****

**Green University of Bangladesh**

**Department of Computer Science and Engineering (CSE)**

**Faculty of Sciences and Engineering**

**Semester: (Fall, Year:2023), B.Sc. in CSE (Day)**

**Lab Report NO: 03**

**Course Title : Algorithms Lab**

**Course Code : CSE 204**

**Section : 221 D9**

**Lab Experiment Name: Find the number of distinct minimum spanning trees**

**for a given weighted graph using prim's algorithms.**

**Student Details**

|  |  |  |
| --- | --- | --- |
| **Name** | | **ID** |
| **1.** | Jahidul Islam | 221002504 |

**Lab Date : 29-10-2023**

**Submission Date : 28-11-2023**

**Course Teacher’s Name : Md. Abu Rumman Refat**

|  |
| --- |
| **Lab Report Status**  **Marks: ………………………………… Signature:.....................**  **Comments:.............................................. Date:..............................** |

**1. TITLE OF THE EXPERIMENT:**

Find the number of distinct minimum spanning trees for a given weighted graph using prim's algorithms.

**2. INTRODUCTION:**

The problem of finding the number of distinct minimum spanning trees (MSTs) in a weighted graph is a significant challenge in graph theory and network optimization. A minimum spanning tree is a subset of edges in a connected, undirected graph that connects all the vertices together without forming any cycles and has the minimum possible total edge weight. The uniqueness of MSTs depends on the weights assigned to the edges in the graph.

**3. PROCEDURE:**

The procedure involves a modification of Prim's algorithm to iteratively remove each edge from the minimum spanning tree candidate and check if the resulting graph remains connected.

**3.1. Initialization:**

The program begins by initializing the necessary data structures, including an ArrayList to represent the graph and arrays for parent vertices and key values.

Edges are added to the graph along with their corresponding weights.

**3.2. Modified Prim's Algorithm:**

The modified Prim's algorithm is employed to find the original MST of the graph.

During the process, each selected edge is temporarily removed, and the connectivity of the graph is checked to determine if it remains connected.

**3.3. Counting Distinct MSTs:**

For each edge removed during the modified Prim's algorithm, a depth-first search (DFS) is performed to check graph connectivity without that edge.

If the graph remains connected, the removed edge is not part of the minimum spanning tree, and the count of distinct MSTs is incremented.

**3.4. Graph Restoration:**

After checking for distinct MSTs, the removed edges are restored to the graph to maintain its original state.

**4. IMPLEMENTATION:**

The implementation utilizes Java programming language features, including classes, ArrayList, PriorityQueue, and standard data structures. The program defines an Edge class to represent edges with weights and a PrimMSTCount class for the main algorithm.

* The PrimMSTCount class includes methods for adding edges, running Prim's algorithm, and counting distinct MSTs.
* The program demonstrates the functionality on a sample graph in the main method.

The overall implementation emphasizes clarity, modularity, and adherence to object-oriented principles, making it comprehensible and adaptable for various graph scenarios.

