**计算机程序设计基础（C++)**

**实验报告**

专业班级： 软件工程2402

学 号： 8209240223

姓 名： 王仪杰

**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？

2. 本实验中主函数调用函数时采用的是何种传递方式？

1. **算法分析，程序结果**

**1.**#include<iostream>

using namespace std;

int GCD(int& m, int& n)//定义最大公约数函数并使用引用作为参数，这样可以操控用户输入的m,n

{

int temp = 0;

int a = 0;//设置变量并赋初值

for (temp = 1; temp <= m && temp <= n; temp++)//设置循环遍历1到m或n中较大的一个

{

if (m % temp == 0 && n % temp == 0)//判断是否为m,n的公约数

{

a = temp;//把值赋给a，这样经过循环便可以找到最大的a

}

else

{

};

}

return a;

}

int LCM(int& m, int& n)//设置最小公倍数函数

{

int temp1 = 0;

int a1 = 0;

for (temp1 = 1; temp1 <= m \* n; temp1++)//注意这里不用限制tenp1大于谁，不然会直接跳出循环

{

if (temp1 % m == 0 && temp1 % n == 0)

{

a1 = temp1;

break;//直接跳出循环

}

else;

}

return a1;

}

int main()

{

int m = 0;//设置变量

int n = 0;

cout << "请输入m,n的值" << endl;

cin >> m >> n;

int x, y;

x = GCD(m, n);//对函数进行调用

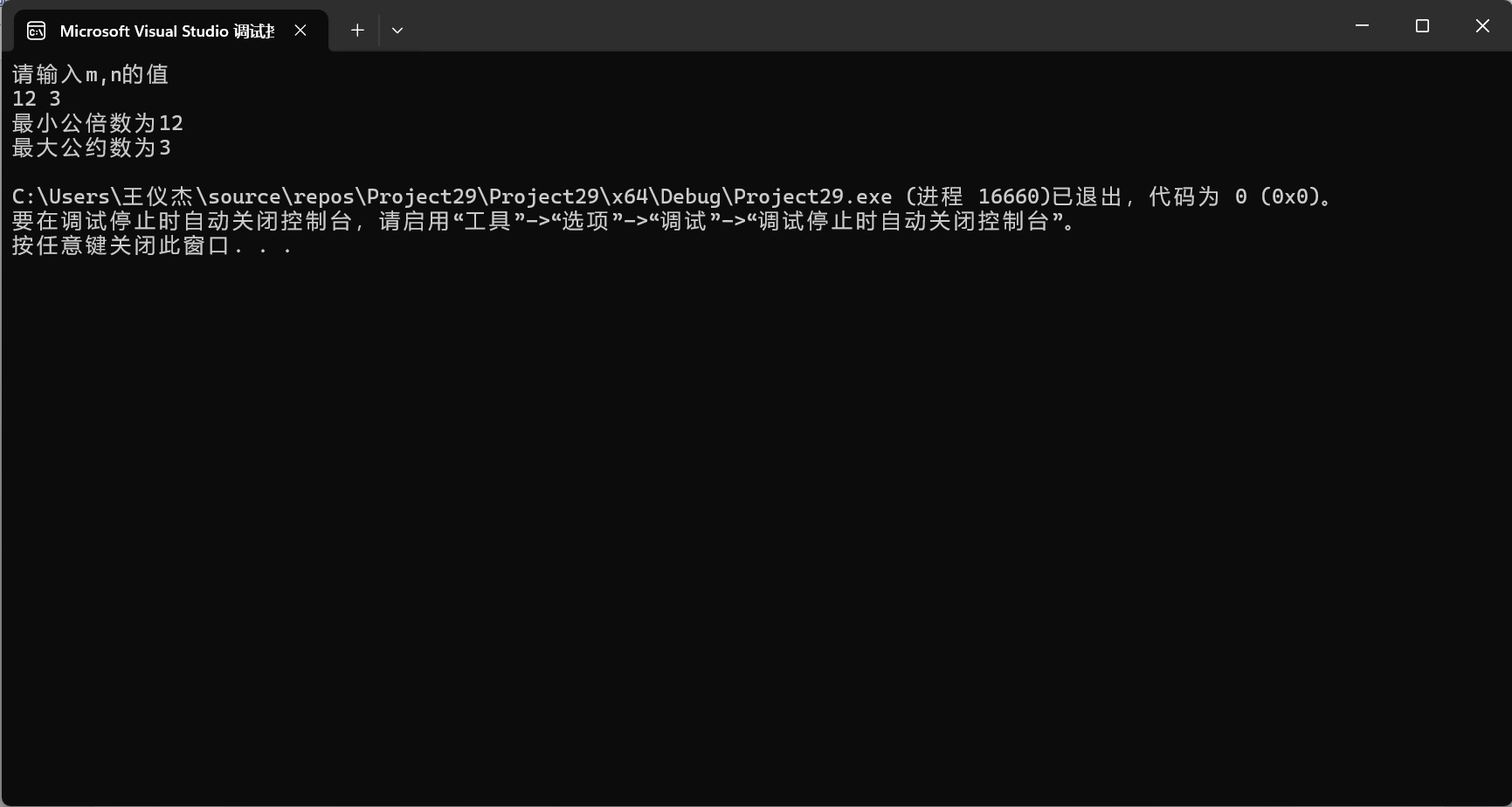
y = LCM(m, n);//对函数进行调用

cout << "最小公倍数为" << y << endl;//对函数进行输出

cout << "最大公约数为" << x << endl;

return 0;

}



**2.**

#include<iostream>

using namespace std;

bool is\_prime(int num)//定义求解素数的函数

{

int p = 0;//引入p变量

if (num == 2)//先把2整出来，因为之后的循环中无法写出包含2的条件

{

p = 1;//返回 true

}

else

{

for (int i = 2; i < num; i++)//因为1不是质数所以这里的i必须要从2开始

{

if (num % i == 0)//因为2是质数所以如果不把2单独写出来2/2==0，会被判断为不是质数

{

p = 0;

break;

}

else

{

p = 1;

}

}

return p;

}

}

int main()

{

int l = 0;//引入l作为确定个数的变量

for (int a = 1; l <= 200; a++)//设置循环结束条件为l<200,即质数的个数为200时结束

{

int x = is\_prime(a);

if (x != 0)

{

cout << a << " ";

l++;//找到一个就把确定质数的变量加一

if (l % 10 == 0)//这个是确定每十个换一行

{

cout << endl;//之前出错是因为把这个if else写到下面的else中去了

}

else

{

}

}

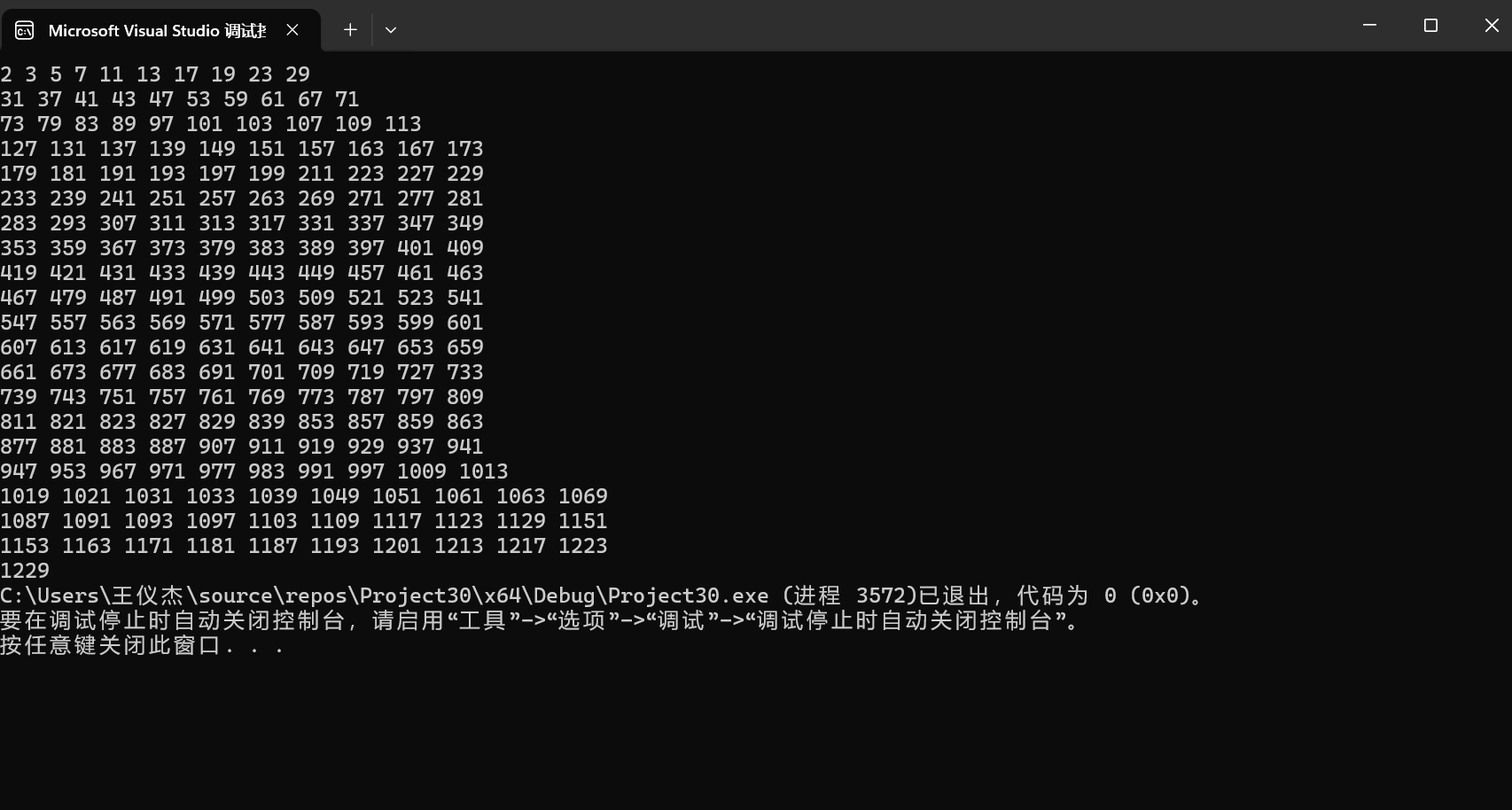
else

{

}

}

return 0;



}

**3.**

**.h的头文件：**

#include<iostream>//这是创建了一个.h的头文件，在其中写了基本的框架

using namespace std;

double celsius\_to\_fah(double cel);//然后写上函数声明，即定义函数时最开头那一行

double fahrenheit\_to\_cels(double fah);

**写函数定义的源文件：**

#include"mytemperature.h"

double celsius\_to\_fah(double cel)//在这个分文件中写了函数的定义

{

double fah = 0;

fah = 32 + cel \* 1.8;

return fah;

}

double fahrenheit\_to\_cels(double fah)

{

double cel;

cel = (fah - 32) / 1.8;

return cel;

}

**调用函数的源文件：**

#include<iostream>

#include<iomanip>//保留小数时使用的头文件

using namespace std;

#include"mytemperature.h"//这个分文件中进行函数的调用

int main()

{

double choose = 0;

double temp = 0;

cout << "请选择转换方式" << endl;

cout << "1.摄氏温度转华氏温度" << endl;//选择切换的方式

cout << "2.华氏温度转摄氏温度" << endl;

cin >> choose;

cout << "请输入温度" << endl;

cin >> temp;

if (choose == 1)

{

double fah = celsius\_to\_fah(temp);

cout <<fixed<<setprecision(2)<< fah << endl;

}

else if (choose == 2)

