Graph

Search explore

Searching Algorithm :

-Blind Search Algorithm (also called an uninformed search) :

is a search that has no information about its domain or nature of the problem.

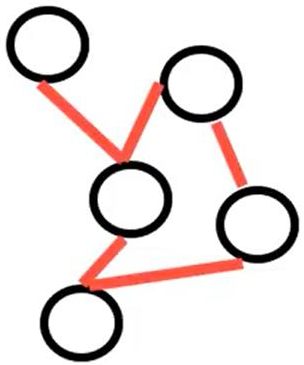
EX:
1.Breadth-First Search
2.Depth-First Search

-Heuristic Search Algorithm (also called an informed or directed search):

have further information about the cost of the path between any state in search space and the goal state.

EX:
1.Best-First Search
2.A-Star (A*)
3.Tabu Search

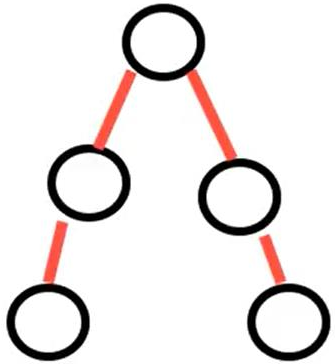
Input → Algorithms → Output



Breadth-First Search

Depth-First Search

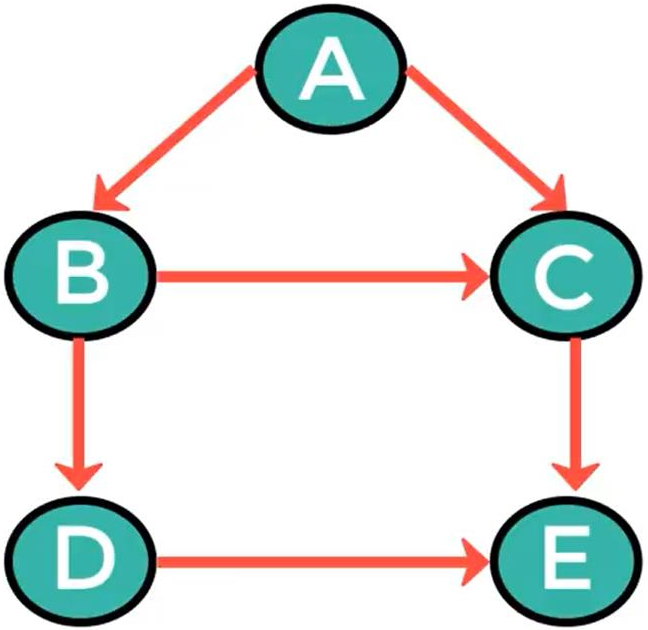
Shortest Path (Dijkstra's)



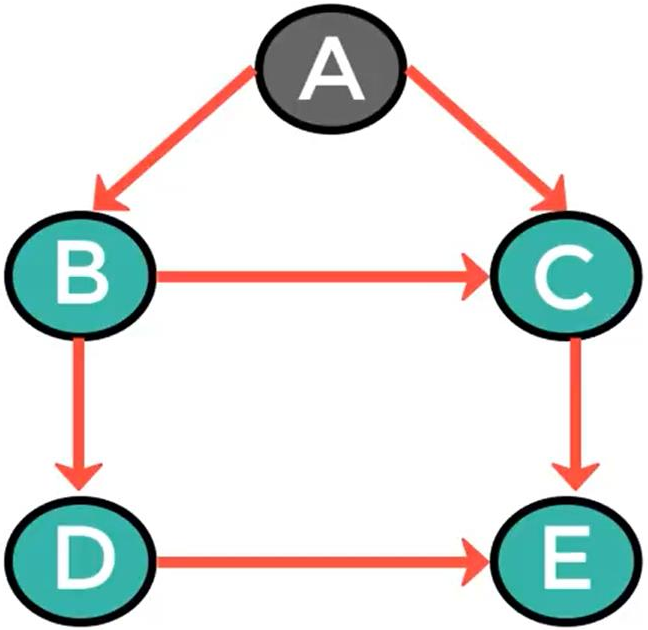
Breadth First Search-

- Breadth First Search or BFS is a graph traversal algorithm.
- It is used for traversing or searching a graph in a systematic fashion.
- BFS uses a strategy that searches in the graph in breadth first manner whenever possible.
- Queue data structure is used in the implementation of breadth first search.