**6. Code in Java**



A screen shot of a computer code

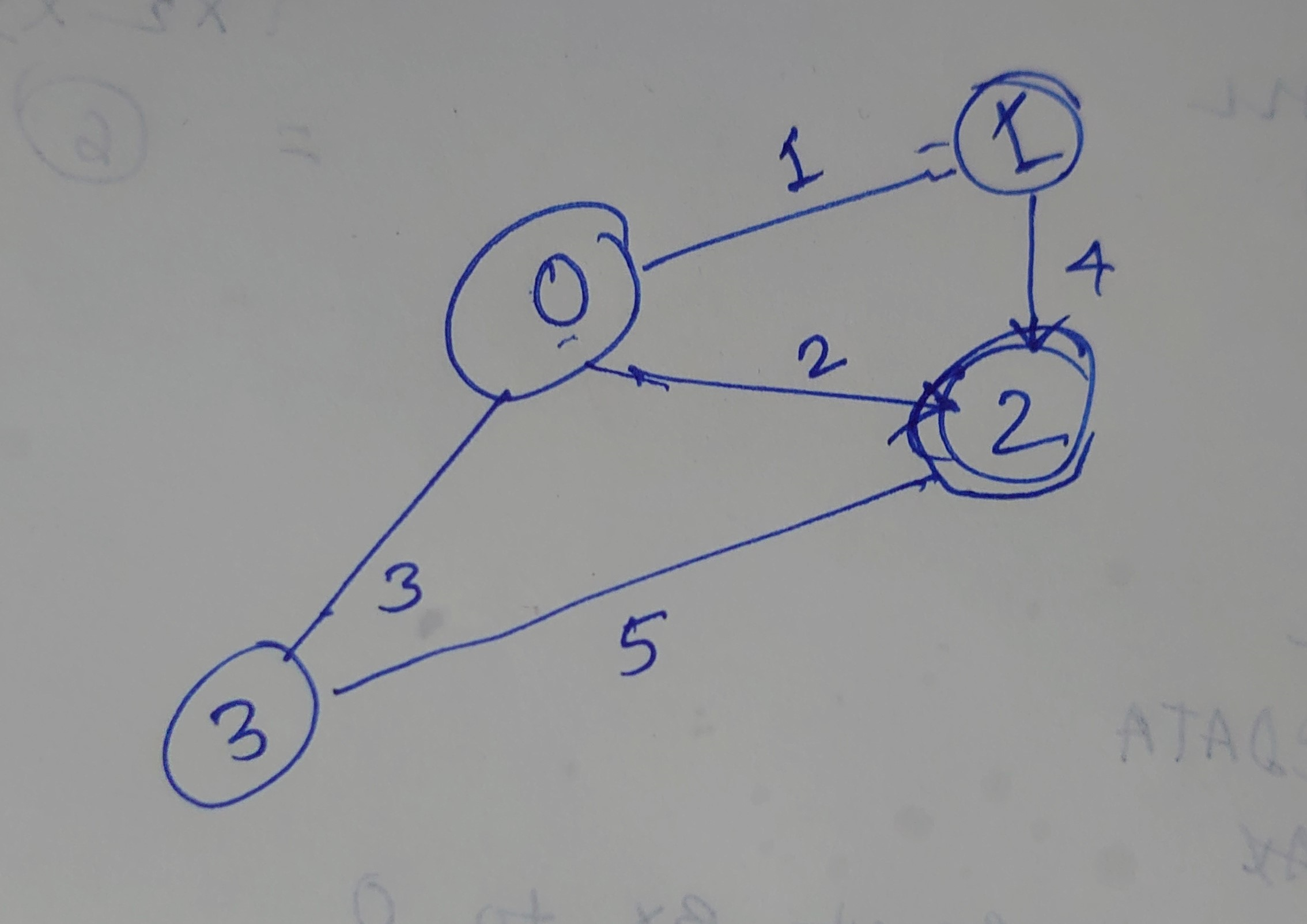
Description automatically generated

A screen shot of a computer

Description automatically generated

**7. INPUT GRAPH:**

* Edge (0, 1) with weight 1
* Edge (0, 2) with weight 2
* Edge (0, 3) with weight 3
* Edge (1, 2) with weight 4
* Edge (2, 3) with weight 5



**8. OUTPUT:**

Number of distinct minimum spanning trees: 2

**9. DISCUSSION:**

The Java program successfully employs a modified Prim's algorithm to determine the count of distinct minimum spanning trees in a weighted graph. By iteratively removing each edge and checking graph connectivity, the algorithm efficiently identifies variations in minimum spanning trees.

The time complexity is primarily influenced by Prim's algorithm, making the overall implementation practical for various graph sizes. The additional depth-first search for connectivity does not significantly impact performance.

**10. CONCLUSION:**

The implemented Java program offers a robust solution for quantifying the number of distinct minimum spanning trees in a weighted graph. Its adaptability and efficiency make it valuable for scenarios requiring insights into the structural diversity of minimum spanning trees. Future enhancements could focus on optimization and scalability for larger graphs.