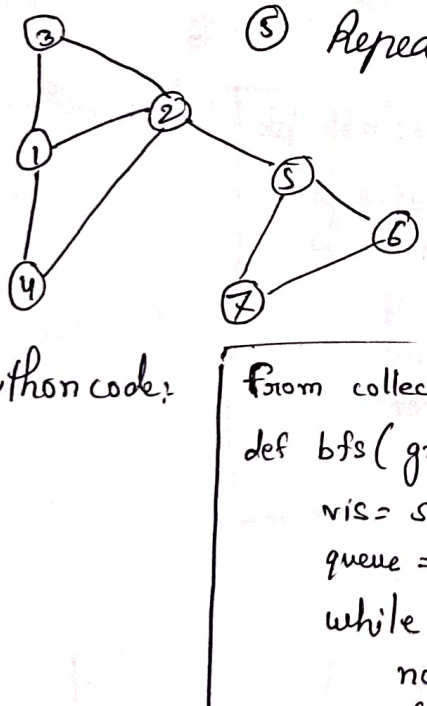
Breadth First Search Algorithm (BFS)

BFS explores all the nodes at the present depth before moving on to the nodes at the next depth level. It uses a queue data structure to accomplish this

Algorithm

1. Start from an arbitrary node
2. Push the starting node to the queue.
3. Deque a node from the and explore its neighbours.
4. If the neighbour hasn't been visited mark it as visited and enqueue it.
5. Repeat 3 & 4 until queue is empty



Possible BFS from (1)

1 → 3 → 2 → 4 → 5 → 6 → 7

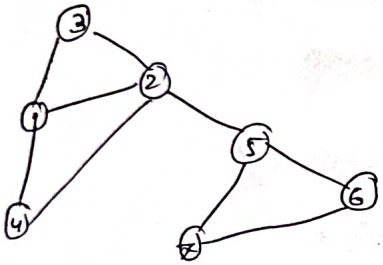
Python code:

```
from collections import deque
def bfs(graph, S):
    vis = set()
    queue = deque([S])
    while queue:
        node = queue.popleft()
        if node not in vis:
            vis.add(node)
            print(node)
            queue.extend(nbr for nbr in graph[node] if nbr not in vis)
```

Depth-First Search (DFS) Algorithm

DFS explores a branch as deep as possible before backtracking.
It uses a stack or recursion for implementation.

Algorithm: ① Start from an arbitrary node.
② Mark it as visited
③ Explore it until unvisited node is found.
④ Repeat 3 until all nodes are visited.



Possible DFS
1 → 4 → 2 → 5 → 6 → 7 → 3

Python code:

```
def dfs(graph, S, vis=set()):
    if S not in vis:
        print(S)
        vis.add(S)
        for nbr in graph[S]:
            dfs(graph, nbr, vis)
```

1 → 4 → 2 → 5 → 6 → 7 → 3
by Stack