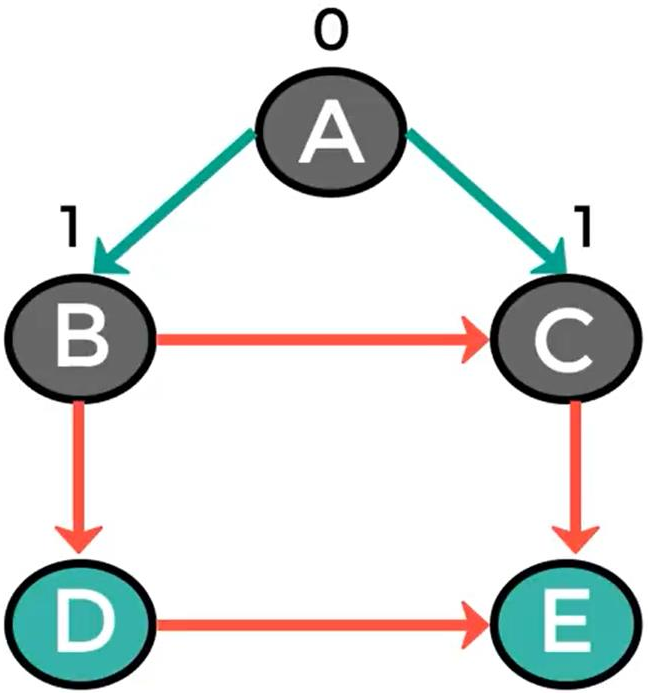
Breadth-First Search



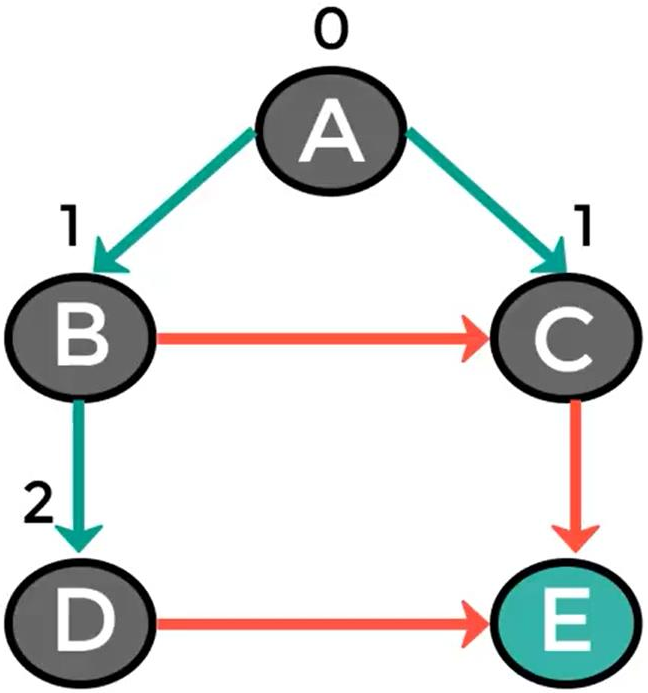
Breadth-First Search



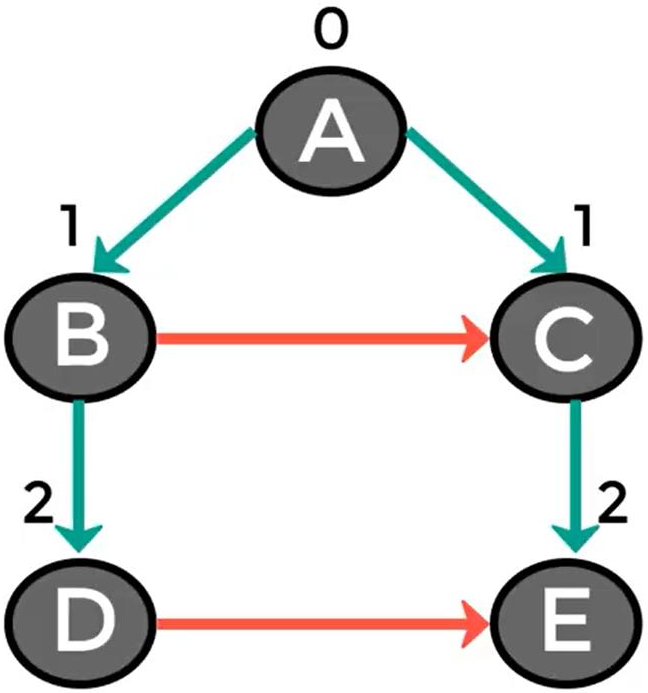
Breadth-First Search



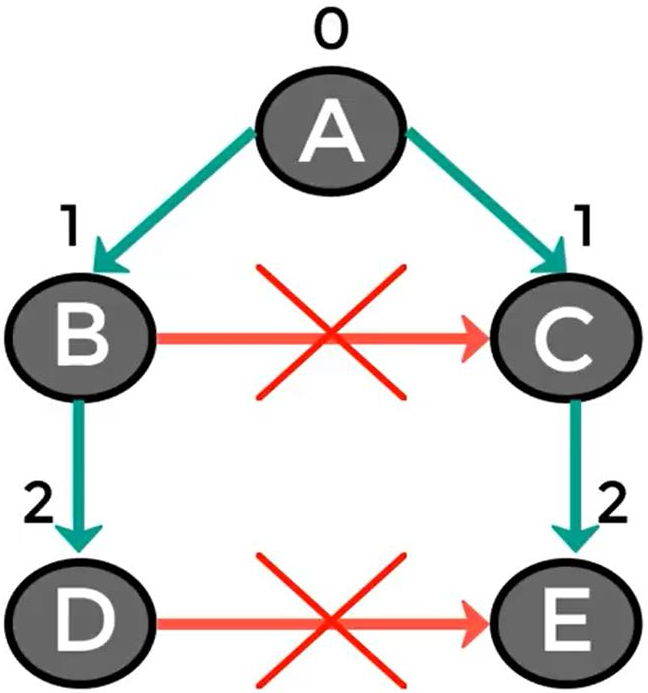
Breadth-First Search



