**计算机程序设计基础（C++）**

**实验报告**

专业班级 软件工程2301班

学 号 8209230104

姓 名 向一洋

**实验报告成绩：**

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

**批阅老师:\_\_\_\_\_\_\_\_**

**实验三 函数**

1. **实验目的与要求**

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。

1. **实验内容**

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。

（2）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. **实验步骤、算法与结果分析**

**第一题:**

定义最大公约数

**算法：**

输入两个数字

**取最小值为t**

t可以被m和n整除

**Y N**

t=t-1

t

最大公约数为t

**最小公倍数：两数相乘除以最小公约数**

**Code:**

#include<iostream>

using namespace std;

int gcd(int m, int n)

{

int t;

t = m < n ? m : n;

for (int i=m<n?m:n;; i--)

{

if ((m % i == 0 && n % i == 0) == 1)

{

t = i;//此处不能再用变量t，否则变量定义重复

break;

}

}

return t;

}//定义取最大公约数的函数

int lcm(int m, int n)

{

int t;

int z;

t = gcd(m, n);

z = m \* n / t;

return z;

}//定义取最小公倍数的函数（两数相乘除以最小公约数）

int main()

{

int m, n;

cout << "please input m,n:" << endl;

cin >> m >> n;

cout << "m与n的最大公约数为：" << gcd(m, n) << endl;

cout << "m与n的最小公倍数为：" << lcm(m, n) << endl;

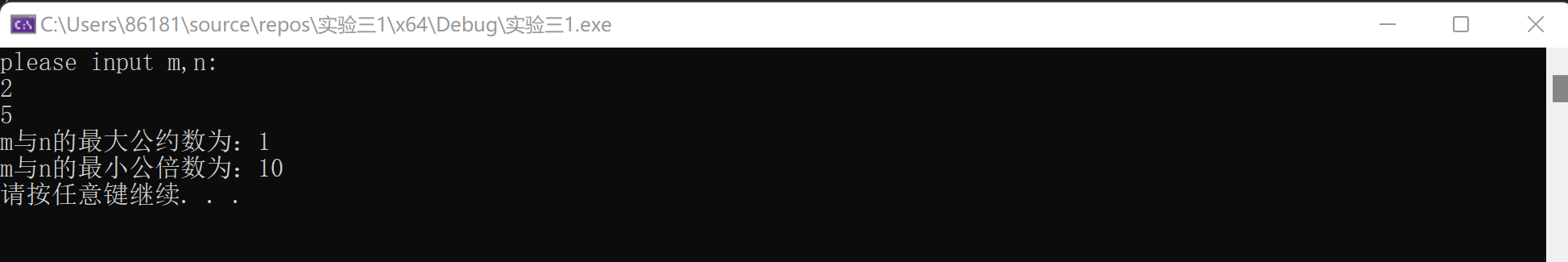
system("pause");

return 0;

}//主函数

**Result:观察输入2、5与20、25时的结果。**

**输入2、5：**

****

**输入20、25：**

****

**第二题:**

**算法：**

定义判断素数的函数

输入num

**num<=1 num>1**

i=2

False

num%i==0

**Y N i<num**

i++

false

**i>=num**

True

**Code:**

#include<iostream>

using namespace std;

bool is\_prime(int num)

{

if (num <= 1)

{

return false;

}//小于等于1的数不是素数

if (num > 1)

{

for (int i = 2; i < num; ++i)

{

if (num % i == 0)

{

return false;

}

}//如果能进入for循环且不退出，能被其他数字整除，则不是素数

return true;

}

}//定义判断是否为素数的函数

int main()

{

int k = 1;

int i = 1;

for (int num = 2;; num++)

{

if (k <= 200)

{

if (is\_prime(num) == 1)//如果是素数

{

if (i % 10 != 0)

{

cout << num <<'\t';//endl换行

k++;

i++;

}

else

{

cout << num<<'\t' << endl;//转义字符加“”

k++;

i++;

