Problem Definition 6:-

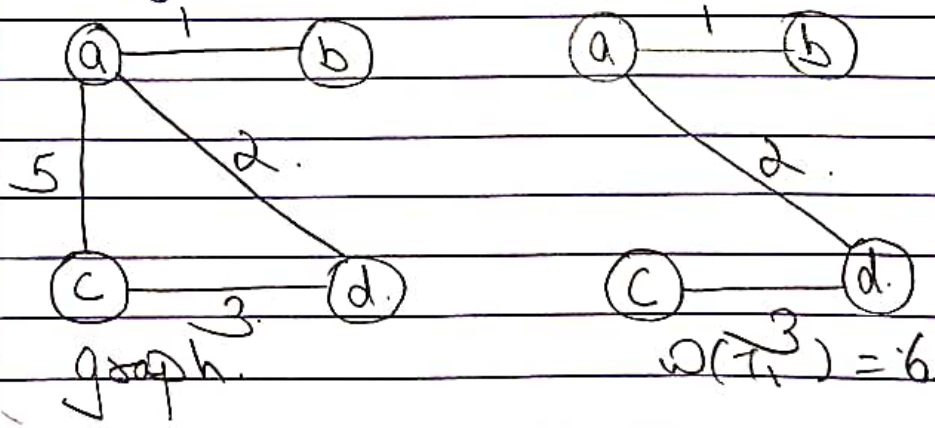
Find the Minimum cost Spanning tree of a given undirected graph using Prim's algorithm.

Objectives of the Experiment:-

1. To understand graph & its Spanning tree.
2. To understand & appreciate the greedy strategy.
3. Present the working of Prim's Algorithm.
4. Learn to write algorithm in standard form.
5. Analyze the algorithm for its time Complexity.

Theory:-

A Spanning tree of a connected graph is its connected acyclic subgraph that contains all the vertices of the graph. A minimum Spanning tree of a weighted connected graph is its Spanning tree of the smallest weight, where the weight of a tree is defined as the sum of the weights on all its edges. The minimum Spanning tree problem is the problem of finding a minimum Spanning tree for a given weighted connected graph.



→ T1 is the minimum Spanning tree.

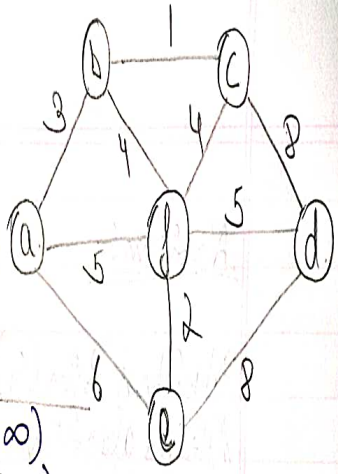
→ Graph & its Spanning tree

Algorithm :-

Algorithm Prim (G)
// Prim's algorithm for constructing a minimum spanning tree.
// Input: A weighted connected graph $G = V, E$
// output: E_T, the set of edges composing a minimum spanning tree of G.

$V_T \leftarrow \{v_0\}$ // the set of tree vertices can be initialized with any vertex
$E_T \leftarrow \phi$
for $i \leftarrow 1$ to $|V| - 1$ do
find a minimum-weight edge $e^* = (v^*, u^*)$ among all the edges (v, u) such that v is in V_T and u is in $V - V_T$
$V_T \leftarrow V_T \cup \{u^*\}$
$E_T \leftarrow E_T \cup \{e^*\}$
return E_T.

Tracing :-



a(-,-) b(a,3) c(-,∞) d(-,∞) e(a,6) f(a,5).

b(a,3) c(b,1) d(-,∞) e(a,6) f(b,4).

c(b,1) d(c,6) e(a,6) f(b,4).

f(b,4) d(f,5) e(f,2)

e(f,2) d(f,5)

d(f,5).

Code:-

```
#include<stdio.h>
#include<stdlib.h>
int a, b, u, v, n, i, j, ne=1;
int visited[10]={0}, min, mincost=0, cost[10][10];
void main()
{
    printf("Enter the number of nodes:");
    scanf("%d", &n);
    printf("\n Enter the adjacency matrix:");
    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);
        minlast += min;
        visited[b]=1;
    }
    cost[a][b] = cost[b][a] = 999;
}
printf("\n Minimum cost = %d\n", minlast);
}
```

Name:Sanket Patil

Usn:2GI20CS132

Date:14/07/2022

Division:C

Term Work:6

Prim's Algorithm

Output:

```
Enter the number of vertices:6
Enter the adjacency matrix:
0 3 999 999 6 5
3 0 1 999 999 4
999 1 0 6 999 4
999 999 6 0 8 5
6 999 999 8 0 2
5 4 4 5 2 0

Edge 1:(1 2) cost:3
Edge 2:(2 3) cost:1
Edge 3:(2 6) cost:4
Edge 4:(6 5) cost:2
Edge 5:(6 4) cost:5
Minimum cost=15

Process returned 0 (0x0)   execution time : 12.227 s
Press any key to continue.
```

Reference:-

* Kenneth Berman, Jerome Paul, Algorithm, langeage language.
* Thomas H Coormen, Charles E. Lierson, Ronal L. Rivest, Clifford Stein, introduction to algorithm PHI 2nd Edition & onwards.

Conclusion:-

In this teamwork we learnt about Prim's algorithm technique & implementation of Prim Algorithm.
We also learned computing time required for recursive & iterative algorithm.