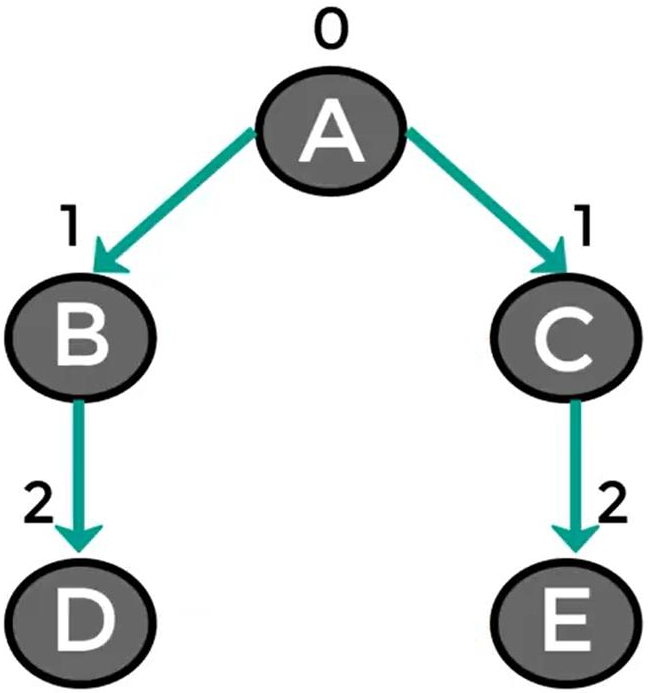
Breadth-First Search



Breadth-First Search



Breadth-First Search



Breadth First Search Algorithm-

BFS (V,E,s)

for each vertex v in V – {s} do color[v] ← WHITE d[v] ← ∞ π[v] ← NIL
color[s] = GREY

d[s] ← 0 π[s] ← NIL

Q ← { }

ENQUEUE (Q,s)

