PART ONE

作者：2014 级电子信息工程--范娇娇

（注明：以下程序均已通过 OJ）

一、计算矩阵边缘元素之和

1、**问题描述：** 输入一个整数矩阵，计算位于矩阵边缘的元素之和。所谓矩阵边缘的元素，就是第一行和 最后一行的元素以及第一列和最后一列的元素。

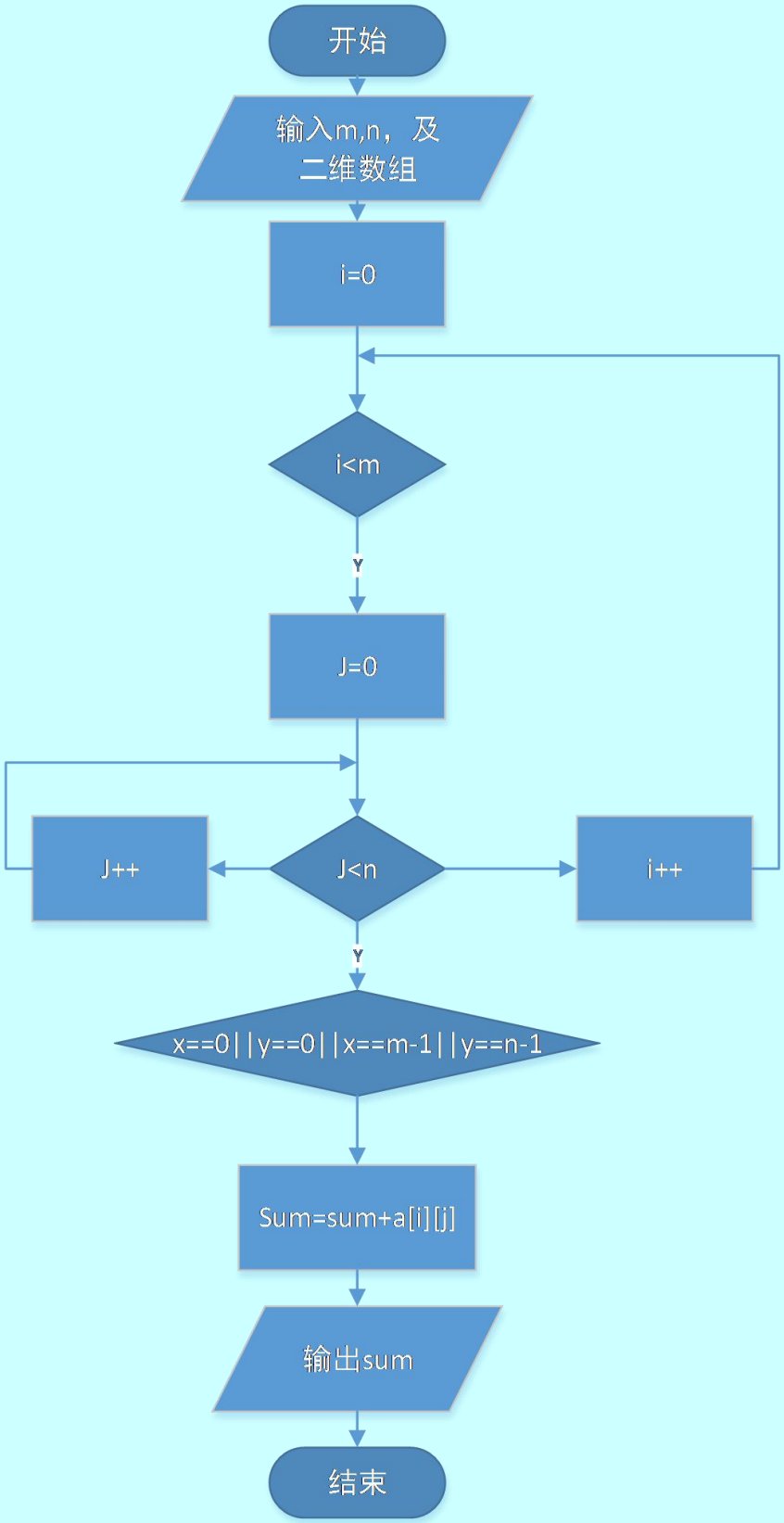
A、输入：

- 第一行为整数 k，表示有 k 组数据。
- 每组数据有多行组成，表示一个矩阵：第一行分别为矩阵的行数 m 和列数 n（m < 100，n < 100），两者之间以空格分隔。 接下来输入的 m 行数据中，每行包含 n 个整数，整数之间以空格作为间隔。

