ASSIGNMENT NO. :

NAME : Karne Shraddha Sunil.

ROLL NO. : 302019.

CLASS :TE Comp 1.

PROGRAM :

#include<iostream>

using namespace std;

#include <stdio.h>

#include <stdlib.h>

#include <omp.h>

int main()

{

//Variables

int nodes,minimum=9999,cost=0,i,j;

// Start timer

double start\_time;

start\_time = omp\_get\_wtime();

nodes=5;

int visited[10]={0},edges[nodes][nodes],nv=1,x,y;

cout<<"\*\*\*\*\* PRIMS ALGORITHM \*\*\*\*\*\*"<<endl;

cout<<"TAKING INPUT ......"<<endl;

//Initializing the adjacency matrix

int graph[5][5] = {{0,6,7,5,3},{6,0,2,9,1},{7,2,0,9,3},{5,9,9,0,6},{3,1,3,6,0}};

cout<<" THE ADJACENCY MATRIX IS :"<<endl;

//Printing the adjacency matrix

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

{

cout<<"\t\t";

for( j=0 ;j<nodes;j++)

{

cout<<graph[i][j]<<"\t";

edges[i][j]=0;

}

cout<<endl;

}

//Select start node and mark it as visited

visited[1]=1;

//Iterate through the loop untill all nodes are visited

while(nv<nodes)

{

minimum =9999;

//Find nearest node

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

{

if(visited[i]==1)

{

#pragma omp parallel for

for( j=0;j<nodes;j++)

{

if(visited[j]==0 && graph[i][j]!=0)

{

#pragma omp critical

if(minimum>graph[i][j])

{

minimum =graph[i][j];

x=i;

y=j;

}

}

}

}

}

//Add nearest node to the tree and mark it visited

//Increase the cost of the tree

//Increment number of visited nodes

cout<<"\n(v"<<x<<",v"<<y<<")->"<<minimum;

cost=cost+minimum;

edges[x][y]=minimum;

visited[y]=1;

nv++;

}

cout<<"\nThe spanning tree is (with weights):\n";

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

{

for( j=0;j<nodes;j++)

cout<<" "<<edges[i][j];

cout<<"\n";

}

cout<<"\n The cost of minimum spanning tree = "<<cost<<endl;

cout<<"\nTotal Execution time :\t"<<omp\_get\_wtime() - start\_time;

cout<<"\n\n";

return 0;

}