While Q is non-empty

do v ← DEQUEUE (Q)

for each u adjacent to v

do if color[u] ← WHITE

then color[u] ← GREY

d[u] ← d[v] + 1

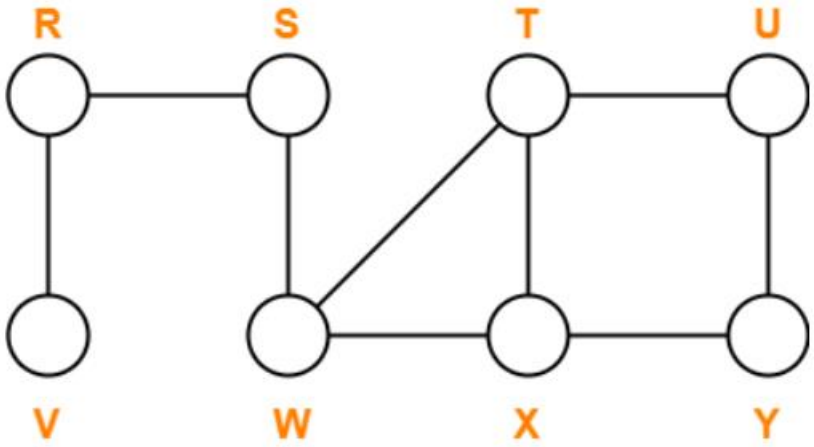
π[u] ← v

ENQUEUE (Q,u)

color[v] ← BLACK

Problem-

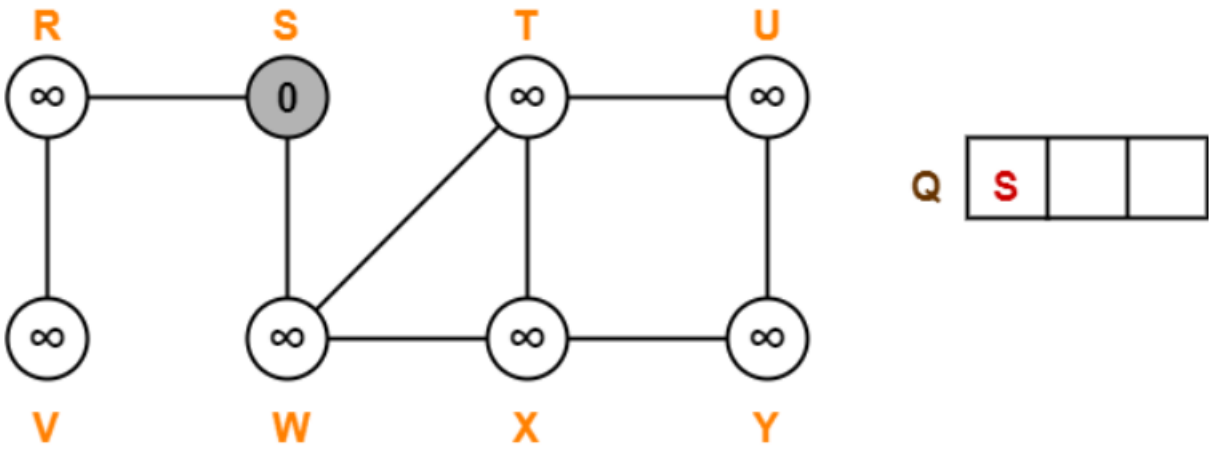
- Traverse the following graph using Breadth First Search Technique-



- Consider vertex S as the starting vertex.

