EXPERIMENT NO - 4:
IMPLEMENTATION AND ANALYSIS OF DFS AND BFS FOR AN APPLICATION

Nikith Kumar Seemakurthi
RA1911003020480

AIM:

To implement and analyze the DFS and BFS for an application.

ALGORITHM:

DFS:

1. Mark the current node as visited and print the node.
2. Traverse all the adjacent and unmarked nodes and call the recursive function with the index of the adjacent node.

BFS:

3. Start by putting any one of the graph's vertices at the back of the queue.
4. Now take the front item of the queue and add it to the visited list.
5. Create a list of that vertex's adjacent nodes. Add those which are not within the visited list to the rear of the queue.
6. Keep continuing steps four and five till the queue is empty.

SOURCE CODE:

Source Code for DFS:

```
# DFS algorithm in Python

# DFS algorithm

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

  if visited is None:

    visited = set()

  visited.add(start)


  print(start)
```

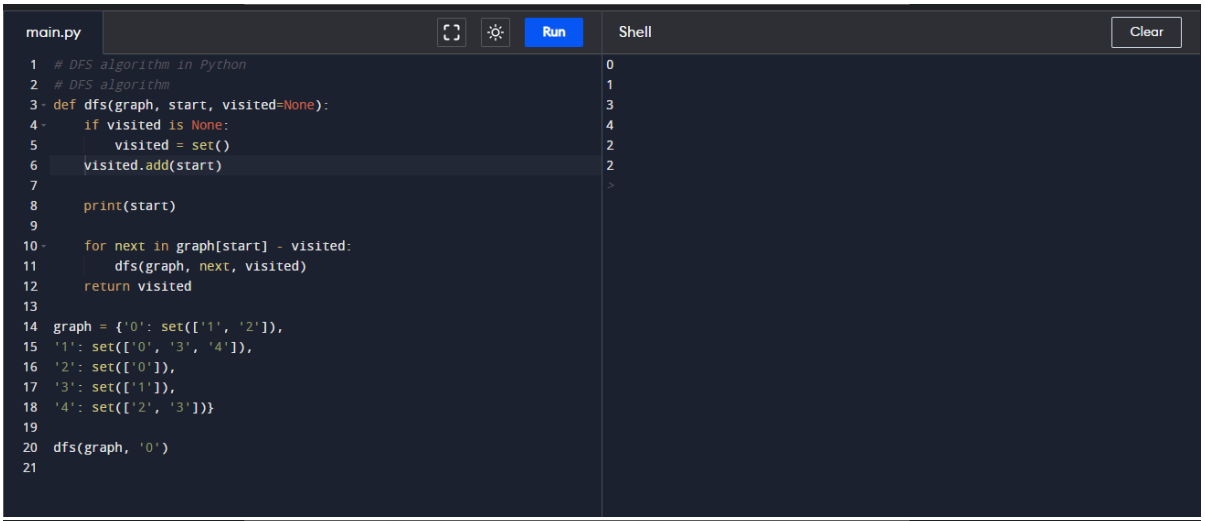
```
    for next in graph[start] - visited:
        dfs(graph, next, visited)
    return visited

graph = {'0': set(['1', '2']),
             '1': set(['0', '3', '4']),
             '2': set(['0']),
             '3': set(['1']),
             '4': set(['2', '3'])}
dfs(graph, '0')
```

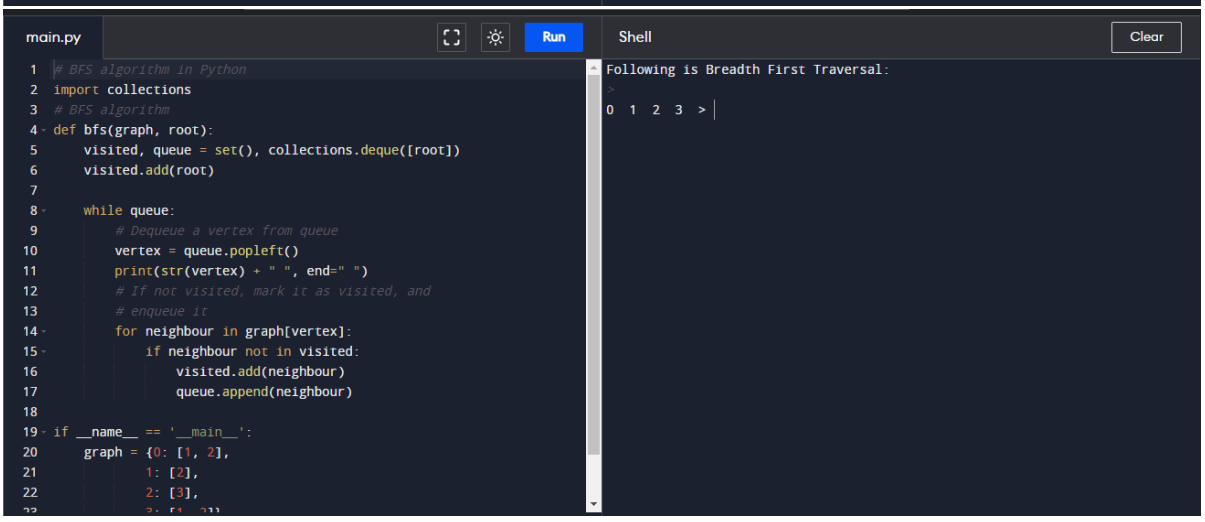
Source Code for BFS:

```
# BFS algorithm in Python
import collections
# BFS algorithm
def bfs(graph, root):
    visited, queue = set(), collections.deque([root])
    visited.add(root)
    while queue:
        # Dequeue a vertex from queue
        vertex = queue.popleft()
        print(str(vertex) + " ", end=" ")
        # If not visited, mark it as visited, and
        # enqueue it
        for neighbour in graph[vertex]:
            if neighbour not in visited:
                visited.add(neighbour)
                queue.append(neighbour)
```

```
if __name__ == '__main__':
    graph = {0: [1, 2],
             1: [2],
             2: [3],
             3: [1, 2]}
    print("Following is Breadth First Traversal: ")
    bfs(graph, 0)
```

OUTPUT:





RESULT:

Hence DFS and BFS for an application is implemented and analyzed.