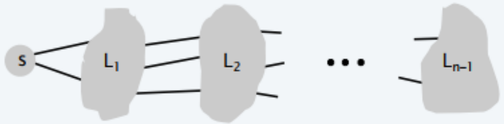
GRAPH TRAVERSAL ALGORITHM: BREADTH FIRST SEARCH (BFS)

WHAT IS BREADTH FIRST SEARCH (BFS)?

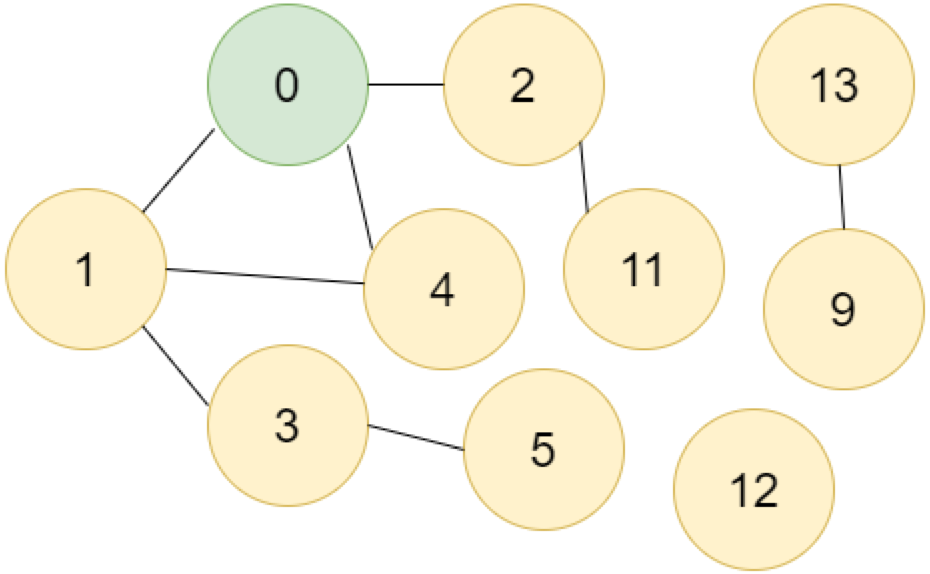
- Starts with the source node and then traverse the adjacent/neighbor nodes.
  - Then traverse the neighbors of neighbors.
    - That is explore each neighbor of the current node before exploring the children of the neighbors
  - That it traverses nodes level by level (or in order their breadth).
    - What do we use to traverse level by level order?
      - Queue
  - First traverses all the nodes which is in one edge distance, then traverses the nodes which have two edge distances from the nodes and so on.
- When processing a node, marks it so that no nodes gets processed more than once.



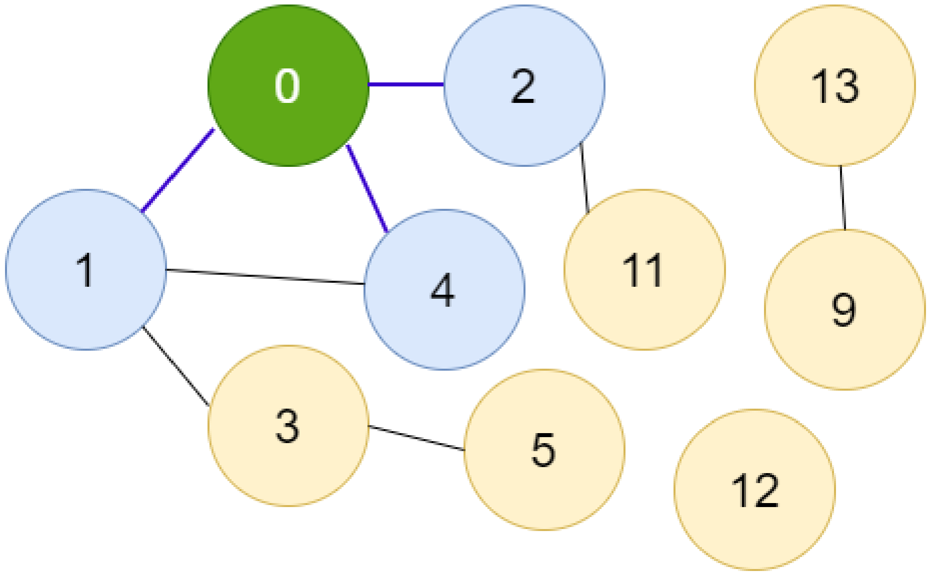
BFS algorithm.