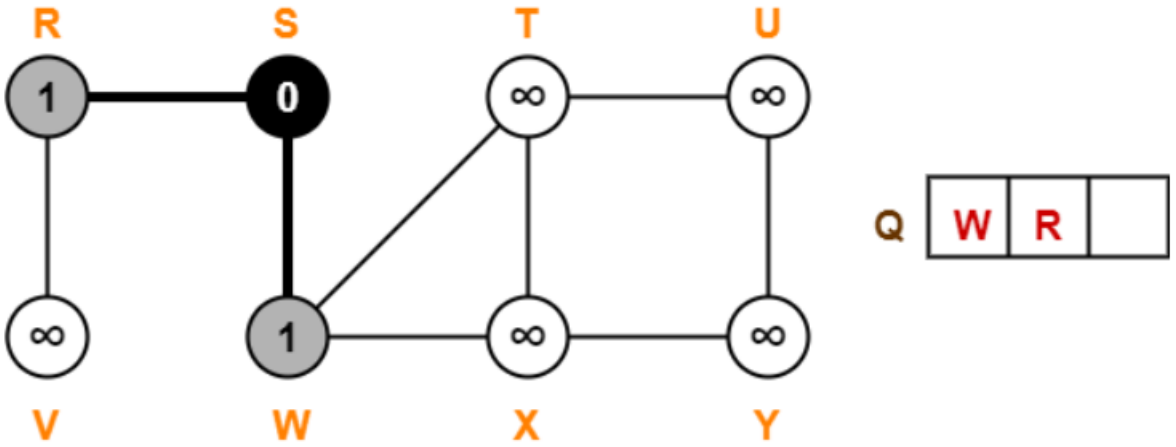
Solution-

- **Step-01:**
- For all the vertices v except source vertex S of the graph, we initialize the variables as-
- color[v] = WHITE , π[v] = NIL , d[v] = ∞
- For source vertex S, we initialize the variables as-
- color[S] = GREY , π[S] = NIL , d[S] = 0
- We enqueue the source vertex S in the queue Q.



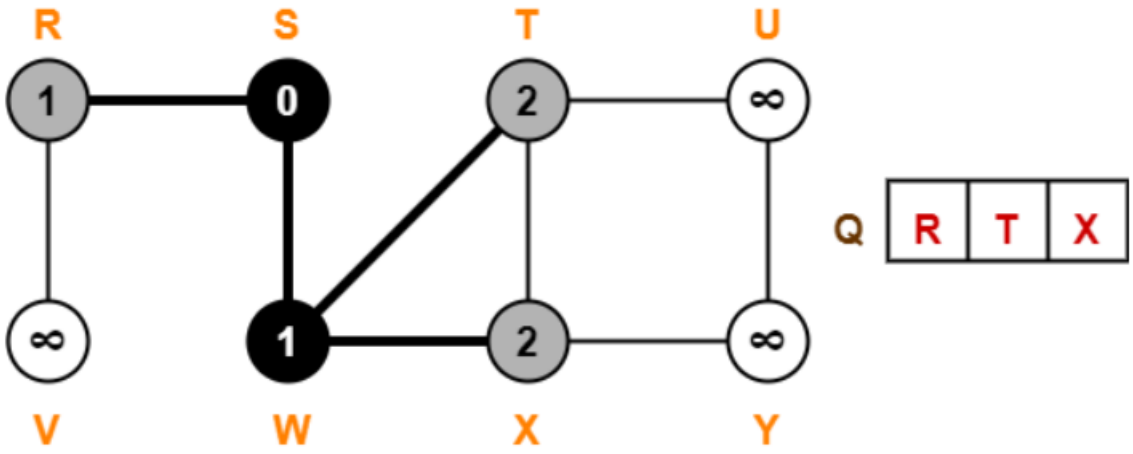
Solution-

- **Step-02:**
- Dequeue vertex S from the queue Q
- For all adjacent white vertices ‘v’ (vertices R and W) of vertex S, we do-
- color[v] = GREY
- d[v] = d[S] + 1 = 0 + 1 = 1
- π[v] = S
- Enqueue all adjacent white vertices of S in queue Q
- color[S] = BLACK