{

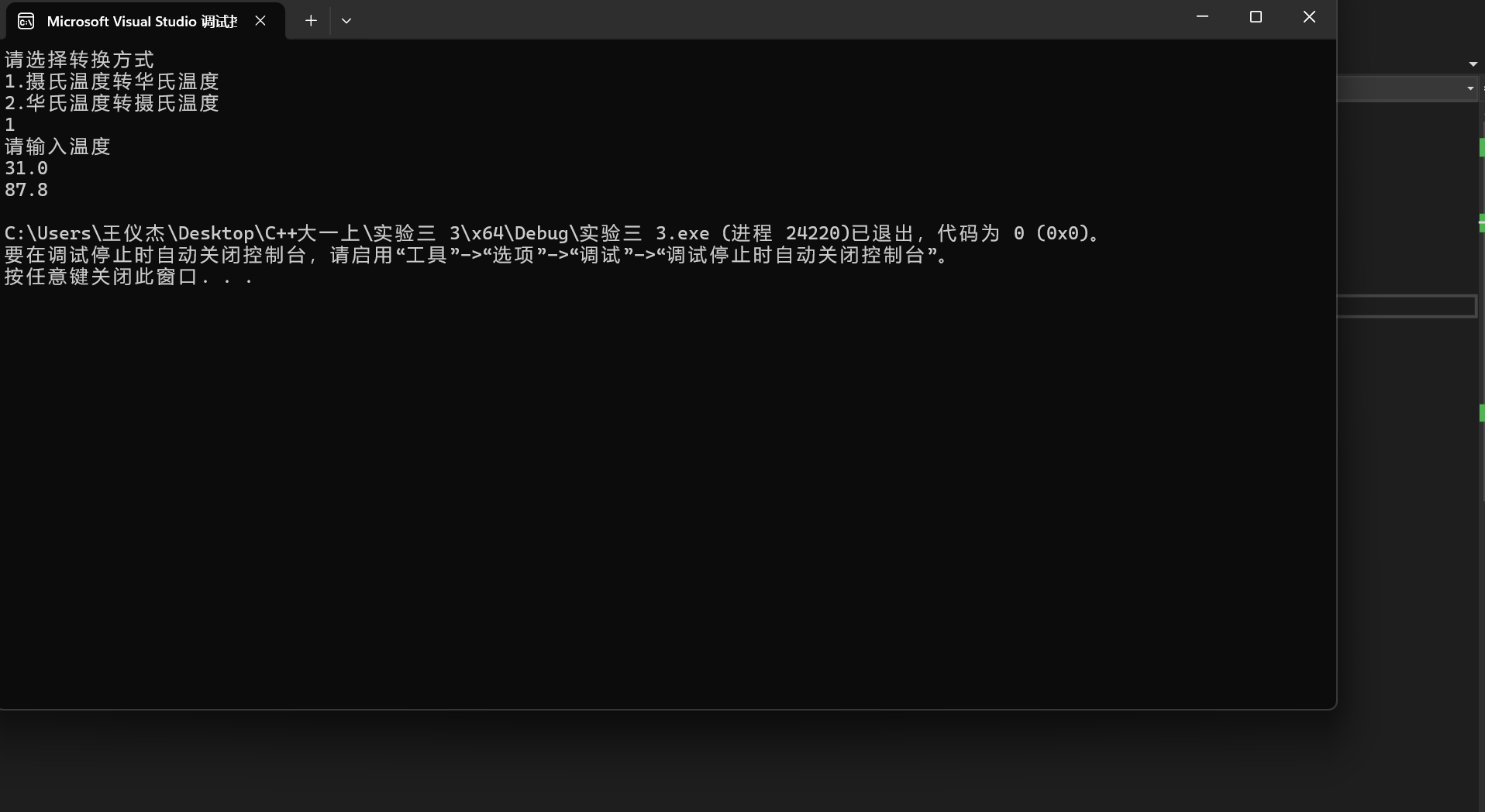
double cel = fahrenheit\_to\_cels(temp);

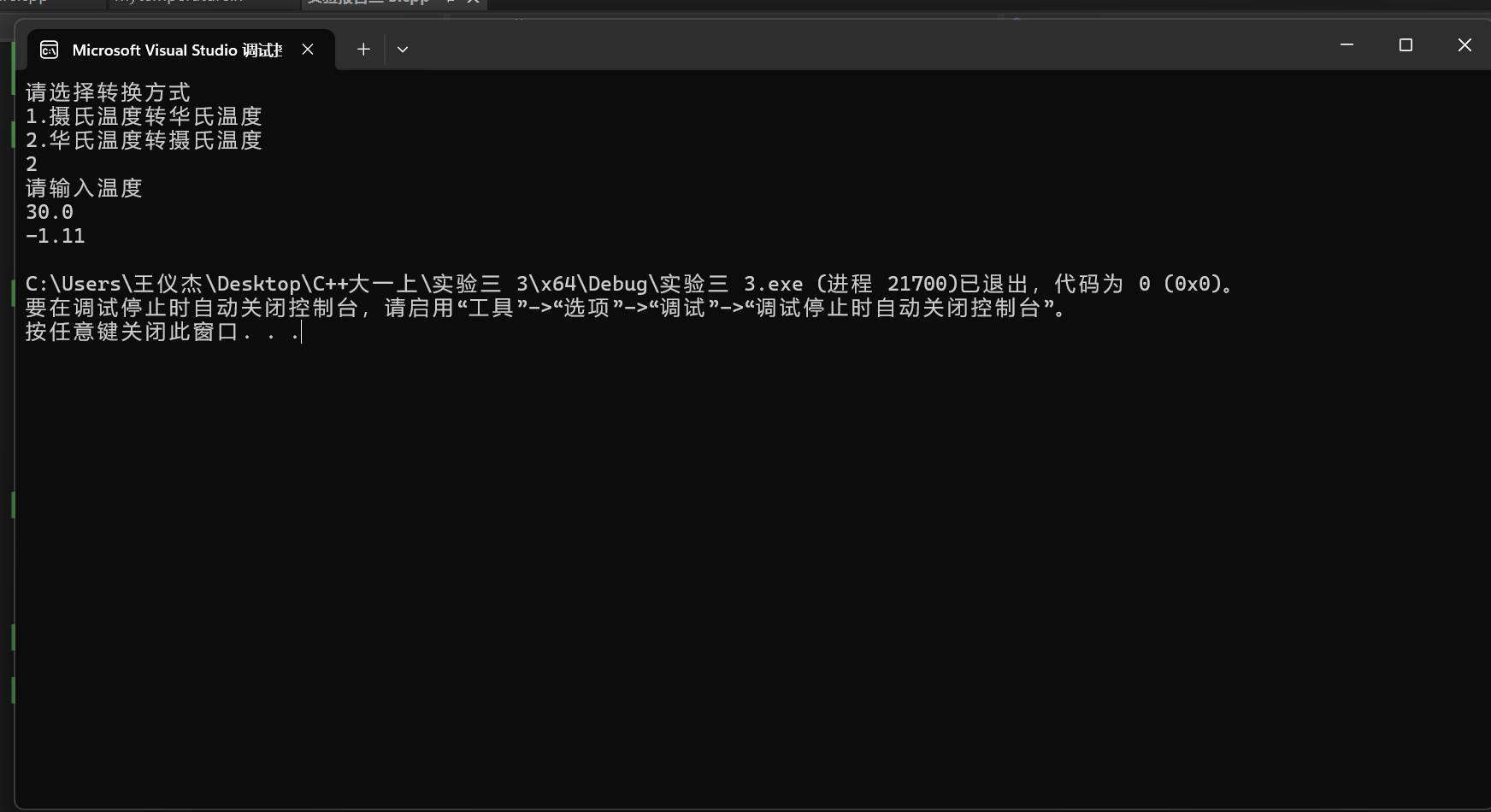
cout <<fixed<<setprecision(2)<< cel << endl;

}

else;//这样便可实现函数分文件的编写

}



****

**5.**#include<iostream>

using namespace std;

int peaches(int n)//n代表天数

{

if (n == 10)//最后一天单独写，因为不存在下一天

{

return 1;

}

else

{

return 2 \*( peaches(n + 1)+1);//在这里写递推公式即可，后一天的加一再乘2就是前一天的

}

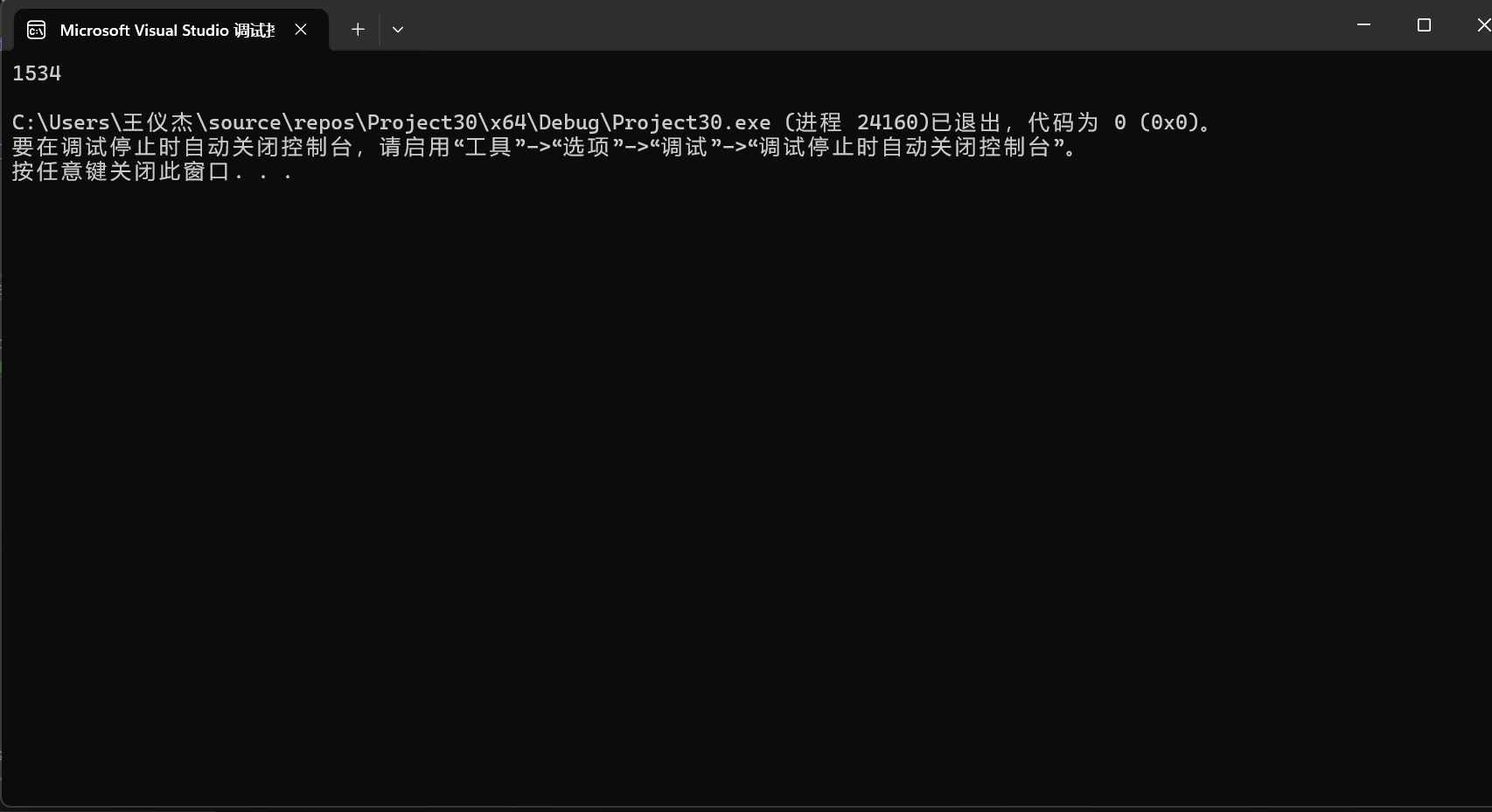
}

int main()

