PartOne

作者：14 电工--范娇娇

一、判断闰年

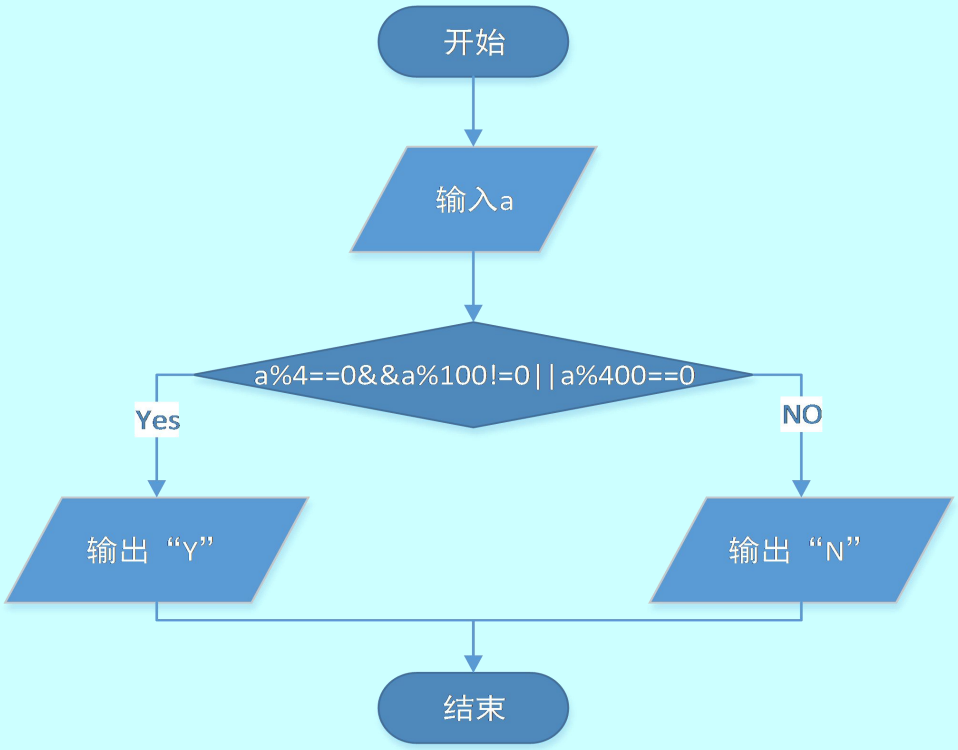
1、**问题描述**：判断某年是否为闰年。

2.、解决方案：此程序的核心算法就是：if（(n%4 == 0 && n%100 != 0) || n %400 == 0）

（关于这个问题，我一直不清楚为什么算法是这样的，因此查了些资料，原因如下）

【关于公历闰年是这样规定的：地球绕太阳公转一周叫做一回归年，一回归年长 365 日 5 时 48 分 46 秒。因此，公历规定有平年和闰年，平年一年有 365 日，比回归年短 0.2422 日，四年共短 0.9688 日，故每四年增加一日，这一年有 366 日，就是闰年。但四年增加一日比四个回归年又多 0.0312 日,400 年后将多 3.12 日,故在 400 年中少设 3 个闰年,也就是在 400 年中只设 97 个闰年，这样公历年的平均长度与回归年就相近似了。由此规定：年份是整百数的必须是 400 的倍数才是闰年，例如 1900 年、2100 年就不是闰年。即四年一闰，百年不闰，四百年再闰。】

3.流程图



4.编程实现（此程序已通过 OJ）

```
#include <iostream>
using namespace std;
int main()
{
    int a;
    cin>>a;
    if(a%4==0&&a%100!=0||a%400==0)
        cout<<"Y"<<endl;
    else
        cout<<"N"<<endl;
    return 0;
}
```

