第四章问题总结

4.1、4.6

部分同学冒泡排序时控制循环边界出错导致数组越界，例如对十个数进行冒泡排序，部分同学写成：

for(i=0;i<10;i++)

for(j=0;j<10-i;j++)

正确代码为

for(i=0;i<9;i++)

for(j=0;j<9-i;j++)

冒泡排序：

i∈**[0,N-1)** //循环N-1遍

j∈**[0,N-1-i)** //每遍循环要处理的无序部分

swap(**j,j+1**) //两两排序（升序/降序）

4.2、4.3、4.4

1、部分同学在定于数组时出错，例如：

int a[3][3]={{111},{222},{333}}—————每行元素间忘记加逗号

int a[3][3]; a={{1,1,1},{2,2,2},{3,3,3}}

int a[3][3]={(1,1,1),(2,2,2),(3,3,3)}—————定义数组有误

2、for语句后的复合语句没有加{}以致运算结果出错。

4.5

1、同学在定义字符数组时没有初始化数组长度，而直接对数组进行输入，例如：char a[]; gets(a);导致编译错误。

fget(a, 10, stdin);

2、使用printf(“%s”)输出字符串时输出变量名应为数组名，而不是数组中的元素。

第五章问题总结

5.2

交换操作时使用的中间元素类型与交换元素类型不符导致结果出错,例如：

void exchange(float a ,float b)

{

int c;

c=a;

a=b;

b=c;

}

void swap(int \*a, int \*b)

{

int temp;

temp = \*a;

\*a = \*b;

\*b = temp;

}

5.3、5.4

部分同学的闰年判断逻辑有误。

正确且简洁的写法为

if( y%4==0 && y%100 !=0 || y%400==0 )

if( (y%4==0 && y%100 !=0) ||y%400==0 )

5.6

定义函数时使用数组作为形参要加数组下标，在调用函数时实际参数应该是数组名，不加数组下标。有些同学在调用函数时在数组名后加下标以致程序无法运行。

例如：

void reverse(int a[4][4])

{

int b[4][4];

….

}

reverse(a[4][4])————错误！！

应改为：

reverse(a)

一位数组的传递方式（常用）：

**void myFunction(int \*param)**

{}

void myFunction(int param[10])

{}

**void myFunction(int param[])**

{}

二维数组的传递方式（还有其他）：

**void Func(int array[3][10]);**

**void Func(int array[ ][10]);**

因为数组的行数无关紧要，所以还可以写成如下形式：

**void Func(int (\*array)[10]);**

注意\*array需要用括号括起来。这种形式的声明参数是一个指针，它指向具有10个元素的一维数组。因为[]的优先级比\*的优先级高，故\*array必须用括号括起来，否则变成了

**void Func（int \*array[10]);**

这时候参数相当于是声明了一个数组，该数组有10个元素，其中每个元素都是一个指向整型对象的指针。