Name	Siddiqui Hawaiza Sabeel
UID no.	2022701010
Experiment No.	6
SUBJECT	DAA

AIM:	**Greedy Method – Prim’s Algorithm**
Program 1	
PROBLEM STATEMENT:	Use the Greedy Programming method to find the minimum weight of a spanning tree.
ALGORITHM/ THEORY:	**Prim's Algorithm** is a greedy algorithm that is used to find the minimum spanning tree from a graph. Prim's algorithm finds the subset of edges that includes every vertex of the graph such that the sum of the weights of the edges can be minimized. Prim's algorithm starts with the single node and explores all the adjacent nodes with all the connecting edges at every step. The edges with the minimal weights causing no cycles in the graph got selected. **How does the prim's algorithm work?** Prim's algorithm is a greedy algorithm that starts from one vertex and continue to add the edges with the smallest weight until the goal is reached. The steps to implement the prim's algorithm are given as follows - ○ First, we have to initialize an MST with the randomly chosen vertex. ○ Now, we have to find all the edges that connect the tree in the above step with the new vertices. From the edges found, select the minimum edge and add it to the tree. ○ Repeat step 2 until the minimum spanning tree is formed. The applications of prim's algorithm are - ○ Prim's algorithm can be used in network designing. ○ It can be used to make network cycles.

- It can also be used to lay down electrical wiring cables.

Algorithm

Step 1: *Determine an arbitrary vertex as the starting vertex of the MST.*

Step 2: *Follow steps 3 to 5 till there are vertices that are not included in the MST (known as fringe vertex).*