- $L_0 = \{ s \}$.
- L_1 = all neighbors of L_0.
- L_2 = all nodes that do not belong to L_0 or L_1, and that have an edge to a node in L_1.
- L_{i+1} = all nodes that do not belong to an earlier layer, and that have an edge to a node in L_i.

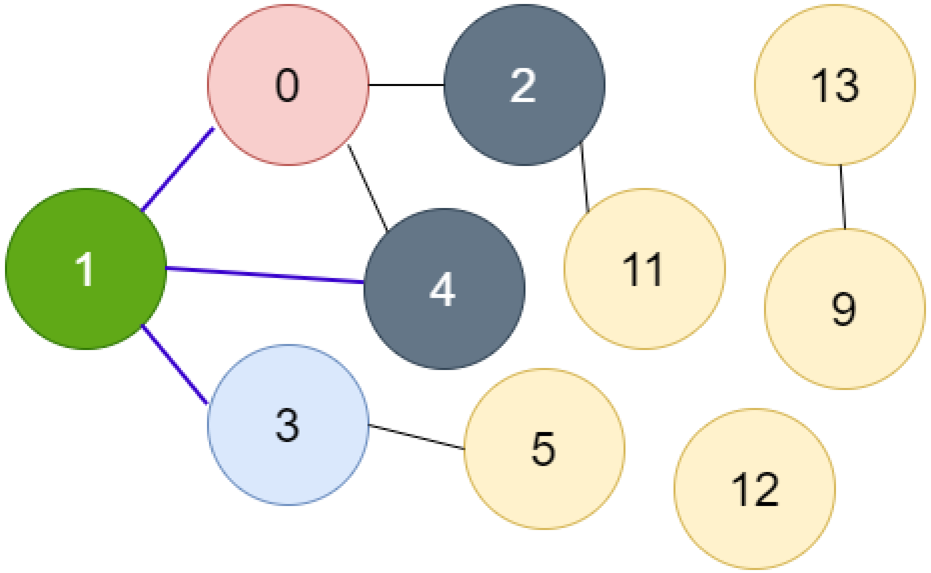
Image Source: KT



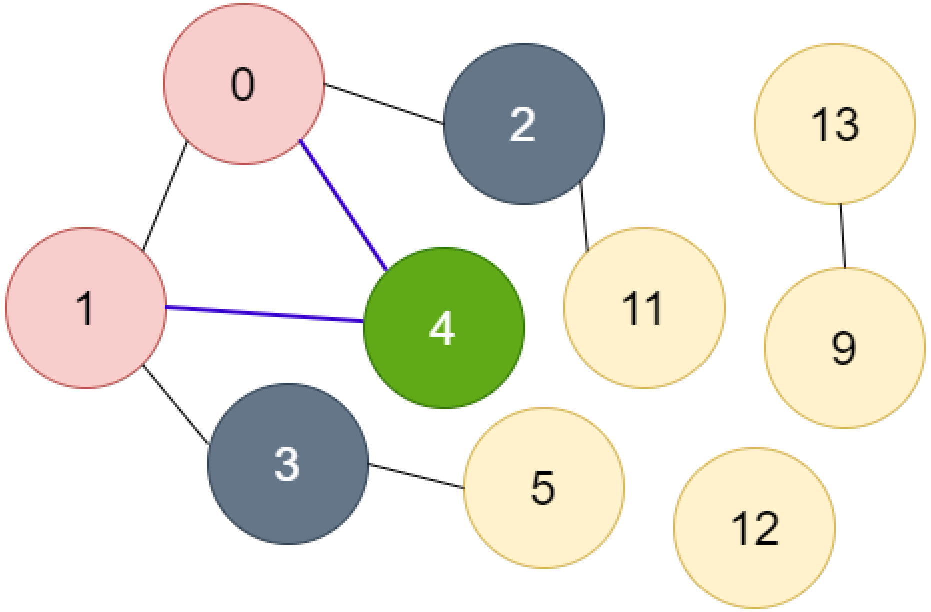
0	end



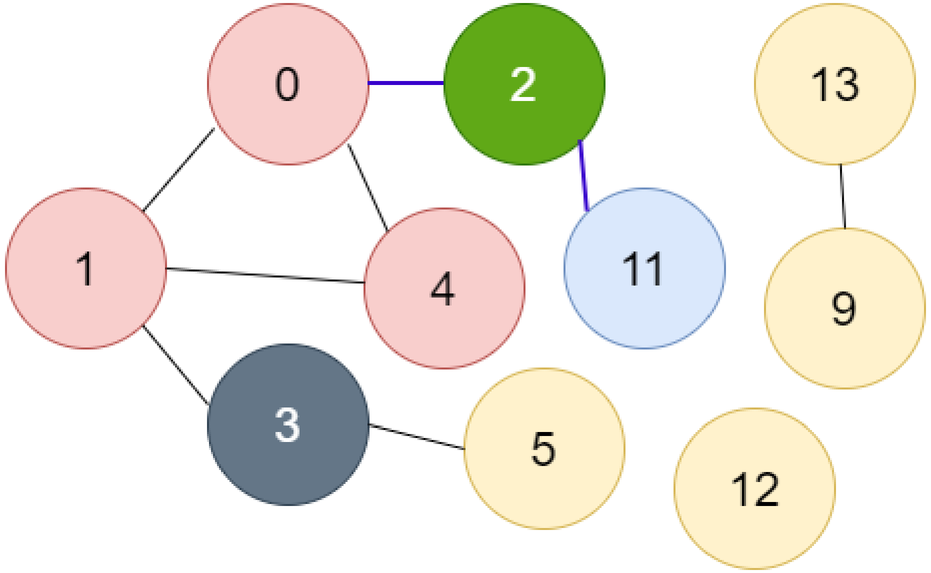
1	4	2	end



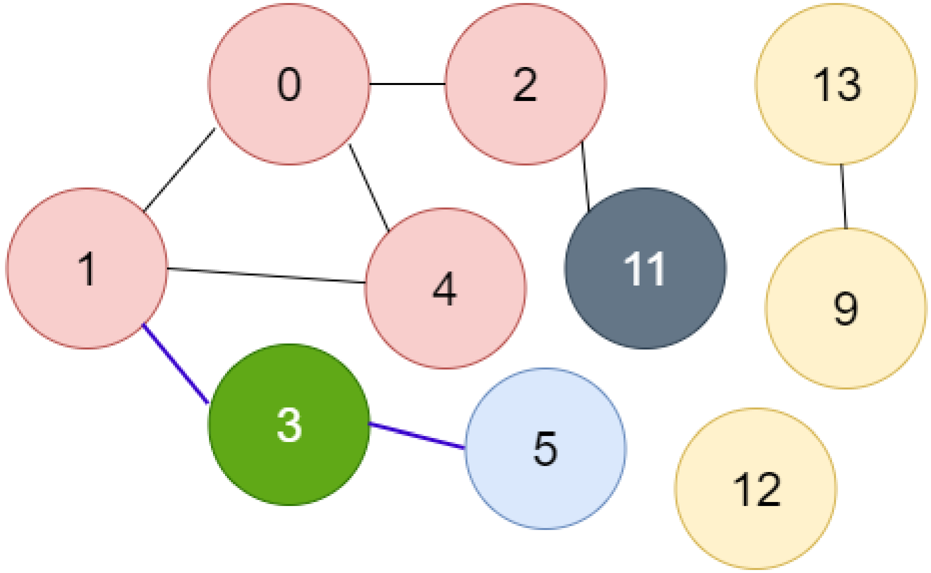
