```
import java.util.*;

public class assign1 {
    int vertices;
    int adjMatrix[][];

    public assign1(int vertices) {
        this.vertices = vertices;
        adjMatrix = new int[vertices][vertices];
    }

    public void addEdge(int u, int v) {
        adjMatrix[u][v] = 1;
        adjMatrix[v][u] = 1;
    }

    public void printMatrix() {
        for (int i = 0; i < vertices; i++) {
            for (int j = 0; j < vertices; j++) {
                System.out.print(adjMatrix[i][j] + " ");
            }
            System.out.println();
        }
    }

    public void DFS(int start) {
        boolean visited[] = new boolean[vertices];
        DFSRecursive(start, visited);
    }

    private void DFSRecursive(int v, boolean[] visited) {
        visited[v] = true;
        System.out.print(v + " ");
        for (int i = 0; i < vertices; i++) {
            if (adjMatrix[v][i] == 1 && !visited[i]) {
                DFSRecursive(i, visited);
            }
        }
    }

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

```
        queue.add(start);
        visited[start] = true;

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

            for (int i = 0; i < vertices; i++) {
                if (adjMatrix[node][i] == 1 && !visited[i]) {
                    queue.add(i);
                    visited[i] = true;
                }
            }
        }
    }

    public static void main(String[] args) {
        assign1 obj = new assign1(5);
        obj.addEdge(0, 1);
        obj.addEdge(0, 2);
        obj.addEdge(1, 3);
        obj.addEdge(1, 4);

        System.out.println("Adjacency Matrix:");
        obj.printMatrix();

        System.out.println("DFS Traversal:");
        obj.DFS(0);

        System.out.println("\nBFS Traversal:");
        obj.BFS(0);
    }
}
```