Day 14

Assignment 6: Depth-First Search (DFS) Recursive

Write a recursive DFS function for a given undirected graph. The function should visit every node and print it out.

A)

Introduction:

Depth-First Search (DFS) is a fundamental graph traversal algorithm used to visit all the nodes in a graph. It starts at a chosen node and explores as far as possible along each branch before backtracking. This traversal method is often implemented using both iterative and recursive approaches. In this example, we'll focus on the recursive implementation of DFS in Java.

Java code for Dfs:-

```
import java.util.*;

public class Graph {
   private int vertices; // Number of vertices
   private LinkedList<Integer>[] adjacencyList; // Adjacency List

   // Constructor
   public Graph(int vertices) {
      this.vertices = vertices;
      adjacencyList = new LinkedList[vertices];
      for (int i = 0; i < vertices; i++) {
         adjacencyList[i] = new LinkedList<>();
      }
   }

   // Add edge to the graph
   public void addEdge(int source, int destination) {
      adjacencyList[source].add(destination);
      adjacencyList[destination].add(source); // Since the graph is undirected
   }

   // DFS helper function
   private void DFSUtil(int vertex, boolean[] visited) {
      // Mark the current node as visited and print it
      visited[vertex] = true;
      System.out.print(vertex + " ");

      // Recur for all the vertices adjacent to this vertex
      for (int adj : adjacencyList[vertex]) {
         if (!visited[adj]) {
```

```
            DFSUtil(adj, visited);
        }
    }
}

// DFS function
public void DFS(int startVertex) {
    // Mark all the vertices as not visited (set as false by default)
    boolean[] visited = new boolean[vertices];

    // Call the recursive helper function to print DFS traversal
    DFSUtil(startVertex, visited);
}

public static void main(String[] args) {
    Graph graph = new Graph(4);

    graph.addEdge(0, 1);
    graph.addEdge(0, 2);
    graph.addEdge(1, 2);
    graph.addEdge(2, 3);

    System.out.println("Depth-First Search (starting from vertex 0):");
    graph.DFS(0);
}
}
```

Explanation:

1. Graph Class:

- Constructor: Initializes the graph with the given number of vertices and creates an adjacency list for each vertex.

- addEdge Method: Adds an edge between two vertices. Since the graph is undirected, edges are added in both directions.

2. DFSUtil Method:

- This is a helper method that performs the recursive DFS traversal.

- It marks the current node as visited, prints it, and recursively visits all unvisited adjacent vertices.

3. DFS Method:

- Initializes a visited array to keep track of visited vertices.

- Calls the DFSUtil method starting from the given vertex.

4. main Method:

- Creates a sample graph, adds edges, and initiates the DFS traversal starting from vertex 0.

When you run the above code, it will perform a DFS traversal of the graph and print the nodes in the order they are visited.

Output:

Depth-First Search (starting from vertex 0):

0 1 2 3