4	2	3	end



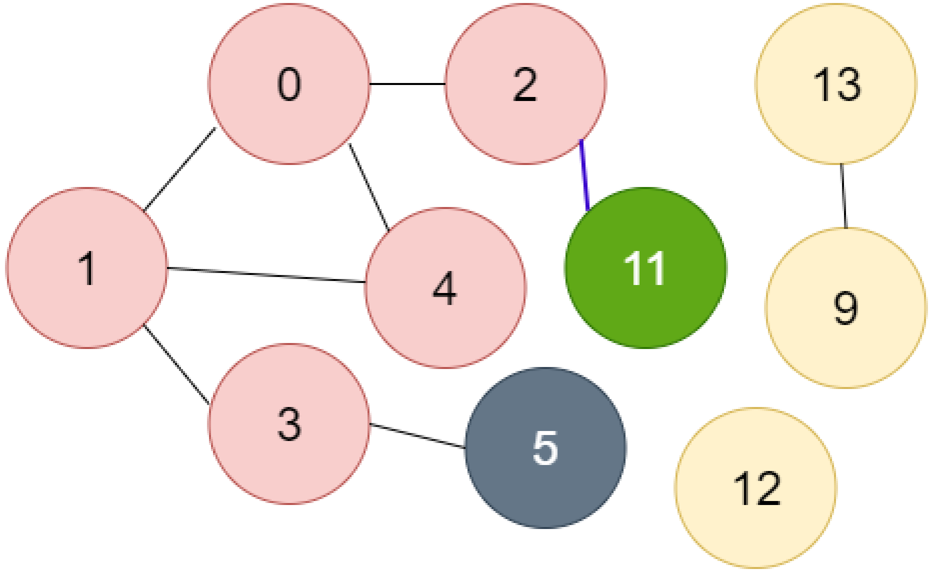
2	3	end



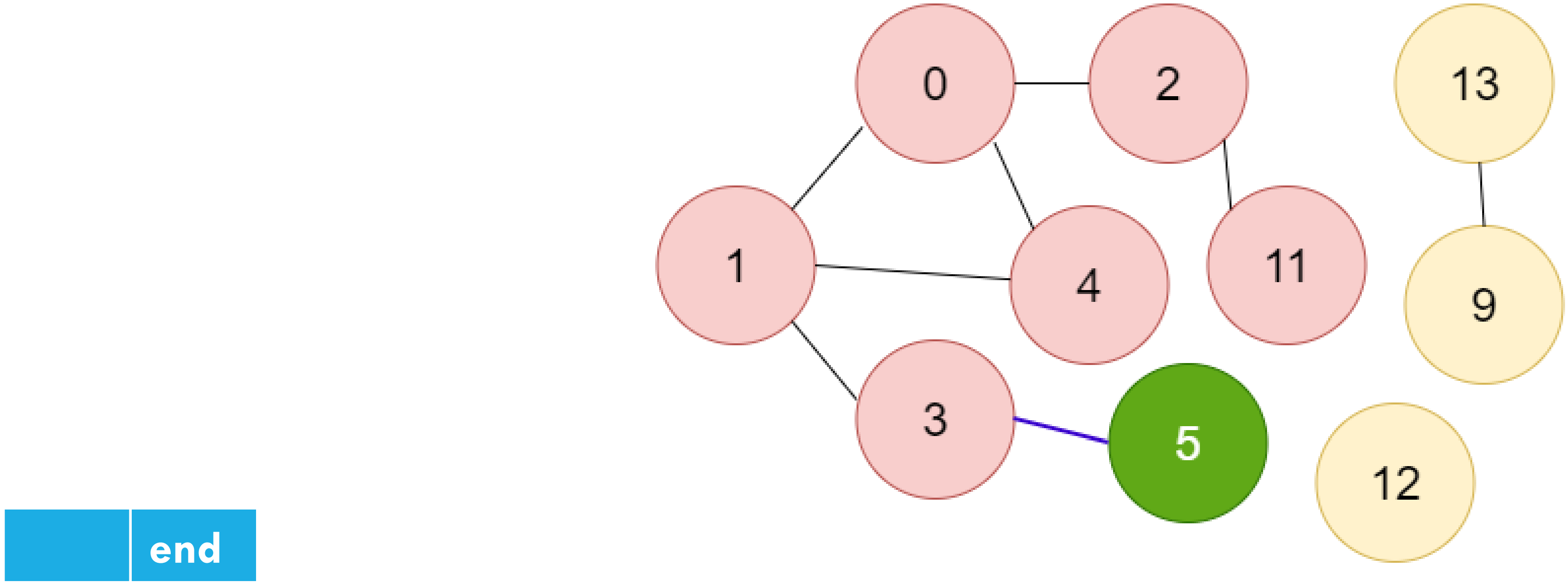
3	11	end

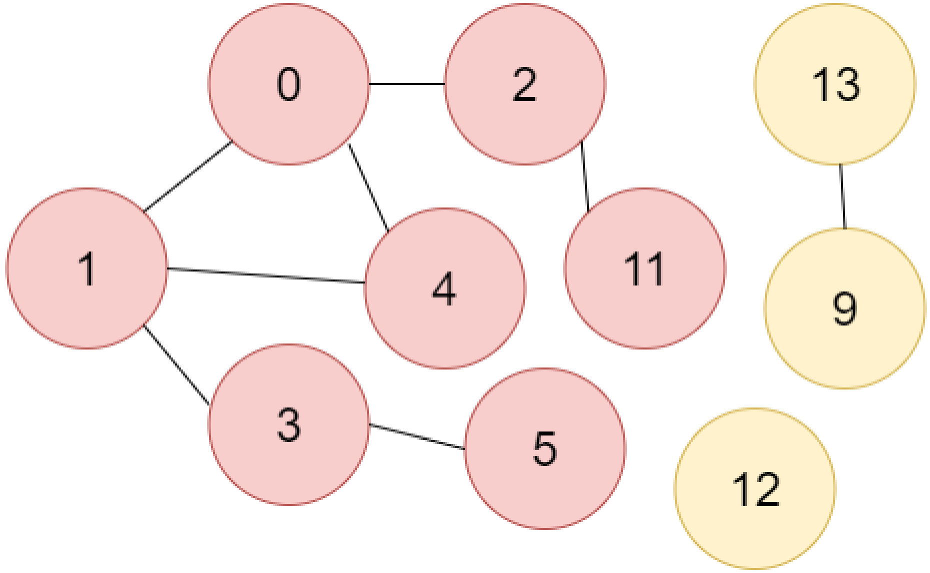


11	5	end

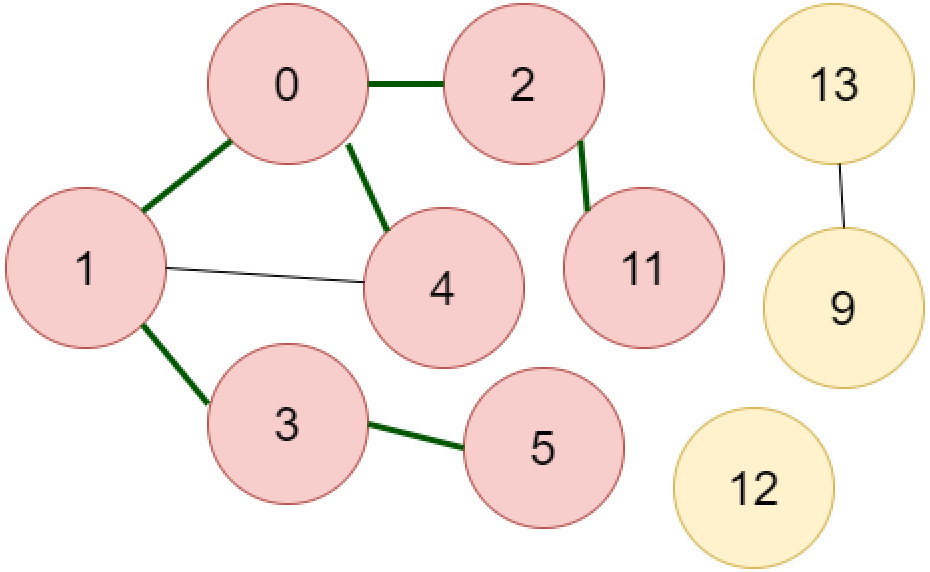


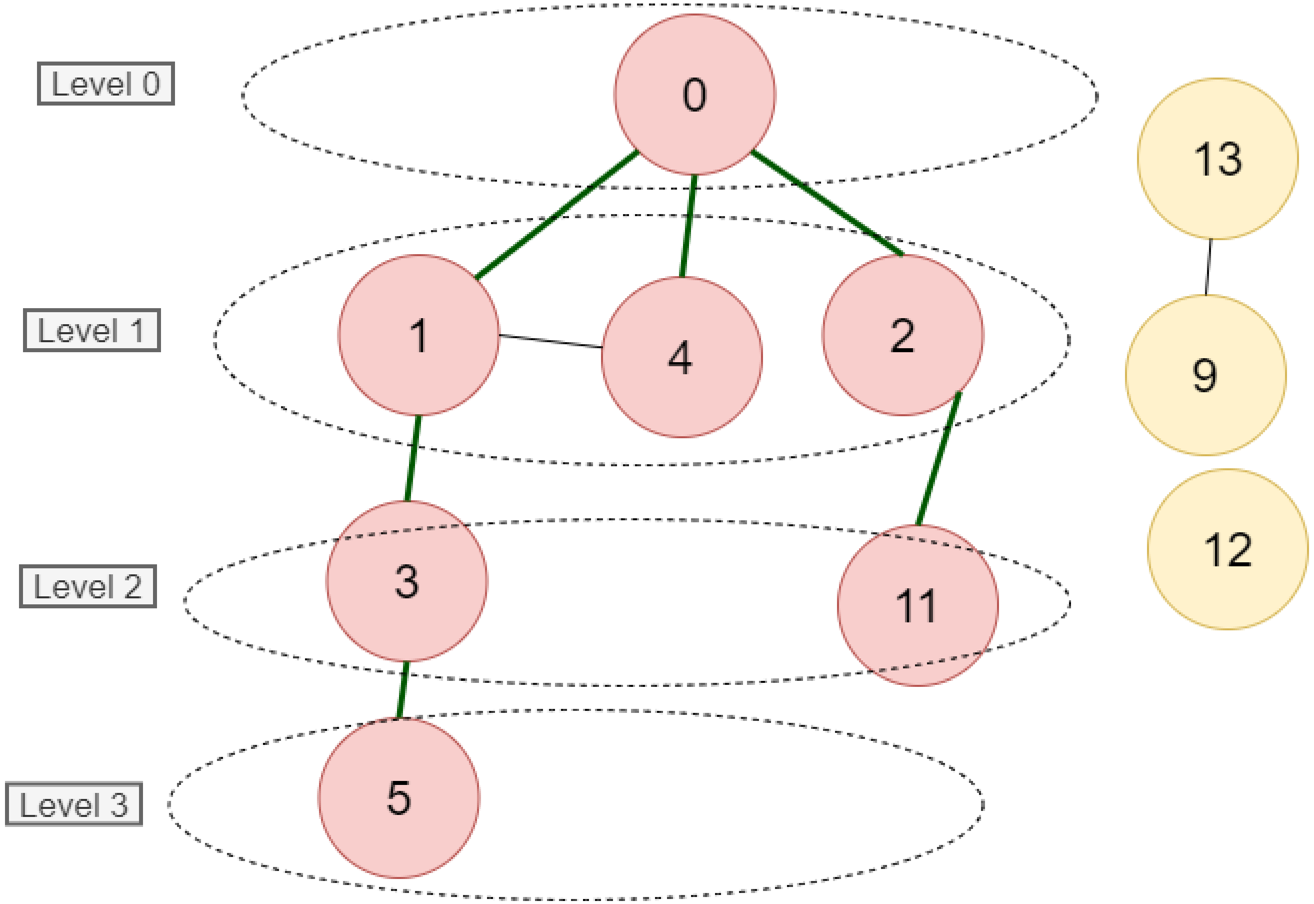
5	end


0
2
13
1
4
11
9
3
5
12
end



end


0
2
13
1
4
11
9
3
5
12


Level 0
0
13
Level 1
1
4
2
9
Level 2
3
11
12
Level 3
5

PSEUDO CODE

BFS (from source vertex s)