B、输出 ：输出对应矩阵的边缘元素和，一个一行。

2、解决方案：首要的是找出矩阵的边缘元素，满足 m 行 n 列的数组边缘元素的条件为（k==0||l==0||k==m-1||l==n-1），找到这些元素后，将其相加求和，得到最终结果。

3、流程图


开始
输入m,n，及
二维数组
i=0
i<m
Y
J=0
J++
J<n
i++
Y
x==0||y==0||x==m-1||y==n-1
Sum=sum+a[i][j]
输出sum
结束

4、编程实现

```
#include<iostream>
using namespace std;
int main()
{
int k;
cin>>k;
while(k!=0)
{
int m,n;
cin>>m>>n;
int a[100][100];
for(int i=0;i<m;i++)
{
    for(int j=0;j<n;j++)
    {
       cin>>a[i][j];
    }
}
int *p;
int sum=0;
for(int x=0;x<m;x++)
{
    for(int y=0;y<n;y++)
    {
if(x==0||y==0||x==m-1||y==n-1)
      {
p=&a[x][y];
sum=sum+(*p);
      }
    }
}
cout<<sum<<endl;
k--;
}
return 0;
}
```

```
#include <iostream>
using namespace std;
int add()
{
    int m, n;
    int sum = 0;
    cin >> m >> n;
    int a[100][100];
    for (int i = 0; i<m; i++)
    {
        for (int j = 0; j<n; j++)
        {
            cin >> a[i][j];
        }
    }

    for (int k = 0; k<m; k++)
    {
        for (int l = 0; l<n; l++)
        {
            if (k==0||l==0||k==m-1||l==n-1)
                sum=sum+(*(&a[k][l]));
        }
    }
    cout << sum << endl;
    return 0;
}
int main()
{
    int x;
    cin >> x;
    for (int h = 0; h<x; h++)
    {
        add();
    }
    return 0;

}
```

5、结果分析：左一的程序为常规写法，关键是找出（k==0||l==0||k==m-1||l==n-1）这个条件，则整个程序的编辑则很很容易了；经过思考，我觉得还可以练习使用一下自定义函数和指针，于是右一的程序就产生了，虽然是简单的调用，也不乏为一种新的方式；在右一函数编写过程中将输入循环条件错写为

```
for (int i = 0; i<m; i++){
for (int j = 0; j<n; j++){cin >> a[i][j];
```

导致程序只能计算 m=n 的数组，否则则会出现数组越界的情况，比如调试过程如下：

```
for(int x=0;x<m;x++)
{
    for(int y=0;y<n;y++)
    {
if(x==0||y==0||x==m-1||y==n-1)
     {
      cout<<x<<" "<<y<<" "<<a[x][y]<<endl;
p=&a[x][y];
sum=sum+(*p);
     }
    }
}
```

```
1
2 3
1 1 1
1 1 1
0 0 1
0 1 1
0 2 -858993460
1 0 1
1 1 1
1 2 -858993460
-1717986916
```

6、总结分析：从拿到这个问题，到解决这个问题的过程中，有突然开窍的时候，也有突然脑子秀逗的时候，对于同一问题，要多方面考虑，活跃思维，对所学到的知识要做到灵活运用，同时，细节很重要，比如，自己检查自己的程序很容易忽视细节性的错误，比如我在输入循环条件上犯的错误，就让我浪费了一部分时间来找出问题所在，细节决定成败!!!