}//if条件判断实现换行

}

}//if条件判断实现200素数

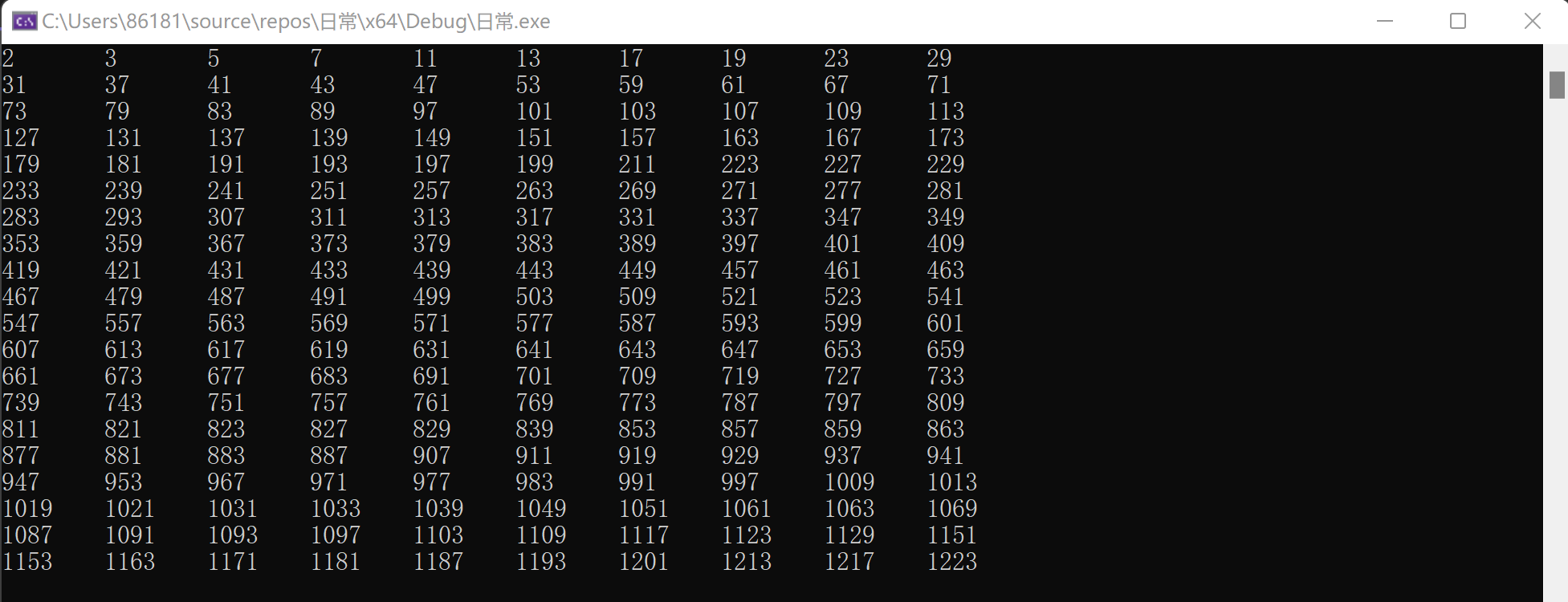
}

system("pause");

return 0;

}

**Result:**

****

**第四题:**

输入三边长度

**算法：**

构成三角形

**Y N**

Wrong

输出面积

**Code:**

#include<iostream>

using namespace std;

bool is\_valid(double side1, double side2, double side3)//定义判断是否构成一个三角形的函数

{

if (side1 + side2 > side3 && side1 + side3 > side2 && side2 + side3 > side1)

{

return 1;

}

else

{

return 0;

}

}

double double\_area(double side1, double side2, double side3)//定义求三角形面积的函数

{

int s, area;

s = (side1 + side2 + side3) / 2;

area = sqrt(s \* (s - side1)\*(s - side2)\*(s - side3));

return area;