Put s onto a FIFO queue, and mark s as visited.

Repeat until the queue is empty:

- remove the least recently added vertex v
- add each of v's unvisited neighbors to the queue, and mark them as visited.

Image Source: RS, KW

PSEUDO CODE

BFS

Input: graph $G = (V, E)$ in adjacency-list representation, and a vertex $s \in V$.

Postcondition: a vertex is reachable from s if and only if it is marked as "explored."

---

mark s as explored, all other vertices as unexplored
Q := a queue data structure, initialized with s
while Q is not empty **do**
    remove the vertex from the front of Q, call it v
    for each edge (v, w) in v's adjacency list **do**
        if w is unexplored **then**
            mark w as explored
            add w to the end of Q

Image Source: T. RoughGarden

Complexity: $O(V + E)$
For adjacency List

Complexity: $O(V^2)$
For adjacency matrix

PSEUDO CODE

```
procedure bfs(G, s)
Input:    Graph G = (V, E), directed or undirected; vertex s ∈ V
Output:   For all vertices u reachable from s, dist(u) is set
          to the distance from s to u.

for all u ∈ V:
   dist(u) = ∞

dist(s) = 0
Q = [s]  (queue containing just s)
while Q is not empty:
   u = eject(Q)
   for all edges (u, v) ∈ E:
      if dist(v) = ∞:
         inject(Q, v)
         dist(v) = dist(u) + 1
```

Image Source: DPV

PSEUDO CODE

```
BFS(G, s)
for each vertex u ∈ G.V − {s}
    u.color = WHITE
    u.d = ∞
    u.π = NIL
s.color = GRAY
s.d = 0
s.π = NIL
Q = ∅
ENQUEUE(Q, s)
while Q ≠ ∅
    u = DEQUEUE(Q)
    for each v ∈ G.Adj[u]
        if v.color == WHITE
            v.color = GRAY
            v.d = u.d + 1
            v.π = u
            ENQUEUE(Q, v)
    u.color = BLACK
```

Figure 22.3 illustrates the progress of BFS on a sample graph.

