**计算机程序设计基础（C++)**

**实验报告**

专业班级： 软件工程2404班

学 号： 8209240420

姓 名： 李雅琴

**实验报告成绩：**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **实验** | **实验一** | **实验二** | **实验三** | **实验四** | **实验五** | **总评** |
| **成绩** |  |  |  |  |  |  |

**批阅教师：**

**实验三 函数**

**一、实验目的**

本实验主要培养、训练学生对函数的理解，要求：

1. 掌握函数的定义、声明的方法；

2. 掌握函数的编写要求；

3. 掌握函数的调用方法；

4. 掌握函数参数的传递方法；

5. 掌握变量的作用域；

6. 掌握多文件编程方法。

**二、实验内容与要求**

1、输入自然数m和n，

（1）求他们的最大公约数（或称最大公因数）。

要求输入、输出在主函数中进行，求公约数由函数实现。

1. 在函数中求最大公约数与最小公倍数。（提示：使用引用参数）

2. 编写程序满足：声明一个函数，判断一个整数是否为素数，使用如下函数头：

bool is\_prime(int num) ,如果num是素数函数返回true，否则返回false；

利用函数is\_prime找出前200个素数，并按每行10个输出：

     2     3      5      7    11    13    17    19    23    29

3、编程实现摄氏温度到华氏温度的转换：

编写一个头文件，包含下面两个函数：

double celsius\_to\_fah(double cel)    //摄氏温度到华氏温度

double fahrenheit\_to\_cels(double fah) //华氏温度到摄氏温度

实现头文件，并编写测试程序，调用函数显示如下结果：

Celsius    Fahrenheit   |   Fahrenheit       Celsius

40.0       105.0        |   120.0            48.89

39.0       102.0        |   110.0            43.33

……       ……        |   ……             ……

31.0        87.8        |   30.0             -1.11

（测试程序为主模块，即main( )函数所在的CPP文件，头文件mytemperature.h只有函数声明；函数定义写在另一CPP文件mytemperature.cpp）

4、创建名为mytriangle.h的头文件，包括：

bool is\_valid(double side1,double side2,double side3)

double\_area(double side1,double side2, double side3)

面积=sqrt(s(s-side1)(s-side2)(s-side3))

其中s=(side1+side2+side3)/2

写测试程序：读取三角形三边长，如输入合法，计算面积，否则输出错误信息。

（测试程序为主模块，即main( )函数所在的CPP文件，头文件mytriangle.h只有函数声明；函数定义写在另一CPP文件mytriangle.cpp）

**3与4选一个完成**

5、猴子吃桃：猴子第一天摘若干桃子，当即吃了一半，还不过瘾，又吃了一个。第二天又将剩下的桃子吃掉一半，又多吃一个，以后每天如此，到第10天，发现只剩最后一个桃子，问，第一天猴子共摘多少桃子（用递归实现）。

