Program:

def dfs(graph, start, visited=None):

if visited is None:

visited = set()

visited.add(start)

print(start, end=' ')

for neighbor in graph[start]:

if neighbor not in visited:

dfs(graph, neighbor, visited)

def bfs(graph, start):

visited = set([start])

queue = [start] # Use list as queue

while queue:

vertex = queue.pop(0) # pop from front (inefficient but works)

print(vertex, end=' ')

for neighbor in graph[vertex]:

if neighbor not in visited:

visited.add(neighbor)

queue.append(neighbor)

# Example graph as adjacency list

graph = {

0: [1, 4],

1: [0, 2, 3, 4],

2: [1, 3],

3: [1, 2, 4],

4: [0, 1, 3]

}

print("DFS traversal starting at node 0:")

dfs(graph, 0)

print("\nBFS traversal starting at node 0:")

bfs(graph, 0)

output:

DFS traversal starting at node 0:

0 1 2 3 4

BFS traversal starting at node 0:

0 1 4 2 3