Task 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.

1. **JAVA CODE:**

```
import java.util.*;
public class DFSUndirectedGraph {
   private int numVertices; // Number of vertices in the graph
   private LinkedList<Integer> adjList[]; // Adjacency list

  public DFSUndirectedGraph(int vertices) {

      numVertices = vertices;
      adjList = new LinkedList[vertices];
      for (int i = 0; i < vertices; ++i)
         adjList[i] = new LinkedList<>();
   }
   // Function to add an edge into the graph
   void addEdge(int v, int w) {
      adjList[v].add(w); // Add w to v's list.
      adjList[w].add(v); // Add v to w's list since it's undirected
   }
   void DFSUtil(int v, boolean visited[]) //recursive function
    {
      visited[v] = true;
      System.out.print(v + " ");
      // Recur for all the vertices adjacent to this vertex
      for (int n : adjList[v]) {
         if (!visited[n]) {
            DFSUtil(n, visited);          }       }   }
   void DFS(int v) {
      boolean visited[] = new boolean[numVertices];
      DFSUtil(v, visited);   }
   public static void main(String args[]) {
      DFSUndirectedGraph g = new DFSUndirectedGraph(6);
      g.addEdge(0, 1);
      g.addEdge(0, 2);
      g.addEdge(1, 3);
      g.addEdge(2, 4);
      g.addEdge(3, 4);
      g.addEdge(3, 5);
      g.addEdge(4, 5);
      System.out.println("Following is Depth First Traversal " +
                         "(starting from vertex 0):");
      g.DFS(0);   }   }
```

Explanation:

1. **Graph Initialization:**
   - numVertices holds the number of vertices.
   - adjList is an array of linked lists where each list represents the adjacency list of a vertex.
   - The constructor initializes the adjacency list for each vertex.
2. **Adding Edges:**
   - The addEdge method adds an edge between vertex v and vertex w.
   - Since the graph is undirected, the edge is added in both directions.
3. **DFS Utility Method:**
   - DFSUtil is a recursive method that performs the DFS.
   - It marks the current node v as visited and prints it.
   - It then recursively visits all the adjacent vertices that haven't been visited.
4. **DFS Method:**
   - The DFS method initializes the visited array and calls the recursive DFSUtil method starting from vertex v.
5. **Main Method:**
   - Creates a graph and adds some edges to represent an undirected graph.
   - Calls the DFS method starting from vertex 0 and prints the traversal order.