**三、实验思考题**

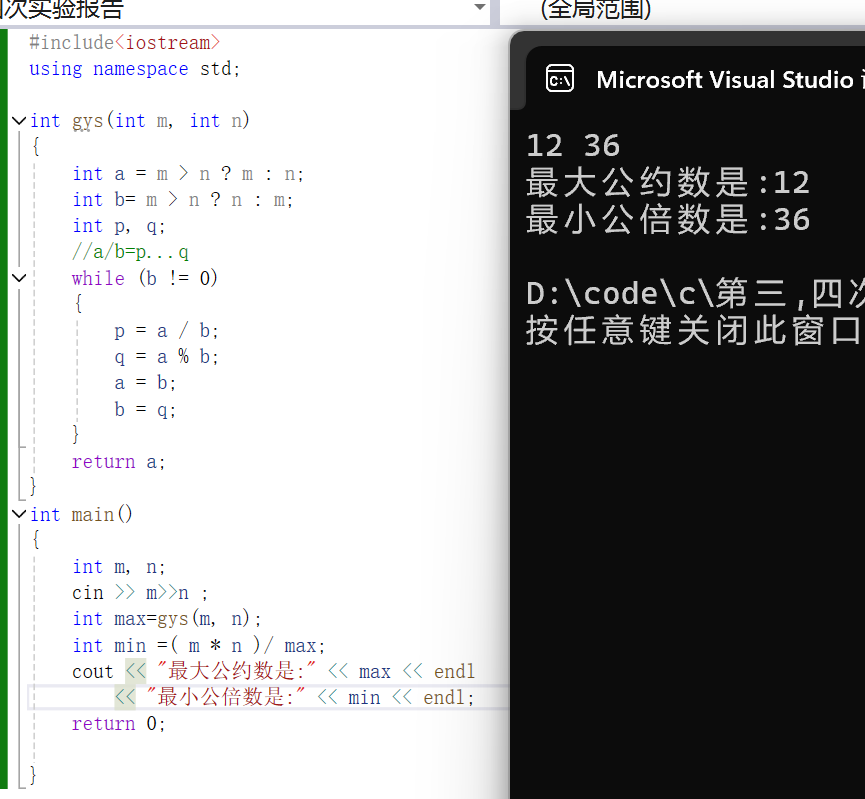
1. 本实验中函数中返回的值为什么与函数类型一致？

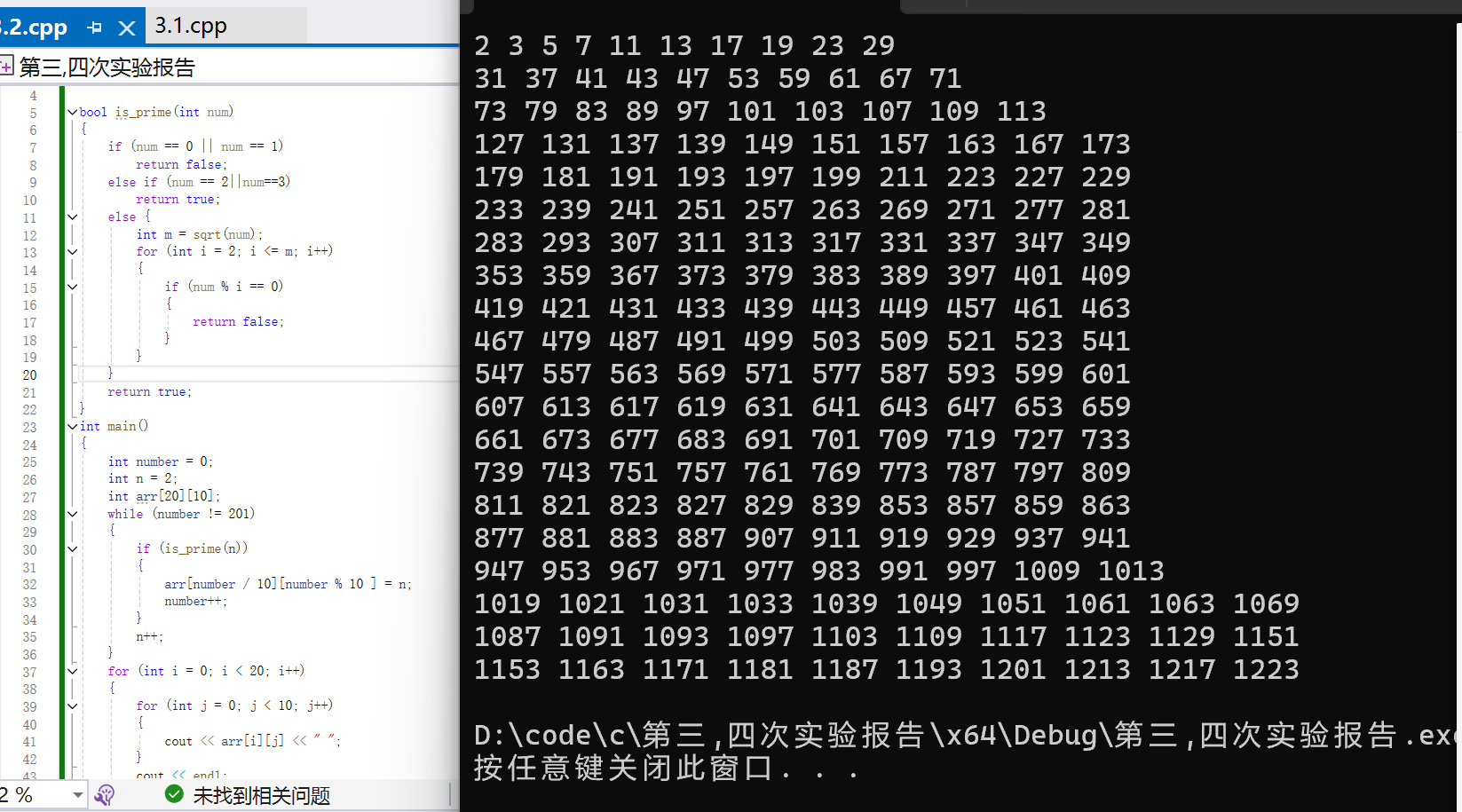
原因:函数声明时指定的类型决定了该函数预期返回值的类型

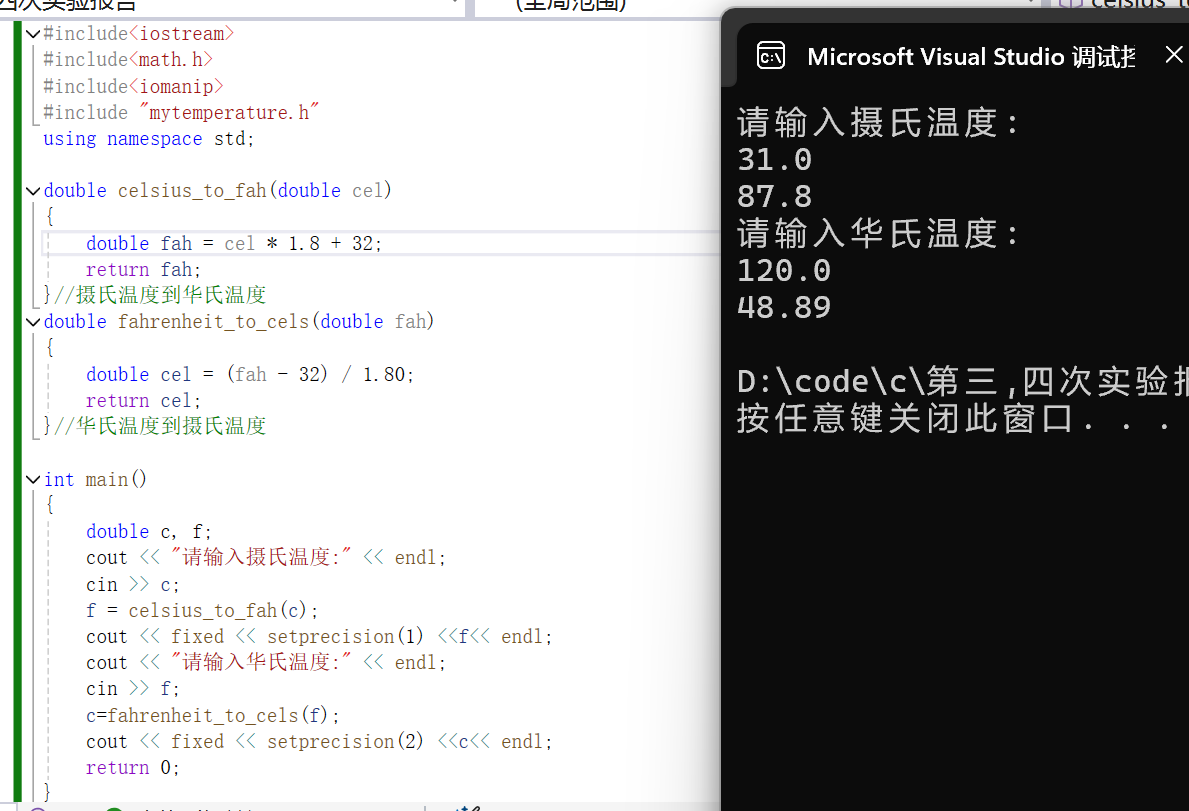
1. 本实验中主函数调用函数时采用的是何种传递方式？

方式:值传递

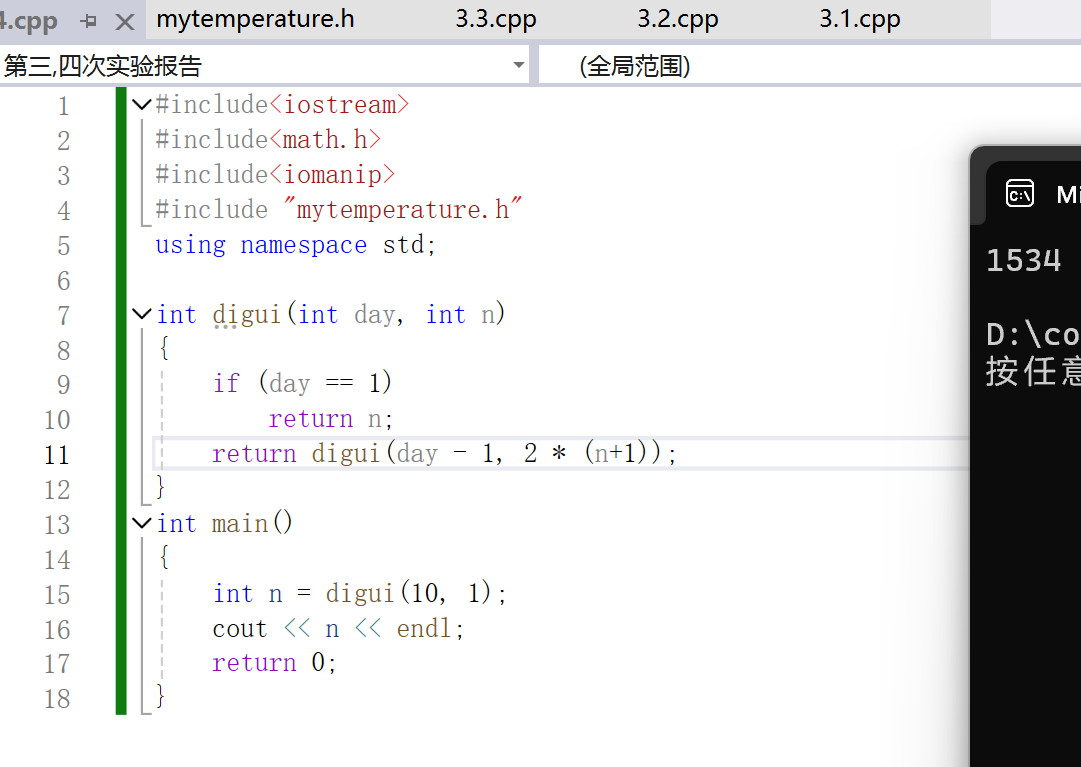
1. **算法分析，程序结果**

**3.1**

**3.2**

**3.3**

**3.5**

****

1. **遇到的问题与解决方法**
2. **问题:最大公约数的算法**

**解决方法:通过网络学习了辗转相除法**

1. **问题:不知道怎么自己写头文件**

**解决方法:通过网络学习了如何自创头文件并且引入**

1. **问题:不理解递归思想**

**解决方法:通过不断测试,理解并正确写出相对应的递归程序**

1. **问题:素数算法偏复杂**

**解决方法:学习了如何优化素数算法**

**六、体会**

《中南大学软件工程专业 C++实验课第三次实验体会》

在本次 C++实验课中，我经历了从困惑迷茫到逐步掌握的过程，收获了许多宝贵的知识与经验，也深刻体会到了自主学习在编程领域的重要性。

实验初期，最大公约数的算法成为了我面临的首个难题。对于如何高效地计算两个数的最大公约数，我最初毫无头绪。但通过在网络上深入学习辗转相除法，我仿佛打开了一扇新的大门。这种基于不断取余和迭代的方法，其简洁性与高效性让我惊叹。从反复琢磨其原理到成功运用代码实现，我不仅掌握了一种实用的算法，更明白了在编程中遇到难题时，积极借助网络资源拓宽知识视野的重要性。

紧接着，创建自己的头文件也给我带来了不小的挑战。不知道从何下手的我，再次求助于网络学习。在这个过程中，我逐渐理解了头文件对于代码模块化和组织性的关键意义。通过精心设计头文件，将函数声明等内容合理放置其中，我感受到了代码结构变得更加清晰有序，这为后续大型项目的开发奠定了良好的基础，也让我意识到良好的代码架构设计是提升编程效率和代码质量的重要环节。

递归思想曾一度像一团迷雾笼罩着我。然而，通过不断地编写测试代码，反复地调试与分析，我逐渐拨开了这团迷雾。从最初的一知半解到能够熟练运用递归解决诸如阶乘计算、斐波那契数列等问题，我深刻体会到递归的强大与精妙之处。它能够以一种简洁而优雅的方式解决一些看似复杂的问题，但同时也需要我们对问题的逻辑和终止条件有极为清晰的把握，这无疑锻炼了我的逻辑思维能力。

最后，面对复杂的素数算法，我没有退缩，而是积极学习优化方法。学习埃拉托斯特尼筛法的过程，让我领略到了算法优化的魅力和价值。通过巧妙地标记和筛选，能够大幅提高素数判断的效率。这使我明白在编程中不能满足于基本的实现，而应不断追求更高效、更优质的解决方案。

总之，本次实验课让我在知识技能和学习方法上都取得了显著的进步。它让我明白，编程学习之路充满挑战，但只要保持积极的学习态度，善于利用各种资源，勇于不断尝试和探索，就一定能够攻克难关，逐步提升自己的编程水平，向着成为一名优秀的软件工程专业人才不断迈进。

**实验四 数组与指针**

【**实验目的**】

1、进一步加深对数组的理解，掌握数组的定义方法；

2、掌握数组的处理方法、数组作为函数参数的使用方法，以及搜索与排序的应用。

3、掌握指针的概念、指针变量定义格式以及指针的运算；

4、掌握指针与数组、函数的关系；

5、理解内存动态分配的含义、熟练掌握内存动态分配方法；

6、掌握递归函数的定义方法。

【实验内容与步骤】

**（一）数组**

1、打印不同的数：

编写一个程序，读入10个数，输出其中不同的数（即如果一个数出现多次，只打印一次）。

提示：读入的数如果是一个新的值，则将其存入一个数组。否则，将其丢弃。输入完毕后，数组中保存的就是不同的数。

下面是一个运行样例：

Enter

Enter ten numbers: 1 2 3 2 1 6 3 4 5 2

The distinct numbers are: 1 2 3 6 4 5

2、起泡排序：

利用起泡排序算法编写一个排序函数。起泡排序算法分若干趟对数组进行处理。每趟处理中，对相邻元素进行比较。若为降序，则交换；否则，保持原顺序。此技术被称为起泡排序（bubble sort）或下沉排序（sinking sort），因为较小的值逐渐地“冒泡”到上部，而较大值逐渐下沉到底部。

算法可描述如下：

bool changed = true;

do

{

changed = false;

for (int j = 0; j < listSize – 1; j++)

if (list[j] > list[j+1])

{

swap list[j] with list[j+1];

changed = true;

}

} while (changed);

很明显，循环结束后，列表变为升序。容易证明do循环最多执行listSize – 1次。

编写测试程序，读入一个含有10个双精度数字的数组，调用函数并显示排列后的数字。

3、游戏：存物柜问题：

一个学校有100个存物柜，100个学生。开学第一天所有存物柜都是关闭的。第一个学生（记为S1）来到学校后，打开所有的存物柜。第二个学生S2，从第二个存物柜（记为L2）开始，每隔两个存物柜，将它们关闭。第三个学生S3从第三个存物柜L3开始，每隔三个，将它们的状态改变（开着的关上，关着的打开）。学生S4，从L4开始，每隔四个改变它们的状态。学生S5，从L5开始，每隔五个改变状态。依此类推，直至学生S100改变L100的状态。

