**SVKM’s NMIMS**

**Mukesh Patel School of Technology Management & Engineering**

**Computer Engineering Department**

Program: B.Tech. Sem V

**Course: Design and Analysis of Algorithms**

**List of Experiments**

w.e.f. 1st Jul 2020

**Faculty:** Abhay Kolhe.

LAB Manual

PART A

(PART A : TO BE REFFERED BY STUDENTS)

**Experiment No.05**

**A.1 Aim:**

Implementation of Greedy Technique Algorithm Design.

Write a program to implement Prim’s Algorithm for finding Minimum Spanning Tree (MST).

**A.2 Prerequisite:**

1. Concepts of Greedy Technique of algorithm design.

2. Knowledge of Graph Handling.

3. Knowledge of Minimum Spanning Tree.

**A.3 Outcome:**

**After successful completion of this experiment students will be able to**

1. Design & implement a solution using Greedy Technique.
2. Identify different problems that can be solved by using Greedy Technique.
3. Identify applications of Minimum Spanning Tree.

**A.4 Theory:**

**A.4.1.**

**General Concept of Greedy Technique of Algorithm Design**

* An optimization problem is one in which you want to find, not just *a* solution, but the *best* solution
* A “greedy algorithm” sometimes works well for optimization problems
* A greedy algorithm works in phases. At each phase:
  + You take the best you can get right now, without regard for future consequences
  + You hope that by choosing a *local* optimum at each step, you will end up at a *global* optimum

**A Simple Example**

* **Problem:** Pick k numbers out of n numbers such that the sum of these k numbers is the **largest.**
* **Algorithm:**

FOR i = 1 to k

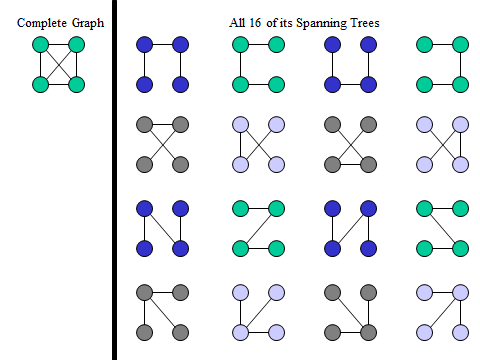
pick out the largest number and

delete this number from the input.

ENDFOR

**A spanning tree** of a graph is just a subgraph that contains all the vertices and is a tree.

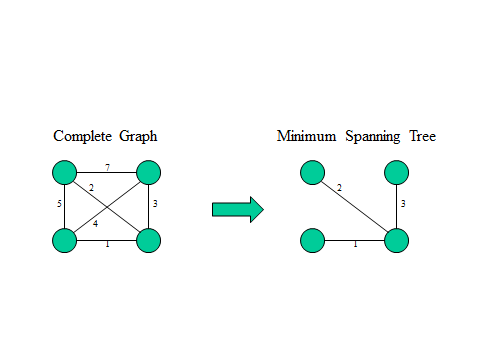
A graph may have many spanning trees**.** Different possible spanning trees for a given graph are as shown in Fig. 1.



**Figure 1: Graph & Its Spanning Trees**

**Minimum Spanning Trees**

The Minimum Spanning Tree for a given graph is the Spanning Tree of minimum cost for that graph.

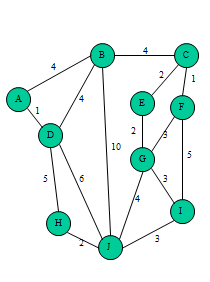
****

**Figure 2: A graph & Its spanning tree.**

**Prim's Algorithm**

This algorithm starts with one node. It then, one by one, adds a node that is unconnected to the new graph, each time selecting the node whose connecting edge has the smallest weight out of the available nodes’ connecting edges.

For the graph shown in Fig. 3, we proceed as follows.

****

**Figure 3: Complete Graph**

The starting Point (node) is A.

Then we add, edge<A, D> of weight 1.

Next we add, edge <A,B> of weight 4.

Next we add, edge <B,C> of weight 4.

Next we add, edge <C, F> of weight 1.

Next we add, edge <C, E> of weight 2.

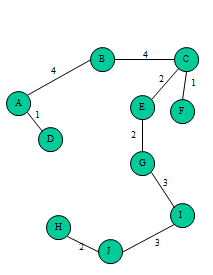
Next we add, edge <E, G> of weight 2.

Next we add, edge <G, I> of weight 3.

Next we add, edge <I, J> of weight 3.

Next we add, edge <J, H> of weight 2.

