**EXERCISE -** 8

**Spanning Tree**

**Aim:** Write a java program to implement Minimum cost spanning tree problem using Prim's algorithm .

**File name:** Prims.java

**Program:**

**import java.util.Scanner;**

**public class Prims**

**{**

**public static void main(String[] args){**

**int cost[][]=new int[10][10];**

**int i, j, mincost = 0;**

**Scanner in = new Scanner(System.in);**

**System.out.println("-------- Prims Algorithm--------");**

**System.out.println("Enter the number of nodes");**

**int n = in.nextInt();**

**System.out.println("Enter the cost matrix");**

**for(i=1; i<=n; i++){**

**for(j=1; j<=n; j++){**

**cost[i][j] = in.nextInt();**

**}**

**}**

**System.out.println("The entered cost matrix is");**

**for(i=1; i<=n; i++){**

**for(j=1; j<=n; j++){**

**System.out.print(cost[i][j]+"\t");**

**}**

**System.out.println();**

**}**

**System.out.println("Minimum Spanning Tree Edges and costs are");**

**mincost=prims(cost,n,mincost);**

**System.out.print("The minimum spanning tree cost is:");**

**System.out.print(+mincost);**

**}**

**static int prims(int cost[][],int n,int mincost){**

**int nearV[]=new int[10],t[][]=new int[10][3],u = 0,i,j,k;**

**for(i=2; i<=n; i++)**

**nearV[i]=1;**

**nearV[1]=0;**

**for(i=1; i<n; i++){**

**int min=999;**

**for(j=1;j<=n;j++){**

**if(nearV[j]!=0 && cost[j][nearV[j]]<min){**

**min=cost[j][nearV[j]];**

**u=j;**

**}**

**}**

**t[i][1] = u;**

**t[i][2] = nearV[u];**

**mincost += min;**

**nearV[u] = 0;**

**for(k=1; k<=n; k++){**

**if(nearV[k] != 0 && cost[k][nearV[k]] > cost[k][u])**

**nearV[k] = u;**

**}**

**System.out.print(i+") Minimum edge is ("+t[i][1]);**

**System.out.println(","+t[i][2]+") and its cost is :"+min);**

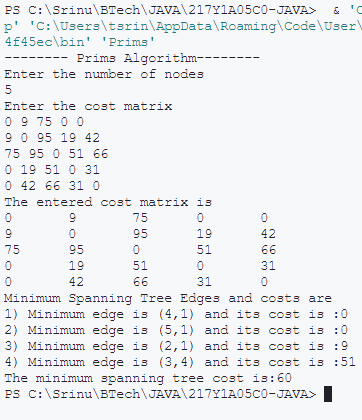
**}**

**return mincost;**

**}**

**}**

**Output:**

****

**Aim:** Write a java program to implement Minimum cost spanning tree problem using Kruskal's algorithm.

**File name:** KRUSKAL.java

**Program:**

**import java.util.Scanner;**

**public class KRUSKAL{**

**public static void main(String[] args)**

**{**

**int cost[][]=new int[10][10];**

**int i, j,mincost=0;**

**Scanner in = new Scanner(System.in);**

**System.out.println("\*\*\*\*\*\*\*\*\* KRUSKAL'S ALGORITHM \*\*\*\*\*\*\*");**

**System.out.println("Enter the number of nodes: ");**

**int n = in.nextInt();**

**System.out.println("Enter the cost matrix");**

**for(i=1;i<=n;i++){**

**for(j=1;j<=n;j++){**

**cost[i][j] = in.nextInt();**

**}**

**}**

**System.out.println("The entered cost matrix is");**

**for(i=1;i<=n;i++){**

**for(j=1;j<=n;j++){**

**System.out.print(cost[i][j]+"\t");**

**}**

**System.out.println();**

**}**

**mincost=kruskals(n,mincost,cost);**

**System.out.println("The minimum spanning tree cost is:");**

**System.out.println(mincost);**

**}**

**static int kruskals(int n,int mincost,int cost[][] )**

**{**

**int ne = 1,a=0,u=0,b=0,v=0,min;**

**int parent[]=new int[10];**

**while(ne < n){**

**min=999;**

**for(int i=1; i<=n; i++)**

**{**

**for(int j=1; j<=n; j++)**

**{**

**if(cost[i][j] < min){**

**min = cost[i][j];**

**a=u=i;**

**b=v=j;**

**}**

**}**

**}**

**while(parent[u]>0)**

**u = parent[u];**

**while(parent[v]>0)**

**v = parent[v];**

**if(u != v)**

**{**

**System.out.print((ne++)+">minimum edge is :");**

**System.out.println("("+a+","+b+") and its cost is:"+min);**

**mincost += min;**

**parent[v] = u;**

**}**

**cost[a][b] = cost[b][a] = 999;**

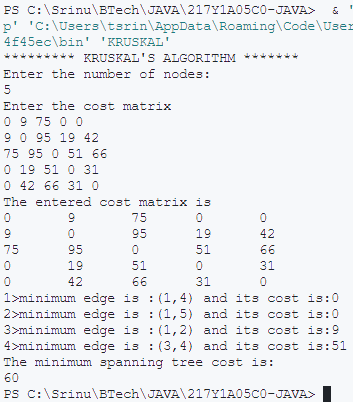
**}**

**return mincost;**

**}**

**}**

**Output:**

****