当所有学生完成这个过程，那些存物柜是开着的？编写一个程序求解此问题，显示所有开着的柜子号码，号码之间用一个空格隔开。

提示：使用一个100个布尔型元素的数组，每个元素代表存物柜是开（true）或关（false）。最初所有的储物柜都是关闭的。

4、合并两个排列好的数组：

编写如下函数，合并两个排列好的数组，形成一个新的排列好的数组。

void merge(const int list1[], int size1, const int list2[], int size2, int list3[])

使用size1+size2次比较实现函数。编写测试程序，提示用户输入两个排列好的数组，并显示合并以后的数组。下面是一个运行样例。注意，输入数据的第一个数字是数组的元素数，而不是数组的一部分。假定数组大小不超过80。

Enter

Enter

Enter list1: 5 1 5 16 61 111

Enter list1: 4 2 4 5 6

The merged list is 1 2 4 5 5 6 16 61 111

5、检验子串：

编写如下函数，检验C字符串s1是否是C字符串s2的子串。如果匹配，返回s1在s2中的下标，否则返回–1。

int indexOf(const char s1[], const char s2[])

编写测试程序，读入两个C字符串，检验C字符串s1是否是C字符串s2的子串。下面是程序的运行样例：

Enter

Enter

Enter

Enter the first string: welcome

Enter the second string: We welcome you!

indexOf(“welcome”, “We welcome you!”) is 3

Enter

Enter the first string: welcome

Enter the second string: We invite you!

indexOf(“welcome”, “We invite you!”) is –1

6、字符串中每个字母出现的次数：

请使用如下函数头编写函数，数出字符串中每个字母出现的次数。

void count(const char s[], int counts[])

counts是一个有26个元素的整数数组。const[0]，const[1]，…，const[25]分别记录a，b，…，z出现的次数。字母不分大小写，例如字母A和字母a都被看作a。

编写测试程序，读入字符串并调用count函数，显示非零的次数。下面是程序的一个运行样例：

Enter

Enter a string: Welcome to New York!

c: 1 times

e: 3 times

k: 1 times

l: 1 times

m: 1 times

n: 1 times

o: 3 times

r: 1 times

t: 1 times

w: 2 times

y: 1 times

**（二）指针**

1、上机验证下列程序的运行结果（有错误的话自己补充完善）

(1) void main()

｛

int i,j,\*pi,\*pj; //此处的\*表示定义指针变量，而非间接运算符

pi=&i;

pj=&j;

i=5;j=7;

cout<<i<<’\t’<<j<<’\t’<<pi<<’\t’<<pj;

cout<<&i<<’\t’<<\*&i<<’\t’<<&j<<’\t’<<\*&j;

}

运行结果：

上述结果中，pi与&i,pj与&j是地址值，随编译程序而变化，不确定。

(2) int main() //C语言程序，要了解

{

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

int \*p,i;

p=a; //将数组a首地址送给p

for (i=0;i<3;i++)

printf("%d,%d,%d,%d\n",a[i],p[i],\*(p+i),\*(a+i)); //与cout功能差不多

}

运行结果：

1,1,1,1

2,2,2,2

3,3,3,3

通过这两道题目，希望学生掌握数组元素与指向数组的指针的不同。

a[i]表示数组中下标为i的元素。

a[i]←p[i]←\*(p+i)←\*(a+i)

a是数组名，表示数组首地址，(p+i)表示数组中第i个元素的地址，\*(p+i) 相当于a[i]。

(3)通过如下的问题理解递归函数的定义与调用（递归未讲，可以后做）

//#include “stdio.h”

void f(char \*st,int i)

{

st[i]=’\0’;

cout<<st; // printf(“%s\n”,st);

if (i>1) f(st,i-1);

}

void main()

{

char st[]=”abcd”;

f(st,4);

}

补充完整，运行时输出为\_\_\_abcd abc ab a\_\_\_\_\_

(4)下面程序的主函数中能保证p[0]输出1，p[1]输出2吗？如何修改以保证之（提示：在函数f中使用new生成动态数组；在main中用delete释放。）

#include<iostream>

using namespace std;

int \*f()

{

int list[]={1,2,3,4};

return list;

}

void main()

{

int \*p=f();

cout<<p[0]<<endl;

cout<<p[1]<<endl;

}

2、程序设计

(1)编写函数检查字符串s1是否为字符串s2的子串，若是，返回第一次匹配的下标，否则返回-1。在主程序中输入字符串s1与s2，调用函数实现。

函数原型：int indexof(const char \*s1,const char \*s2);

(2)编写一个函数将以字符串形式表示的一个16进制数转换为10进制数，并在主函数中测试。函数原型 int parseHex(const char \*const hexString);

