## Experiment: 3

Print all nodes reachable from a starting node using either a directed or undirected graph

**Student Name:** Nitish Rai

**UID:** 23MCA20326

**Branch:** Computers Application

**Section/Group:** 4(A)

**Semester:** 2nd

**Date of Performance:** 12/02/2024

**Subject Name:** Design and Analysis of Algorithms

**Subject Code:** 23CAH-511

## A. The task is to:

Implement an algorithm to print all nodes reachable from a given starting node in a graph (directed or undirected). Use either Depth-First Search (DFS) or Breadth-First Search (BFS) to traverse the graph, keeping track of visited nodes to avoid cycles.

## B. Steps of Experiment:

- Understand the problem and divide it into parts.
- Install JDK: If Java isn't installed on your system, download and install it.
- Open IDE( Integrated development environment) like VS code.
- Start by creating a new Java file with a .java extension (e.g., program.java).
- Write your code in a structural manner taking care of indentation to maintain readability.
- Execute your code

**C. Practical Code:**

```
import java.util.LinkedList;
import java.util.Queue;

public class ReachableNodes {

  public static void main(String[] args) {
    int[][] graph = {
      { 0, 1, 1, 0 },
      { 0, 0, 1, 1 },
      { 0, 0, 0, 1 },
      { 0, 0, 0, 0 },
    };

    int startNode = 0; // Starting node

    System.out.println("Reachable nodes from node " + startNode + ":");
    printReachableNodes(graph, startNode);
  }

  public static void printReachableNodes(int[][] graph, int startNode) {
    boolean[] visited = new boolean[graph.length];
    Queue<Integer> queue = new LinkedList<>();

    queue.add(startNode);
    visited[startNode] = true;

    while (!queue.isEmpty()) {
      int currentNode = queue.poll();
      System.out.print(currentNode + " ");

      for (int neighbor = 0; neighbor < graph.length; neighbor++) {

        if (!visited[neighbor] && graph[currentNode][neighbor] == 1) {
          queue.add(neighbor);
          visited[neighbor] = true;
        }
      }
    }
  }
}
```

## D. Output:

```
PS C:\Users\Nitish\Desktop\JAVA\javaComeBack> cd "c:\Users\Nitish\Desktop\JAVA\javaComeBack\" ; if ($?) { javac ReachableNodes.java } ; if ($?) { java ReachableNodes }
Reachable nodes from node 0:
0 1 2 3
PS C:\Users\Nitish\Desktop\JAVA\javaComeBack>
```

## E. Time Complexity:

O(V+E)

## Learning outcomes:

- Understand the concepts of functions, classes and BFS.
- Understand how to use DFS.
- Understand the concept of the creation of a graph.
- Understanding how to deal with errors..