**Subset problem**

import java.util.Scanner;

public class subset {

static int sol=0;

public static void main(String[] args)

{

int set[]=new int[10];

int n,d,i,sum=0;

int x[]=new int[10];

Scanner sc=new Scanner(System.in);

System.out.println("enter the number of elements:");

n=sc.nextInt();

System.out.println("enter the elements in increasing order:");

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

set[i]=sc.nextInt();

System.out.println("enter the value of d:");

d=sc.nextInt();

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

sum=sum+set[i];

System.out.println("sum="+sum);

if(sum<d||set[0]>d)

{

System.out.println("subset is not possible");

System.exit(0);

}

subset(0,0,sum,x,set,d);

if(sol==0)

System.out.println("subset is not possible");

}

static void subset(int cs,int k,int r,int x[],int set[],int d)

{

x[k]=1;

if(cs+set[k]==d)

{

sol++;

System.out.print("\n subset"+sol+"is{");

for(int i=0;i<=k;i++)

if(x[i]==1)

{

System.out.print(set[i]+",");

}

System.out.print("}");

}

else if((cs+set[k]+set[k+1])<=d)

subset(cs+set[k],k+1,r-set[k],x,set,d);

if((cs+r-set[k])>=d&&(cs+set[k+1]<=d))

{

x[k]=0;

subset(cs,k+1,r-set[k],x,set,d);

}

}

}

**Kruskal’s algorithm**

import java.util.Scanner;

public class kruskal

{

int parent[]=new int[10];

int find(int m)

{

int p=m;

while(parent[p]!=0)

p=parent[p];

return p;

}

void union(int i,int j)

{

if(i<j)

parent[i]=j;

else

parent[j]=i;

}

void krkl(int[][]a, int n)

{

int u=0,v=0,min,k=0,i,j,sum=0;

while(k<n-1)

{

min=99;

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

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

if(a[i][j]<min&&i!=j)

{

min=a[i][j];

u=i;

v=j;

}

i=find(u);

j=find(v);

if(i!=j)

{

union(i,j);

System.out.println("("+u+","+v+")"+"="+a[u][v]);

sum=sum+a[u][v];

k++;

}

a[u][v]=a[v][u]=99;

}

System.out.println("The cost of minimum spanning tree = "+sum);

}

public static void main(String[] args)

{

int i,j;

System.out.println("Enter the number of vertices of the graph");

Scanner sc=new Scanner(System.in);

int n;

n=sc.nextInt();

int a[][]=new int[n][n];

System.out.println("Enter the wieghted matrix");

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

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

a[i][j]=sc.nextInt();

kruskal k=new kruskal();

k.krkl(a,n);

}

}