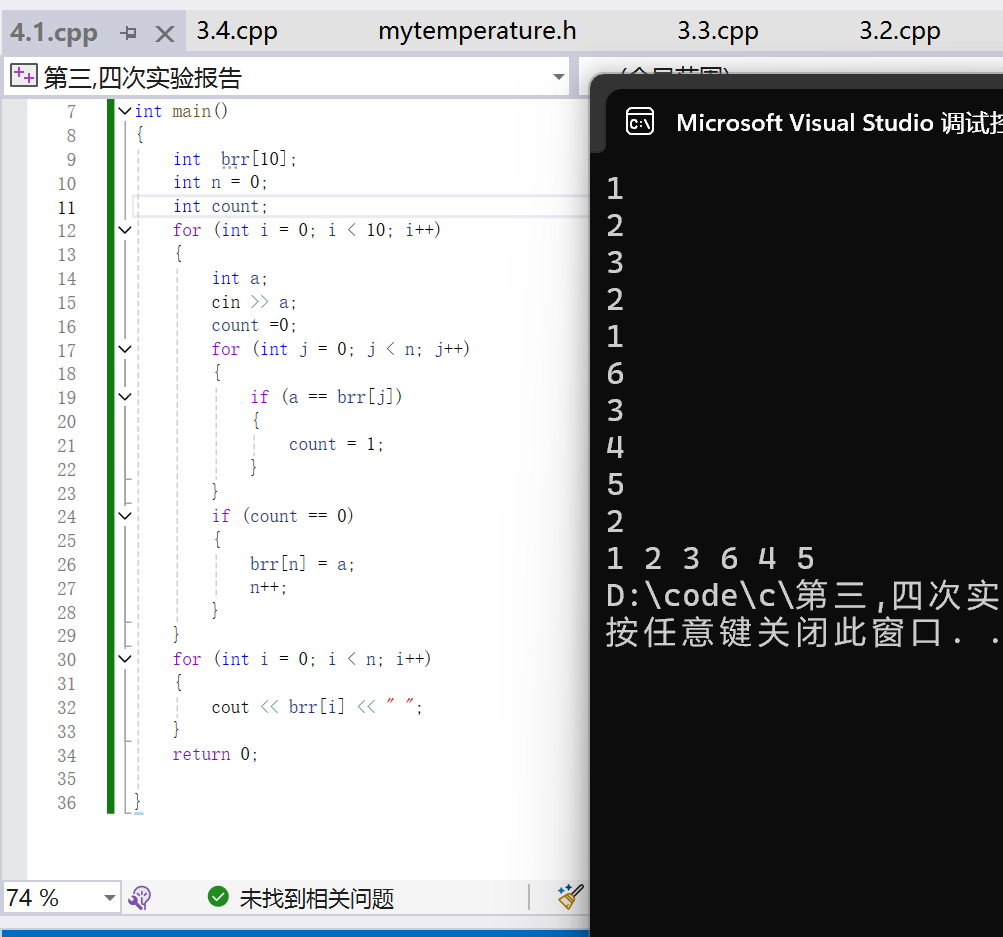
如：调用函数 parseHex(“A5”);返回165

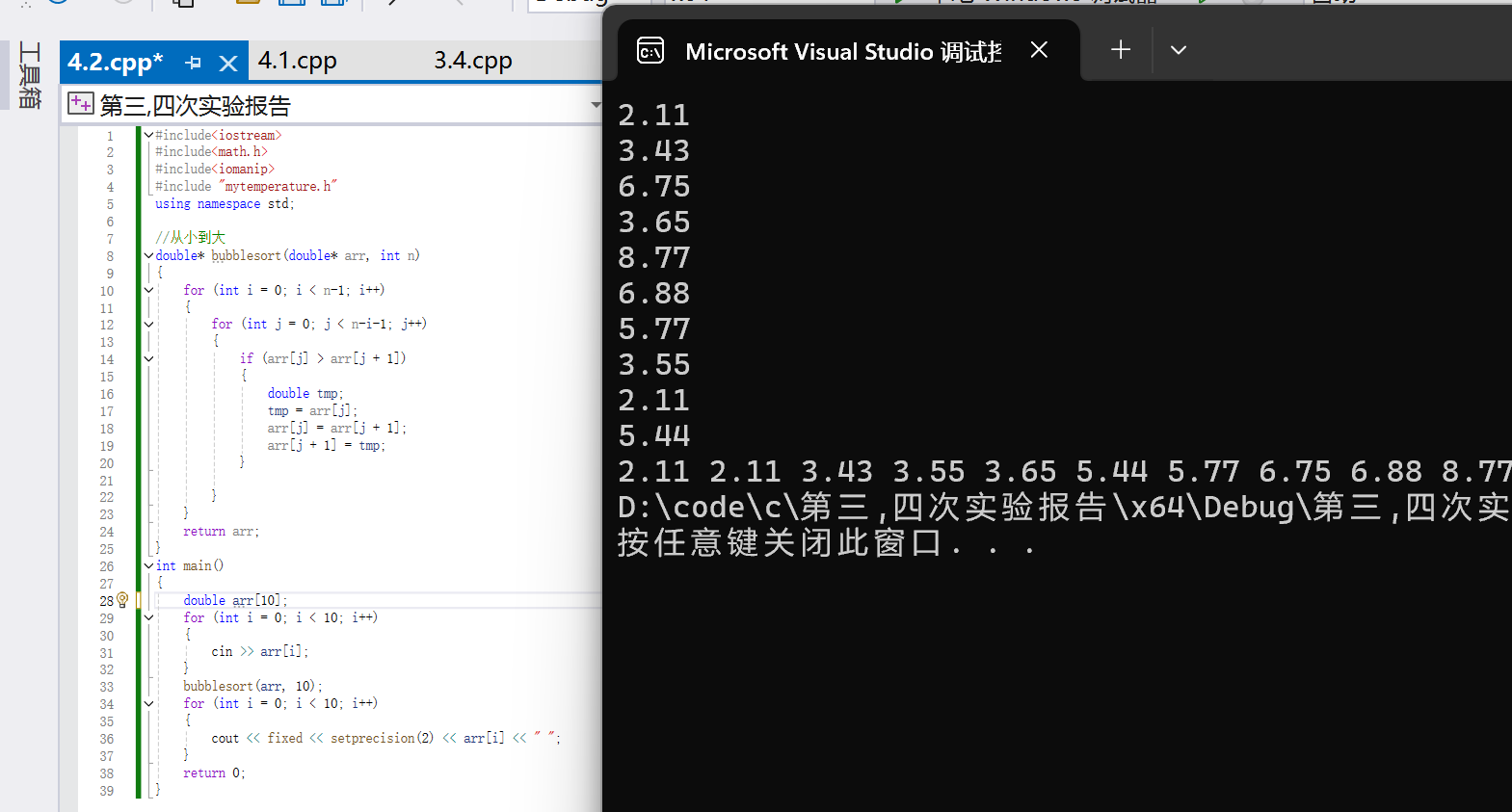
1. 主程序中建立一动态数组（使用new），数组元素及元素个数由键盘输入，动态调试观察指针及指针指向的内容；设计一个函数对数组由小到大排序；主程序中用指针方式输出数组元素；最后释放数组内存（delete）。

