实验二

编程题 2 （总分：100.00）

#	题目	分值
1	**实验2-1: 整数划分问题**	50.00

Description

将正整数n表示成一系列正整数之和：n=n1+n2+…+nk，其中n1≥n2≥…≥nk≥1，k≥1。

正整数n的这种表示称为正整数n的划分。求正整数n的不同划分个数。

例如正整数6有如下11种不同的划分：

6；

5+1；

4+2，4+1+1；

3+3，3+2+1，3+1+1+1；

2+2+2，2+2+1+1，2+1+1+1+1；

1+1+1+1+1+1。

Input

输入包含n+1行；

第一行是一个整数n，表示有n个测试用例；

第2至n+1每行一个正整数。

Output

对应每组输入，输出正整数n的不同划分个数。

Sample Input

2

5

6

Sample Output

7

11

参考答案:

```
#include <stdio.h>
   int split(int n, int m)
   {
      if(n < 1 || m < 1) return 0;
      if(n == 1 || m == 1) return 1;
      if(n < m) return split(n, n);
      if(n == m) return (split(n, m - 1) + 1);
      if(n > m) return (split(n, m - 1) + split((n - m),
m));
  }
int main()
{
     int k,i;
     int a[100];
     scanf("%d",&k);
     for(i=0;i<k;i++)
     {
       scanf("%d",&a[i]);
     }
     for(i=0;i<k;i++)
     {
       printf("%d\n",split(a[i],a[i]));
     }
}
```

2 实验2-2: 邮局选址问题 50.00

Description

在一个按照东西和南北方向划分成规整街区的城市里，n个居民点散乱地分布在不同的街区中。用x 坐标表示东西向，用y坐标表示南北向。各居民点的位置可以由坐标(x,y)表示。街区中任意2 点(x1,y1)和(x2,y2)之间的距离可以用数值|x1-x2|+|y1-y2|度量(注意这里的距离度量方式)。居民们希望在城市中选择建立邮局的最佳位置，使n个居民点到邮局的距离总和最小。

编程任务： 给定n 个居民点的位置,编程计算n 个居民点到邮局的距离总和的最小值。

Input

输入由多组测试数据组成。 每组测试数据输入的第1 行是居民点数n，1≤n≤10000。接下来n 行是居民点的位置，每行2 个整数x 和y，-10000≤x，y≤10000。

Output

对应每组输入，输出的第1 行中的数是n 个居民点到邮局的距离总和的最小值。

Sample Input

5

1 2

2 2

1 3

3 -2

3 3

Sample Output

10

参考答案:

```
#include<iostream>
using namespace std;
void QuickSort(int arry[],int l,int h);
int main()
{
    int i,j,n,l,h,x,y,a,b;
        int sum1=0,sum2=0;
        cin >> n;
        l=0;
        h=n-1;
        int arry[10000][2];
        int *x1=new int[n];
        int *y1=new int[n];

    for(i=0;i<n;i++)
                for(j=0;j<2;j++)
                {
                        scanf("%d",&arry[i][j]);
                }
        for(i=0;i<n;i++)
        {
                x1[i]=arry[i][0];
                y1[i]=arry[i][1];
        }
    QuickSort( x1,l, h);
        QuickSort( y1,l, h);
        x=x1[(n -1)/2];
        y=y1[(n -1)/2];
        for(i=0;i<n;i++)
        {
                a=arry[i][0]-x;
                if((arry[i][0]-x)<0)
                {
                        a=x-arry[i][0];
                }
```

```
                b=arry[i][1]-y;
                if((arry[i][1]-y)<0)
                {
                        b=y-arry[i][1];
                }
                sum1+=a;
                sum2+=b;
        }
        cout<<sum1+sum2<<endl;
        return 0;
}
void QuickSort(int arry[],int l,int h)
        {
                int i=l, j=h; //低LOW , 高HIGH
                int temp = arry[l]; //取第一个做标准数据元书的
                while(i<j)
                {
                        while(i<j && temp <=arry[j])j--;//
右端扫描
                        if(i<j)
                        {
                        arry[i]=arry[j];
                        i++;
                        }
                        while(i<j && arry[i] < temp)i++;
                        if(i<j)
                        {
                        arry[j]=arry[i];
                        j--;
                        }
                }
                arry[i]=temp;
                if(l<i)QuickSort( arry, l,i-1);
                if(i<h)QuickSort( arry, j+1,h);
        }
```