样例输入

```
1900
```

样例输出

```
N
```

5.**总体分析：**对程序的算法理解有助于对程序的整体把握。

二、能被 3、7、5 整除的数

1.**问题描述**：输入一个整数，判断其能否被 3、5、7 整除

要求：（1）输入一个数字；

（2）输出：a.能同时被 3，5，7 整除（直接输出 3 5 7，每个数中间一个空格）；

b.能被其中两个数整除（输出两个数，小在前大在后，用空格分隔）

c.能被其中一个数整除（输出这个除数）

d.不能被任何数整除；（输出小写字符'n',不包括单引号）

2.解决方案 1：可根据题意，将每种情况及其输出情况枚举出来，如下：（**此程序已通过 OJ**）

```
if(n%3==0&&n%5==0&&n%7==0)    cout<<"3"<<" "<<"5"<<" "<<"7"<<endl;
if(n%3==0&&n%5==0&&n%7!=0)    cout<<"3"<<" "<<"5"<<endl;
if(n%3==0&&n%7==0&&n%5!=0)    cout<<"3"<<" "<<"7"<<endl;
if(n%5==0&&n%7==0&&n%3!=0)    cout<<"5"<<" "<<"7"<<endl;
if(n%3==0&&n%5!=0&&n%7!=0)    cout<<"3"<<endl;
if(n%5==0&&n%3!=0&&n%7!=0)    cout<<"5"<<endl;
if(n%7==0&&n%5!=0&&n%3!=0)    cout<<"7"<<endl;
if(n%3!=0&&n%5!=0&&n%7!=0)    cout<<"n"<<endl;
```

解决方案 2：可以先判断输入的这个数可以被 3、5、7 中的几个数整除，再结合 switch case 语句判断可以被哪几个数整除，具体实现如下：（**此程序未通过 OJ,但在其他编译器正常运行**）

```
#include <iostream>
using namespace std;
int main()
{
int N,nu=0;
cin>>N;
   {
if(N%3==0) nu++;
if(N%5==0) nu++;
if(N%7==0) nu++;
switch(nu)
      {
       case 0:cout<<"n"<<endl;break;

       case 1:
                if(N%3==0) cout<<"3"<<endl;
                if(N%5==0) cout<<"5"<<endl;
                if(N%7==0) cout<<"7"<<endl;
       break;

       case 2:
                 if(N%3==0) cout<<"3 ";
                 if(N%5==0) cout<<"5 ";
                 if(N%7==0) cout<<"7 ";
                 break;

       case 3:cout<<"3"<<" "<<"5"<<" "<<"7"<<endl;
       break;
      }
   }
}
```

