1. **Searching in Graph :**
   How to search or traverse in a Graph

2. **Recap Queue :**
   push(), pop() : Time = O(1), Space = O(n)

3. **BFS Simulation - 1** [Tree] **:** Breadth First Seach / Level Order Traversal
   <u>STEP :</u>
   a. Select a Node
   b. Explore a Node { Visit All Adjacent Nodes }

~~2~~	4	~~5~~	~~6~~	~~9~~

Fig : Queue

Output : 2, 3, 4, 5, 9

4. **BFS Simulation part - 2 [**Cyclic**] :**
   STEP : Same as part 1 + **We Need Extra Visited Array**

5. **BFS Pseudocode :**

```
a. Input—> Graph and source
b. Output—> Traverse all node and print
c. BFS(graph, source)
d. Initiate Visited Array and an empty Queue
e. Mark visited[source] = 1 and q.push(source)
f. while(!Queue.empty()) :
     i. Head = queue.front()
    ii. q.pop()
   iii. Print the head
    iv. For all adjacent node of head
          1. If visited[adj_node] == 0: continue
          2. If visited[adj_node] == 1: q.push(adj_node)
```

6. **Complexity Analysis :**
   Time Complexity : O (V) + O (2E) == O (V + E), Worst Case : O (V^2)
   Space Complexity : O (V)