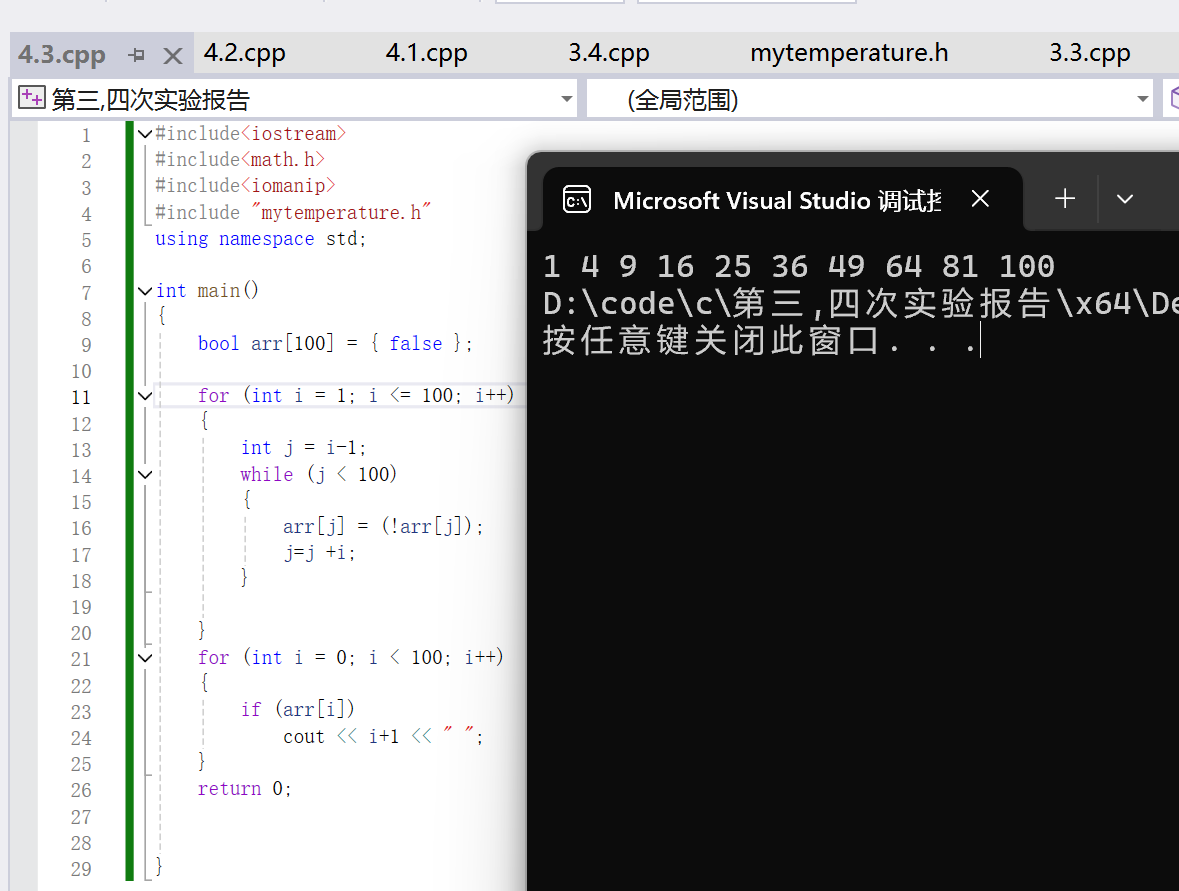
【完成实验报告】

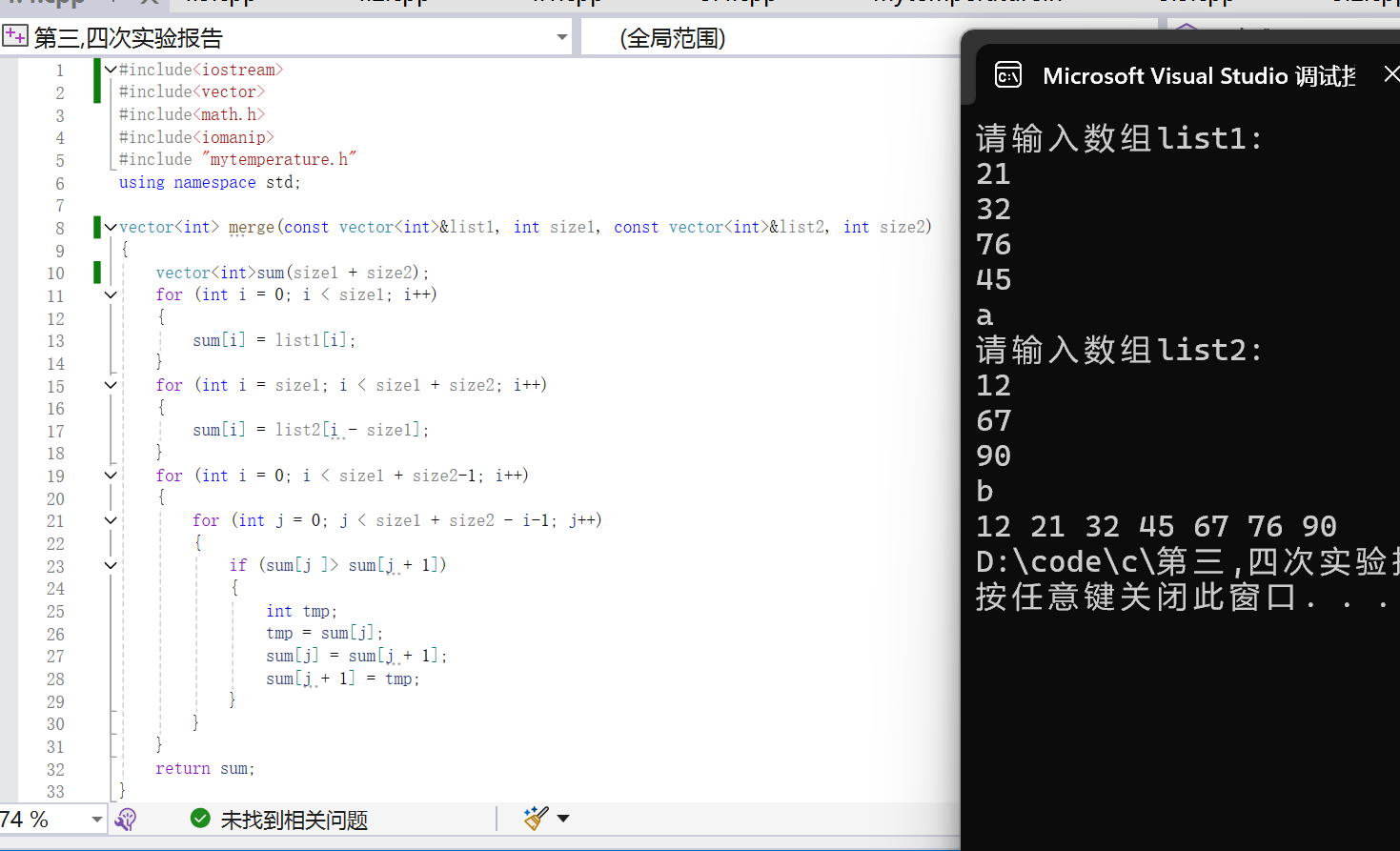
**实验报告只要求写程序设计部分**

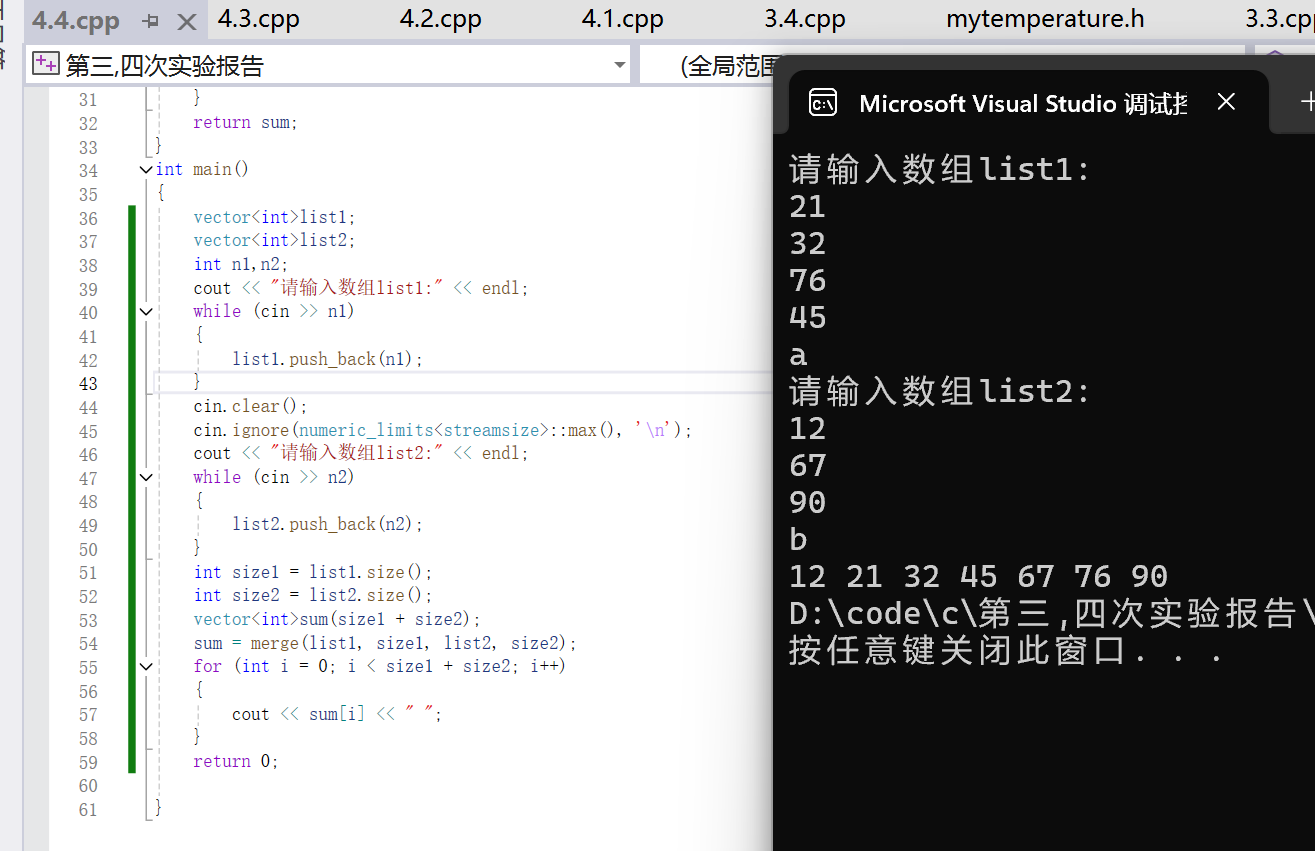
**三、算法分析，程序结果**

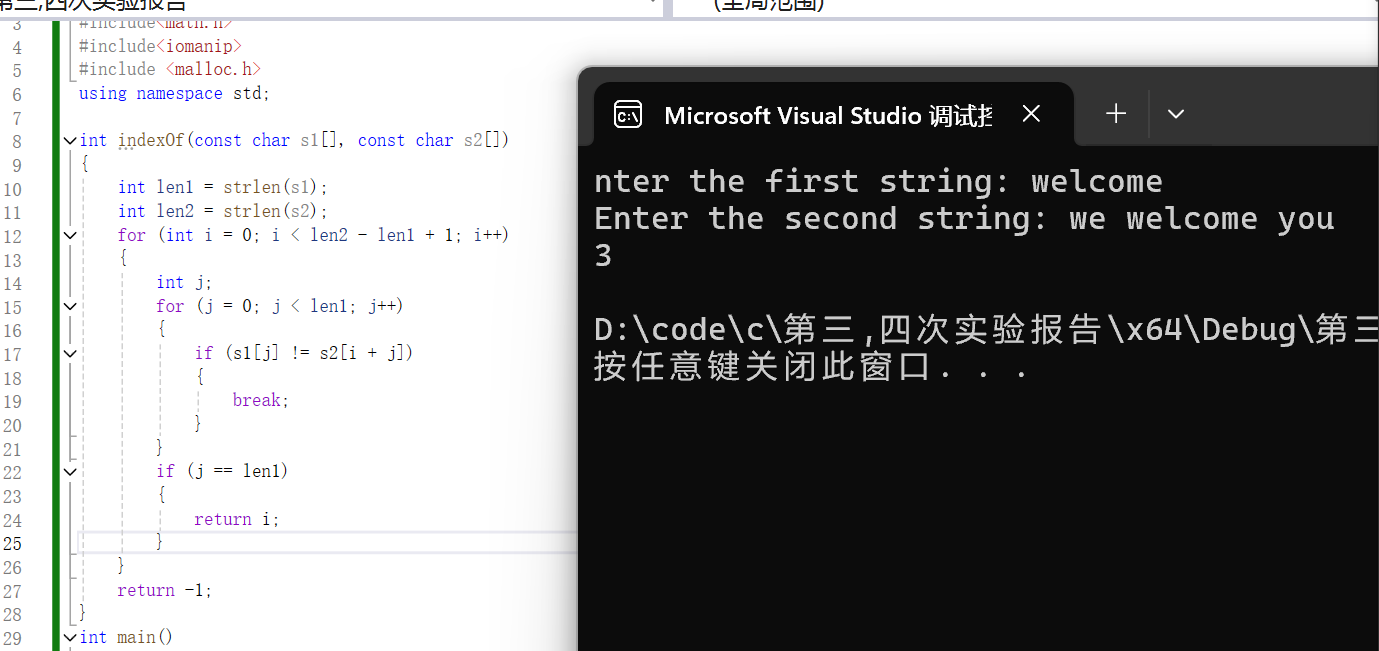
**4.1 **

**4.2**

**4.3**

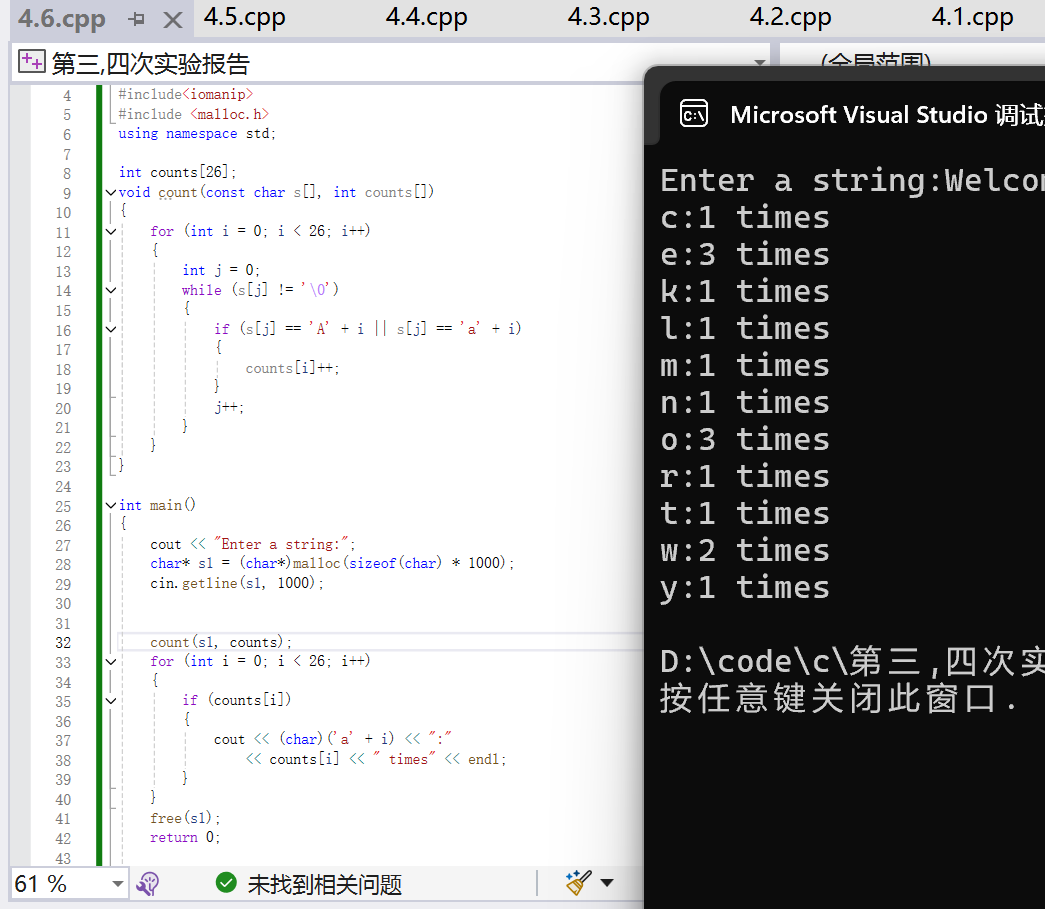
**4.4**

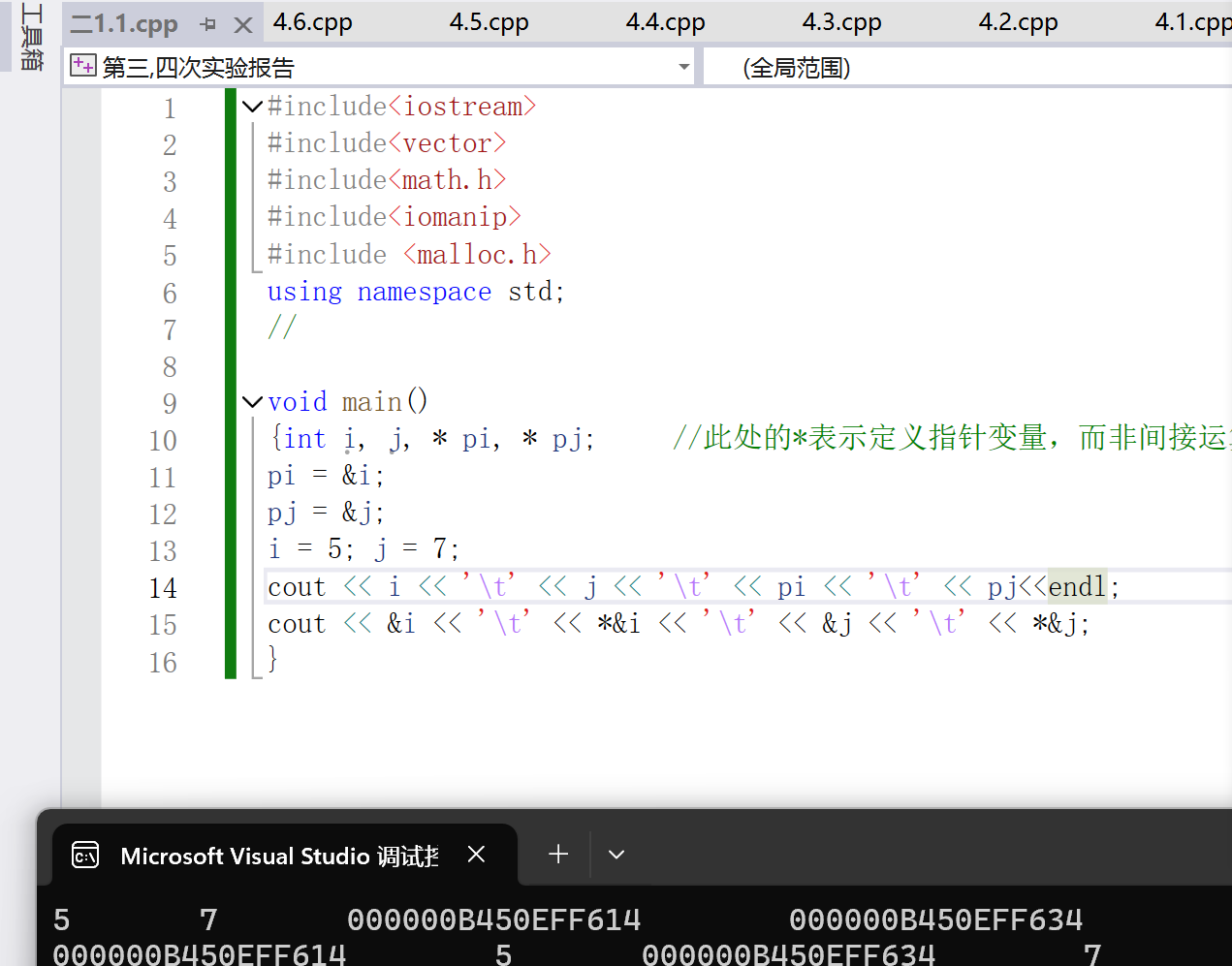
****

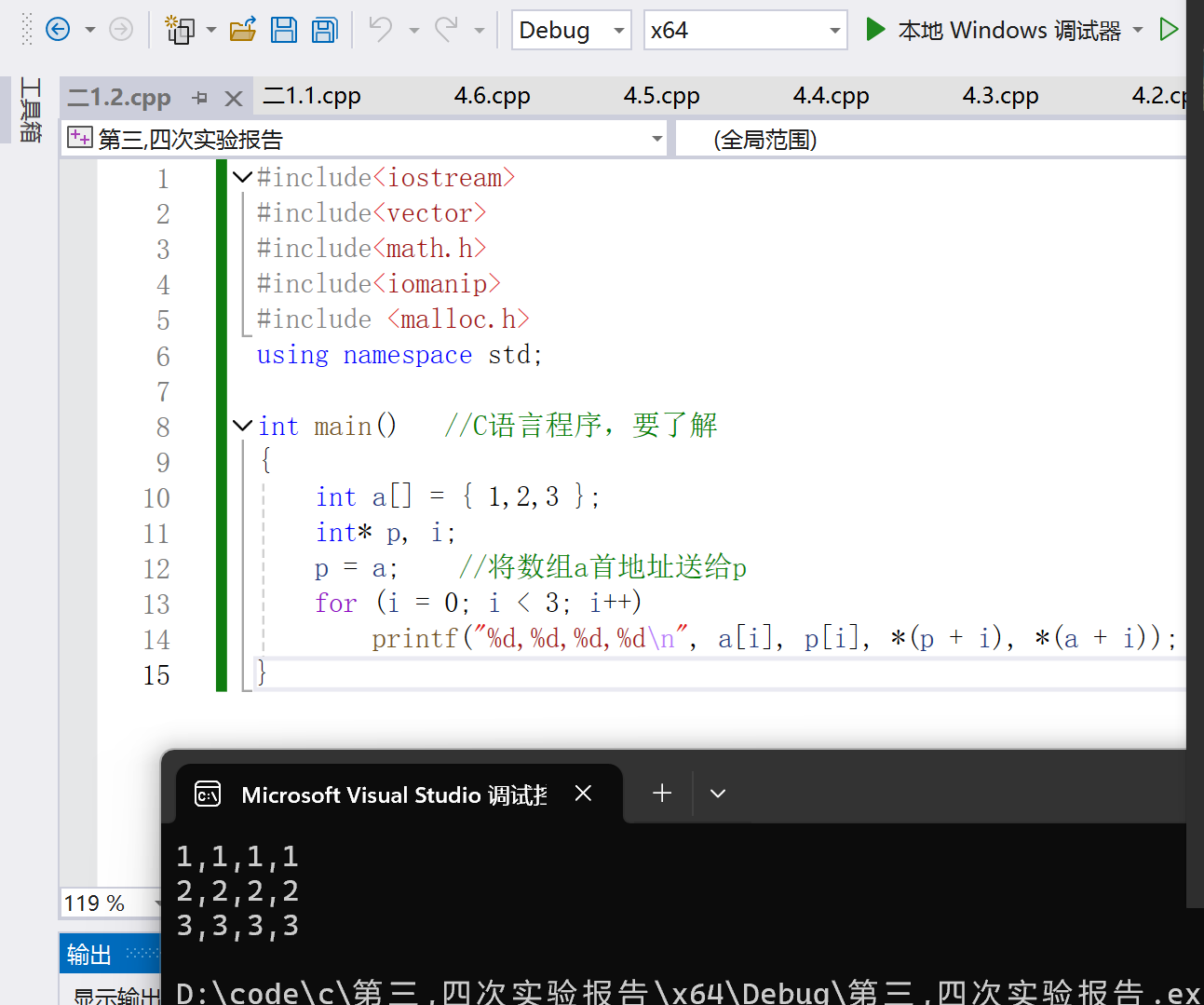
**4.5**

****

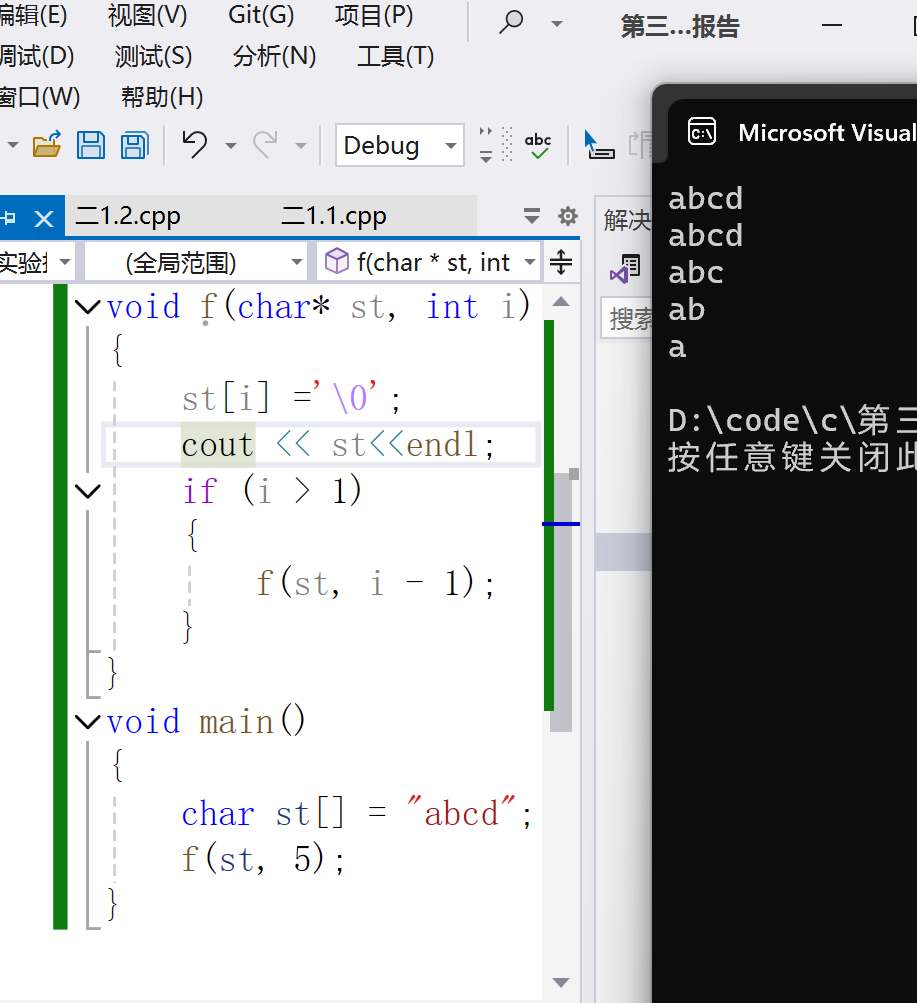
**4.6**

****

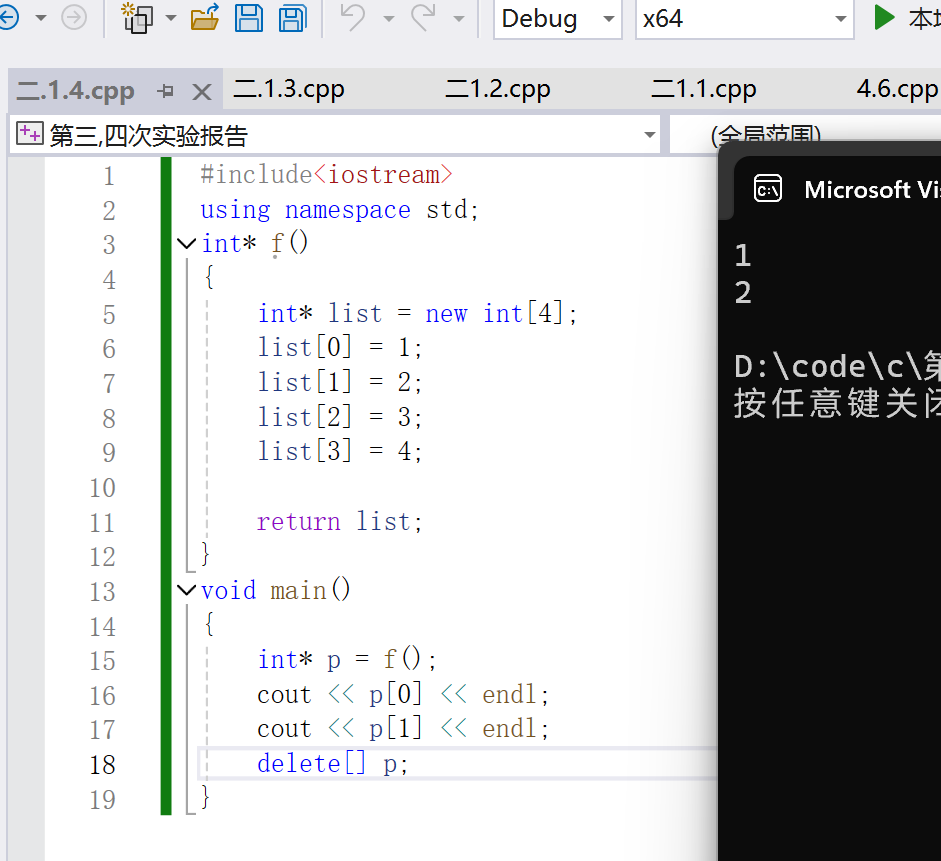
1. **二1.1**

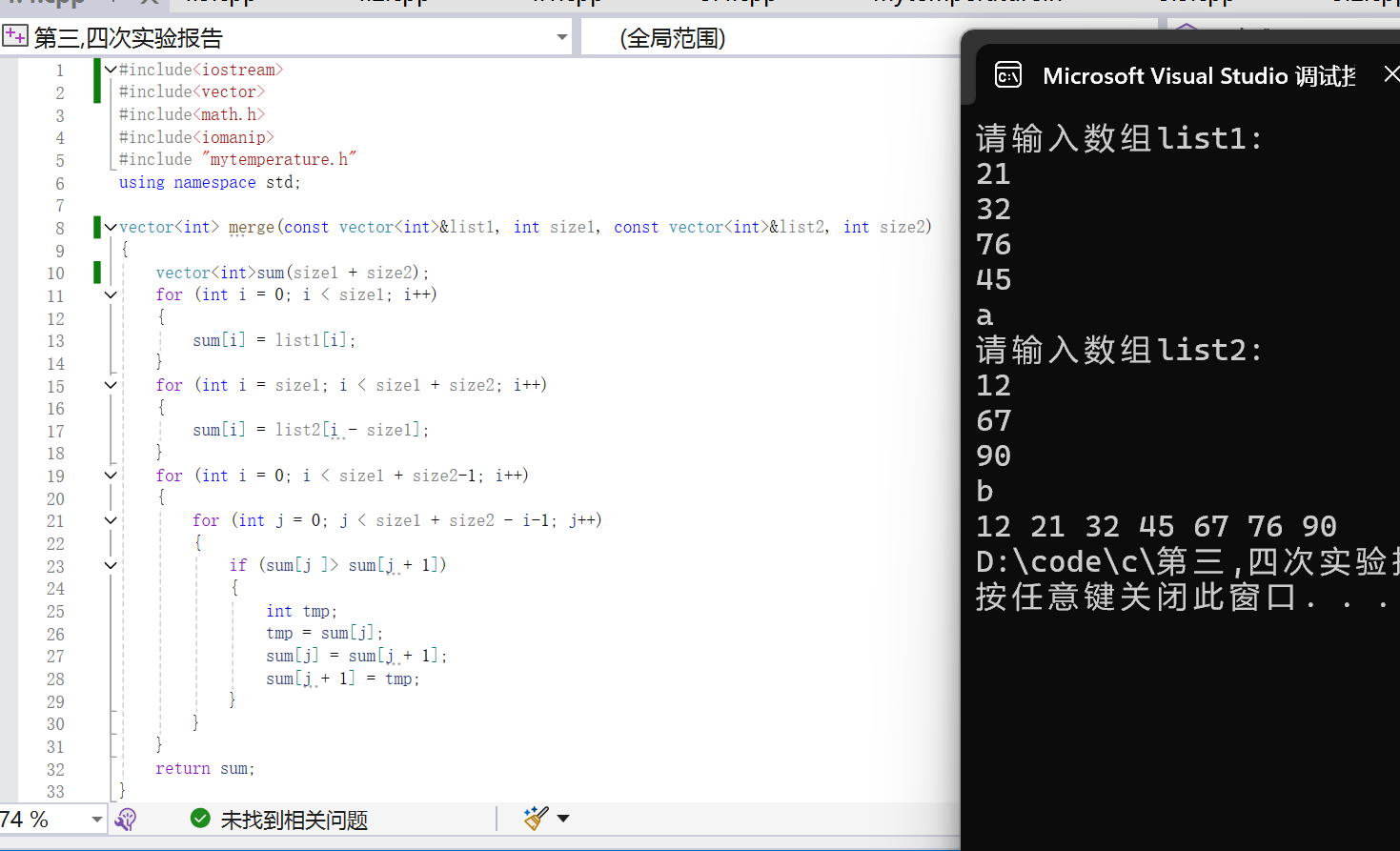