{

cout << peaches(1) << endl;//把初始的数据传入即可

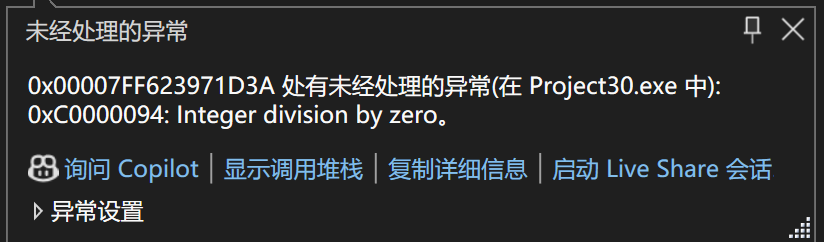
}//递归相关的问题时先考虑递归公式然后在函数中调用函数即可



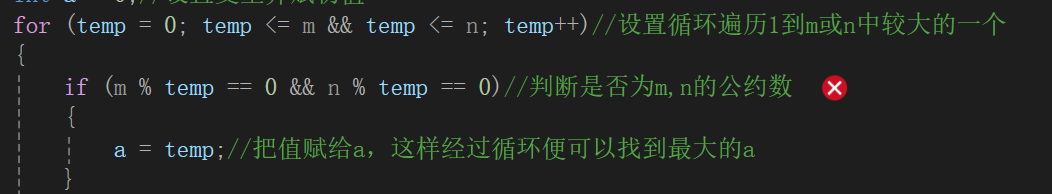
**五、遇到的问题与解决方法**

**三．1**

**这里出现的仍然是之前出现过的问题即除数等于0的问题**

****

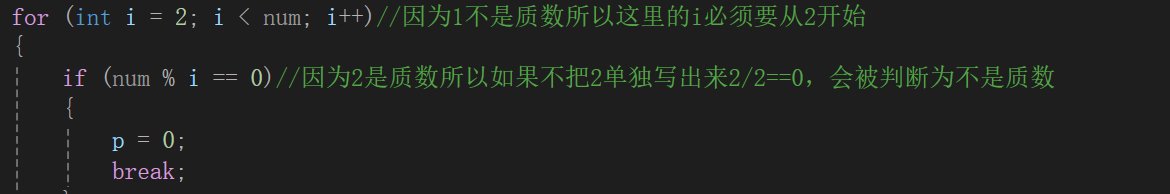
**原因是在于：**

****

**这里设初值时包含了0，到致判定余数是否为零时除数为0，因此报错，在进行取余运算时应多注意。**

**三．2**

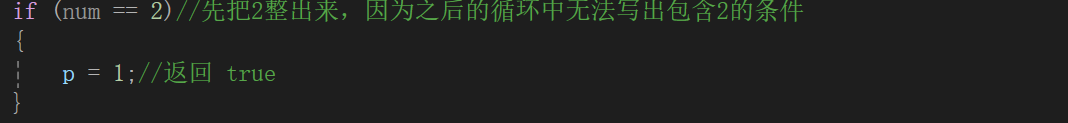
1. **在设置循环条件时总是无法把2包含进去**

****

**如图，首先不能设置i=1 因为不管谁除以一的余数都为0，会发生错误。**

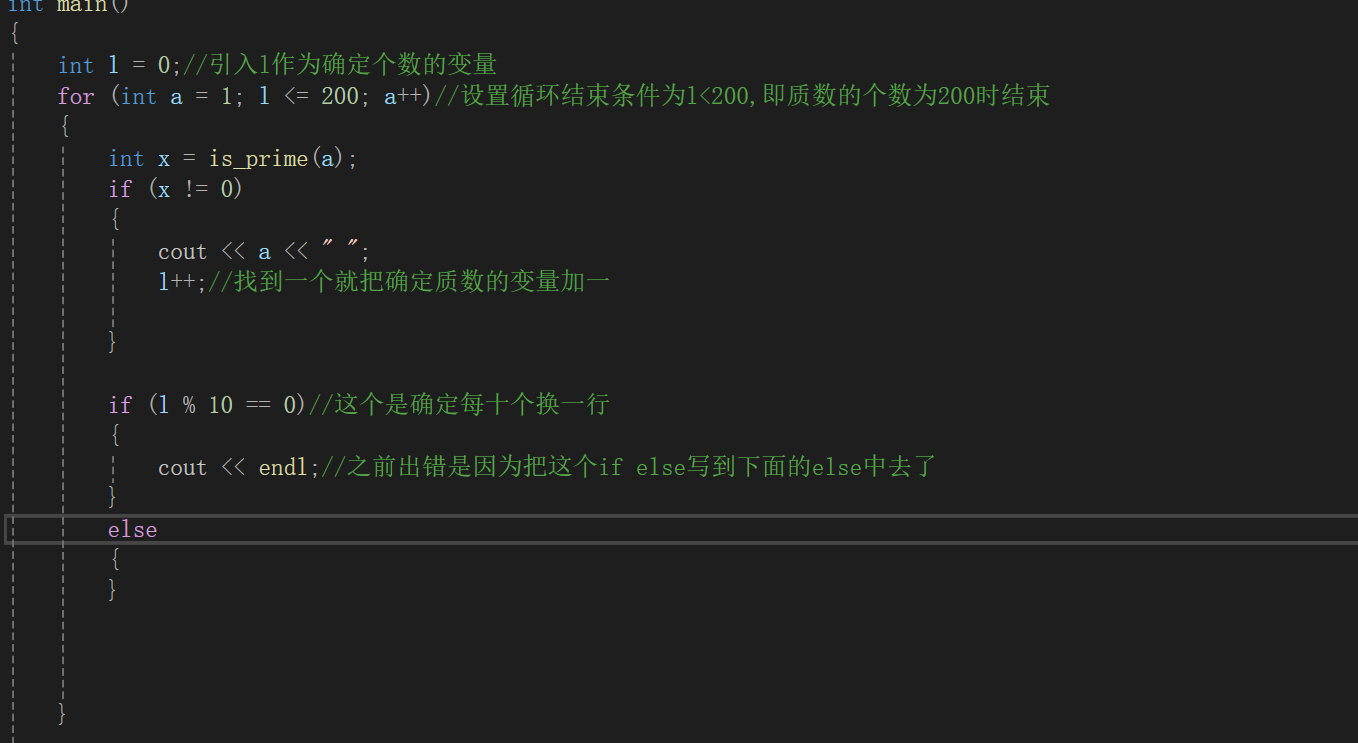
**其次，这样的设置是不包含对2的判断的，num==2时会直接跳出循环，若设置i<=num,**

**则也无法包含2，因为2/2余数为0，所以只能把2 单独设置一个if条件进行判断。**

****

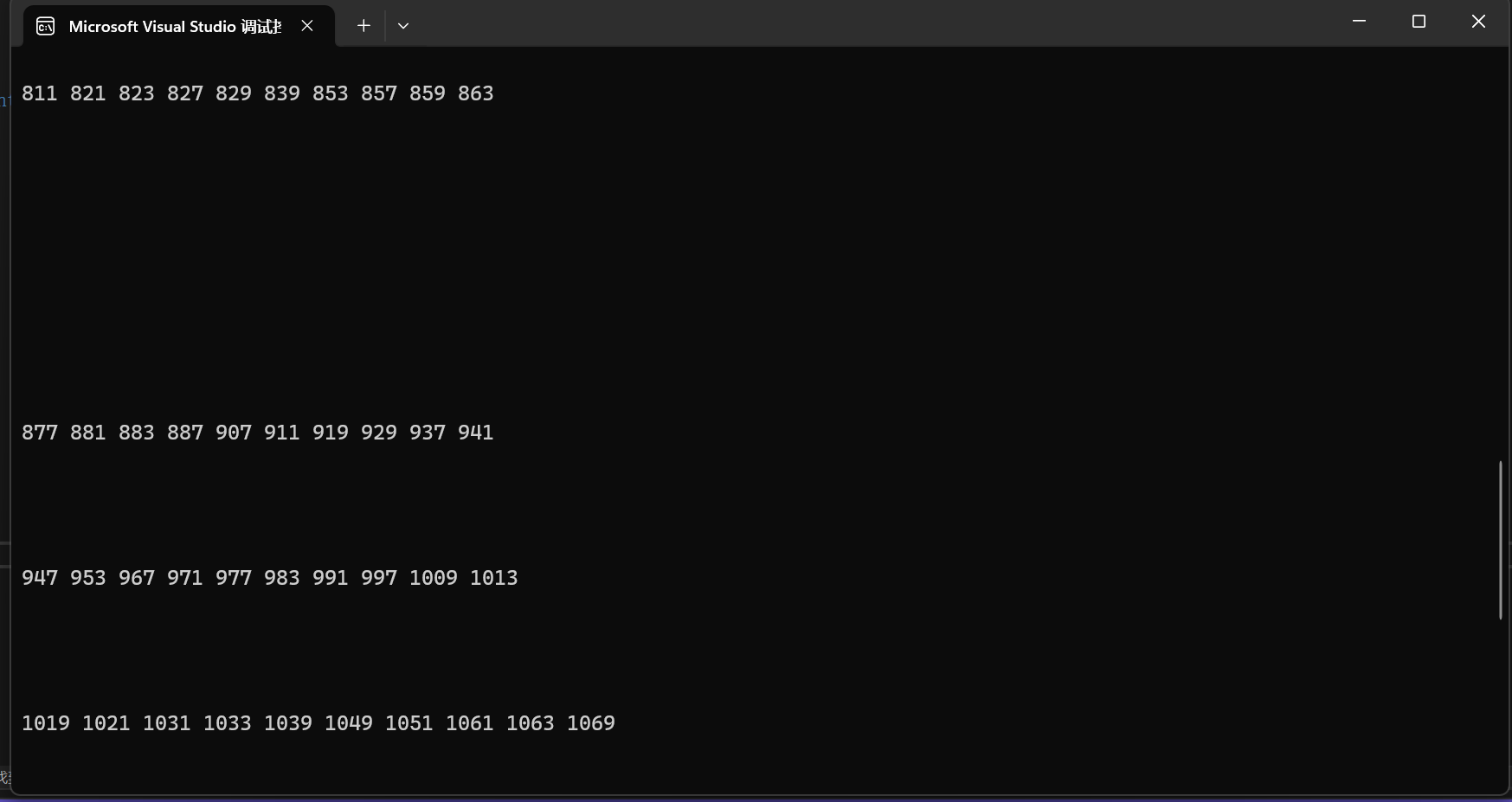
**2.**

**原来写的代码如图所示：**

****

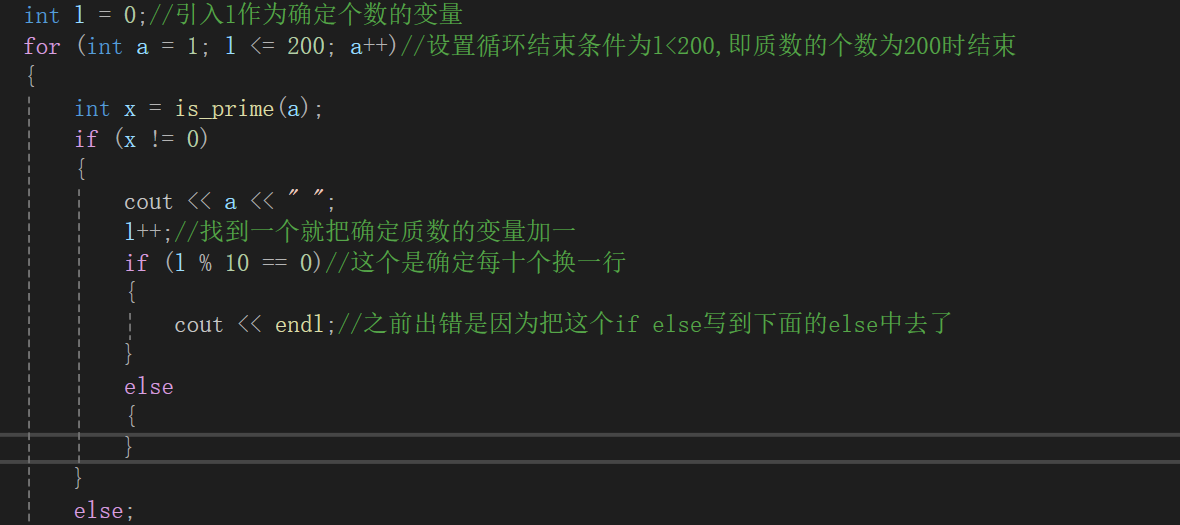
**每找到一个质数就把l加一 ，然后在进行条件判断看是否到达十个，如果到十个数了就换一行。**

**但是得到的结果如图：**

****

**发现每十个数之间空的行数并不相同经过利用F11进行逐行检测时可以发现利用循环进行枚举时，也就是说在某次找到一个素数后，满足j%10==0时，执行一次换行，然后执行下次循环，而下次循环没有找到素数，j的计数不变，也就是说，依旧满足j%10==0，所以还会进行一次换行，以此类推，在每行之间就有可能存在很多的空行。**

**将其改为**

****

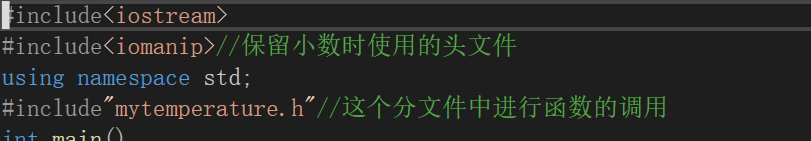
**这样将if 放入判断是否为质数的那个if中便可以只在找到质数的条件下才进行换行否则不换行。**

**三.3**

**学会函数分文件的编写方式，即先创建后缀为.h的头文件，在创建后缀为.cpp的源文件**

**并在头文件中写函数声明，在源文件中写函数的定义，并可以在新建一个源文件用于函数的调用。**

**注意：1.在写.h的头文件时要加上函数的基本框架。2.源文件直接用.h头文件的声明即可这样就可以不用再写函数的框架了。**

****

**另外注意保留小数的函数**

**屏幕截图 2024-12-08 110658**

**一定要加fixed不然会以科学计数法的方式保留两位小数。**

**三．5**

**主要使用的是递归的方法只要弄清楚递归公式并在函数中调用函数即可。调用函数是输出最终的数据即第一天的桃子数量。**

**六．体会**

**1.对逻辑的判断要更加清晰**

**首先在使用for循环时要弄清楚不同语句间的逻辑关系，清楚理解每个for循环的具体作用并加以分析在实验三 5中我由于将换行判断单独放在了循环里面，导致换行的指令被重复执行，最后直接作用到了输出的内容中：行与行之间会存在大量的空行。这一问题也大大地降低了我的编程效率，在寻找这一问题以及思考怎么解决时，耗费了我大量的时间以及精力，这就是由于我在书写循环以及判断的过程中，对逻辑不够细心造成的。**

**2.对各种错误的判断要更加明确和敏感**

**我经常在代码没有报错而输出错误或着有一部分代码没有经过调用，这是我应该根据输出结果迅速判断时那个地方出现错误，避免因为检查哪里出错而浪费大量时间，当实在检查不出来时在采取利用f11逐行检查的方法来找出是哪里出现了问题。另外我还应该对各类型的报错更加熟悉，避免出现报错时却不知道是哪里错了而去查找导致大量时间浪费。**

**3.对于函数的分文件编写要更加熟悉**

**在使用函数量较大时，可以分文件进行编写，在头文件中把所有函数进行定义，随后对各函数进行功能划分，把他们各自分进其他源文件中，最后在主函数所在源文件调用函数，这样编写代码可读性及逻辑性很强。**

**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<iostream>

using namespace std; //#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\_\_abcdabcaba\_

(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**

1. **算法分析，程序结果**
2. **数组**

**四．1**

#include<iostream>//这个实验首先可以看出是要通过两层循环判断来实现

using namespace std;

int main()

{

int arr[10];//定义一个数组来承载最终的数据

int arr1[10];//定义一个临时数组来存放用户输入的数据

int num = 0;//定义最终去掉重复的数后剩余的数

int temp;//创建一个临时变量

for (int i = 0; i < 10; i++)//这个for循环用于进行数据的输入

{

cin >> arr1[i];

}

for (int x = 0; x < 10; x++)//这个for循环对整个数组进行遍历

{

temp = 0;//初始化临时变量

for (int y = 0; y < x; y++)//这个for用于判断这个数arr[x]与这个数之前的数是否存在相等的情况，不把arr[x]与整个数组进行比较是因为

