C15- You have a business with several offices; you want to lease phone lines to connect them up with each other; and the phone company charges different amounts of money to connect different pairs of cities. You want a set of lines that connects all your offices with a minimum total cost. Solve the problem by suggesting appropriate data structures.

```
#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()
{
	cout<<"Enter the 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: ";
	cin>>e;

	for(i=0;i<e;i++)
	{
		cout<<"Enter the end branches of connections:  "<<endl;
		cin>>l>>u;
		cout<<"Enter the phone company charges for this 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()
{
        int p=0,q=0,total=0,min;
        visited[0]=1;
        for(int count=0;count<(v-1);count++)
        {
                min=999;
                for(i=0;i<v;i++)
                {
                        if(visited[i]==1)
                        {
                                for(j=0;j<v;j++)
                                {
                                        if(visited[j]!=1)
                                        {
                                                if(min > a[i][j])
                                                {
                                                        min=a[i][j];
                                                        p=i;
                                                        q=j;
                                                }
                                        }
                                }
                        }
                }
                visited[p]=1;
                visited[q]=1;
                total=total + min;
                cout<<"Minimum cost connection is"<<(p+1)<<" -> "<<(q+1)<<"  with charge :
"<<min<< endl;

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

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;
}
```