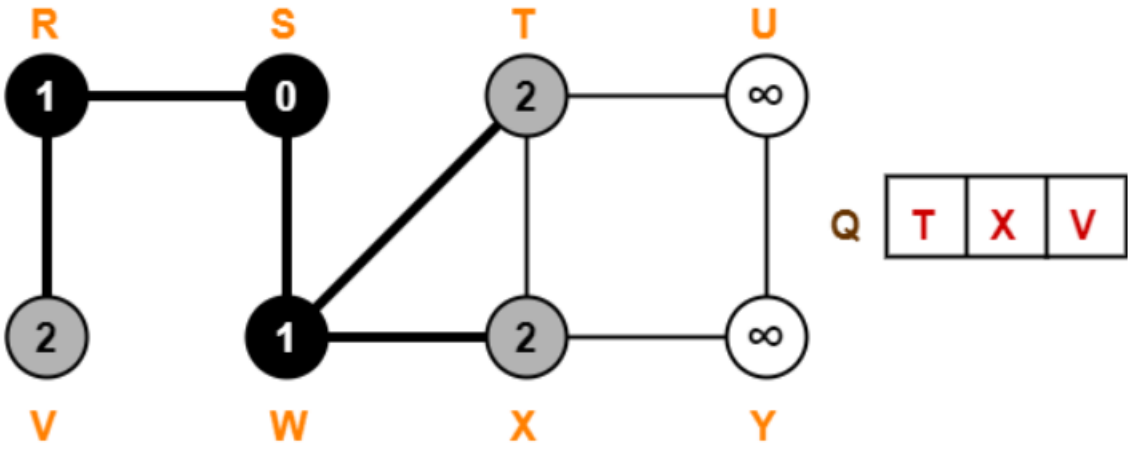
Solution-

- **Step-03:**
- Dequeue vertex W from the queue Q
- For all adjacent white vertices ‘v’ (vertices T and X) of vertex W, we do-
- color[v] = GREY
- d[v] = d[W] + 1 = 1 + 1 = 2
- π[v] = W
- Enqueue all adjacent white vertices of W in queue Q
- color[W] = BLACK



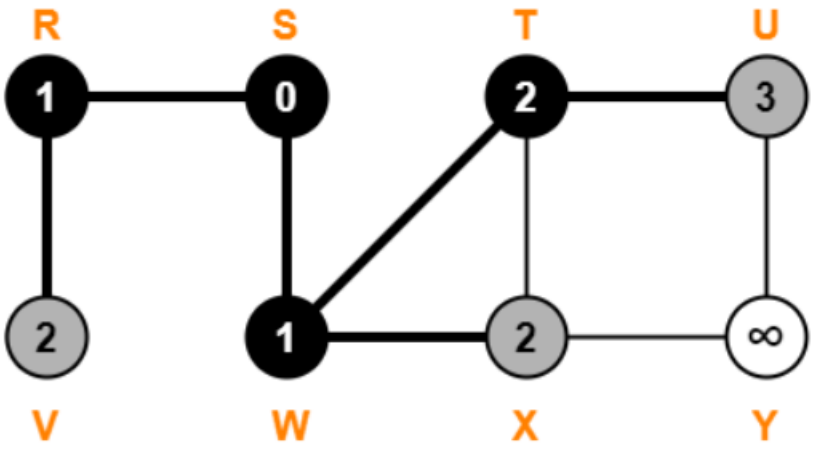
Solution-

- **Step-04:**
- Dequeue vertex R from the queue Q
- For all adjacent white vertices ‘v’ (vertex V) of vertex R, we do-
- color[v] = GREY
- d[v] = d[R] + 1 = 1 + 1 = 2
- π[v] = R
- Enqueue all adjacent white vertices of R in queue Q
- color[R] = BLACK



