Prim's Algorithm (Minimum Spanning Tree)
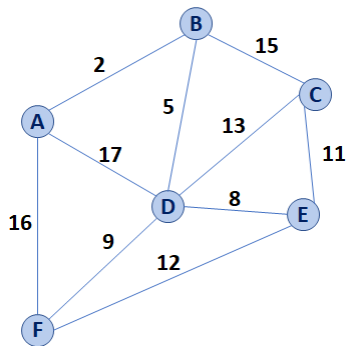


```
                      Pseudocode for Prim's MST Algorithm
1    PrimMST(G, s):
2      Input: G, Graph;
3             s, vertex in G, starting vertex of algorithm
4      Output: T, a minimum spanning tree (MST) of G
5
6      foreach (Vertex v : G):
7        d[v] = +inf
8        p[v] = NULL
9      d[s] = 0
10
11     PriorityQueue Q    // min distance, defined by d[v]
12     Q.buildHeap(G.vertices())
13     Graph T            // "labeled set"
14
15     repeat n times:
16       Vertex m = Q.removeMin()
17       T.add(m)
18       foreach (Vertex v : neighbors of m not in T):
19         if cost(v, m) < d[v]:
20           d[v] = cost(v, m)
21           p[v] = m
22
23     return T
```

	Adj. Matrix	**Adj. List**
Heap		
Unsorted Array		

Running Time of MST Algorithms

- Kruskal's Algorithm:

- Prim's Algorithm:

Q: What must be true about the connectivity of a graph when running an MST algorithm?

...what does this imply about the relationship between **n** and **m**?
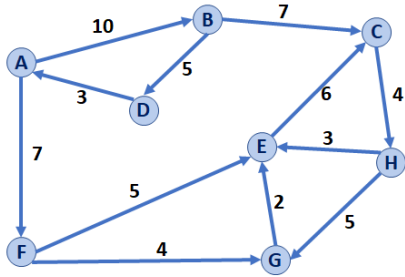

Q: Suppose we built a new heap that optimized the decrease-key operation, where decreasing the value of a key in a heap updates the heap in amortized constant time, or O(1)*. How does that change Prim's Algorithm runtime?

Shortest Path Home:

Dijkstra's Algorithm (Single Source Shortest Path)


Dijkstra's Algorithm Overview:
- The overall logic is the same as Prim's Algorithm
- We will modify the code in only two places – both involving the update to the distance metric.
- The result is a directed acyclic graph or DAG

	Pseudocode for Dijkstra's SSSP Algorithm
1	`DijkstraSSSP(G, s):`
2	`  Input: G, Graph;`
3	`         s, vertex in G, starting vertex of algorithm`
4	`  Output: T, DAG w/ shortest paths (and distances) to s`
5	
6	`  foreach (Vertex v : G):`
7	`    d[v] = +inf`
8	`    p[v] = NULL`
9	`  d[s] = 0`
10	
11	`  PriorityQueue Q   // min distance, defined by d[v]`
12	`  Q.buildHeap(G.vertices())`
13	`  Graph T          // "labeled set"`
14	
15	`  repeat n times:`
16	`    Vertex m = Q.removeMin()`
17	`    T.add(m)`
18	`    foreach (Vertex v : neighbors of m not in T):`
19	`      if _____ < d[v]:`
20	`        d[v] = _____`
21	`        p[v] = m`
22	
23	`  return T`

Dijkstra: What if we have a negative-weight cycle?
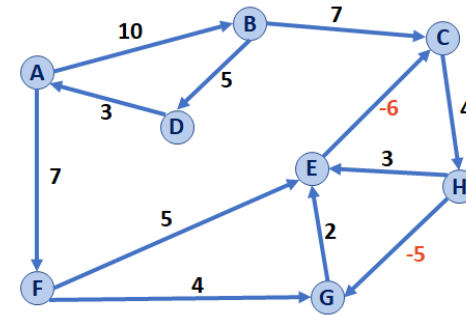
Dijkstra: What if we have a minimum-weight edge, without having a negative-weight cycle?
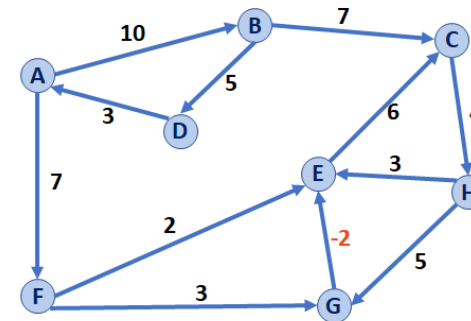
Dijkstra makes an assumption:

Dijkstra: What is the running time?

CS 225 – Things To Be Doing:
1. Final Exam runs Thursday, May 3 – Thursday, May 10
2. MP7 is released; EC due tonight, Monday, April 23th
3. Final lab, **lab_ml**, released Wednesday
4. This week is the last week of POTDs *(last POTD is Friday!)*