二、二维数组右上左下遍历

1、问题描述：给定一个 row 行 col 列的整数数组 array，要求从 array[0][0] 元素开始，按从左上到右下 的对角线顺序遍历整个数组。

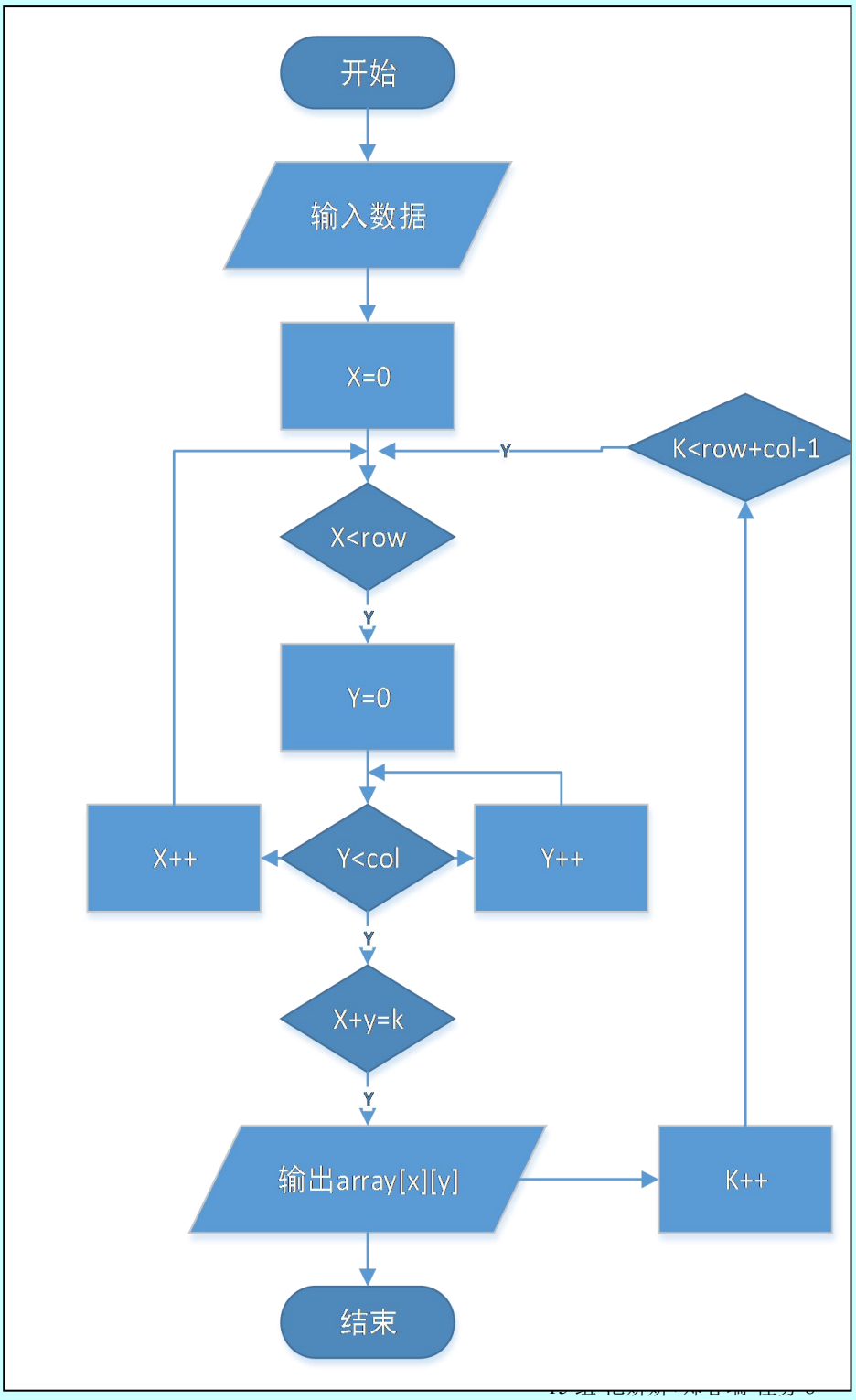
- 输入的第一行上有两个整数，依次为 row 和 col。
- 余下有 row 行，每行包含 col 个整数，构成一个二维整数数组。

（注：输入的 row 和 col 保证 0 < row < 100，0 < col < 100）

- 输出 按遍历顺序输出每个整数。每个整数占一行。

2、解决方案：仔细分析题目要求，可总结出如下规律：以第一行的每个元素为首，其小标和与其对角线上的每一个数组元素下标之和相等，根据此规律，解题思路如下文流程图所示

3、流程图



4、编程实现

```
#include <iostream>
using namespace std;
int main()
{
	int row,col,k=0;
	int*p;
	int array[100][100];
	cin>>row>>col;
	for(int i=0;i<row;i++)
	{
		for(int j=0;j<col;j++)
		{
			cin>>array[i][j];
		}
	}

while(k<row+col-1)
{
for(int x=0;x<row;x++)
	{
for(int y=0;y<col;y++)
		{
if((x+y)==k)
		{
	cout<<array[x][y]<<endl;
		}
		}
	}
k++;
}
return 0;
}
```

5、问题分析：在开始分析这道题的时候，除了注意到下标和的关系，还注意到将曲线拉直的话，其下标排列是

00 01 10	02 11 20	03 12 21 30	04 13 22 31 40	05 14 23 32 41 50

一维下标从 0 递增，二维下标从高位递减；且如果从相隔元素个数来看的话，没两个连续的对角线上的元素之间相隔的元素个数是有一定规律的；但这两种算法和下标和的算法比较起来都复杂很多，所以选择简单易懂的算法解决问题，节省时间，且程序可读性强。

