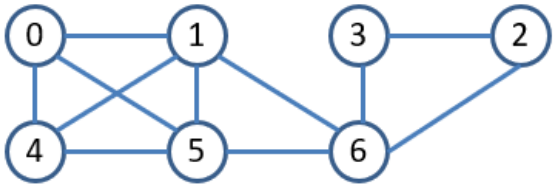
BFS-Visit(G,s) // search graph G starting from vertex s.

1. *For each vertex u of G*
    1. *color[u] = WHITE // undiscovered*
    2. *dist[u] = inf // distance from s to u*
    3. *pred[u] = NIL // predecessor of u on the path from s to u*
2. *color[s] = GRAY // s is being processed*
3. *dist[s] = 0*
4. *pred[s] = NIL*
5. *Initialize empty queue Q*
6. *put(Q,s) // s goes to the end of Q*
7. *While Q is not empty*
    1. *u = get(Q) // removes u from the front of Q*
    2. *For each v adjacent to u //explore edge (u,v) // in increasing order*
        1. *If color[v] == WHITE*
            1. *color[v] = GRAY*
            2. *dist[v] = dist[u]+1*
            3. *pred[v] = u*
            4. *put(Q,v)*
    3. *color[u] = BLACK*

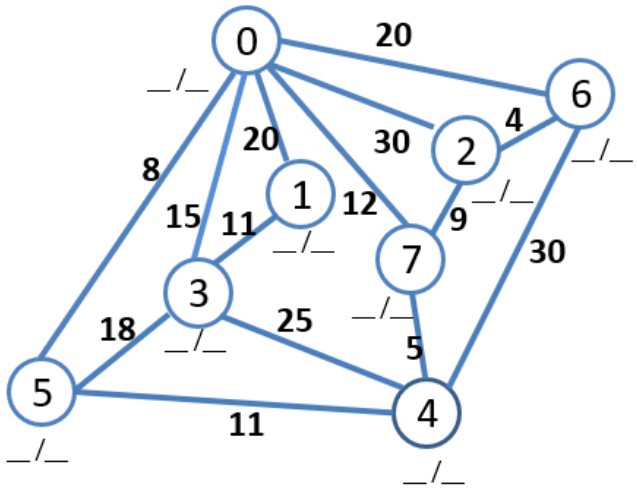
Representation	*BFS time complexity*	*BFS space complexity*
Adj LIST	*O(_____)*	
Adj MATRIX	*O(_____)*	



Vertex	*0*	*1*	*2*	*3*	*4*	*5*	*6*
Work (dist and parent updates for nodes)							

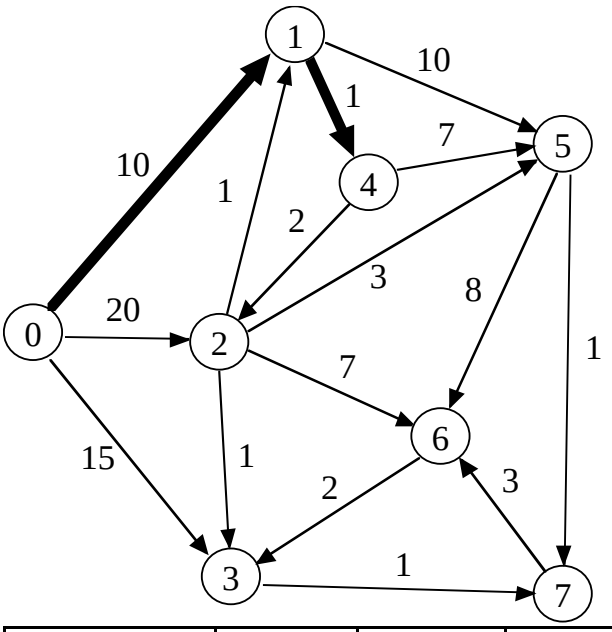
```
MST_Prim(G,w,s) // N = |V|
1   int d[N], p[N]
2   For v =0 -> N-1
3       d[v]=inf    //min weight of edge connecting v to MST
4       p[v]=-1     //MST vertex, s.t. w(p[v],v) =d[v]
5   d[s]=0
6   Q = PriorityQueue(G.V, d)
7   While notEmpty(Q)
8       u = removeMin(Q,d)
9       for each v adjacent to u
10          if v in Q and w(u,v)<d[v] {
11              p[v]=u
12              d[v] = w(u,v);
13              decreasedKeyFix(Q,v,d)  //v is neither index nor key
14          }
15  }
```

Assume adjacency list representation. TC: SC:



Vertex	0	1	2	3	4	5	6	7
Work (dist and parent updates for nodes)								

```
Dijkstra(G,w,s) // N = |V|
1   int d[N], p[N]
2   For v =0 -> N-1
3       d[v]=inf    //total weight from s to v
4       p[v]=-1   //v's predecessor on path s to v
5   d[s]=0
6   Q = PriorityQueue(d)
7   While notEmpty(Q) {
8       u = removeMin(Q,w)
9       for each v adjacent to u
10          if v in Q and (d[u]+w(u,v))<d[v]{
11              p[v]=u
12              d[v] = d[u]+w(u,v);
13               decreasedKeyFix(Q,v,d)
14          }
15  }
```



Vertex	0	1	2	3	4	5	6	7
Work (dist and parent updates for nodes)								