**Experiment: 1.3**

**Student Name:** SANJIV GUPTA **UID:** 21BCS-*3478*

**Branch:** CSE **Section/Group:** 21BCS-IOT-602B

**Semester:** 5th **Date:** 31/08/23

**Subject Name**: AIML Lab **Subject Code:** 21CSH-316

1. **AIM:** *Implement the BFS algorithm and analyze its performance and characteristics*
2. **Objective:** *The objective of this experiment is to implement the Breadth-First Search (BFS) algorithm and analyze its performance and characteristics.*
3. **Tools/Resource Used:**

*1. Python programming language.*

*2. VS Code.*

1. **Algorithm:**

* *Create a queue data structure to store the vertices to be visited.*
* *Mark the source vertex as visited and enqueue it.*
* *While the queue is not empty, do the following:*
* *Dequeue a vertex from the queu. Process the dequeued vertex (e.g., print it or perform any required operations).*
* *Enqueue all the adjacent vertices of the dequeued vertex that are not visited and mark them as visited.*
* *Repeat steps 3 until the queue becomes empty.*

1. **Program Code:**

*from collections import deque*

*def bfs(graph, source):*

*visited = set()*

*queue = deque([source])*

*visited.add(source)*

*while queue:*

*vertex = queue.popleft()*

*print(vertex)*

*for neighbor in graph[vertex]:*

*if neighbor not in visited:*

*queue.append(neighbor)*

*visited.add(neighbor)*

*graph = {*

*'A': ['B', 'C'],*

*'B': ['A', 'D', 'E'],*

*'C': ['A', 'F'],*

*'D': ['B'],*

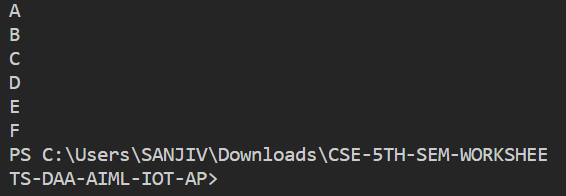
*'E': ['B', 'F'],*

*'F': ['C', 'E']*

*}*

*bfs(graph, 'A')*

1. **Output/Result:**

****

1. **Learning Outcomes:**
2. *Implement a (BFS) algorithm on a graph data structure.*
3. *Understand the concept of graph traversal and its importance in various applications.*
4. *Use recursion effectively to navigate through graph nodes and explore their connections.*