样例输入

```
0
5
15
105
```

样例输出

```
3 5 7
5
3 5
3 5 7
```

3.问题分析：（1）解决方案 2 适用于更加复杂的程序设计；

（2）解决方案 2 没有通过 OJ 的原因一部分为格式错误，另一部分原因还在调试。

三、 最远距离

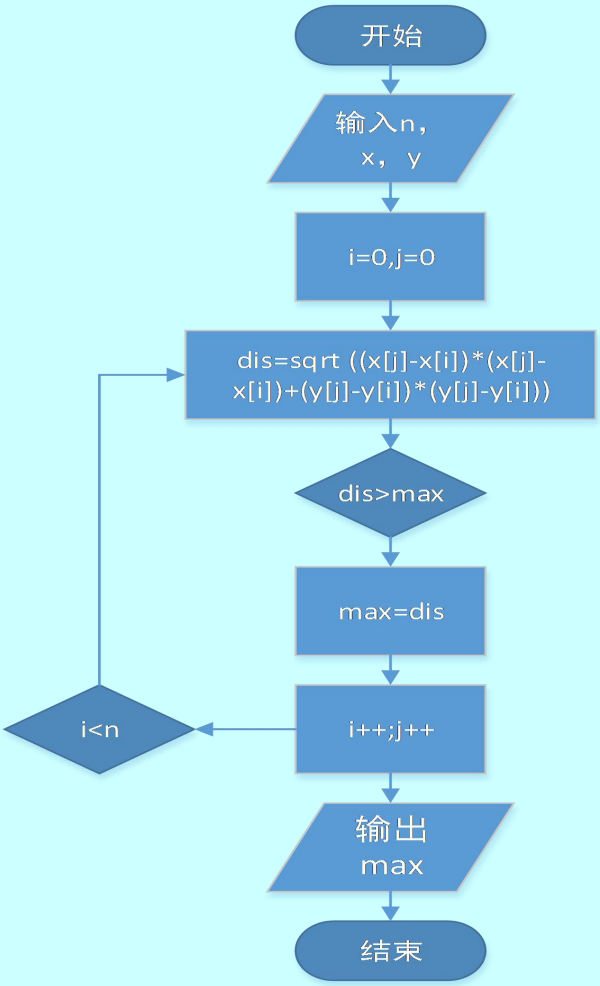
1.问题描述： 给定一组点(x,y)，求距离最远的两个点之间的距离。

要求：输入：第一行是点数 n（n 大于等于 2），接着每一行代表一个点，由两个浮点数 x y 组成。
输出：输出一行是最远两点之间的距离，并精确到小数点后 4 位。

2.解决方案：调用 cmath 库里的 sqrt 函数，则两点之间距离公式为：sqrt（（x2-x1）*(x2-x1)+(y2-y1)*(y2-y1)）

结合循环语句，计算每一个点与其他点的距离，找出最大距离。

3.流程图



4.编程实现（此程序已通过 OJ）

```
#include <iostream>
#include<cmath>
#include <iomanip>
using namespace std;
int main()
{
	int n;
	cin>>n;
	double max=0;
	float x[1000],y[1000];

	for(int k=0;k<n;k++)
	{
	cin>>x[k]>>y[k];
	}

	for(int i=0;i<n;i++)
	{
		for(int j=0;j<n;j++)
		{
			double dis=sqrt ((x[j]-x[i])*(x[j]-x[i])+(y[j]-y[i])*(y[j]-y[i]));
				if (dis>max)
				max=dis;
	}
	}
	cout << fixed << setprecision(4) << max << endl;
	return 0;
}
```

样例输入

```
6
34.0    23.0
28.1    21.6
14.7    17.1
17.0    27.2
34.7    67.1
29.3    65.1
```