}

#include<cmath>

using namespace std;

bool is\_valid(double side1, double side2, double side3);//函数的声明

double double\_area(double side1, double side2, double side3);//函数的声明

#include<iostream>

#include"mytriangle.h"

using namespace std;

int main()

{

double side1, side2, side3;

cout << "please input side1,side2 and side3:" << endl;

cin >> side1 >> side2 >> side3;

if (is\_valid(side1, side2, side3) == 1)

{

cout << "the area is:" << double\_area(side1, side2, side3) << endl;

}//如果可以构成三角形就输出面积

else

{

cout << "the side length is not wrongful" << endl;

}//如果无法构成三角形就显示错误

system("pause");

return 0;

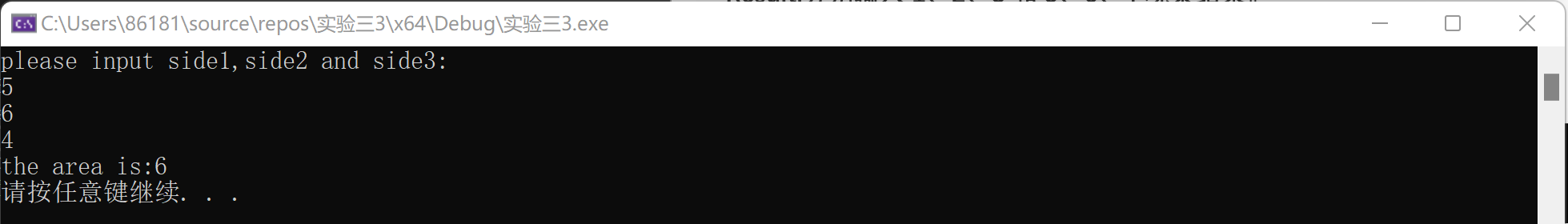
}

**Result:分别输入1、2、3和5、6、4观察结果。**

**输入1、2、3：**

****

**输入5、6、4：**

****

**第五题:**

**算法：**

X个桃子

**第一天**

**第二天**

**…………**

**第十天**

**Code:**

#include<iostream>

using namespace std;

int main()

{

int i = 1;

for (int k = 0; k < 9; k++)

{

i = (i + 1) \* 2;

}

cout << i << endl;

system("pause");

return 0;

}

**Result:**

****

1. **遇到的问题和解决方法**

第一个实验中，当在某个函数的for循环外已经定义了变量t，在for中就要引用一个新的变量i，再将t的值赋给i，而不可以直接再int t造成重复定义（i相当于一个桥梁）。第四个实验中注意数学的严谨性（即丢分点是判断输入的三个边是否可以构成一个三角形），此外在写第四个实验时最后报错的原因实在是找不到了，最后通过询问同学才知道是源文件中没有写包括头文件的代码造成的错误。

**五、体会**

在写实验三的过程中，我明白了细节决定成败。很多时候，程序运行不出来也不是算法没搞懂，而是细节没有处理好。例如在本次实验中，第四个实验中判断是否构成三角形和包括头文件都是容易丢分的点，就像在定义字符串时要写#include<string>一样，很容易被人忽略。因此，在今后的学习过程中，不仅仅对C++适用，而是对生活中的很多方面都适用，学习中遇到的所有问题都要细心对待。

**实验四 数组与指针**

**一、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。

**二、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 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 Enter 10 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 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 the first string: welcome

Enter the second string: We welcome you!

