Assignment No 7

Input:

```
#include<iostream>
using namespace std;

class tree
{
	int a[20][20],l,u,w,i,j,v,e,visited[20];
public:
		void input();
		void display();
		void minimum();
};

void tree::input()			//To take input from the user
{
	cout<<"Enter the no. of branches: ";	//ask the user about no. of branches as v=no. of branches
	cin>>v;

	for(i=0;i<v;i++)
	{
		visited[i]=0;
		for(j=0;j<v;j++)
		{
			a[i][j]=999;
		}
	}

	cout<<"\nEnter the no. of connections: ";		//ask for no. of connections required as e
	cin>>e;

	for(i=0;i<e;i++)
	{
		cout<<"Enter the end branches of connections:  "<<endl;	//ask for two end point connections
		cin>>l>>u;
		cout<<"Enter the phone company charges for this connection:  ";	//ask for the charges for that connection
		cin>>w;
		a[l-1][u-1]=a[u-1][l-1]=w;
	}
}

void tree::display()
```

```
{
        cout<<"\nAdjacency matrix:";
        for(i=0;i<v;i++)
        {
                cout<<endl;
                for(j=0;j<v;j++)
                {
                        cout<<a[i][j]<<"   ";
                }
                cout<<endl;
        }
}

void tree::minimum()                    //to find minimum cost using prims algo
{
        int p=0,q=0,total=0,min;    //declare and initialise the variables
        visited[0]=1;                   //visited first city as1
        for(int count=0;count<(v-1);count++)  //count variable declare in for loop itself
        {
                min=999;                        //initially min=999
                for(i=0;i<v;i++)
                {
                        if(visited[i]==1)      //if first node is visited(i is show source city)
                        {
                                for(j=0;j<v;j++)       //(j is show destination city)
                                {
                                        if(visited[j]!=1)     //second node is not visited
                                        {
                                                if(min > a[i][j])  //then compare their wt with
min var.& if it is less then
                                                {
                                                        min=a[i][j];    //put that wt in min
                                                        p=i;            //update that min node p as i
                                                        q=j;          // q as j
                                                }
                                        }
                                }
                        }
                }
                visited[p]=1;  //make p & q as visited node by inserting 1
                visited[q]=1;
                total=total + min;  //add min to their total cost
                cout<<"Minimum cost connection is"<<(p+1)<<" -> "<<(q+1)<<"  with
charge : "<<min<< endl;
                                //print that cost with their nodes
        }  //continue search until all are visited
```

```
	cout<<"The minimum total cost of connections of all branches is: "<<total<<endl;	// print total cost of connection
}

int main()
{
	int ch;
	tree t;
	do
	{
		cout<<"==========PRIM'S ALGORITHM================="<<endl;
		cout<<"\n1.INPUT\n \n2.DISPLAY\n \n3.MINIMUM\n"<<endl;
		cout<<"Enter your choice :"<<endl;
		cin>>ch;

	switch(ch)
	{
	case 1: cout<<"*******INPUT YOUR VALUES*******"<<endl;
		t.input();
		break;

	case 2: cout<<"*******DISPLAY THE CONTENTS********"<<endl;
		t.display();
		break;

	case 3: cout<<"*********MINIMUM***********"<<endl;
		t.minimum();
		break;
	}

	}while(ch!=4);
	return 0;
}
```

Output:

```
==========PRIM'S ALGORITHM=================
1.INPUT
2.DISPLAY
3.MINIMUM
Enter your choice :
1
*******INPUT YOUR VALUES*******
Enter the no. of branches: 5
Enter the no. of connections: 10
Enter the end branches of connections:
2
```

3
Enter the phone company charges for this connection: 8
Enter the end branches of connections:
3
4
Enter the phone company charges for this connection: 9
Enter the end branches of connections:
5
6
Enter the phone company charges for this connection: 4
Enter the end branches of connections:
7
6
Enter the phone company charges for this connection: 3
Enter the end branches of connections:
8
2
Enter the phone company charges for this connection: 5
Enter the end branches of connections:
6
9
Enter the phone company charges for this connection: 4
Enter the end branches of connections:
8
1
Enter the phone company charges for this connection: 1
Enter the end branches of connections:
6
8
Enter the phone company charges for this connection: 6
Enter the end branches of connections:
8
5
Enter the phone company charges for this connection: 7
Enter the end branches of connections:
7
5
Enter the phone company charges for this connection: 10
==========PRIM'S ALGORITHM==========
1.INPUT
2.DISPLAY
3.MINIMUM
Enter your choice :
2
*******DISPLAY THE CONTENTS********

Adjacency matrix:
999 999 999 999 999

999 999 8 999 999

999 8 999 9 999

999 999 9 999 999

999 999 999 999 999
==========PRIM'S ALGORITHM================
1.INPUT
2.DISPLAY
3.MINIMUM
Enter your choice :
3
*********MINIMUM***********
Minimum cost connection is1 -> 1 with charge : 999
Minimum cost connection is1 -> 1 with charge : 999
Minimum cost connection is1 -> 1 with charge : 999
Minimum cost connection is1 -> 1 with charge : 999