Image Source: CLRS

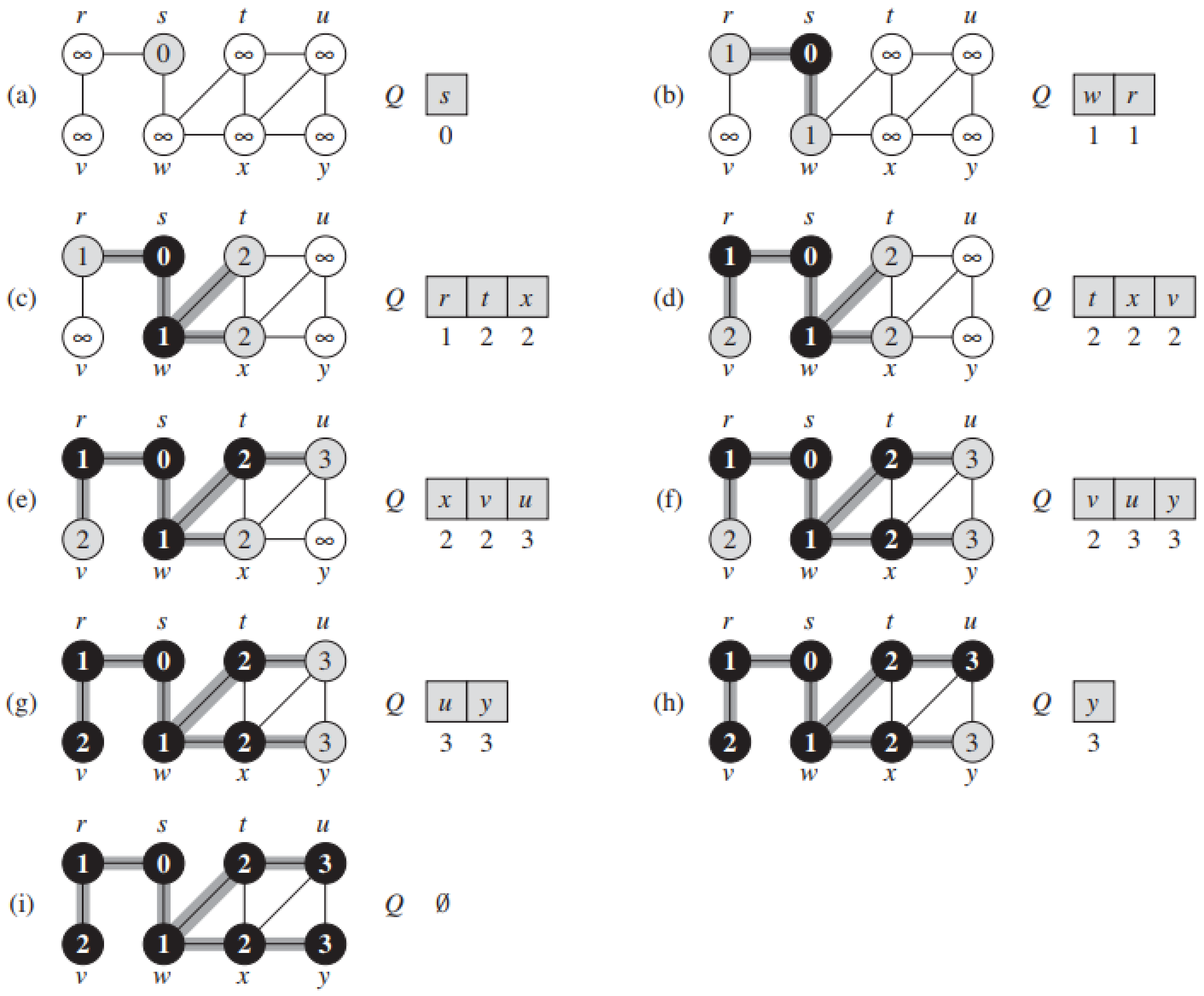


Image Source: CLRS

Figure 22.3 The operation of BFS on an undirected graph. Tree edges are shown shaded as they are produced by BFS. The value of $u.d$ appears within each vertex u. The queue Q is shown at the beginning of each iteration of the **while** loop of lines 10–18. Vertex distances appear below vertices in the queue.

NEXT TOPIC?

- STL Priority Queue (Heap)
- Single Source Shortest Path (SSSP) Problem
  - Dijkstra’s Algorithm (Greedy)