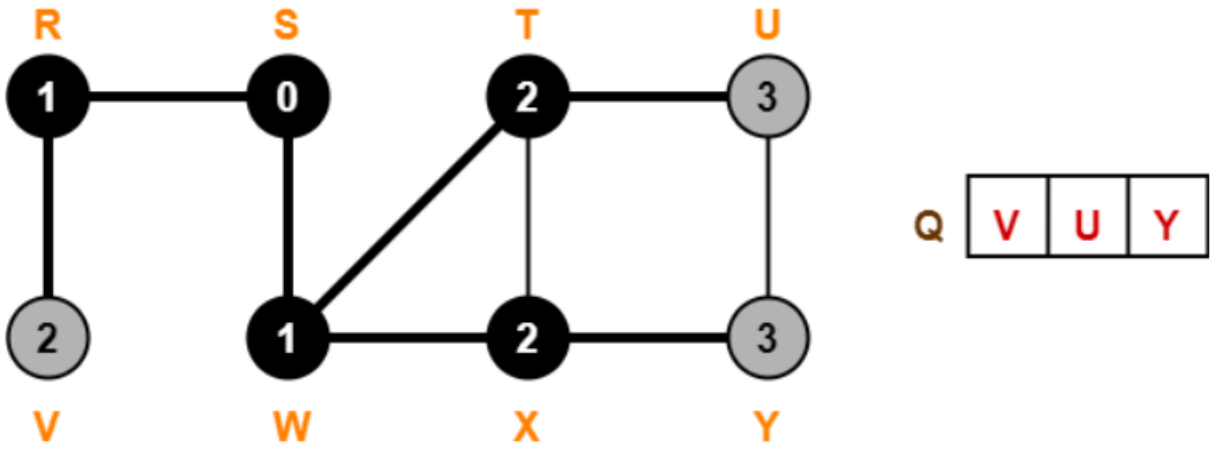
Solution-

- **Step-05:**
- Dequeue vertex T from the queue Q
- For all adjacent white vertices ‘v’ (vertex U) of vertex T, we do-
- color[v] = GREY
- d[v] = d[T] + 1 = 2 + 1 = 3
- π[v] = T
- Enqueue all adjacent white vertices of T in queue Q
- color[T] = BLACK



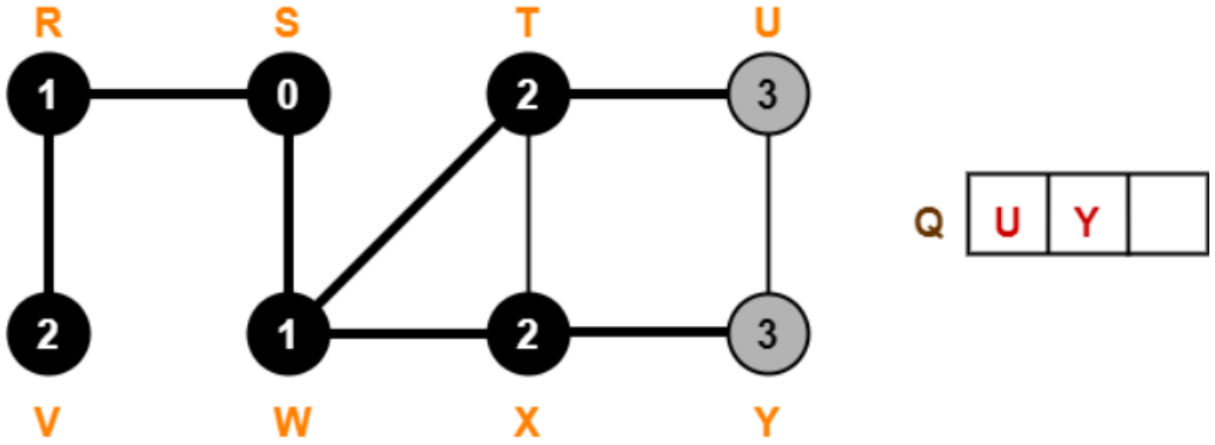
Solution-

- **Step-06:**
- Dequeue vertex X from the queue Q
- For all adjacent white vertices ‘v’ (vertex Y) of vertex X, we do-
- color[v] = GREY
- d[v] = d[X] + 1 = 2 + 1 = 3
- π[v] = X
- Enqueue all adjacent white vertices of X in queue Q
- color[X] = BLACK