{//遍历整个数组时会遍历到自身，这样全都不符合下面的if条件判断于是一个数也输出不出来

if (arr1[x] == arr1[y])

{

temp++;//不能在这里面嵌套for循环进行赋值，因为要考虑外面嵌套循环的影响，所以在这里面创建一个临时变量

}

}

if (temp == 0)

{

arr[num] = arr1[x];//将临时数组中的数赋值给正式数组

num++;//num++计算出正式数组中有多少数字，便于后面输出

}

}

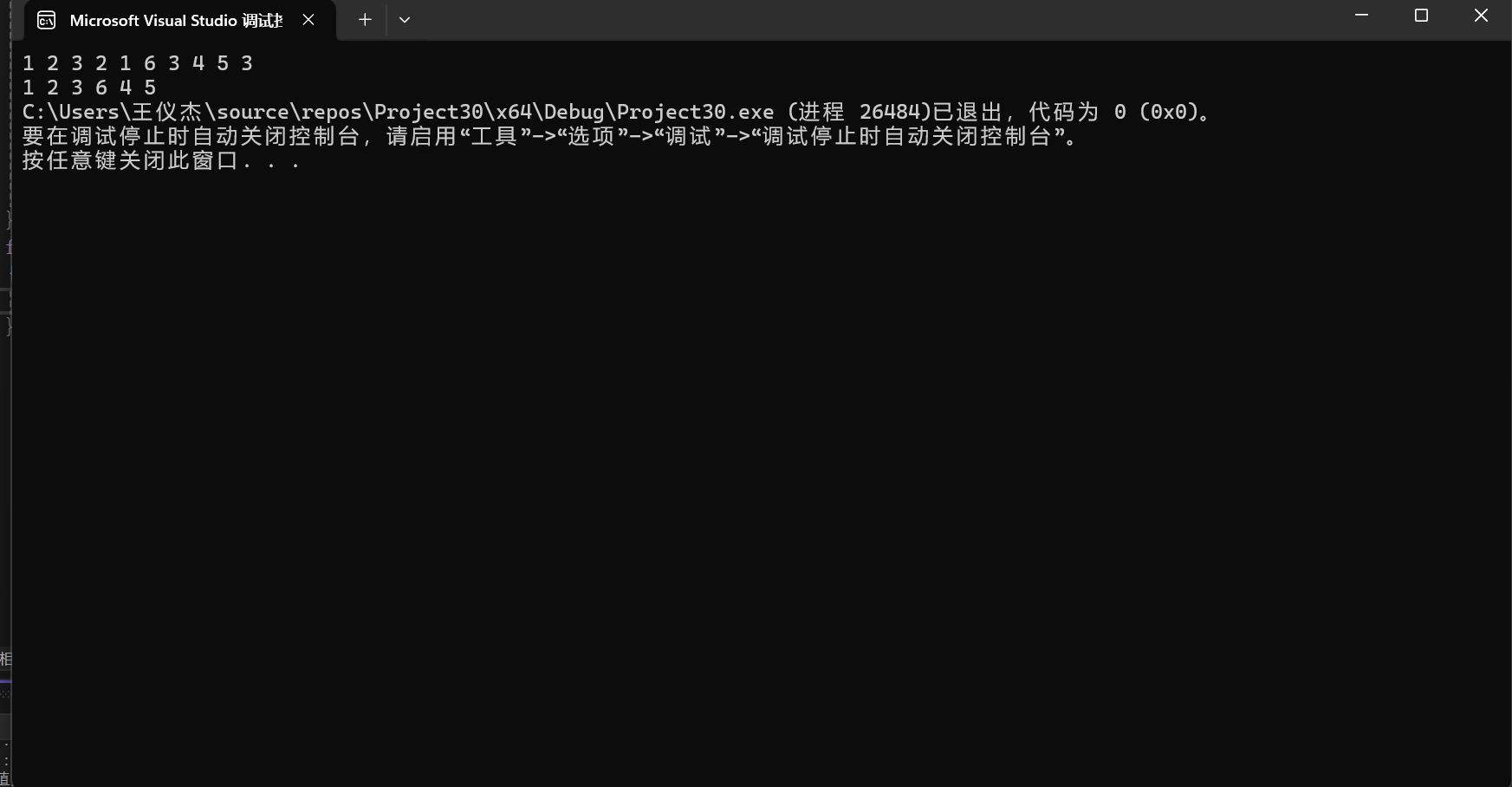
for (int a = 0; a < num; a++)

{

cout << arr[a] <<" ";

}

}



**四．2**

#include<iostream>

using namespace std;

void bubblesort(double\* arr, int length)//用指针可以使形参和实参同时改变；这里利用冒泡排序时需要同时传入数组的地址和它的长度

{

for (int m = 0; m < length - 1; m++)

{

for (int n = 0; n < length - 1 - m; n++)

{

if (arr[n] > arr[n + 1])

{

int temp = arr[n];

arr[n] = arr[n + 1];

arr[n + 1] = temp;

}

else;//这就是一个冒泡排序的函数

}

}

}

int main()

{

double arr[10];//创建一个双精度的数组

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

{

cin >> arr[i];//利用循环进行输入

}

int length = sizeof(arr) / sizeof(arr[1]);

bubblesort(arr, length);//调用函数后要加分号

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

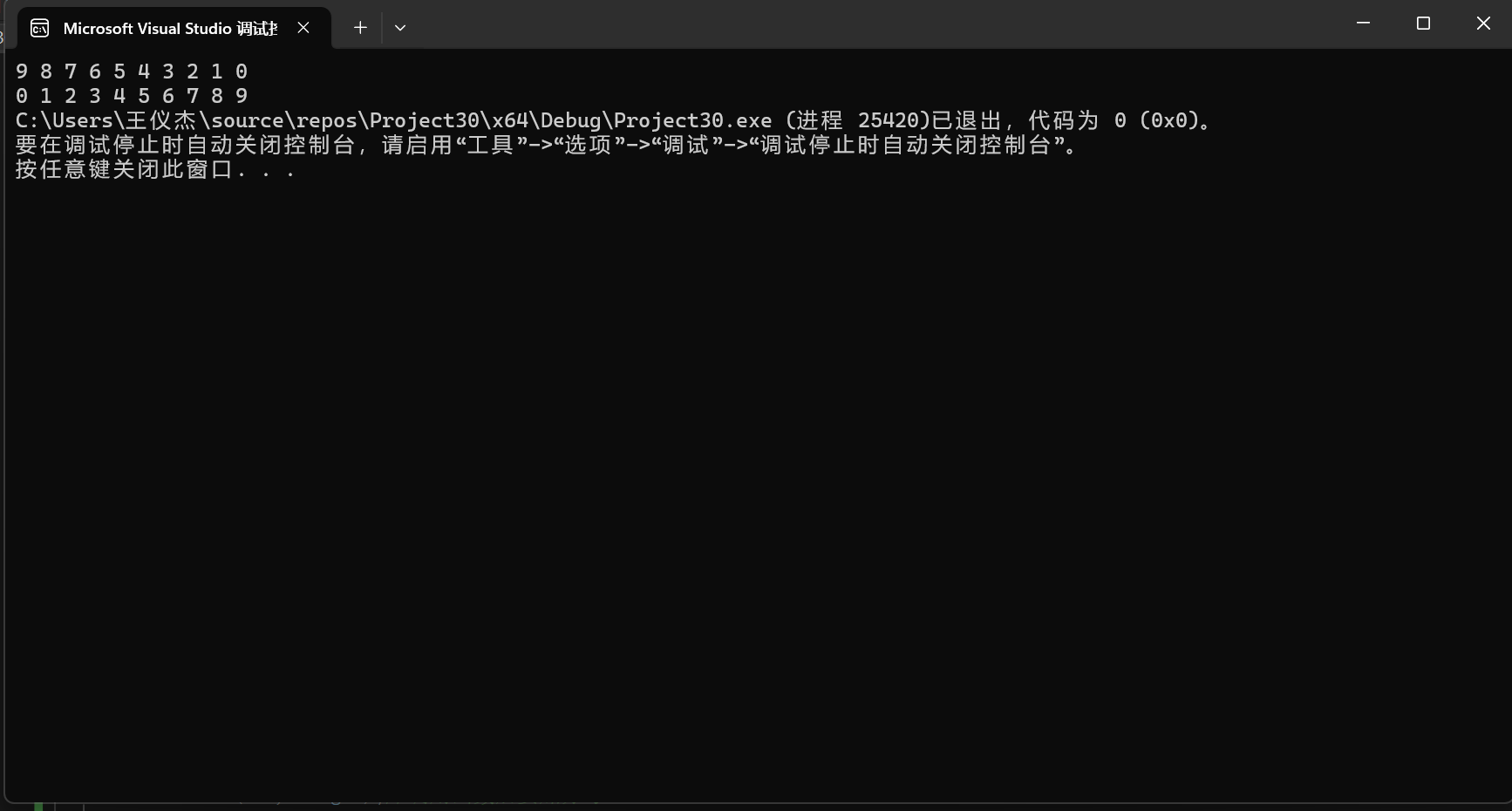
{

cout << arr[i] << " ";//因为是利用指针把地址直接传到函数里，所以形参和实参都会变

}//直接把排序后的输出出来即可

return 0;

}

****

**四．3**

#include<iostream>

using namespace std;

void change(bool\* arr, int i, int a = 2)

{

for (a = 2; a <= 100; a++)//每次的间隔数,注意题上说的是每隔两个改变一次状态

{

for (i = a - 1/\*数组的编号比正常的编号少一\*/; i < 100; i += a + 1)//每次的状态改变，i=i+a+1，这个意思就是每次加a+1

{//因为说的是每隔a个改变一次状态不是每a个改变一次状态

arr[i] = (!arr[i]);//注意取反的符号

}

}

}

int main()

{

bool arr[100];

for (int i = 0; i < 100; i++)

{

arr[i] = true;//先完成第一个学生的操作，因为第一个学生不是每隔一个改变一次状态，是直接改变了所有的状态，和其他人不一样

}

for (int i = 1; i < 100; i++)

{

change(arr, i);//调用函数

}

for (int i = 0; i < 100; i++)

{

if (arr[i] == true)//一定是写成==而不是写成赋值符号=

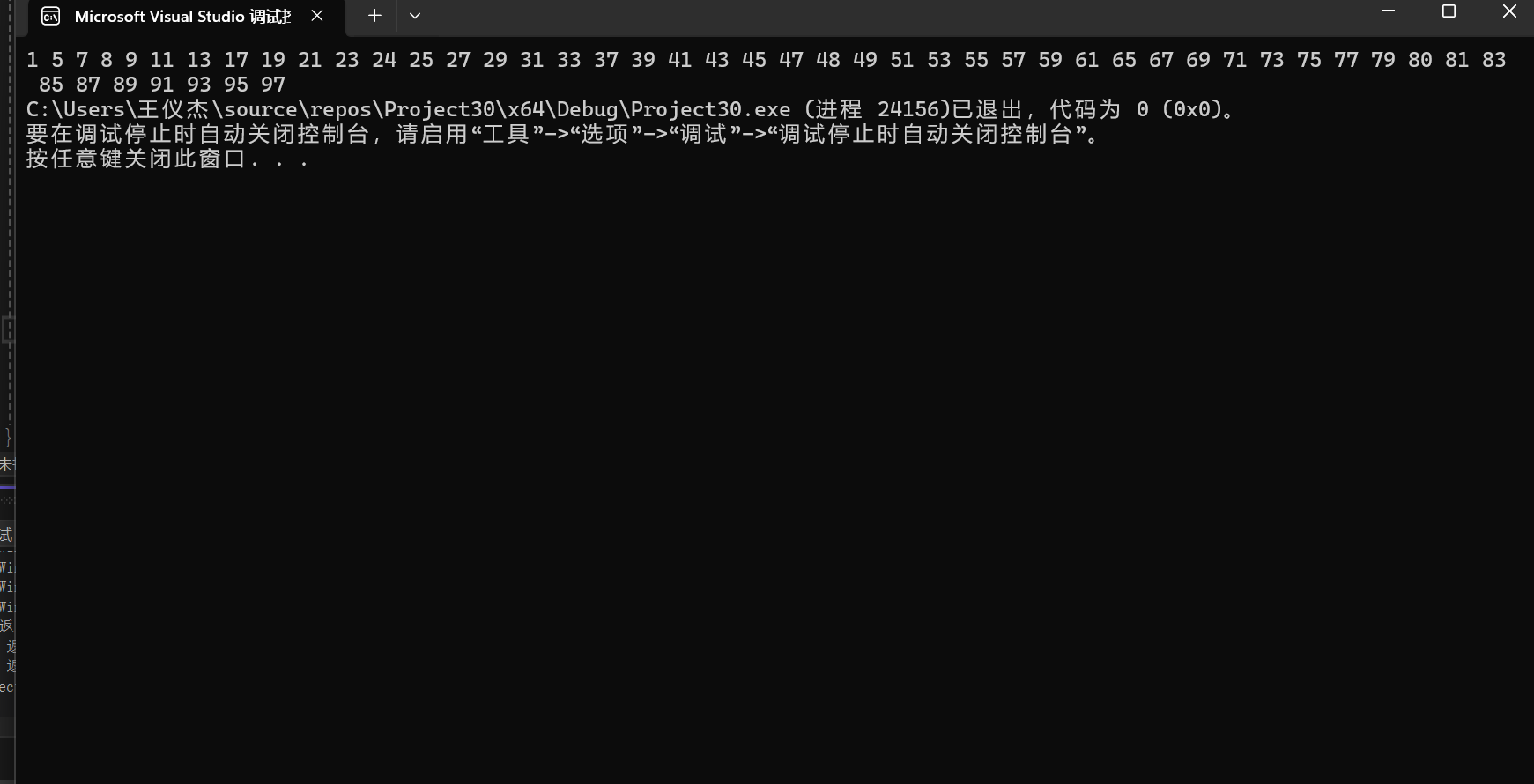
{

cout << i + 1 << " ";//注意输出的不是数组的编号而是存物柜的编号

}

}

}



**四．4**

#include<iostream>

using namespace std;

void merge(const int list1[], int size1, const int list2[], int size2, int list3[])//加const 代表不能函数内部不能对数组进行修改