Step 3: *Find edges connecting any tree vertex with the fringe vertices.*

Step 4: *Find the minimum among these edges.*

Step 5: *Add the chosen edge to the MST if it does not form any cycle.*

Step 6: *Return the MST and exit*

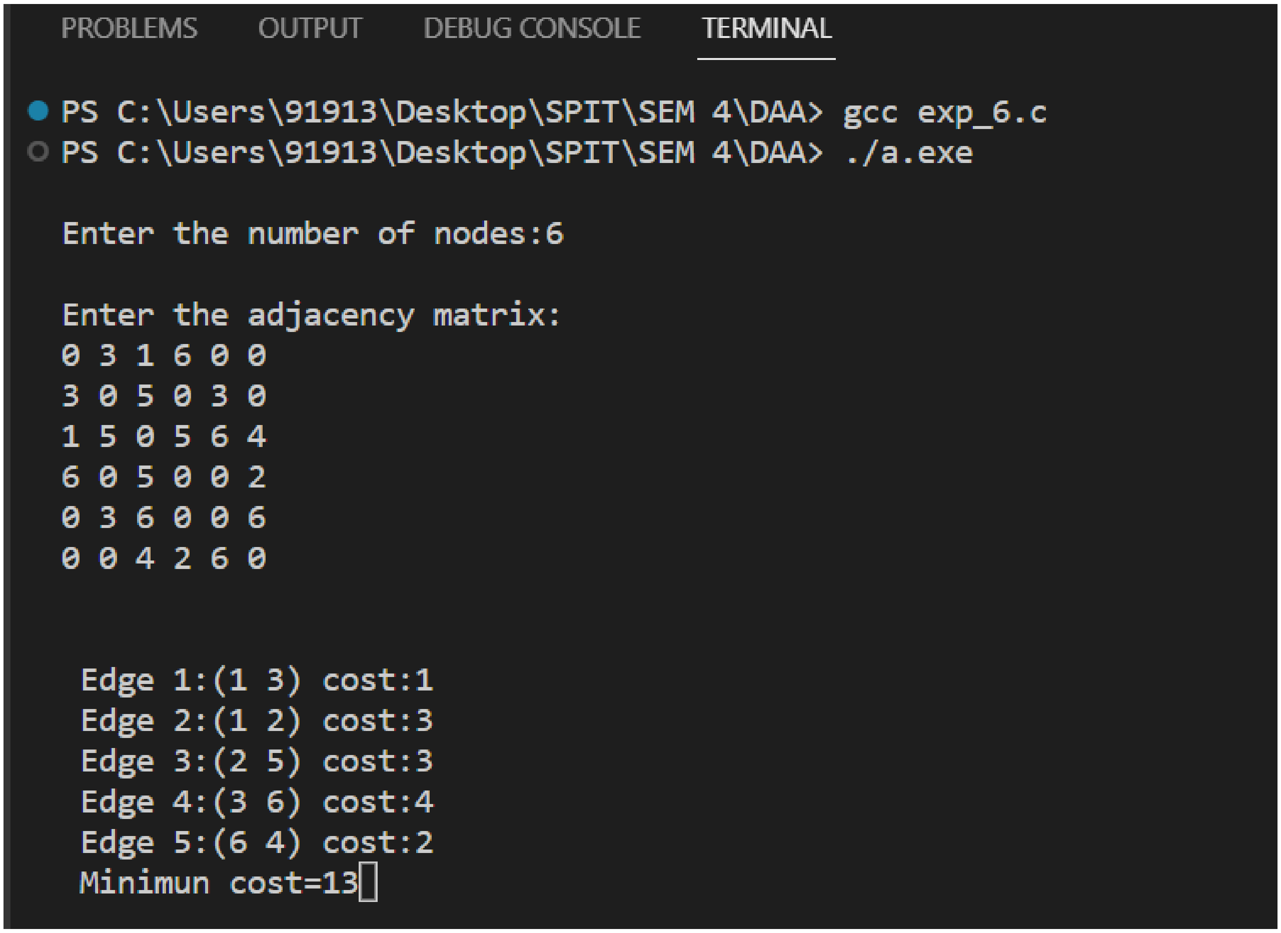
- **Time Complexity**

Data structure used for the minimum edge weight	**Time Complexity**
Adjacency matrix, linear searching	$O(\|V\|^2)$
Adjacency list and binary heap	$O(\|E\| \log \|V\|)$
Adjacency list and Fibonacci heap	$O(\|E\| + \|V\| \log \|V\|)$

PROGRAM:

```
#include <stdio.h>

#include <conio.h>

int a, b, u, v, n, i, j, ne = 1;

int visited[10] = {0}, min, mincost = 0, cost[10][10];

void main()

{
    printf("\nEnter the number of nodes:");
    scanf("%d", &n);
    printf("\nEnter the adjacency matrix:\n");
    for (i = 1; i <= n; i++)
        for (j = 1; j <= n; j++)
        {
            scanf("%d", &cost[i][j]);
            if (cost[i][j] == 0)
                cost[i][j] = 999;
        }
    visited[1] = 1;
    printf("\n");

    while (ne < n)
    {
        for (i = 1, min = 999; i <= n; i++)
            for (j = 1; j <= n; j++)
                if (cost[i][j] < min)
                    if (visited[i] != 0)
                    {
                        min = cost[i][j];
                        a = u = i;
                        b = v = j;
                    }
        if (visited[u] == 0 || visited[v] == 0)
        {
            printf("\n Edge %d:(%d %d) cost:%d", ne++, a, b, min);
            mincost += min;
            visited[b] = 1;
        }
        cost[a][b] = cost[b][a] = 999;
    }
    printf("\n Minimun cost=%d", mincost);
    getch();
}
```

RESULT:



```
● PS C:\Users\91913\Desktop\SPIT\SEM 4\DAA> gcc exp_6.c
○ PS C:\Users\91913\Desktop\SPIT\SEM 4\DAA> ./a.exe

  Enter the number of nodes:6

  Enter the adjacency matrix:
  0 3 1 6 0 0
  3 0 5 0 3 0
  1 5 0 5 6 4
  6 0 5 0 0 2
  0 3 6 0 0 6
  0 0 4 2 6 0


   Edge 1:(1 3) cost:1
   Edge 2:(1 2) cost:3
   Edge 3:(2 5) cost:3
   Edge 4:(3 6) cost:4
   Edge 5:(6 4) cost:2
   Minimun cost=13
```

CONCLUSION:	I understood the Greedy approach and implemented it in Prims Algorithm