**4.二1.2**

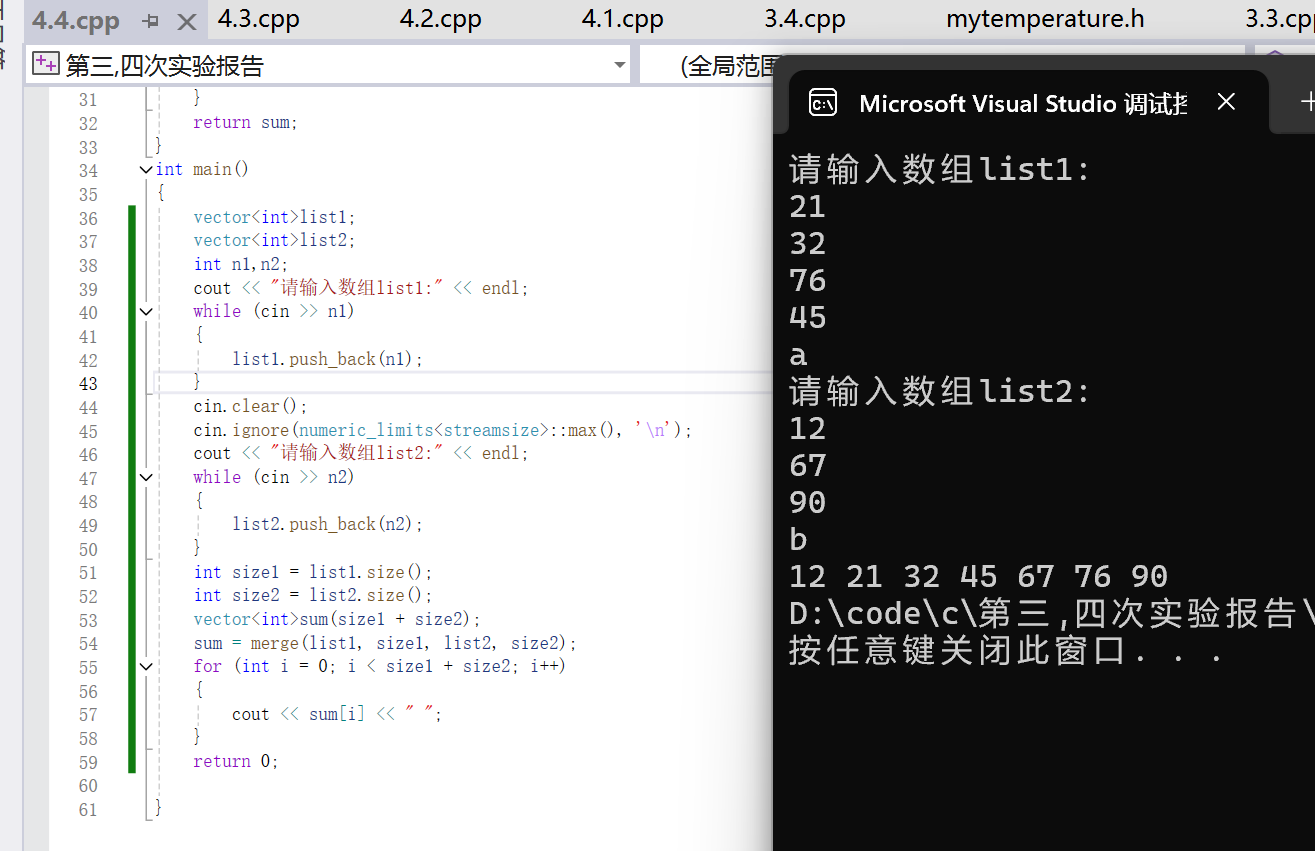
**4.二1.3**

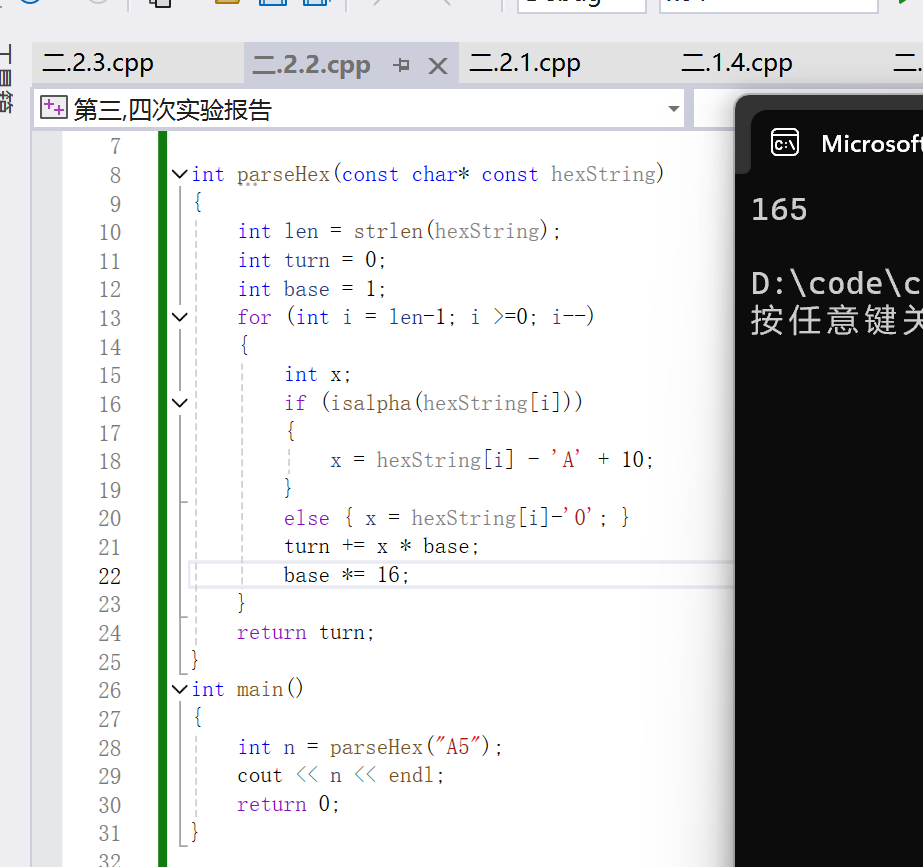
****

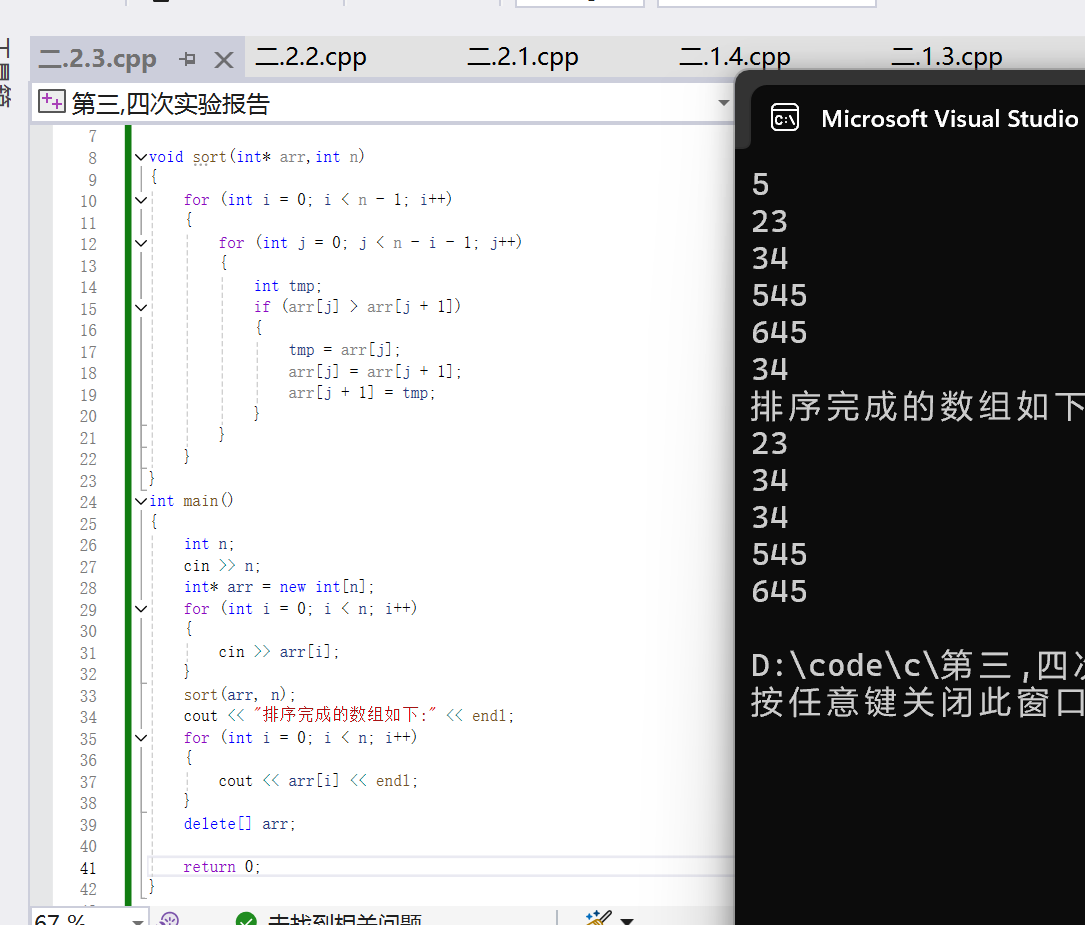
**4.二1.4**

****

**4.二2.1**

****

**4.二2.2**

**4.二2.3**

**四、遇到的问题与解决方法**

1. **问题:不知道排序问题如何完成**

**解决方法:通过网络和老师注解学会了冒泡排序**

1. **问题:不知道如何设定储物柜状态改变**

**解决方法:思考后知道用!直接解决**

1. **问题:不知道如何在不知道长度的情况下输入两组数组**

**解决方法:通过学会了动态数组的写法,包括malloc和new**

1. **问题:不太清楚指针的具体用法**

**解决方法:通过编写程序,不断深刻了解指针的使用和意义**

1. **问题:不明白十六进制如何转化,以及如何表示16次方**

**解决方法:通过网络找到了pow函数**

1. **体会**