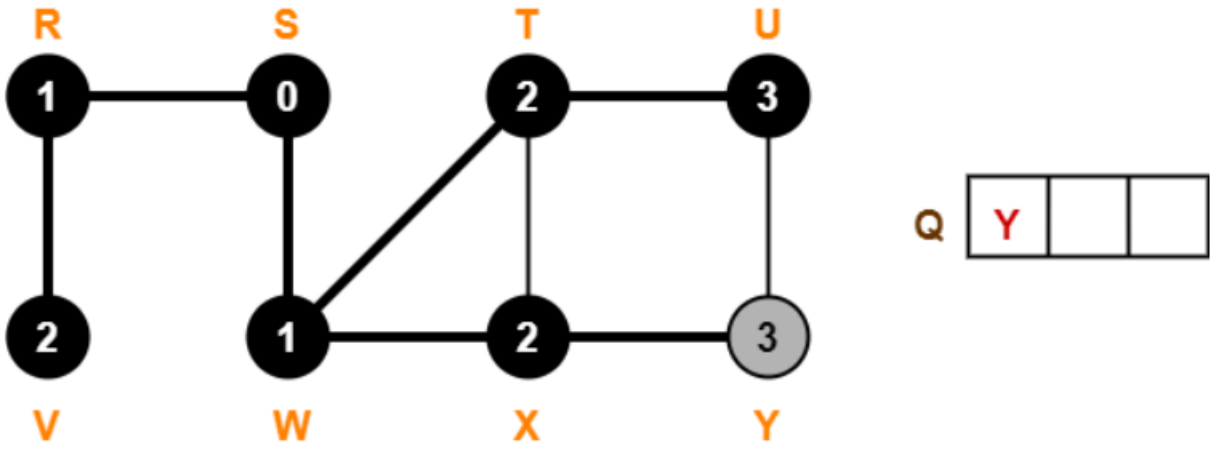
Solution-

- **Step-07:**
- Dequeue vertex V from the queue Q
- There are no adjacent white vertices to vertex V.
- color[V] = BLACK



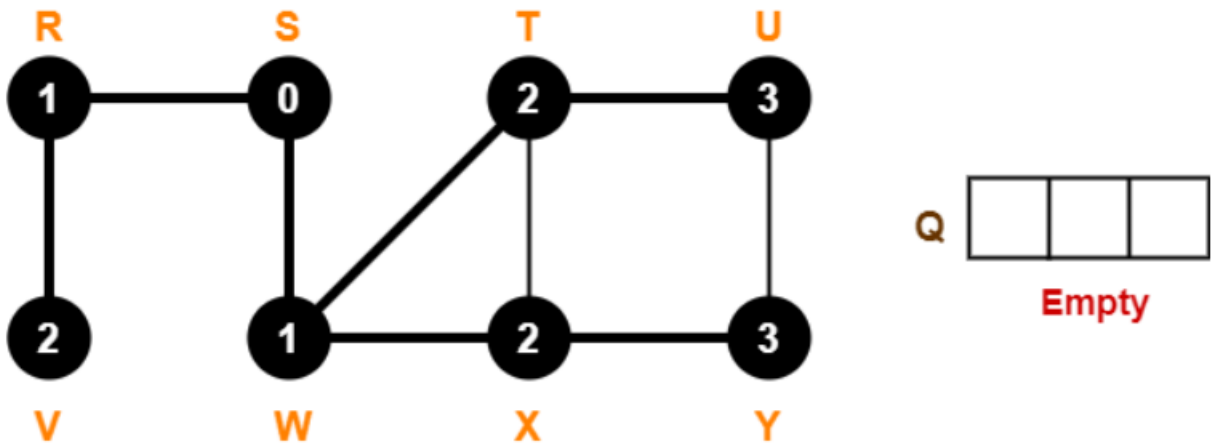
Solution-

- **Step-08:**
- Dequeue vertex U from the queue Q
- There are no adjacent white vertices to vertex U.
- color[U] = BLACK



Solution-

- **Step-09:**
- Dequeue vertex Y from the queue Q
- There are no adjacent white vertices to vertex Y.
- color[Y] = BLACK



- Since, all the vertices have turned black and the queue has got empty, so we stop.
- This is how any given graph is traversed using Breadth First Search (BFS) technique