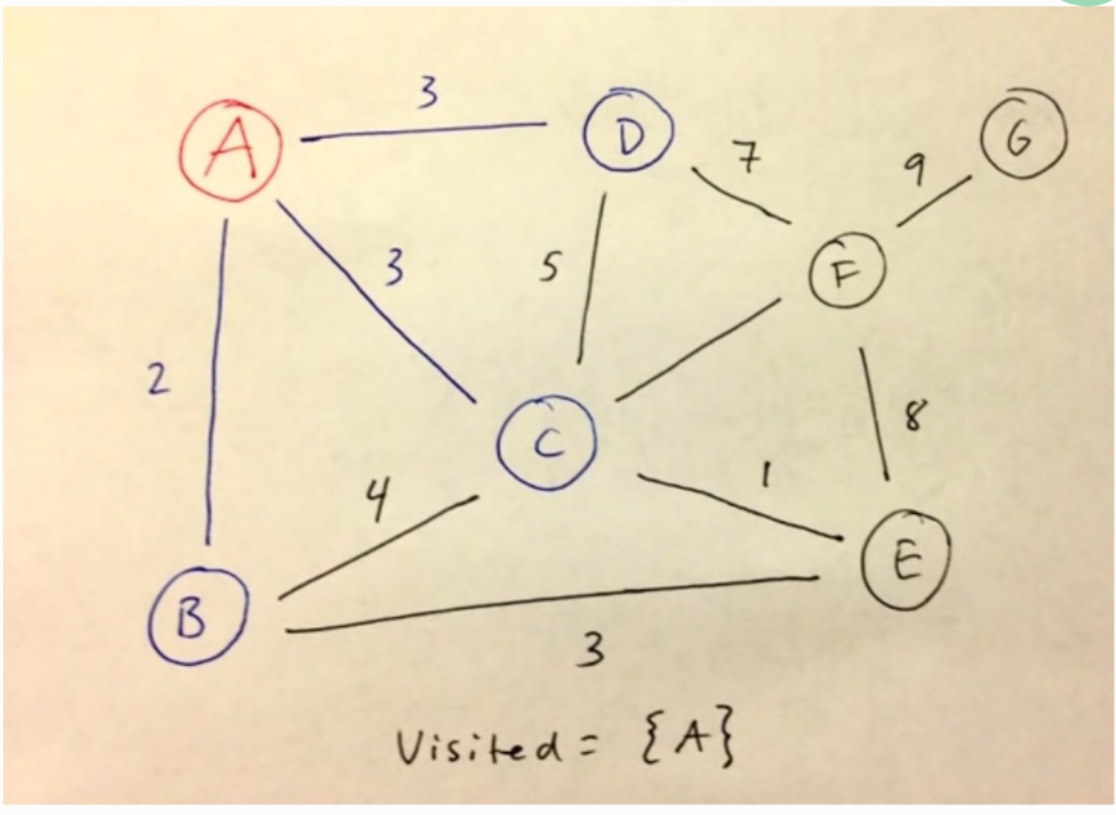
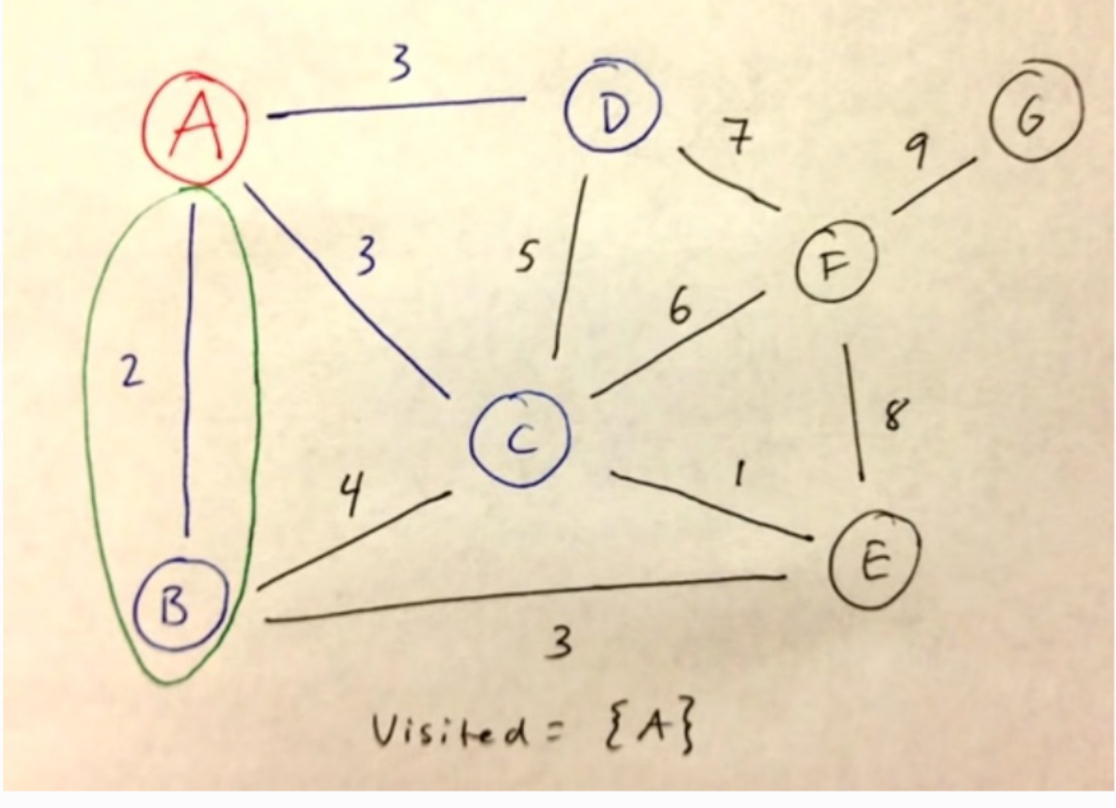
