Kruskal's Algorithm

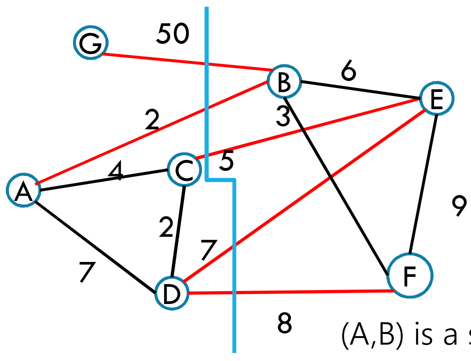
```
KruskalMST(Graph G)
   initialize each vertex to be its own component
   sort the edges by weight
   foreach(edge (u, v) in sorted order){
      if(u and v are in different components){
         add (u,v) to the MST
         Update u and v to be in the same component
      }
   }
```

Prim's Algorithm

```
PrimMST(Graph G)
     initialize costToAdd to ∞
     mark source as costToAdd 0
     mark all vertices unprocessed, mark source as processed
     foreach(edge (source, v) ) {
          v.costToAdd = weight(source,v)
          v.bestEdge = (source,v)
     }
     while(there are unprocessed vertices){
          let u be the cheapest to add unprocessed vertex
          add u.bestEdge to spanning tree
          foreach(edge (u,v) leaving u){
               if(weight(u,v) < v.costToAdd AND v not processed){
                    v.costToAdd = weight(u,v)
                    v.bestEdge = (u,v)
               }
          }
    mark u as processed
     }
```

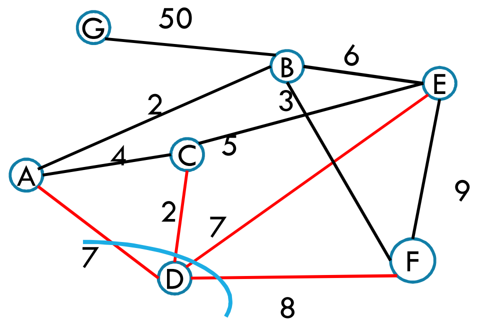
Safe Edge

Call an edge, e, a "**safe edge**" if there is some cut $(S, V \setminus S)$ where e is the minimum edge spanning that cut



(A,B) is a safe edge

(C,D) is a safe edge



What about Kruskal's?

Exchange argument:

General outline:

Suppose, you didn't find the best one.

Suppose there's a better MST

Then there's something in the algorithm's solution that doesn't match OPT. (an edge that isn't a safe edge/that's heavier than it needs to be)

Swap (exchange) them, and finish the proof (arrive at a contradiction or show that your solution is equal in quality)!