6、总结体会:每个问题所对应的算法都会有很多种，但应该斟酌选择较简单易懂的，不仅节省时间，且增强程序的可读性，也有利于程序的后期维护。

PART TWO

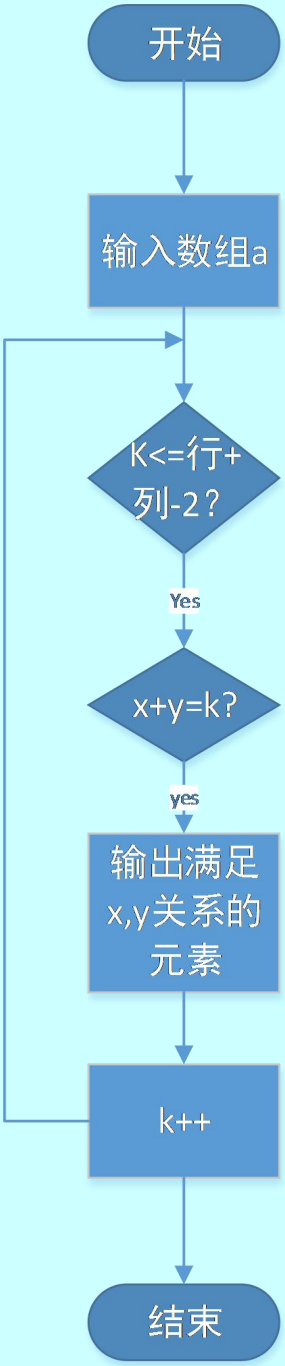
作者：14-电子信息工程-郑春瑞

一、 二维数组右上左下遍历

1、问题描述：给定一个 row 行 col 列的整数数组 array，要求从 array[0][0] 元素开始，按从左上到右下的对角线顺序遍历整个数组。

2、解决方案：输出的对角线上有这样的规律，行列的和依次为 0,1,2,3 · · · 行+列-2，所以可以根据这个规律利用数组和循环语句实现。

3、流程图： **4、程序代码：（通过ＯＪ）**




```
#include <iostream>
using namespace std;
int main()
{
	int row,col,k=0;
	int array[100][100];
	cin>>row>>col;
	for(int i=0;i<row;i++)
	{
		for(int j=0;j<col;j++)
		{
			cin>>array[i][j];
		}
	}

while(k<=row+col-2)
{
for(int x=0;x<row;x++)
	{
for(int y=0;y<col;y++)
		{
if((x+y)==k)
		{
	cout<<array[x][y]<<endl;
		}
		}
	}
k++;
}
return 0;
}
```

