|  |
| --- |
| **Subject: DAA Class: S.E.(Comp)**    **Practical No.: 7 Date:** |

**AIM: Write program to find out a minimum spanning tree of a simple connected undirected graph by applying Prim’s algorithm.**

**Title:** Program to find out a minimum spanning tree of a simple connected undirected graph by applying Prim’s algorithm.

**Theory:**

Prim's Algorithm 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.

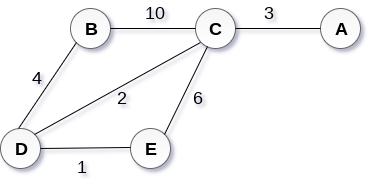
Prim's algorithm starts with the single node and explore all the adjacent nodes with all the connecting edges at every step. The edges with the minimal weights causing no cycles in the graph got selected.

## Algorithm

* **Step 1:** Select a starting vertex
* **Step 2:** Repeat Steps 3 and 4 until there are fringe vertices
* **Step 3:** Select an edge e connecting the tree vertex and fringe vertex that has minimum weight
* **Step 4:** Add the selected edge and the vertex to the minimum spanning tree T  
  [END OF LOOP]
* **Step 5:** EXIT

### Example :

Construct a minimum spanning tree of the graph given in the following figure by using prim's algorithm.



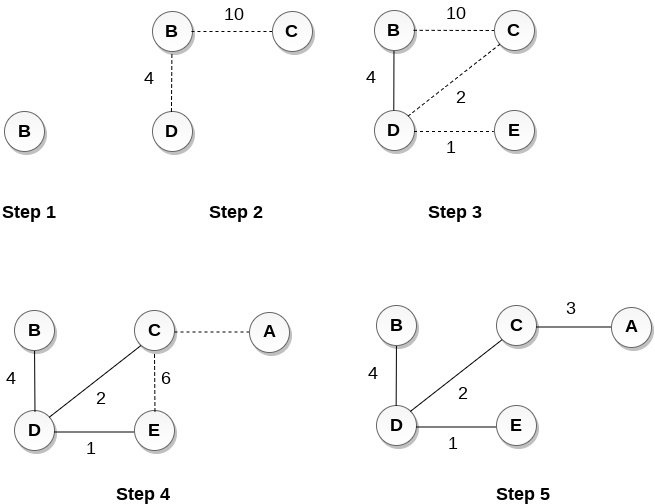
## Solution:

* **Step 1 :** Choose a starting vertex B.
* **Step 2:** Add the vertices that are adjacent to A. the edges that connecting the vertices are shown by dotted lines.
* **Step 3:** Choose the edge with the minimum weight among all. i.e. BD and add it to MST. Add the adjacent vertices of D i.e. C and E.
* **Step 3:** Choose the edge with the minimum weight among all. In this case, the edges DE and CD are such edges. Add them to MST and explore the adjacent of C i.e. E and A.
* **Step 4:** Choose the edge with the minimum weight i.e. CA. We can't choose CE as it would cause cycle in the graph.

The graph produces in the step 4 is the minimum spanning tree of the graph shown in the above figure.

The cost of MST will be calculated as;

cost(MST) = 4 + 2 + 1 + 3 = 10 units.



**Source Code:**

#include <stdio.h>

#include <limits.h>

#define vertices 5

int minimum\_key(int k[], int mst[])

{

int minimum = INT\_MAX, min,i;

for (i = 0; i < vertices; i++)

if (mst[i] == 0 && k[i] < minimum )

minimum = k[i], min = i;

return min;

}

void prim(int g[vertices][vertices])

{

int parent[vertices];

int k[vertices];

int mst[vertices];

int i, count,u,v;

for (i = 0; i < vertices; i++)

k[i] = INT\_MAX, mst[i] = 0;

k[0] = 0;

parent[0] = -1;

for (count = 0; count < vertices-1; count++)

{

u = minimum\_key(k, mst);

mst[u] = 1;

for (v = 0; v < vertices; v++)

if (g[u][v] && mst[v] == 0 && g[u][v] < k[v])

parent[v] = u, k[v] = g[u][v];

}

for (i = 1; i < vertices; i++)

printf("%d %d %d \n", parent[i], i, g[i][parent[i]]);

}

void main()

{

int g[vertices][vertices] = {{3, 2, 1, 9, 0},

{5, 1, 2, 10, 4},

{0, 4, 1, 0, 9},

{8, 10, 0, 2, 10},

{1, 6, 8, 11, 0},

};

prim(g);

}

**Output:**