{//在函数外仍可以修改

int size3 = size1 + size2;//因为list1和list2是不可修改的两个数组，因此只能作为传入的两个数组使用

for (int i = 0; i < size2; i++)

{

list3[i] = list2[i];//list3作为合并后的数组

}

for (int i = 0; i < size1; i++)

{

list3[size2 + i] = list1[i];

}//这两个for用于把两个给定的数组合并到list3中去

//下面用冒泡排序排一下

for (int i = 0; i < size3 - 1; i++)//循环的总轮数

{

for (int m = 0; m < size3 - i - 1; m++)//每一次循环需要交换的次数，这里是减去i也就是减去轮数再减1，即可算出每轮需要交换的次数

{

if (list3[m] > list3[m + 1])

{

int temp = list3[m];

list3[m] = list3[m + 1];

list3[m + 1] = temp;

}

else;

}

}

for (int i = 0; i < size3; i++)//输出list3[]中的所有元素

{

cout << list3[i] << " ";

}

}

int main()

{

int size1;

int size2;

int size3;

cout << "请输入数组长度" << endl;

cin >> size1;//第一个数先输入长度

cout << "enter list 1:" << endl;

int\* list1 = new int[size1];//用new不要求数组的长度是一个变量,直接定义一个数组时必须写出数组中含有几个数，或者写出数组的内容

for (int i = 0; i < size1; i++)//使用new定义一个数组时是没有自己的名字的，必须用指针的名字

{

cin >> list1[i];

}

cout << "请输入数组长度" << endl;

cin >> size2;

cout << "enter list 2" << endl;

int\* list2 = new int[size2];//和上一个相同

for (int i = 0; i < size2; i++)

{

cin >> list2[i];

}

size3 = size1 + size2;

int\* list3 = new int[size3];

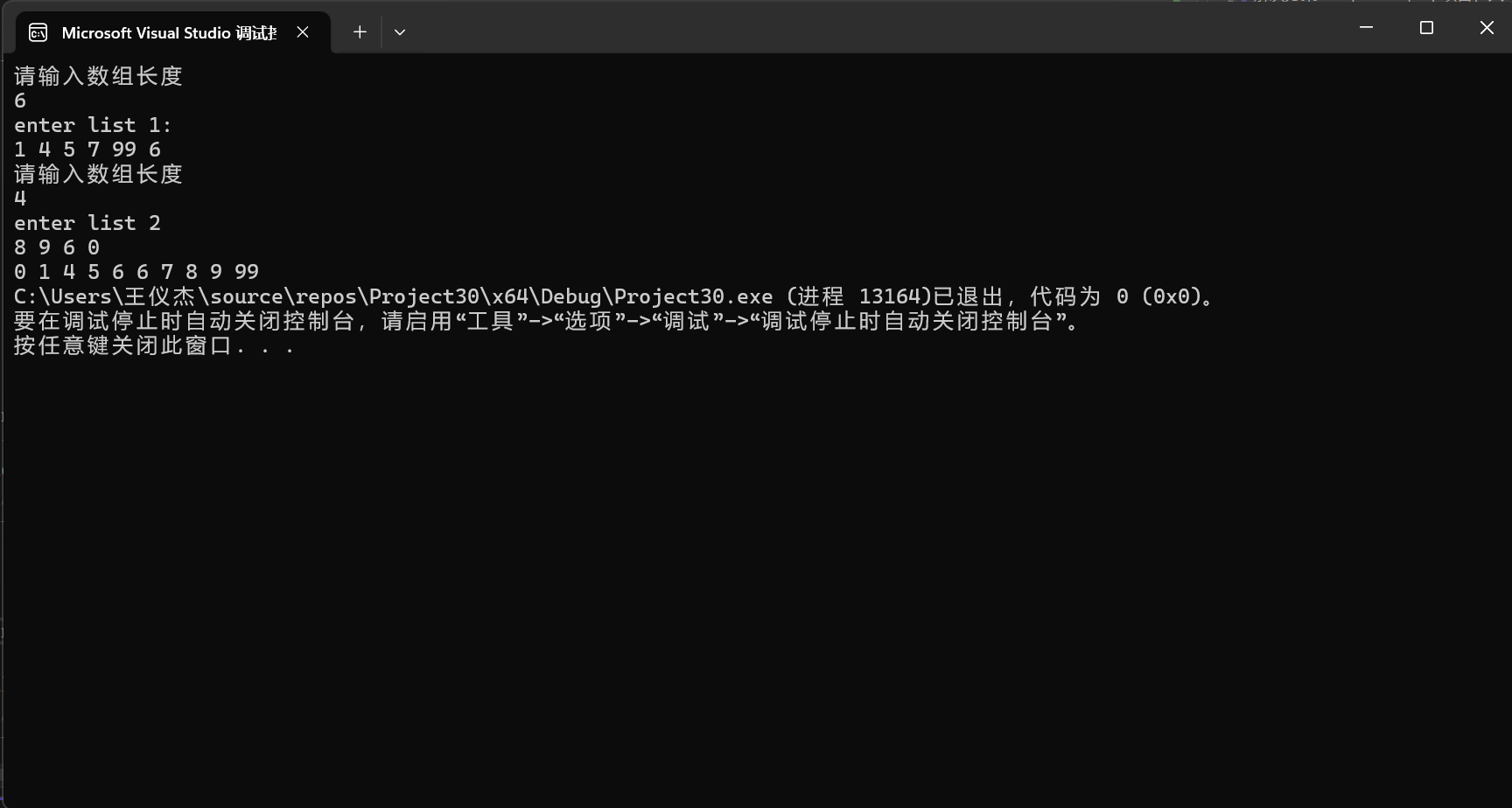
merge(list1, size1, list2, size2, list3);//调用函数只用传数组名进去就行了

delete[] list1;//new出来的空间最后要手动删除

delete[] list2;

delete[] list3;

}



**四．5**

/\*#include<iostream>

#include<string>

using namespace std;

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

{

int len\_s1 = strlen(s1);//求s1长度

int len\_s2 = strlen(s2);//求s2长度

for (int i = 0; i<len\_s1; i++)

{

for (int a = 0; a<len\_s2; a++)

{

if (s1[i] == s2[a])

{

cout <<i;

break;

}

}

}

return -1;

}

int main()

{

char s1[100];

char s2[100];

cout << "enter the first string " << endl;

cin.getline(s1, 100);//这种写法不要求写成string 类型

//getline(cin, s1);不能识别这种写法，因为这种写法只适用于string s1

cout << "enter the second string " << endl;

cin.getline(s2, 100);

int index=indexOf(s1, s2);错了

}\*/

}#include<iostream>

using namespace std;

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

{

//用于计算字符串长度的函数strlen

int len\_s1 = strlen(s1);

int len\_s2 = strlen(s2);

//判断如果s1比s2还长那么就不可能是s2的子串

if (len\_s1 > len\_s2)

{

return -1;

}

else;

int a = 0;

for (int i = 0; i <= len\_s2 - len\_s1/\*这里的目的是遍历s2中所有可能作为起始点的字符并判断s1是否为其子串\*/; i++)

{

bool match = true;

for (a = 0; a < len\_s1; a++)//这个循环的目的是遍历S1

{

if (s2[i + a] != s1[a])//

{

match = false;//这里引入这个match的原因是else无法写到for函数外面，因此需要把他放到一个数里面，到for函数外在进行判断

break;

}

}

if (match)//如果为真

{

return i;

}

}

return -1;//return -1;写在这里代表的是上面的return i如果没有任何值能返回那么才会返回-1

}

int main()

{

char s1[100];//因为题中一定要用char所以getline(cin,input)和input.length()函数都用不了；这两个只能用与string 类型

char s2[100];

cout << "Enter the first string:" << endl;

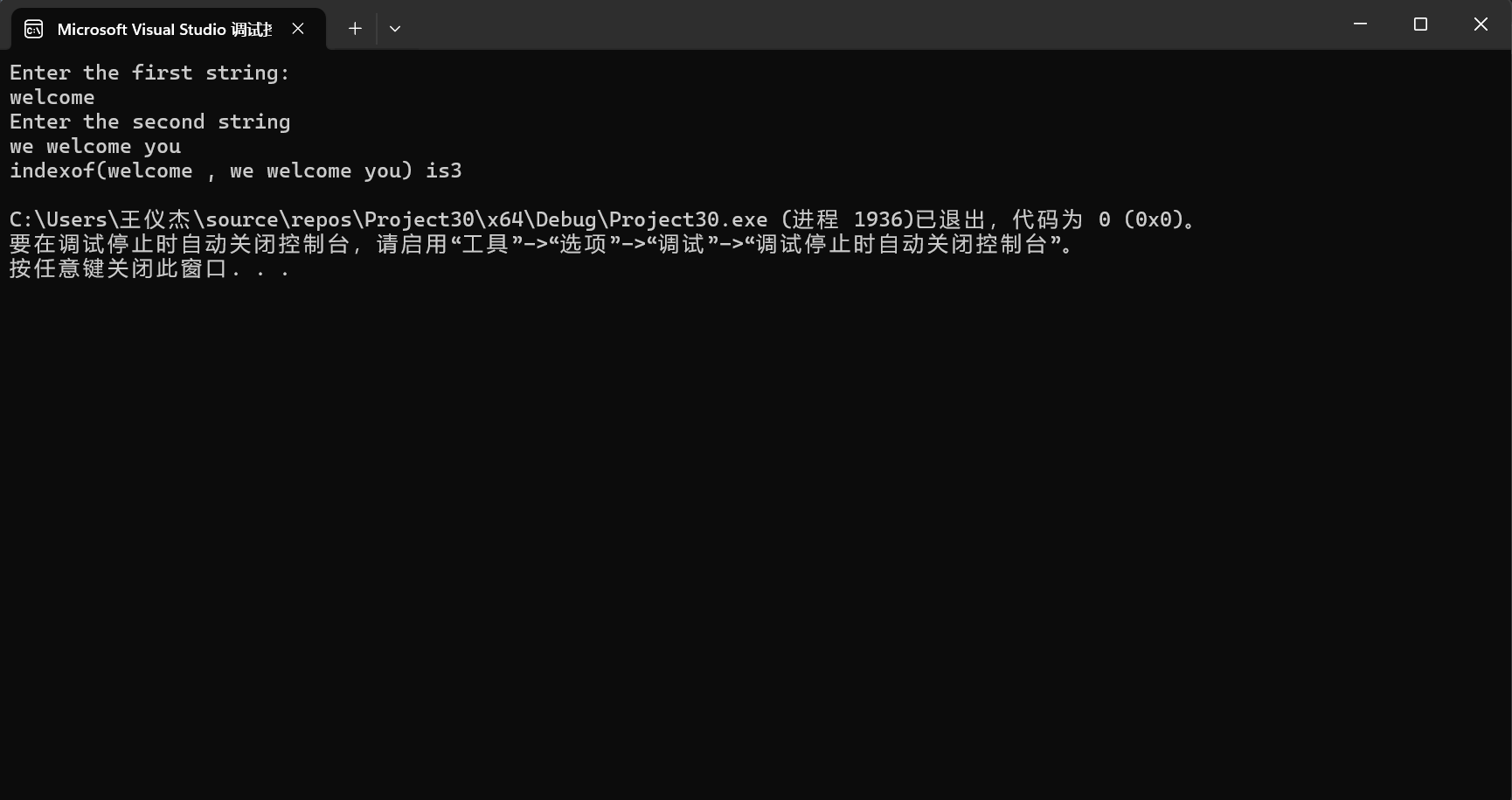
cin.getline(s1, 100);//用cin.getline可以不用于string并且不用包含头文件