5、总结体会：我觉得这个程序对于循环和数组的要求比较高，一开始有点摸不到头绪，

后来在纸上画了画有了点想法，但是在编程的时候还是有点感觉乱乱的，后来参考网上的程序才理清了些思绪。

二、文字排版

1、问题描述：将一段英文短文重新排版，单词之间以空格分隔（每个单词包括其前后紧邻的标点符号），要求如下：

1）每行不超过 80 个字符；

2）每个单词居于同一行上；

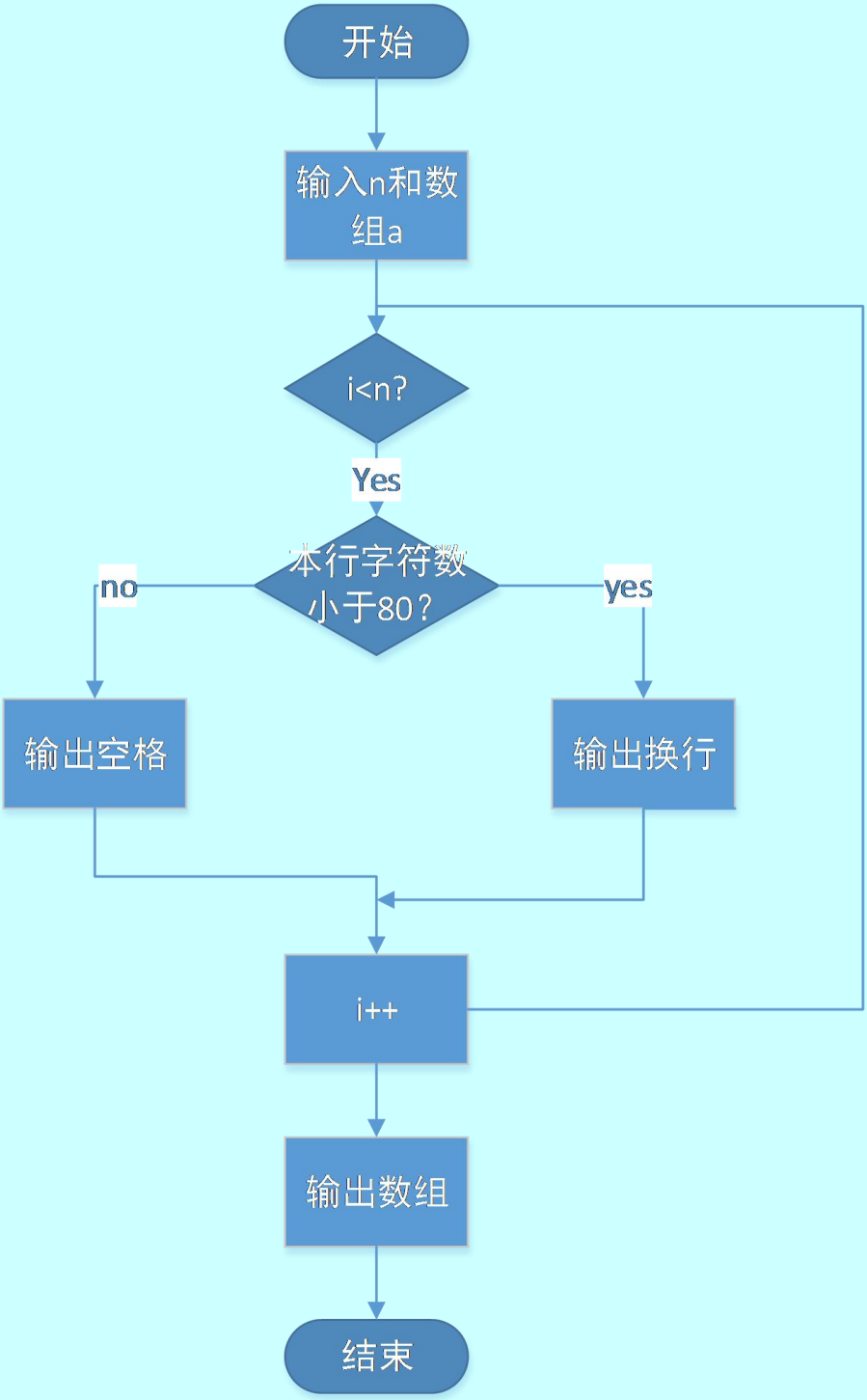
3）在同一行的单词之间以一个空格分隔；

4）行首和行尾都没有空格。

总的来说就是将比较长的一横行变成题目所要求的一段。并满足输入的第一行是一个整数 n，表示英文短文中单词的数目。其后是 n 个以空格分隔的英文单词（单词包括其前后紧邻的标点符号，且每个单词长度都不大于 40 个字母）；输出之后，排版后的多行文本，每行文本字符数最多 80 个字符，单词之间以一个空格分隔，每行文本首尾都没有空格。

2、解决方案：使用数组进行输入，每一行如果算上空格的长度大于 80 个字符，那么就要输出换行。其中每个单词之间的空格是要保留的，并且行首行尾都没有空格，所以需要考虑输出换行和空格分别为什么情况。

3、流程图：



4、编程实现：（通过 OJ）

```
#include<iostream>
#include<cstring>
using namespace std;
int main()
{
	char a[100];
	int m=0,n;
	cin>>n;
	for(int i=0;i<n;i++){
		cin>>a;
		if(m+1+strlen(a)>80){
			cout<<endl;
			m=0;
		}
		else if(i>0){
			cout<<" ";
			m++;
		}
		cout<<a;
		m+=strlen(a);
	}
	return 0;
}
```

5、**总结：**这个程序主要还是考察数组的运用，以及 strlen 函数的应用，满足 80 的时候换行是这个题一个比较关键的解决点，当然对于 if 语句和循环语句也要求灵活运用。

四、实验总结

本次实验感觉应该需要用指针的，但是不知道该在什么时候用，就像当时不知道在什么时候用函数一样。程序一次比一次的难度变大，每次刚看到题目的时候都有些摸不到头绪，甚至有时候连题目也有些读不懂，好多时候都需要借助网上的程序才能完成，感觉自己对于程序的练习和理解还不够，需要进一步的加强。通过这一次的作业，也发现在纸上先写写画画整理一下程序整体思想是很重要的。