Name: Vijay Misal
Div: C Batch: C3
Roll No: 233073
PRN No: 22320079

Practical No: 4

Title: Create a graph using adjacency list representation. Perform graph traversal using BFS and DFS.

Code:

```java
import java.util.*;

class Graph {
    private int vertices;
    private LinkedList<Integer>[] adjacencyList;

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

    public void addEdge(int source, int destination) {
        adjacencyList[source].add(destination);
    }

    public void bfsTraversal(int startVertex) {
        boolean[] visited = new boolean[vertices];
        Queue<Integer> queue = new LinkedList<>();

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

        while (!queue.isEmpty()) {
            int currentVertex = queue.poll();
            System.out.print(currentVertex + " ");
            for (int neighbor : adjacencyList[currentVertex]) {
                if (!visited[neighbor]) {
                    visited[neighbor] = true;
                    queue.add(neighbor);
                }
            }
        }
    }

    public void dfsTraversal(int startVertex) {
        boolean[] visited = new boolean[vertices];
        dfsUtil(startVertex, visited);
    }

    private void dfsUtil(int currentVertex, boolean[] visited) {
        visited[currentVertex] = true;
        System.out.print(currentVertex + " ");

        for (int neighbor : adjacencyList[currentVertex]) {
            if (!visited[neighbor]) {
                dfsUtil(neighbor, visited);
            }
        }
    }
}

public class Practical4 {
    public static void main(String[] args) {
        int vertices = 6;
        Graph graph = new Graph(vertices);

        graph.addEdge(0, 1);
        graph.addEdge(0, 2);
        graph.addEdge(1, 3);
        graph.addEdge(1, 4);
        graph.addEdge(2, 4);
        graph.addEdge(3, 5);
        graph.addEdge(4, 5);

        System.out.println("BFS Traversal:");
        graph.bfsTraversal(0);
        System.out.println("\nDFS Traversal:");
        graph.dfsTraversal(0);
    }
}
```

Output:

```
BFS Traversal:

0 1 2 3 4 5

DFS Traversal:

0 1 3 5 4 2
```