```
BFS(G, s)
 1   for each vertex u ∈ V[G] − {s}
 2         do color[u] ← WHITE
 3            d[u] ← ∞
 4            π[u] ← NIL
 5   color[s] ← GRAY
 6   d[s] ← 0
 7   π[s] ← NIL
 8   Q ← ∅
 9   ENQUEUE(Q, s)
10   while Q ≠ ∅
         do u ← DEQUEUE(Q)
            for each v ∈ Adj[u]
                do if color[v] = WHITE
                      then color[v] ← GRAY
                           d[v] ← d[u] + 1
                           π[v] ← u
                           ENQUEUE(Q, v)
18          color[u] ← BLACK
```

G means Graph -> Adjacency Matrix, S means Source (Integer)

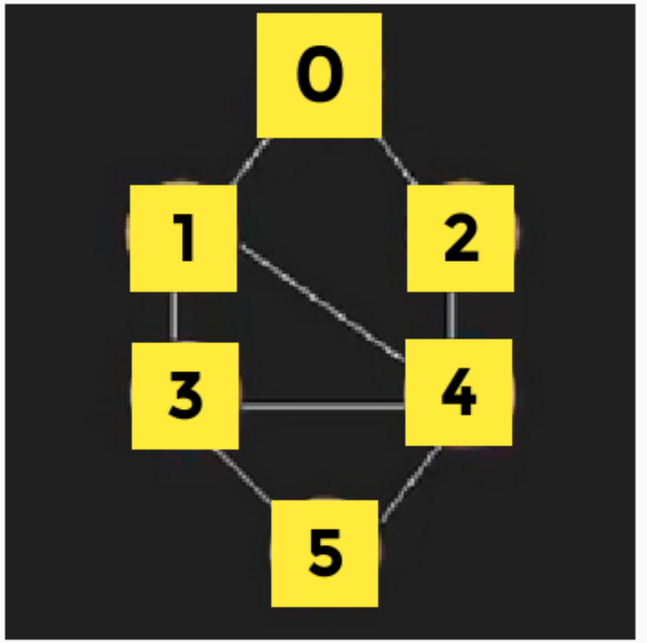
You need to have three arrays. Color, D (Distance/Level), Parent.

```
Queue<Integer> q = new LinkedList<Integer>();
```

White means 0 (Undiscovered). Gray means 1 (Visited but not processed). Black means 2 (Processed).

q.poll()

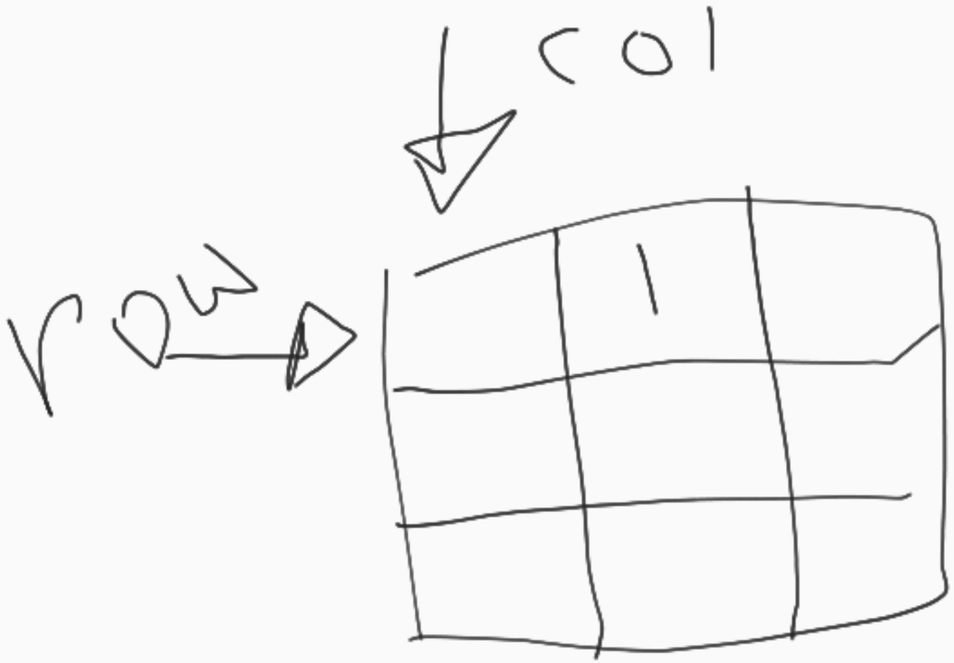
```
while(!q.isEmpty())
```



6 8 (0,1) (0,2) (1,3) (1,4) (2,4) (3,4) (4,5) (3,5)