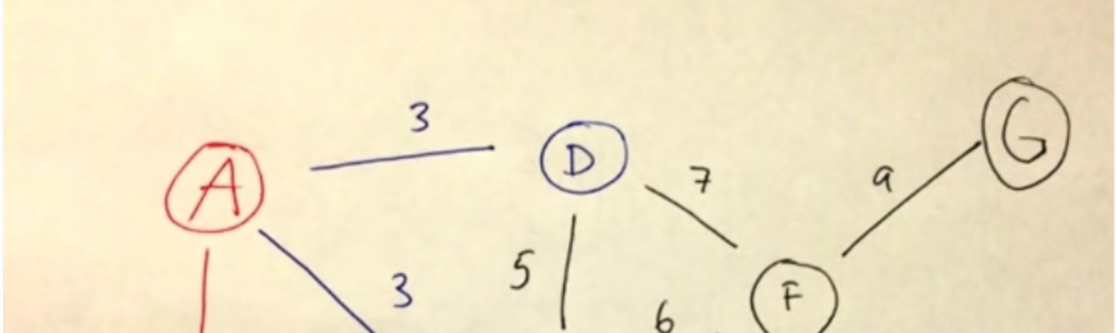
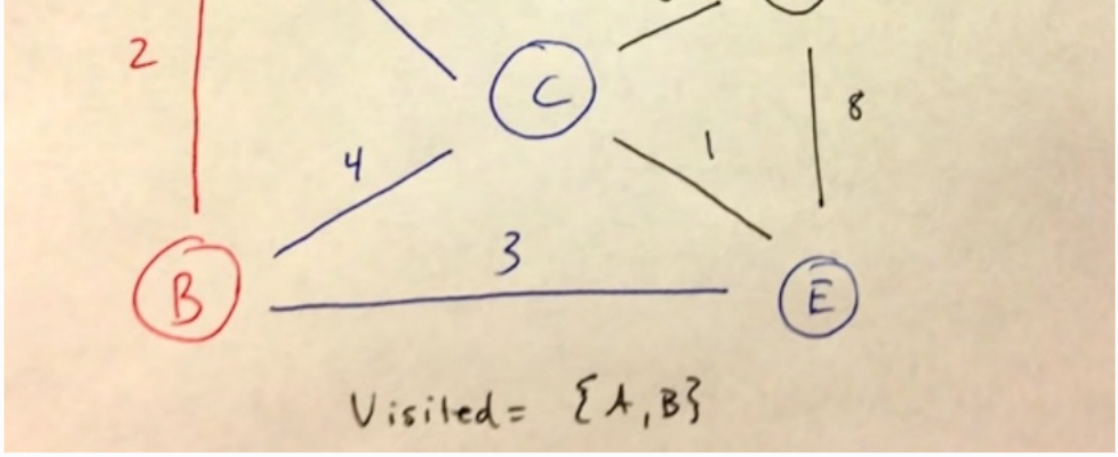
