BFS-visit(G,2) V vertices E edges DFS, DFS-Visit

002

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 -1 // 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 -1*
5. *Initialize empty queue Q // O(1)*
6. *put(Q,s) // s goes to the end of Q // O(1)*
7. *While Q is not empty -> O(V) u = 2*
   1. *u = get(Q) // removes u from the front of Q // O(1)*
   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 // O(1)*

O(V) + O(1) + O(V²) O(E) O(1) O(1) O(1)

mandatory. O(V) optional

node at most 1 time in Q

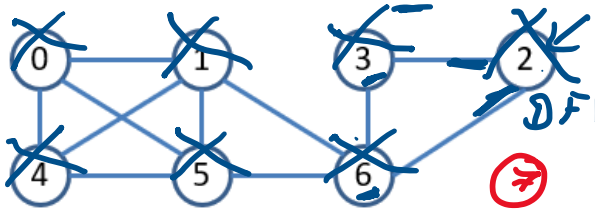
Θ(V) matrix Θ(deg(u)) adj list

Representation	*BFS time complexity*	*BFS space complexity*
Adj LIST	*O(V+E)*	O(V)
Adj MATRIX	*O(V²)*	O(V)

if graph is connected: V ? E

white = undiscovered
gray = discovered, in queue (being processed)
black = out of the queue

6 → 0 2 → 6 → 1 → 0



DFW

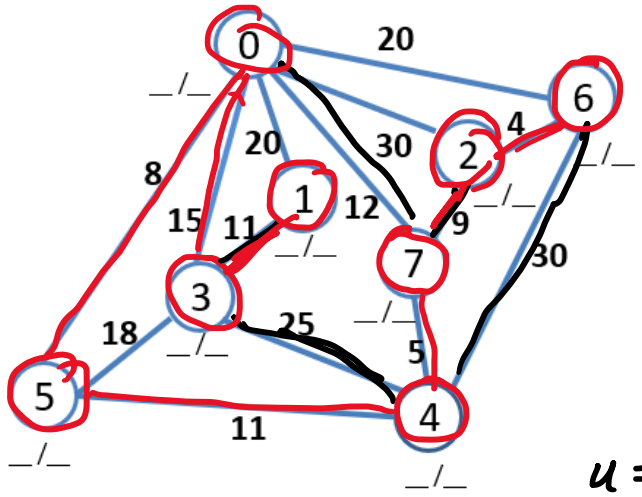
Q: 2, 3, 6, 1, 5, 0, 4

u = 2 3 6 1 5 0 4

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

```
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:



MST_Prim(G, 7)

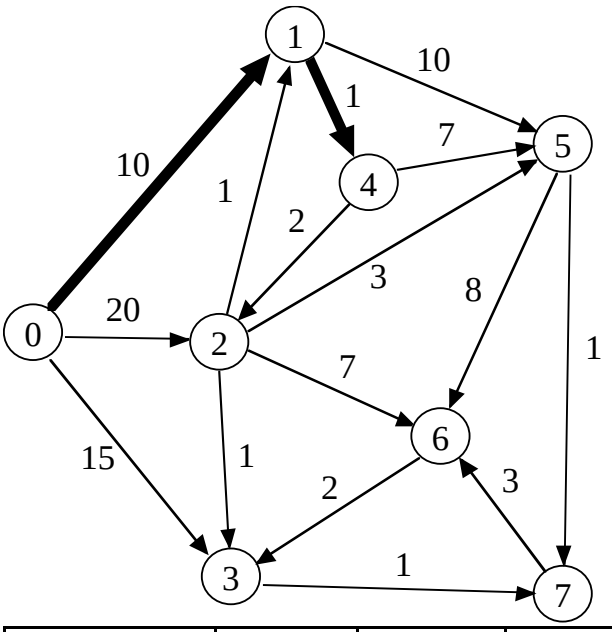
u = 7 4 2 6 5 0 3, 1

Vertex	0	1	2	3	4	5	6	7
Work (dist and parent updates for nodes)	∞/-1 → 12/7 → 30>12 not update → 8/5	∞/-1 → 20/4 → 20/0 → 11<20 → 11/3	∞/-1 → 9/7	∞/-1 → 25/4 → 18<25 → 18/5 → 15<18 → 15/0	∞/-1 → 5/7	∞/-1 → 11/4	∞/-1 → 30/4 → 4<30 → 4/2	0/-1

(5,0) (3,1) (7,2), (0,3) (7,4) (4,5) (3,6),

8 11 2

```
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)								