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

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 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\_\_\_\_\_\_\_\_

(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]<<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

(3) 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）。

**三、算法分析、程序与结果**

**（一）数组**

1. 算法：

开始

依次输入数字

循环判断输入的数字是否与前面的数字相同

**Y**

**N N**

输出此数字

**Y**

结束

程序：

#include<iostream>

using namespace std;

int main()

{

int i = 0;

const int SIZE = 10;//规定数组中元素个数

int arr[SIZE];

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

{

cout << i+1 << ":";

cin >> arr[i];

cout << endl;

}//输入数组中十个元素并放到数组中

cout << arr[0];//第一个元素一定会被输出

//接下来的思路是：两个for循环嵌套，拿某个元素前面的元素与此元素比较，若都不相等则输出这个元素，否则不输出这个元素

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

{

for (int m = i - 1; m>=0; m--)

{

if (arr[m] == arr[i])

{

break;

}//相同就不输出正在检索的数字

else

{

if (m == 0)

{

cout << arr[i];

}

}//不同就输出正在检索的数字

}//看arr[i]以前的数字是否与arr[i]相同

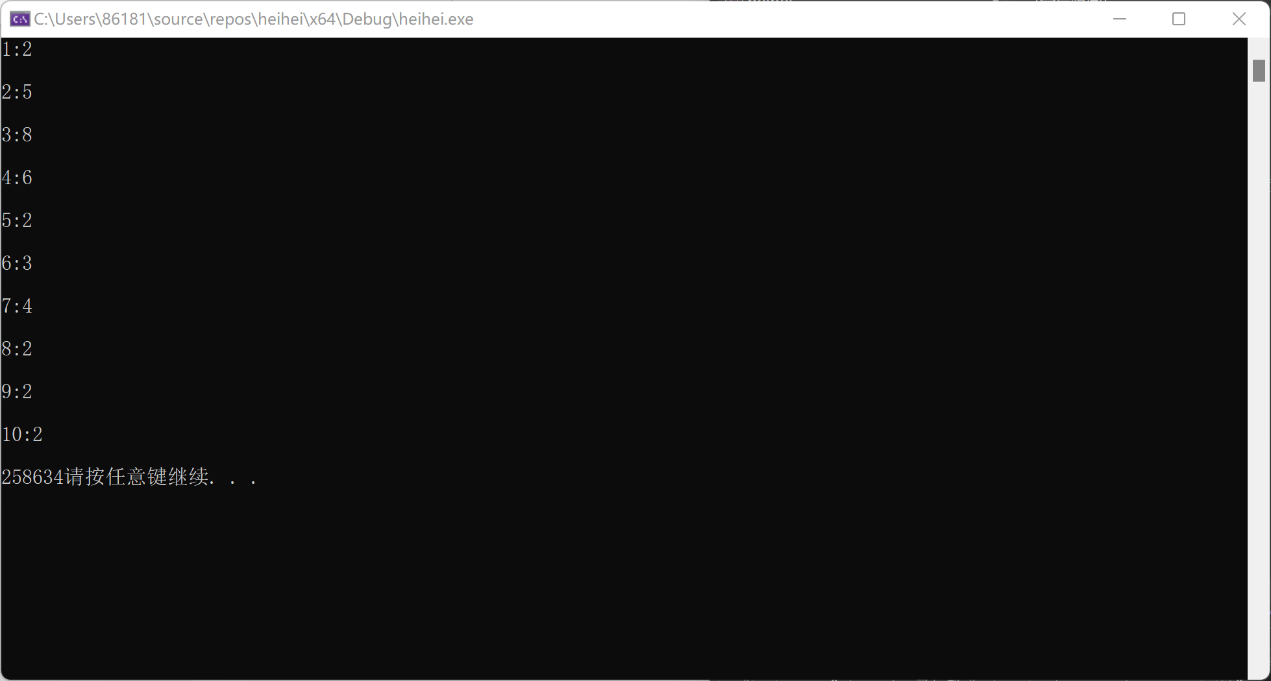
}

system("pause");

return 0;

}

结果：输入2586234222



1. **算法：**

开始

输入乱序数组

循环下列操作size-1次

**N**

判断输入数字>下一个数字

交换两个数字位置

输入顺序数组

结束

**程序：**

//起泡排序

#include<iostream>

using namespace std;

void sort(int n,double p[])//n为元素的个数 p为数组名

{

for (int j=0; j < n-1; j++)//j表示比较多少轮

{

for (int i = 0; i < n - j-1; i++)//i表示每一轮比较的次数

{

double temp = 0;

if (p[i] > p[i + 1])