样例输出

```
53.8516
```

5.总结分析：也可将各个点之间的距离计算自定义为一个函数来简化程序。

四、实验总结

1.在着手解决一个问题时，先摸索理解透彻其算法，对编程的实现有很大的帮助；

2.在已经可以轻松解决当前的编程问题的前提下，下一步可以考虑如何简化程序；

3.当然，理解算法的来源也是对解决问题有很大的帮助，比如前文提到的判断闰年的算法来源。

PartTwo

作者：14 电工-郑春瑞

一、简单计算器

1.问题描述：

设计一个计算器的程序，可以进行加减乘除（+，-，*，/）的运算。

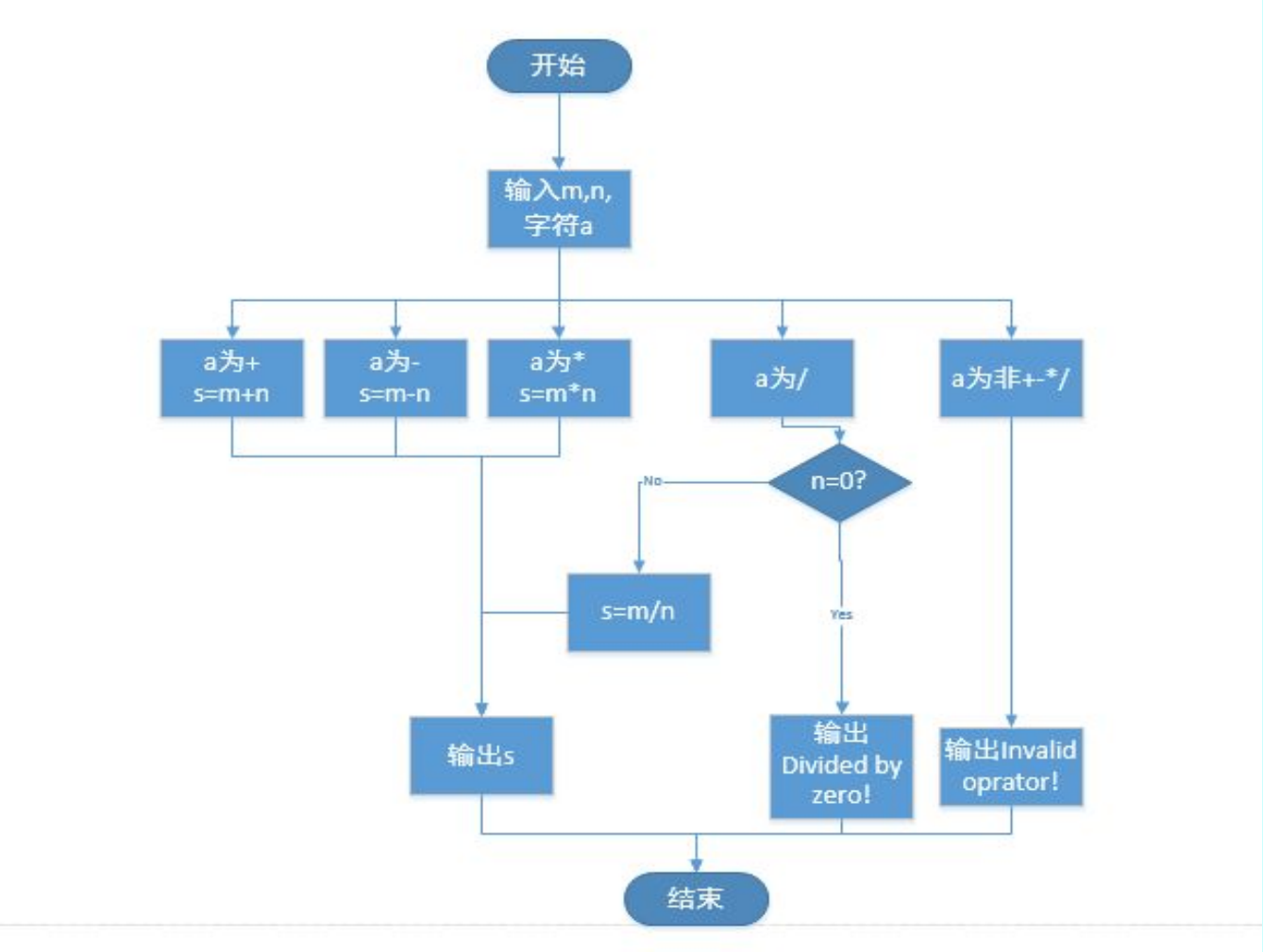
要求：1）输入三个参数，第 1,2 个是整数，第三个为操作符；

2）如果除数为 0 或输入无效的操作符，输出特定语句。

2.解决方案：

用 if 语句实现除数是否为 0 的循环， 用 switch 语句实现+-*/的运算。

3.流程图：



4.编程实现：（此程序通过 OJ）

```
#include<iostream>
using namespace std;
int main()
{
        int m,n,s;
        cin>>m>>n;
        char a;
        cin>>a;
        switch(a){
        case'+':s=m+n;
                cout<<s<<endl;
                break;
        case'-':s=m-n;
                cout<<s<<endl;
                break;
        case'*':s=m*n;
                cout<<s<<endl;
                break;
        case'/':if(n==0)
                        {       cout<<"Divided by zero!"<<endl;break;}
                else
                s=m/n;
                cout<<s<<endl;
                break;
default:cout<<"Invalid operator!"<<endl;break;
        }
        return 0;
}
```