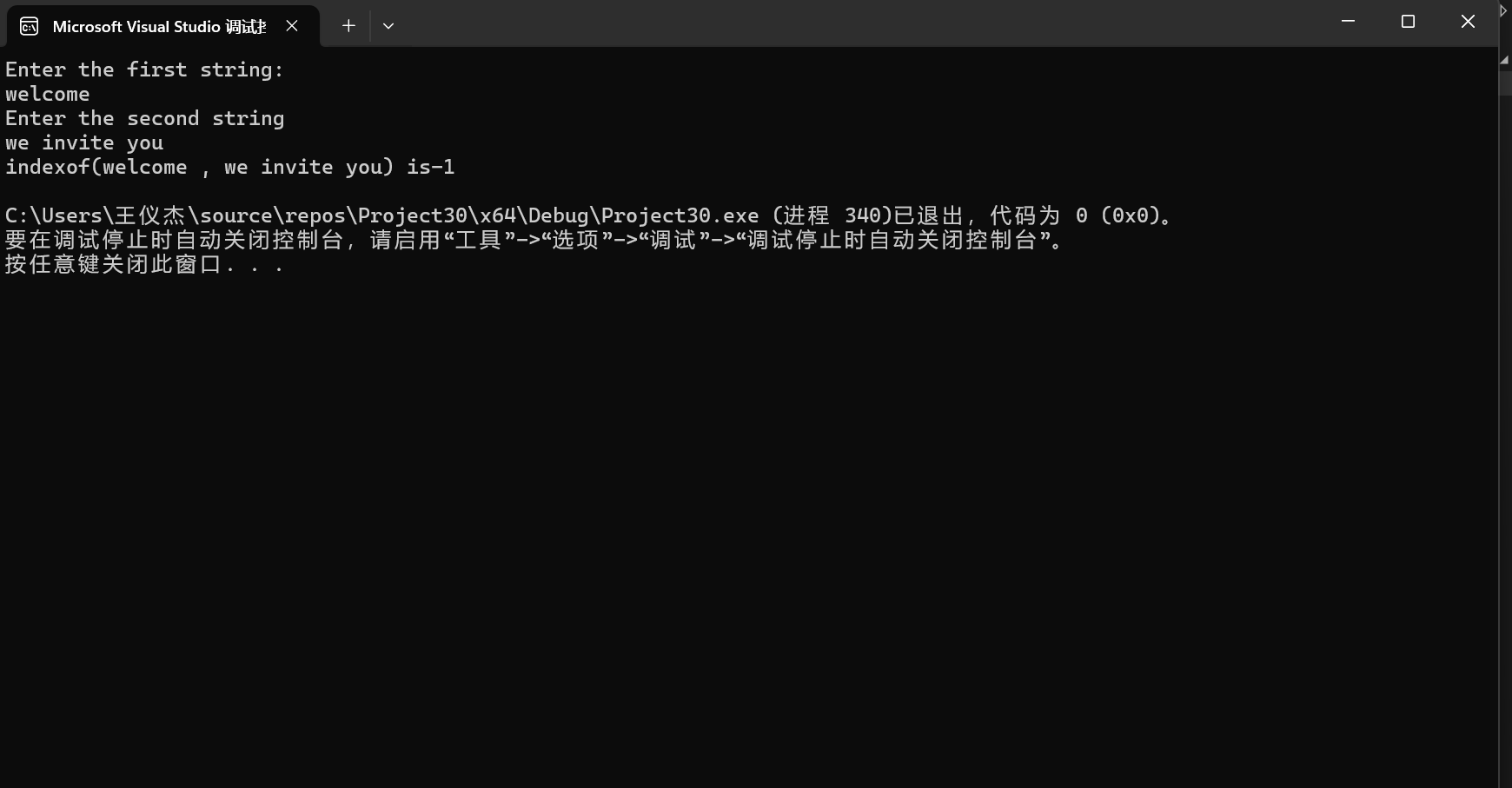
cout << "Enter the second string" << endl;

cin.getline(s2, 100);

int a = indexOf(s1, s2);

cout << "indexof(" << s1 << " , " << s2 << ") is"/\*这只是一个输出的形式\*/ << indexOf(s1, s2) << endl;//最后调用函数并输出即可

****



**四．6**

#include<iostream>

#include<string>

using namespace std;

void count(const char s[], int counts[])//这个数组接受的是一个首地址

{

int length = strlen(s);//记住这个函数，先取一下数组的长度

for (int i = 0; i < 26; i++)

{

counts[i] = 0;//先初始化全部赋值为0

}

/\*for (int i = 65,i1=97; i <= 90||i1<=122; i++,i1++)

{

for (int b = 0, a = 0; b < length && a < 25; b++, a++)//这样的嵌套循环不能有两个条件，因为每次外部训环结束后会进行重置

{

if (s[b] == static\_cast<char>(i)||s[b]==static\_cast<char>(i1))

{

counts[a]++;

}

}

}\*///错误

for (int i = 0; i < length; i++)

{

char ch = s[i];//要学会用一个临时变量来进行条件判断,不用也行但是每次都要遍历一边数组运算量过大

if (ch >= 'a' && ch <= 'z')

{

counts[ch - 'a']++;//ch在这里承接的是一个字母的asc2码，a的asc2码是65，因此把a减掉可以得到对应的数组的索引。例如ch此时承接的是a，为65，减去a后为counts[0]正好为counts[0]应该对应的数

}

else if (ch >= 'A' && ch <= 'Z')

{

counts[ch - 'A']++;

}

}

}

int main()

{

int counts[26];

char s[100];//设置一个足够大的即可

cout << "Enter a string " << endl;

cin.getline(s, 100);

count(s, counts);//如果形参列表中写的是数组的形式，那么只用传一个首地址进去即可

for (int i = 0,a=97; i < 26&&a<=122; i++,a++)

{

if (counts[i] != 0)

{

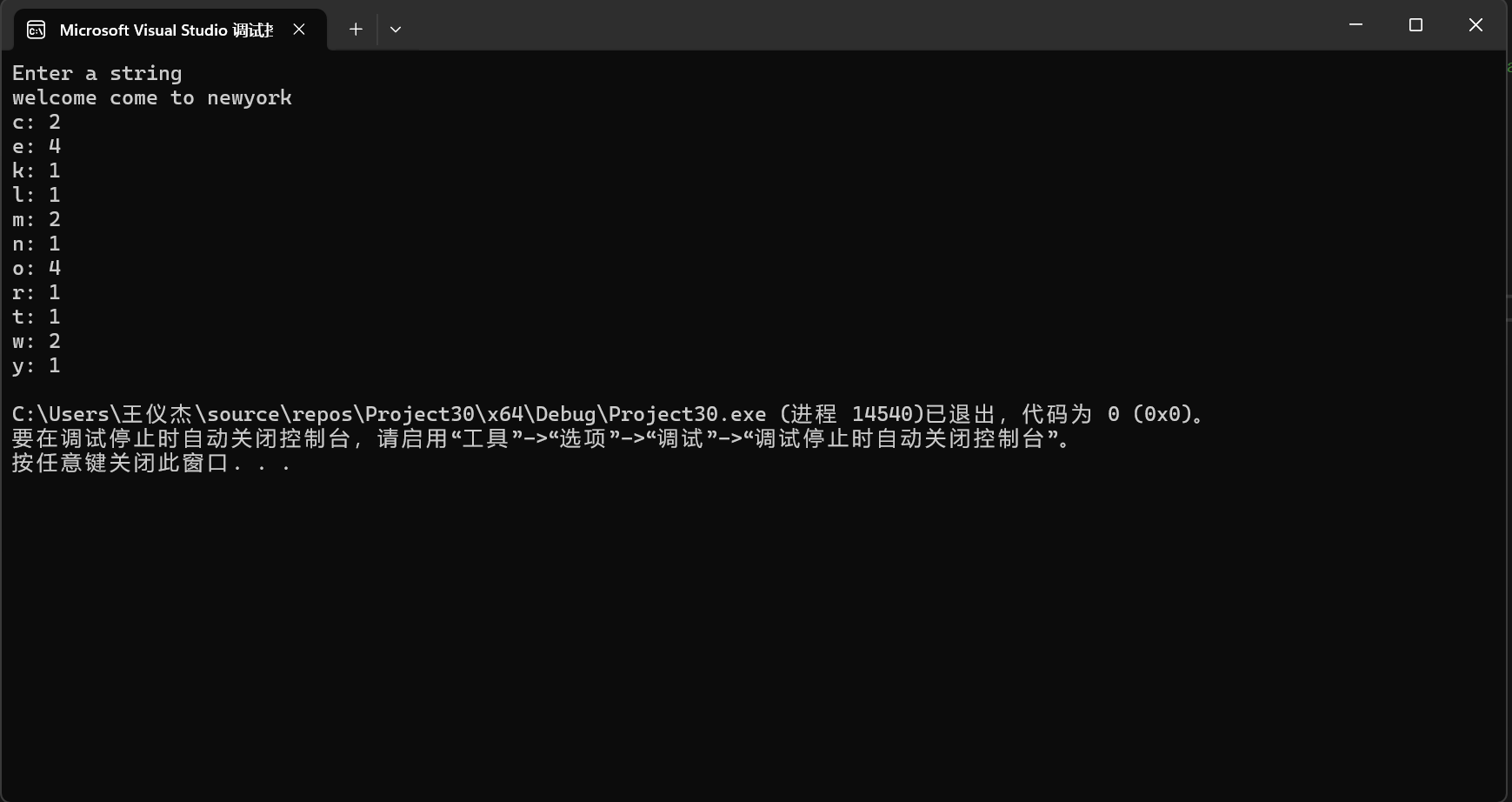
cout << static\_cast<char>(a) << ":" << " ";

cout << counts[i] << endl;

}

}

}



1. **指针**
2. **（1）**

#include<iostream>

using namespace std;

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;

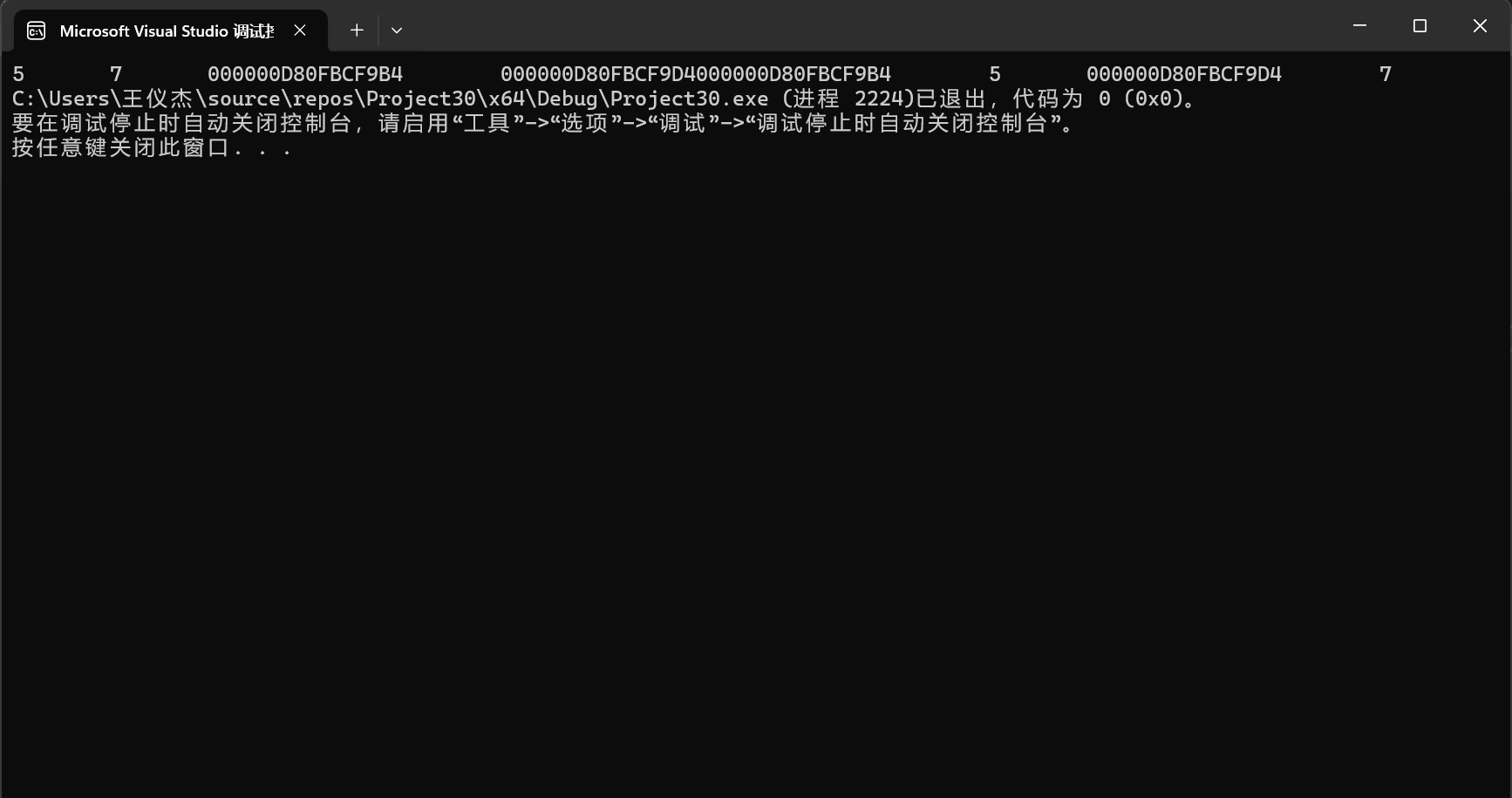
pj = &j;

i = 5; j = 7;

cout << i << '\t' << j << '\t' << pi << '\t' << pj;

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

}



1. **（2）**

#include<iostream>

using namespace std;

int main() //C语言程序，要了解

{

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

int\* p, i;