{

temp = p[i];

p[i] = p[i + 1];

p[i + 1] = temp;

}

}

}

}

int main()

{

double a[10]={};

cout << "please input 10 numbers:";

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

{

cin >> a[i];

cout << endl;

}

sort(10, a);

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

{

cout << a[j];

cout << endl;

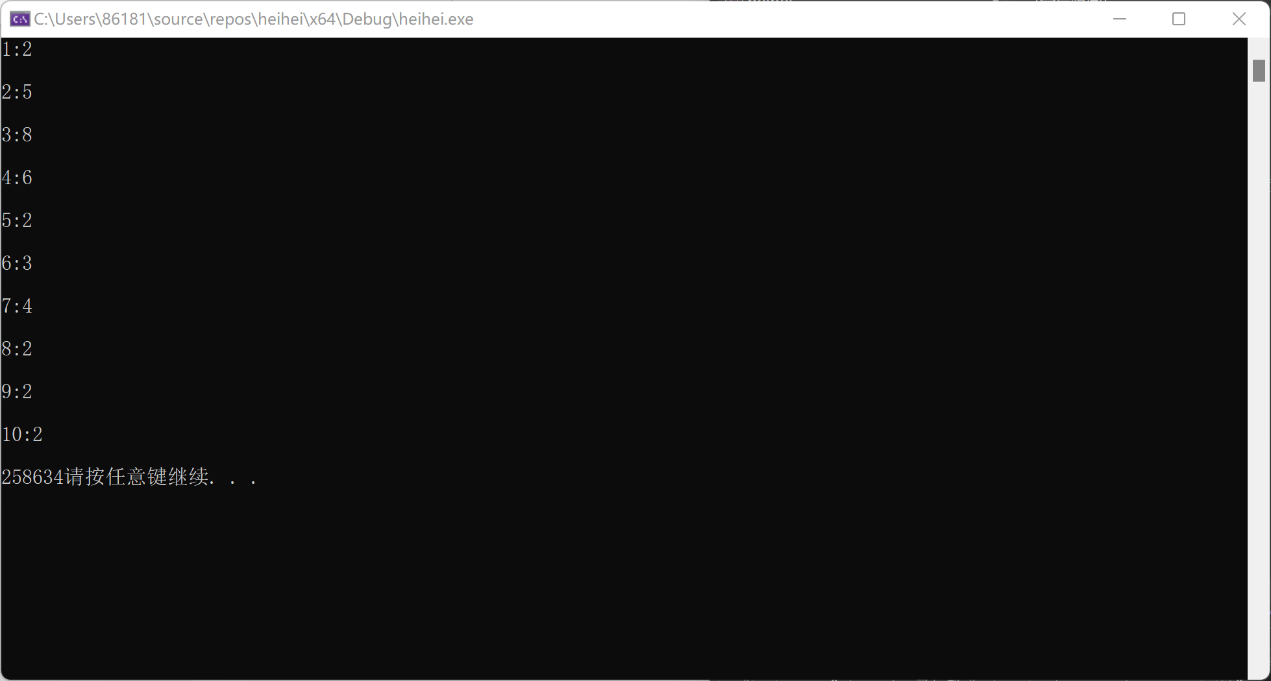
}

system("pause");

return 0;

}

结果：

****

1. **算法：**

输入100个bool数组

开始

所有bool均为false，i=1，n=1，m=1

输入第n学生

**N Y**

Bool list[m]=false

Bool list[m]=true

**Y**

i =n

m =m+i

**Y**

n ++

n <=100

**N**

输出list数组中为true的数字

结束

**程序：**

#include<iostream>

using namespace std;

int main()

{

const int SIZE = 100;

bool arr[SIZE];

for (int k=0; k < SIZE; k++)//?建议参考memset函数

{

arr[k]=false;