样例输入

```
1 2 +
```

样例输出

```
3
```

5.总结分析：

如果 if 语句中 else 前面不加 break，会在执行 if 后继续执行 else 后的内，在输出正确结果后又输出乱码；此程序应比较熟练的运用 switch 和 if 循环语句。

二、字符串输入

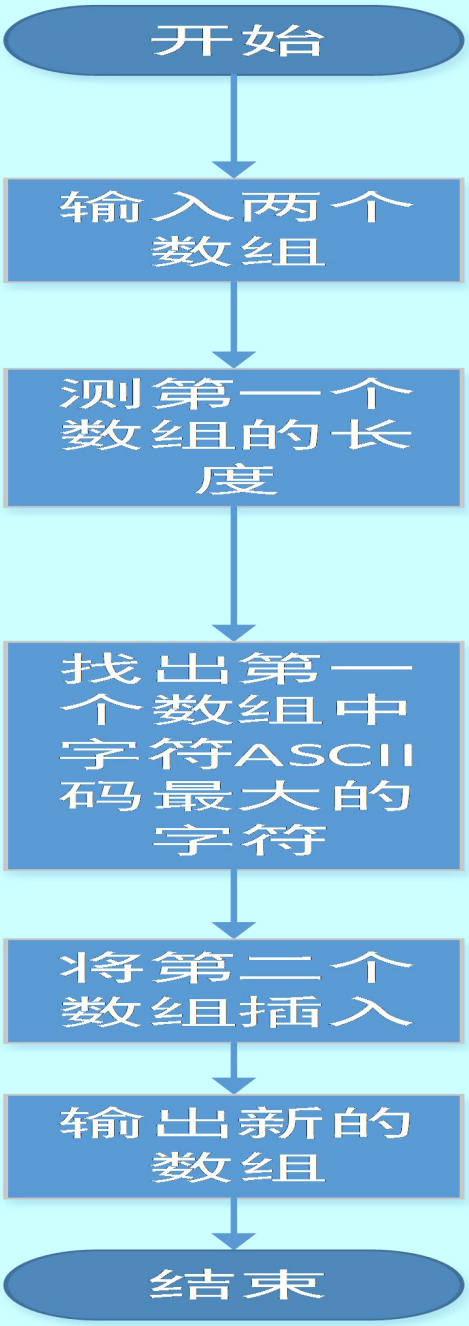
1.问题描述：

有两个字符串 str[11]和 substr[4],将 substr 插入到 str 中 ASCII 码最大的字符后面，如果不止一个最大则只考虑第一个。

2.解决方案:

调用测字符串长度的 strlen 函数，使用 for 循环完成字符的大小比较，在最大的字符处停止，然后插入另一个字符数组。

3.流程图:



4.编程实现（此程序通过 OJ）

```
#include<iostream>
#include<string.h>
using namespace std;
int main()
{
        char str[14],substr[4];
        int c;
                while(cin>>str>>substr)
                {
                        int i,m=0,max=0;
                        c=strlen(str);
                    for(i=0;i<c;i++)
                        {
                                if(str[i]>max)
                                {
                                 max=str[i];
                                 m=i;
                                }
                        }
                        for(i=c+3;i>=m+4;i--)
                        {
                                str[i]=str[i-3];
                        }
                        str[m+1]=substr[0];
                        str[m+2]=substr[1];
                        str[m+3]=substr[2];
                        cout<<str<<endl;
                }
        return 0;
}
```

样例输入

```
abcab eee
12343 555
```

样例输出

```
abceeeab
12345553
```

5.总结分析：

这个程序的 while 循环决定了一次性可以输入几组循环，熟练运用循环语句以及函数调用很重要，对函数也需要一定的理解。

三、实验总结

1）在编程序之前对于程序有一个整体的把握是很重要的，首先应该有大致的思路，需要用到什么样的循环是否需要调用函数（流程图或伪代码，问题分析）是比较重要的，对于编程序也起到了一定的帮助，减少了一些不必要的麻烦。

2）在编写出程序后可以进一步考虑是否有其他的方案，以拓展自己的思维。