9.1. Find a minimum spanning tree using any method Kruskal’s Algorithm or Prim’s Algorithm.

Prim’s Algorithm:

#include<iostream> #include<conio.h> using namespace std; #define ROW 7

#define COL 7

#define infi 99 class prims{

int graph[ROW][COL],nodes; public:

prims();

void createGraph(); void primsAlgo();

};

prims::prims()

{

for(int i=0;i<ROW;i++) for(int j=0;j<COL;j++) graph[i][j]=0;

}

void prims::createGraph()

{

int i,j;

cout<<"Enter Total Nodes: ", cin>>nodes;

cout<<"\n\n Enter Adjacency Matrix: \n"; for(i=1;i<=nodes;i++) for(j=1;j<=nodes;j++)

cin>>graph[i][j]; for(i=1;i<=nodes;i++) for(j=1;j<=nodes;j++) if(graph[i][j]==0) graph[i][j]=infi;

cout<<"Matrix is: "<<endl; for(i=1;i<=nodes;i++){ for(j

=1;j<=nodes;j++){ cout<<" "<<graph[i][j];

}

cout<<endl;

}

}

void prims::primsAlgo()

{

cout<<"Minimum spanning tree is: "; int selected[ROW],i,j,ne;

int f,t,min,x,y;

f=0; t=1;

for(i=1;i<=nodes;i++) selected[i]=f;

selected[1]=t; ne=1; while(ne<=nodes-1)

{

min=infi; for(i=1;i<=nodes;i++)

{

if(selected[i]==t)

{

for(j=1;j<=nodes;j++)

{

if(selected[j]==f)

{

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

{

min=graph[i][j]; x=i;

y=j;

}

}

}

}

}

selected[y]=t; cout<<"\n"<<x<<"--->"<<y; ne=ne+1;

}

}

int

main(){ pri ms mst;

cout<<"\n Prims Algorithm to find Minimum Spanning Tree/n"; mst.createGraph();

mst.primsAlgo(); getch();

}