```
import java.util.*;
class Graphh {
   private int V; // Number of vertices
   private LinkedList<Integer>[] adj; // Adjacency List
// Constructor
   Graphh(int v) {
      V = v;
      adj = new LinkedList[v];
      for (int i = 0; i < v; ++i) {
         adj[i] = new LinkedList();
      }
   }
// Add an edge to the graph
   void addEdge(int v, int w) {
      adj[v].add(w);
   }
// Depth First Search
   void DFS(int v) {
   // Mark all the vertices as not visited
      boolean[] visited = new boolean[V];
   // Call the recursive helper function to print DFS traversal
      DFSUtil(v, visited);
   }
   void DFSUtil(int v, boolean[] visited) {
   // Mark the current node as visited and print it
      visited[v] = true;
      System.out.print(v + " ");
   // Recur for all the vertices adjacent to this vertex
      Iterator<Integer> i = adj[v].listIterator();
      while (i.hasNext()) {
         int n = i.next();
         if (!visited[n]) {
            DFSUtil(n, visited);
         }
```

```
      }

   }}
public class DFSExample {

   public static void main(String[] args) {

      Graphh g = new Graphh(4);

      g.addEdge(0, 1);

      g.addEdge(0, 2);

      g.addEdge(1, 2);

      g.addEdge(2, 0);

      g.addEdge(2, 3);

      g.addEdge(3, 3);

      System.out.println("Depth First Traversal (starting from vertex 2): ");

      g.DFS(2);

   }

}
```