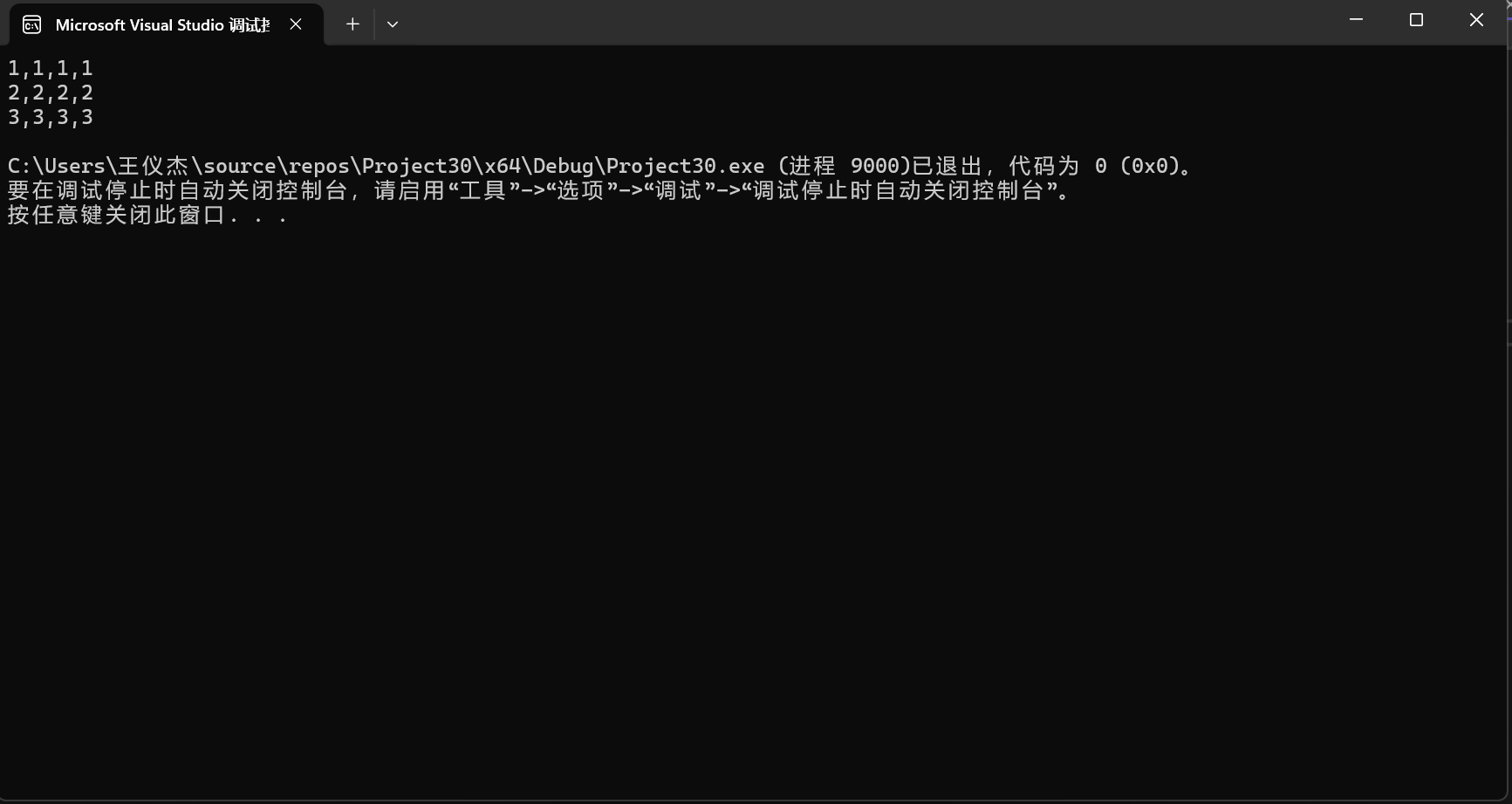
p = a; //将数组a首地址送给p

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

}//当一个指针指向了一个数组的首地址时，用指针名和用数组名可以达到相同的效果a[i],p[i]都指的是数组中的元素

//\*（p+i）\*(a+i)指的是解引用后的数



1. **（3）**

#include<iostream>

using namespace std;

#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);

}



1. **（4）**

#include<iostream>

using namespace std;

int\* f()

{

int\* list = new int[4];

for (int i = 0;i < 4; i++)//new 出来的数组必须通过for 循环来赋值

{

list[i] = i+1;

}//形参存放在栈区会在函数运行完后释放

return list;//返回值会被编译器做一次保留，之后就随机输出东西了

}

void main()

{

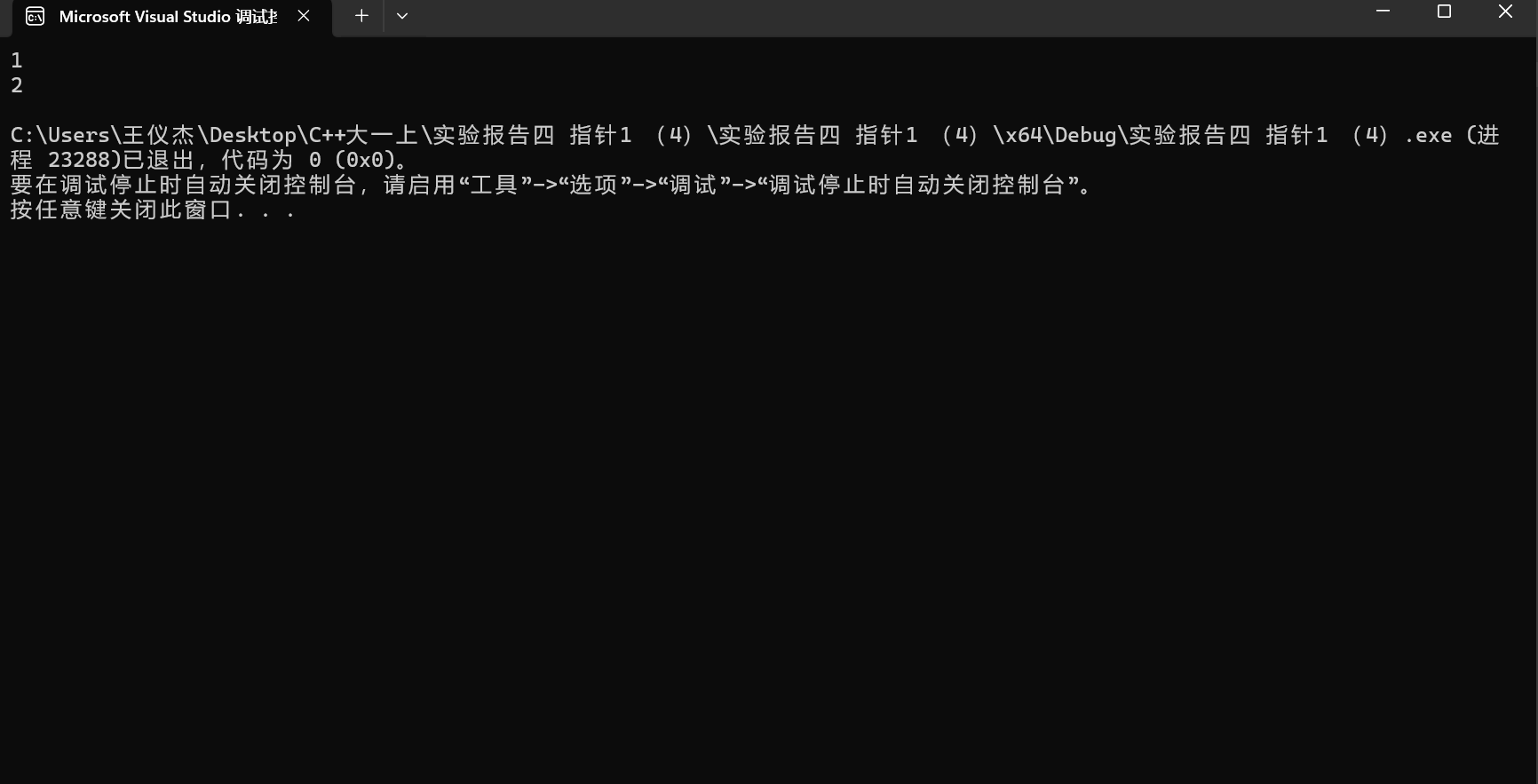
int\* list = f();//new在堆区开辟空间不会被释放

cout << list[0] << endl;

cout << list[1] << endl;

delete[] list;

}



1. **（1）**

#include<iostream>

using namespace std;

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

{

//用于计算字符串长度的函数strlen

int len\_s1 = strlen(s1);

int len\_s2 = strlen(s2);

//判断如果s1比s2还长那么就不可能是s2的子串

if (len\_s1 > len\_s2)

{

return -1;

}

else;

int a = 0;

for (int i = 0; i <= len\_s2 - len\_s1/\*这里的目的是遍历s2中所有可能作为起始点的字符并判断s1是否为其子串\*/; i++)

{

bool match = true;

for (a = 0; a < len\_s1; a++)//这个循环的目的是遍历S1

{

if (s2[i + a] != s1[a])//

{

match = false;//这里引入这个match的原因是else无法写到for函数外面，因此需要把他放到一个数里面，到for函数外在进行判断

break;

}

}

if (match)//如果为真

{

return i;

}

}

return -1;//return -1;写在这里代表的是上面的return i如果没有任何值能返回那么才会返回-1

}

int main()

{

char s1[100];//因为题中一定要用char所以getline(cin,input)和input.length()函数都用不了；这两个只能用与string 类型

char s2[100];

cout << "Enter the first string:" << endl;

cin.getline(s1, 100);//用cin.getline可以不用于string并且不用包含头文件

cout << "Enter the second string" << endl;

cin.getline(s2, 100);

int a = indexOf(s1, s2);

cout << "indexof(" << s1 << " , " << s2 << ") is"/\*这只是一个输出的形式\*/ << indexOf(s1, s2) << endl;//最后调用函数并输出即可

**2.（2）**

#include <iostream>

#include <cmath>

using namespace std;

int parseHex(const char\* const hexString) {

int s\_len = strlen(hexString);//计算输入字符串的长度

int num = 0;//储存最后的十进制数

for (int i = 0; i < s\_len; i++)//遍历字符串中的每个字符

{

char currentChar = hexString[i];//定义当前的字符

if (currentChar >= '0' && currentChar <= '9')

{

num = num + ((currentChar - '0') \* pow(16, s\_len - i - 1));//pow函数的用法16为底数，s\_len为指数

}

else if (currentChar >= 'A' && currentChar <= 'F')

{

num = num + ((currentChar - 'A' + 10) \* pow(16, s\_len - i - 1));

}

else if (currentChar >= 'a' && currentChar <= 'f')//如果当前字符是a-f之间的小写字母

{

num = num + ((currentChar - 'a' + 10) \* pow(16, s\_len - i - 1));

}

else//如果当前字符不是有效的16进制字符

{

cout << "数值不是16进制" << endl;//提示字符串不合法

}

}

return num;//返回计算得到的10进制数

}

int main() {

char s[50];//定义一个字符数组，用于存储用户输入的16进制字符串

cout << "请输入16进制数字:";//提示用户输入16进制数字

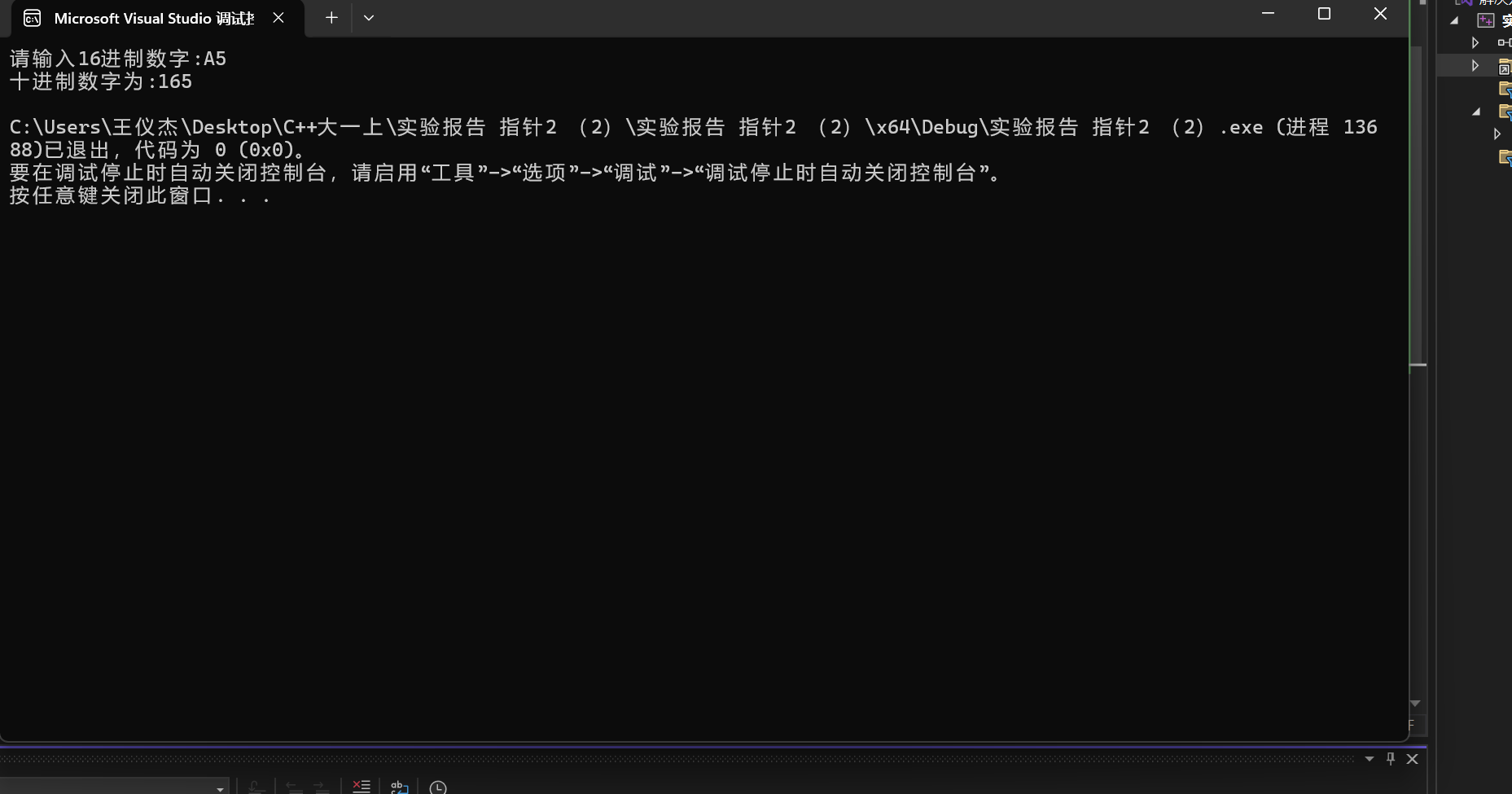
cin.getline(s, 50);

int dec = parseHex(s);//调用parseHex函数

cout << "十进制数字为:" << dec << endl; //输出转换后的10进制数字

return 0;