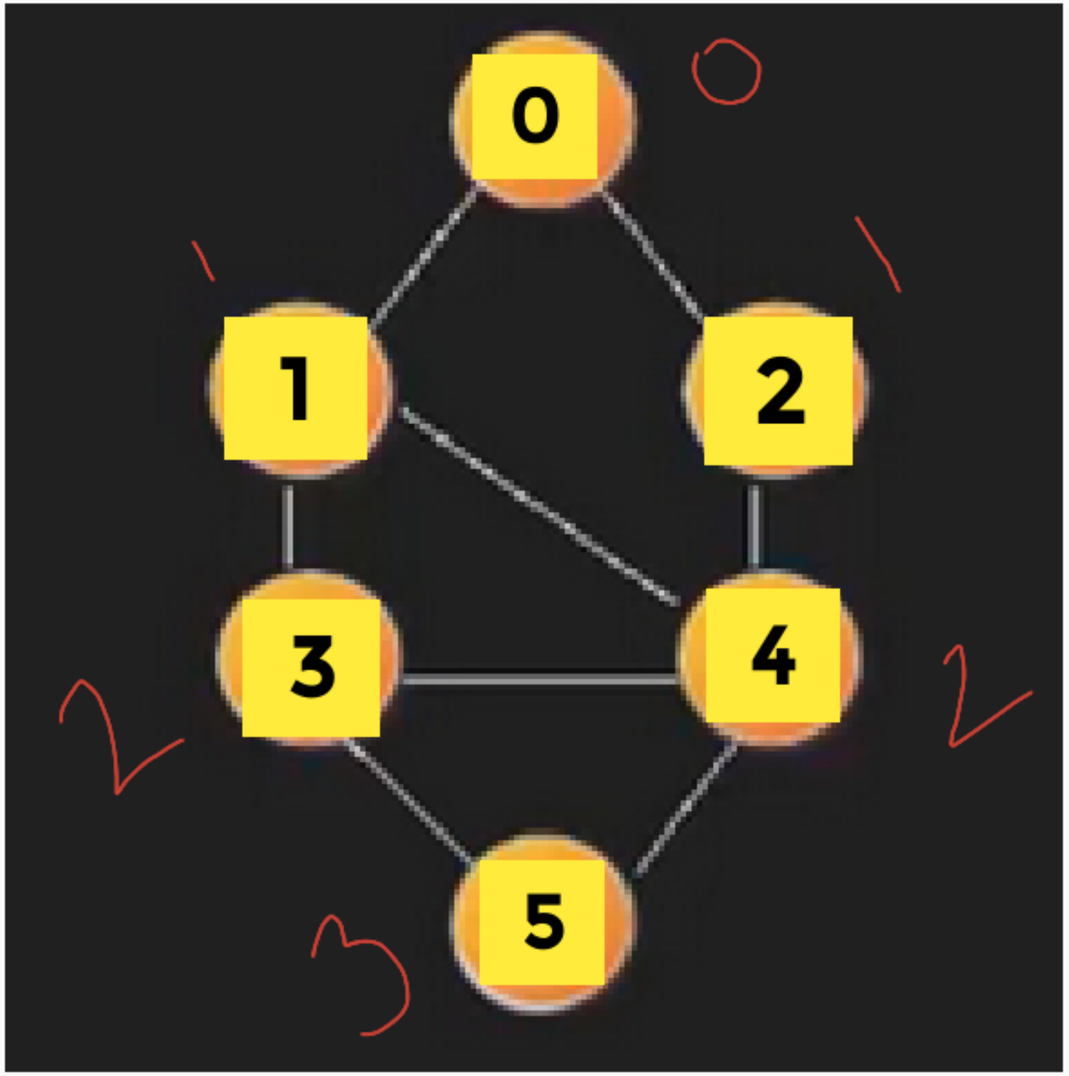
for each v that belongs to Adj[u] -> This means that the for loop will apply to each of the neighbors of u

```
for(int col = 0;
col<mat.length;
col++){
if(mat[u][col] == 1) {
}}
```



mat[row][col]

mat[u]


0
1
2
3
4
5