The resultant Minimum Spanning Tree is as shown in Fig. 4

****

**Figure 4: Minimum Spanning Tree**.

**A.5 Procedure/Algorithm:**

**A.5.1:**

**Prim’s Algorithm**

1. The new graph is constructed - with one node from the old graph.

2. While new graph has fewer than n nodes,

1. Find the node from the old graph with the smallest connecting edge to the new graph,

2. Add it to the new graph

Every step will have joined one node, so that at the end we will have one graph with all the nodes and it will be a minimum spanning tree of the original graph.

procedure Prim(G: weighted connected graph with n vertices)

T := a minimum-weight edge

for i = 1 to n − 2

begin

e := an edge of minimum weight incident to a vertex in T and not forming a circuit

in T if added to T

T := T with e added

end

return(T)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PART B

(PART B : TO BE COMPLETED BY STUDENTS)

|  |  |
| --- | --- |
| Roll No.B032 | Name: Naman Garg |
| Class : B tech CS B | Batch : B2 |
| Date of Experiment: 04/08/2020 | Date of Submission: 7/8/2020 |
| Grade : | Time of Submission: |
| Date of Grading: |  |

**B.1 Software Code written by student:**

# Name: Naman Garg

# Roll No: B032

# Aim: Implement Prim’s Algorithm for finding Minimum Spanning Tree

v = int(input("Enter no of vertices: "))

#adjecency matrix

adj\_matrix = []

for i in range(v):

    #building the adj matrix

    adj\_matrix.append(

        list(map(int, input(f'enter {v} for r{i+1}: ').split())))

#printing the matrix as it is

print("\n Matrix: ")

for rows in adj\_matrix:

    print(\*rows)

#init selected

selected = [0 for i in range(v)]

no\_of\_edges = 0

selected[0] = 1

total\_cost\_of\_mst = 0

print("\n")

print("Edge \t Weight")

#for verices

while(no\_of\_edges < v-1):

    a, b = 0, 0

    min = 99999

    for i in range(v):

        if(selected[i] == 1):

            for j in range(v):

                if(selected[j] == 0 and adj\_matrix[i][j] != 0): #If not Selected and there is an edge

                    if min > adj\_matrix[i][j]:

                        min = adj\_matrix[i][j]

                        a, b = i, j

    print(f"{a} -> {b} \t {adj\_matrix[a][b]}")

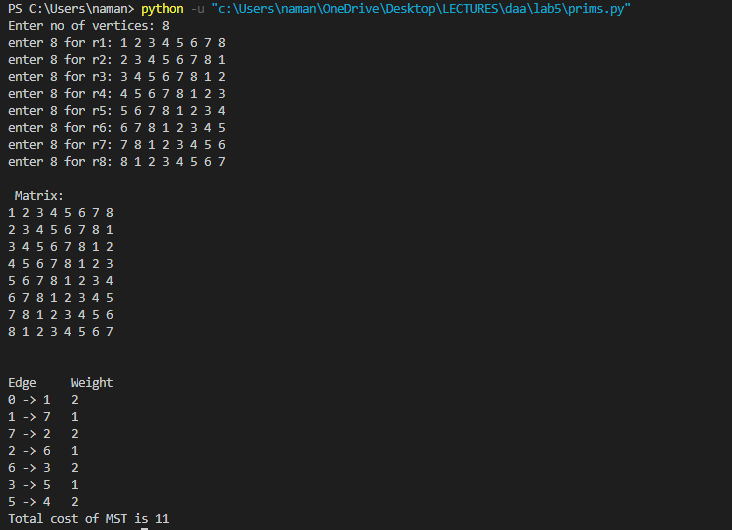
    total\_cost\_of\_mst += adj\_matrix[a][b]

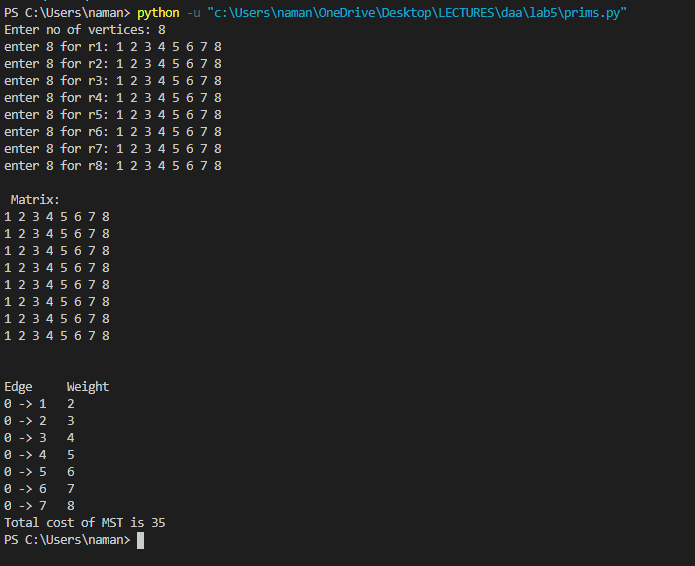
    selected[b] = 1

    no\_of\_edges += 1

print(f"Total cost of MST is {total\_cost\_of\_mst}")

**B.2 Input and Output:**

******

******

**B.3 Observations and learning:**  
greedy technique is useful for problem solving re implemented prims algorithm We breakdown the problem into steps and optimize for each step to produce a greedy approach and solve the problem thus The weight of the minimum spanning tree is minimized we can also use this approach to maximize the weight does depends upon the application

**B.4 Conclusion:**

**1** I understood and implemented the prim's algorithm for minimum spanning tree

2 I understood how to work with graphs and how adjacency matrix represents a graph

I understood the applications of minimum spanning tree

**B.5 Question of Curiosity**

Q.1 Identify & discuss in detail the real life applications Minimum spanning trees.

minimum spanning trees are very useful 2 find the minimum distance between two points these points can be on a map as is done in Google Maps . Computer networks , cluster analysis , traveling salesman, stream carving and image recognition Use such techniques or are based on such techniques and algorithms .

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*