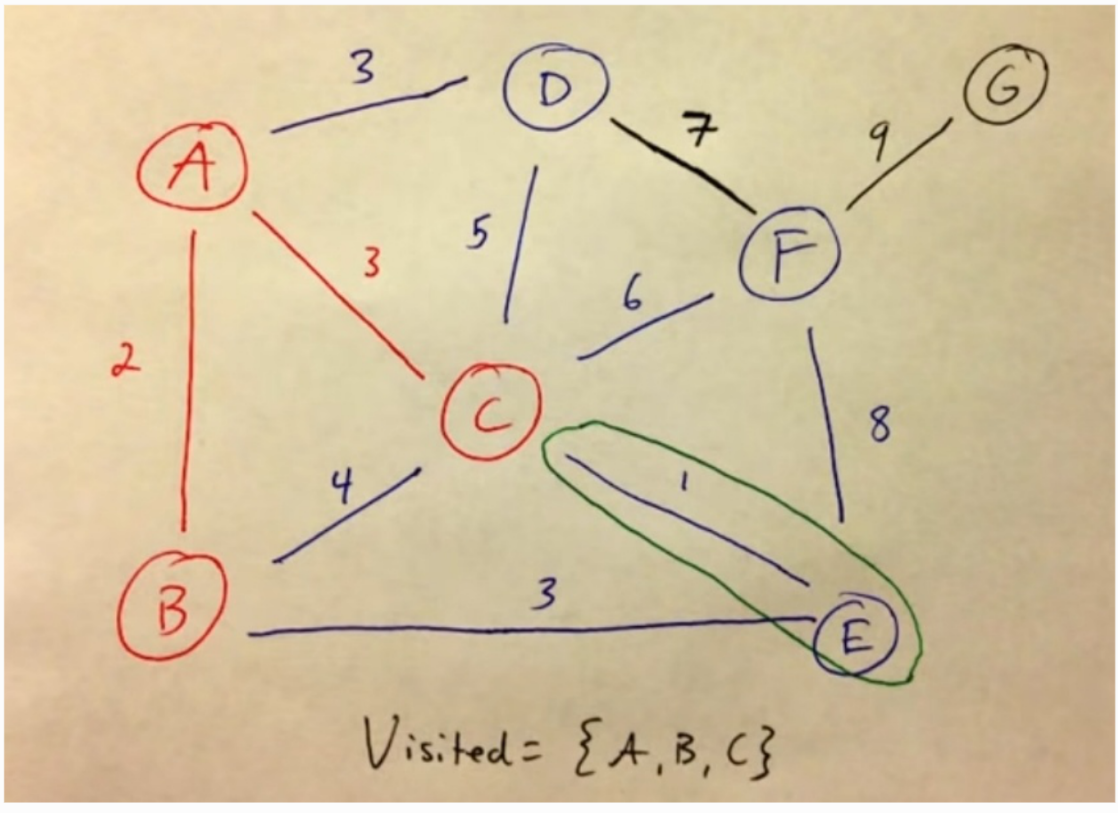
PRIM'S ALGORITHM