}



**2.(3)**

#include<iostream>

using namespace std;

void order(int num,int\* arr)

{

cout << "请输入元素个数" << endl;

for (int i = 0; i < num-1; i++)

{

for (int a = 0; a < num - i - 1; a++)

{

if (arr[a + 1] < arr[a])

{

int temp =arr[a];

arr[a] = arr[a + 1];

arr[a + 1] = temp;

}

}

}

}

int main()

{

int num;

cin >> num;

int\* arr = new int[num];

for (int i = 0; i < num; i++)

{

cin >> arr[i];//这里arr中仍然是具体值

}

order(num, arr);

for (int i = 0; i < num; i++)

{

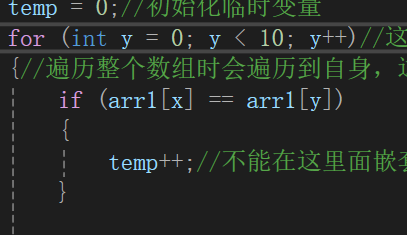
cout << arr[i];

}

****

**四、遇到的问题与解决方法**

**4.1**

****

**在判断数组中的数是否与其他数相等时我直接遍历了所有的数这个for用于判断这个数arr[x]与这个数之前的数是否存在相等的情况，不把arr[x]与整个数组进行比较是因为遍历整个数组时会遍历到自身，这样全都不符合下面的if条件判断于是一个数也输出不出来。**

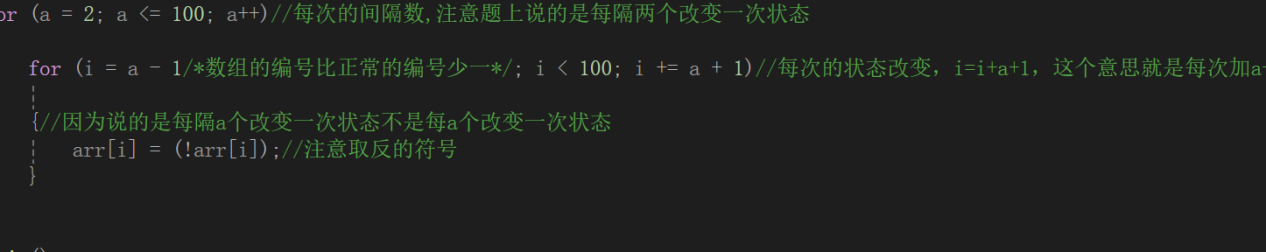
**4.2**



**这里直接用指针指向数组可以使实参和形参一起改变**

**记住冒泡排序的原理**

**4.3**

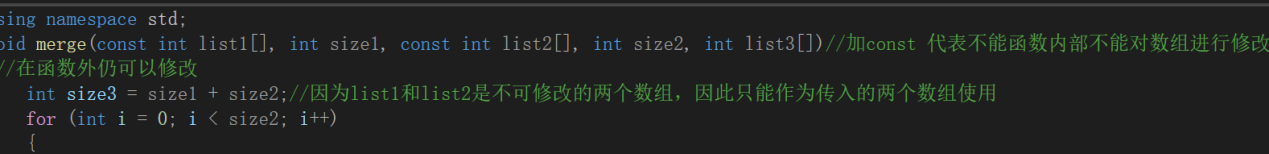


**注意看题，说的是每隔A个改变一次状态而不是每A个改变一次**

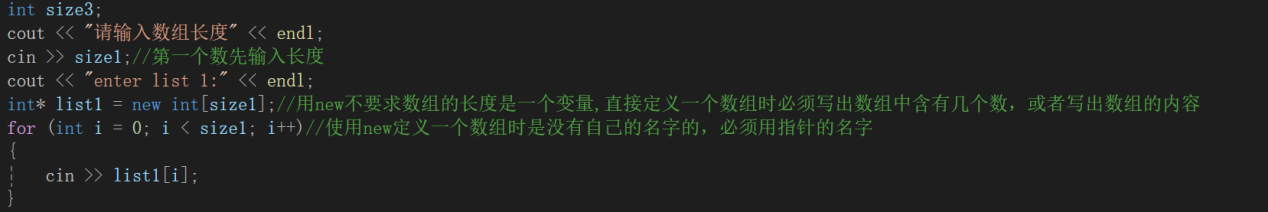
**注意第一个学生和其他的学生的方法是不一样的所以拿出来单独写**

**注意最后输出的是储物柜的编号而不是数组的下标，最后应把A+1在输出**

**4.4**



**加上const的原因是防止在函数内对数组进行误修改导致实参跟着一起变**

****

**（1）这里用new创建一个动态数组,因为此时并不知道数组的长度，直接创建数组会导致不知道具体长度而报错**

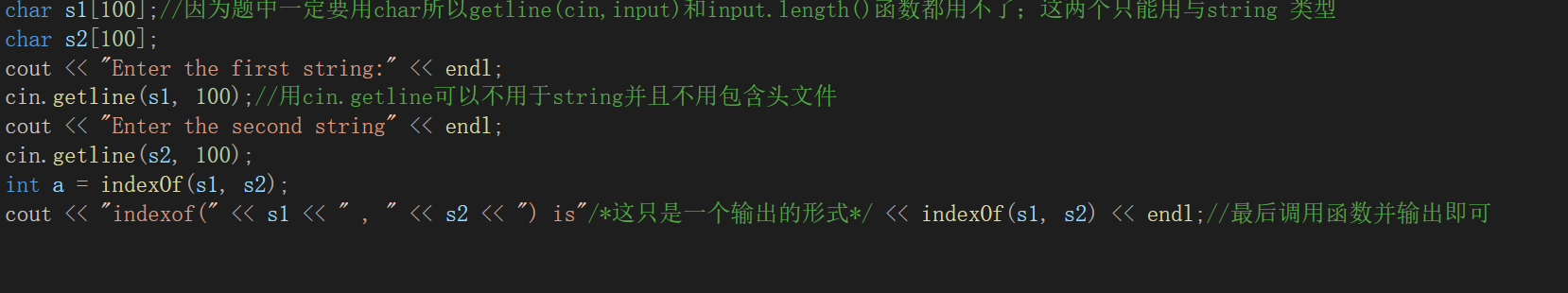
**（2）new创建动态数组后要delete**

**delete[] list1一定要加这个[]告诉编译器要删除的是一个数组**

**（3）new出来的数组没有给定名字，所以直接用对应指针的名字即可**

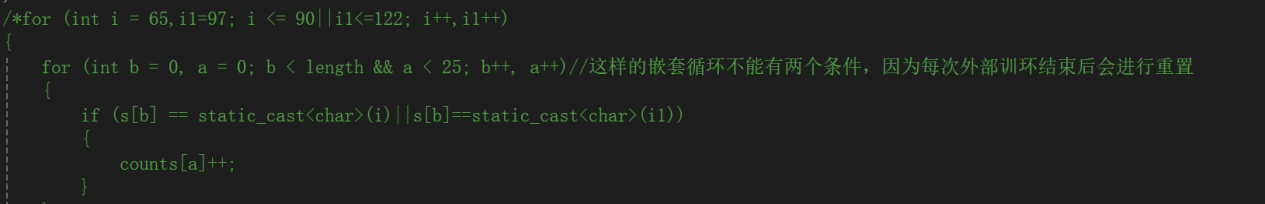
**4.5**

**注意这个题的整体逻辑首先如果s1比s2还要长那么就不可能是s2的子串，其次要遍历s2中所有可能作为是s1起始字符的字符并接着往后判断是否为s1的子串。**

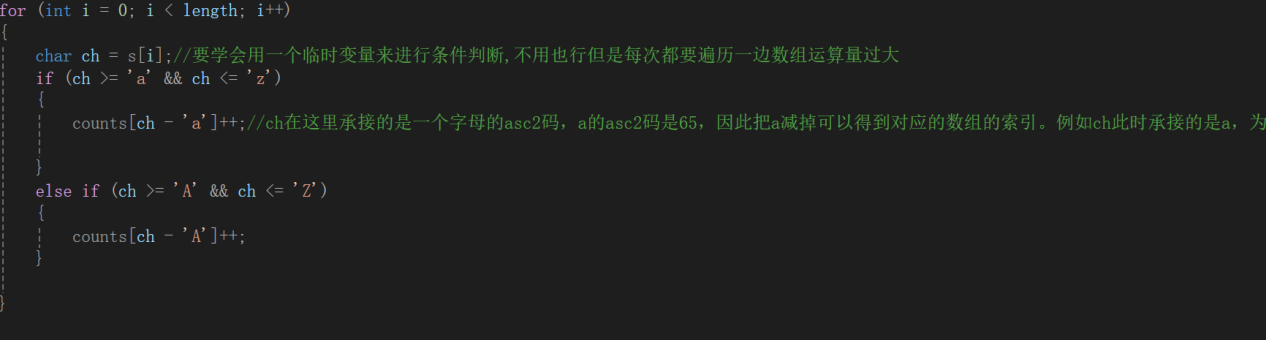
****

**接下来是对cin.getline()的使用，使用这个时char和string 类型好像都能用，并且不用包含任何的头文件**

**4.6**

****

**刚开始我采用这种两个条件的嵌套循环，但最终输出结果错误，经过检查发现每次外部循环结束后都会对a,b两变量进行重置进而回导致最终结果的错误。**

****

**修改为采用if else 进行判断。**

**指针二 （2）**

**整体思路是先把字符串转为整形，在根据16进制的定义将其转化为10进制**

**在asc2码的转化过程中要得到对应的索引只需要对吧此时的字符的asc2码与a的asc2码的区别即可**

**五、体会**

**1.对动态数组的运用应更加熟练**

**数组会与函数，字符串等内容相结合其可以实现的功能也比较丰富。但是如果想要实现相对比较复杂、更多的功能，就要掌握并且理解更多的有关于数组的知识。使用new创建动态数组可以在还不知道数组中元素数量时将数组创建出来并在之后在获取数组的长度**

**在使用了新的动态数组的知识后，运算过程、代码编写的方式都变得更加方便。但是，在使用动态数组时，也要记得在数组使用完毕后通过delete将数组数据删除。**

**2.在理解指针的原理和具体语法后应增强运用的熟练度**

**指针作为一个储存地址的变量有着诸多用处，在动态内存分配时，指针允许程序在运行时根据需要分配和释放内存。这对于处理大小不确定的数据结构非常有用。比如动态数组的创建。可用于数组和字符串操作，指针常用于数组和字符串的处理，因为数组名本身就是一个指向数组首元素的指针。这使得指针成为遍历和操作数组元素的有力工具。在进行函数参数传递时，通过指针，函数可以修改传递给它的变量的值。这在需要函数内部对外部变量进行修改时非常有用，可以同时修改实参和形参的值。**

**指针的诸多用法决定了要完全掌握指针十分困难，这就需要我在平时练习时增加熟练度。**

1. **对于出现的问题应更加灵活的去解决**

**要想达到这一目的仍需大量的代码练习，这样才能在运行结果出现问题是第一时间发现问题并进行解决。如解决的过程中遇到困难，则应该首先利用f11进行逐行检查，进而确定错误类型以及错误发生的位置，如出现看不懂的报错应将出错的代码复制到网络上查询。**

1. **对逻辑运算以及各种数学问题的解决更加熟练**

**应该对计算机可以解决的数学问题更加了解，认识到计算机可以解决的数学问题的特点，比如实验中涉及到将字符串表示的16进制数转为10进制的操做，应更熟悉不同进制之间的转换方式如果不了解进制之间的转化，或者不了解字符ASCII码之间的转化关系，便难以做出相应程序。**

**其次对计算机中的逻辑关系应当有这更深刻的了解，比如了解在进行实验时，先对实验的整体有大概了解并整理出大概思路，之后将逻辑完善，这样才能编写出更简洁的代码，对数学问题有更多的了解也可以拓宽我们的思路，增加我们解决问题的能力。**