《中南大学软件工程专业 C++实验课第四次实验体会》

在此次 C++实验课中，我经历了一场知识与技能的深度磨砺，诸多挑战接踵而至，但在不断探索与学习的过程中，我收获了满满的成长与感悟。

排序问题曾让我陷入困境，面对杂乱无章的数据序列，我茫然不知如何使其有序。幸得网络资源的丰富宝藏以及老师耐心的注解，我接触到了冒泡排序算法。从最初对其原理的懵懂，到逐步理解每一轮比较与交换的精妙之处，再到亲手将代码敲下并成功运行，那一刻，我真切地体会到了算法的魅力与力量。这不仅解决了眼前的排序难题，更让我对程序设计中数据处理的逻辑有了更深刻的认知。

储物柜状态改变的问题看似简单，却也曾让我绞尽脑汁。在苦思冥想之后，突然发现一个简单的逻辑运算符“!”就能巧妙地实现状态的反转，这让我意识到在编程中，有时候解决问题的关键就隐藏在这些基础而又容易被忽视的知识点里。这也提醒我在今后的学习中要更加注重对基础知识的深入理解和灵活运用。

在处理两组未知长度的数组输入时，动态数组成为了我的得力助手。学习 malloc 和 new 的过程充满了挑战与探索，从理解它们如何在内存中开辟空间，到正确地分配和释放内存，每一步都需要小心翼翼。但当我成功地运用动态数组解决了数组长度不确定的问题，实现了数据的有效存储和处理时，我感受到了自己在内存管理和数据结构运用方面的巨大进步。

指针的使用一直是 C++学习中的重难点，起初我对其感到十分困惑，不清楚它在内存中的具体指向和操作方式。然而，通过不断地编写程序，在实践中反复尝试指针的各种用法，我逐渐拨开了指针的神秘面纱。我开始理解指针如何高效地访问和修改数据，以及它在函数参数传递、数组处理等方面的独特作用。这一过程不仅加深了我对 C++语言机制的理解，更培养了我严谨的编程思维和内存管理意识。

十六进制转化及 16 次方表示的问题也曾困扰着我，网络上的 pow 函数如同及时雨般出现。在探索如何运用该函数进行十六进制转换的过程中，我对不同进制之间的关系以及数学函数在编程中的应用有了更深入的理解。

这次实验课犹如一场知识的盛宴，让我在遇到问题、解决问题的循环中不断成长。它让我深刻认识到，编程学习没有捷径可走，唯有不断地探索、学习、实践，才能在面对各种复杂问题时游刃有余。每一个问题都是一次成长的契机，每一次解决都是一次技能的提升。我将带着这份对知识的渴望和对编程的热情，继续在软件工程的学习道路上奋勇前行，努力成为一名更优秀的专业人才。