should have the same cost. When you hand-trace this one, double-check to see that it has the same cost as the Kruskal algorithm

[1] Choose this one to give you a different MST to the example shown in geeksForGeeks for Kruskal