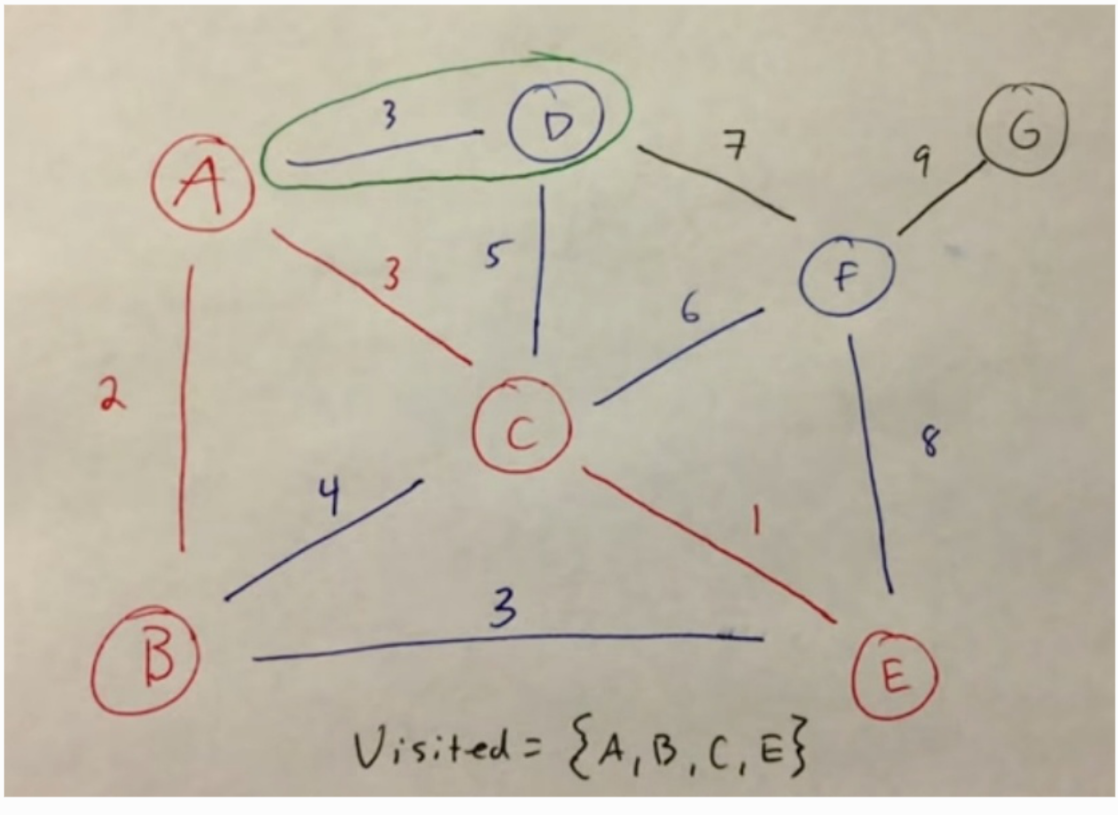


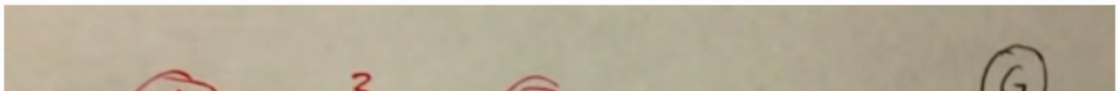


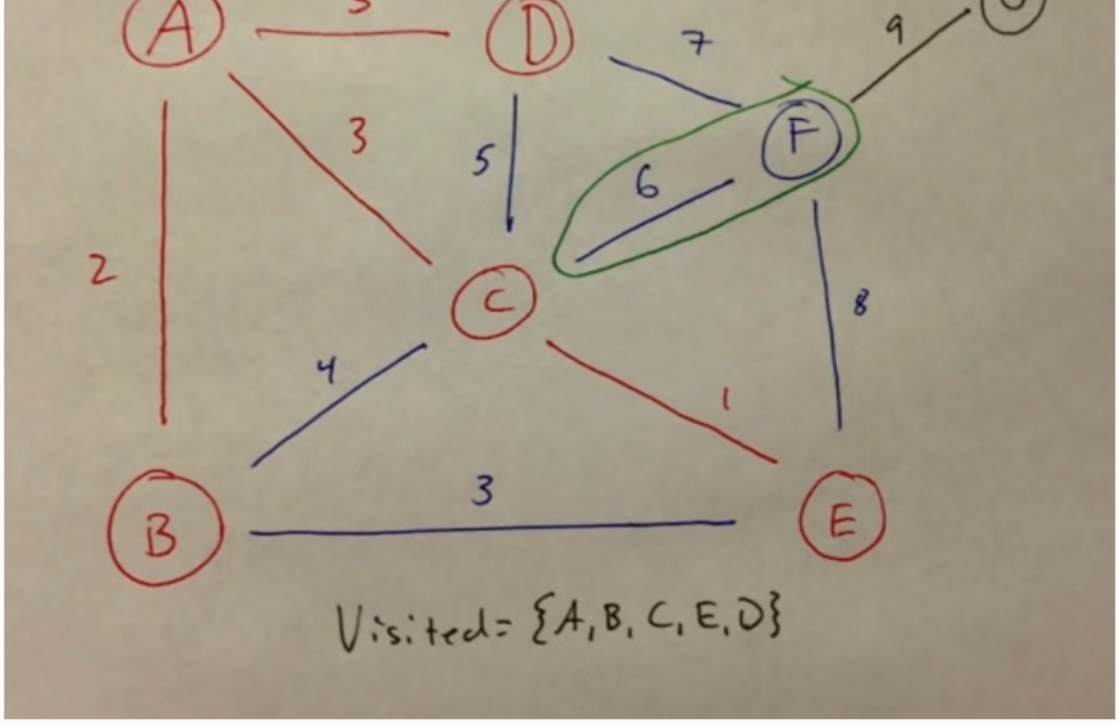


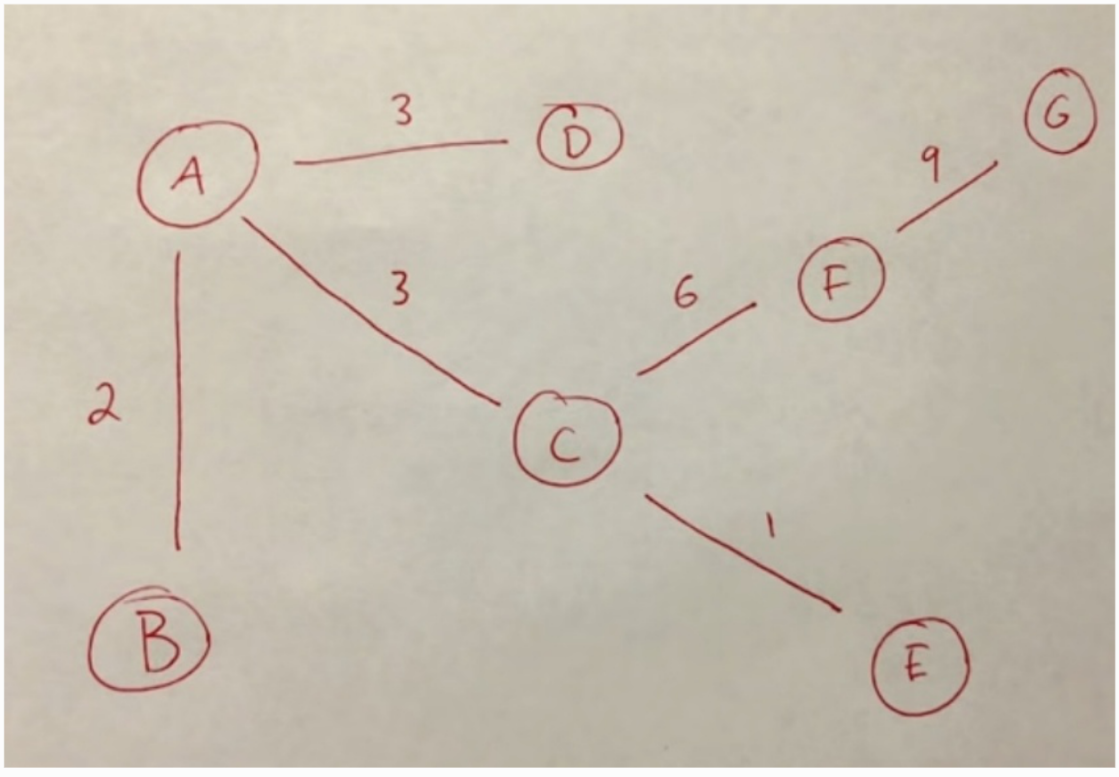

2
C
8
4
1
3
B
E
Visited = {A, B}


3
D
G
7
9
A
5
F
3
6
2
C
8
4
1
B
3
E
Visited = {A, B, C}


3
D
G
7
9
A
5
F
3
6
2
C
8
4
1
B
3
E
Visited = {A, B, C, E}







Time Complexity

Adjacency Matrix, Searching → $O(V^2)$

Binary Heap and Adjacency list → $O(V \log V + E \log V)$

```
procedure prim(G, w)
Input:    A connected undirected graph G = (V, E) with edge weights w_e
Output:   A minimum spanning tree defined by the array prev

for all u ∈ V:
   cost(u) = ∞
   prev(u) = nil
Pick any initial node u_0
cost(u_0) = 0

H = makequeue(V)     (priority queue, using cost-values as keys)
while H is not empty:
   v = deletemin(H)
   for each {v, z} ∈ E:
      if cost(z) > w(v, z):
         cost(z) = w(v, z)
         prev(z) = v
         decreasekey(H, z)
```

Source: http://www.cs.berkeley.edu/~vazirani/algorithms/chap5.pdf