}//输入数组的元素

for (int n=1; n <= SIZE; n++)//n个学生 学生Sn

{

for (int m = n; m <= SIZE; m+=n )//某一个学生进行操作，m为第m个存物柜 //?这里应该不用n+1吧

{

/\*if (arr[m - 1] = false)//？双等号双等号

{

arr[m-1] = true;

}

else

{

arr[m-1] = false;

}\*/

arr[m-1]=arr[m-1]?false:true;//清楚！

}

}//对数组进行一个改变

for (int j = 0; j < SIZE; j++)

{

if (arr[j])//？这个地方一定要注意，if里面的等号一定是双等号！//并且布尔值是可以直接用的

{

//int l = j + 1;

cout << j+1 << "\t";

}

}//输出开着的柜子的号码

system("pause");

return 0;

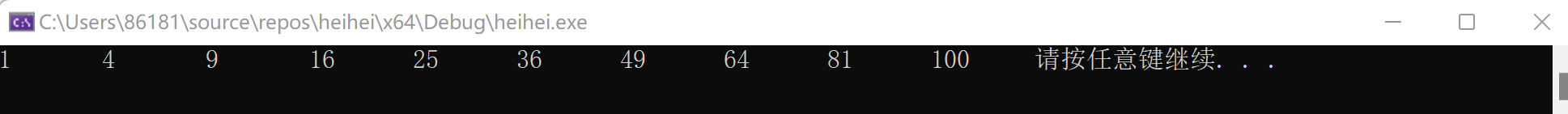
}

//输出了所有的号码？？？

//代码缩进注意一下

//三目运算符的简便使用！

**结果：**

****

1. **算法：**

开始

输入两个数组大小及数值

将前两个数组赋给第三个数组

排序-第三个数组

结束

**程序：**

#include<iostream>

using namespace std;

void merge(int list1[], int size1,int list2[], int size2,int list3[])//将1、2融合入3

{//sizen 是数组元素个数

for (int i = 0; i < size1; i++)//i是指数组元素的角标

{

list3[i] = list1[i];

}

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

{

list3[size1 + j] = list2[j];

}

}//将数组1、2的元素融入到数组3

void arrange(int list3[], int size3)

{

int temp;

for (int j = 0; j < size3 - 1; j++)

{

for (int i = 0; i < size3 - 1; i++)

{

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

{

temp = list3[i];

list3[i] = list3[i + 1];

list3[i + 1] = temp;

}

}

}//从大到小排序

}

//int main()

//{

// int size1, size2, size3;

//

// cout << "enter size1:";

// cin >> size1;

// cout << "enter size2:";

// cin >> size2;

// size3 = size1 + size2;

// const int n = size1;

// const int k = size2;

// const int p = size3;

// int list1[n], list2[k], list3[p];//开始报错：数组定义时，变量不能做下标

//

// cout << "Enter list1:";

// cin >> size1;

// for (int i = 0; i < n; i++)//输入list1数组

// {

// cin >> list1[i] >> "\t";

// }

// cout << "Enter list2:";

// cin >> size2;

// for (int j = 0; j < k; j++)//输入list2数组

// {

// cin >> list2[j] >> "\t";

// }

// merge(list1, n, list2, k, list3);

// arrange(list3, p);

// cout << "The merged list is" << "\t";

// for (int m = 0; m < p; m++)

// {

// cout << list3[m];

// }

// system("pause");

// return 0;

//}

int main()

{

int size1, size2, size3;

cout << "enter size1:";

cin >> size1;

cout << "enter size2:";

cin >> size2;

size3 = size1 + size2;

int\* list1 = new int[size1];

int\* list2 = new int[size2];

int\* list3 = new int[size3];

cout << "Enter list1:";

for (int i = 0; i < size1; i++)//输入list1数组

{

cin >> list1[i];

}

cout << "Enter list2:";

for (int j = 0; j < size2; j++)//输入list2数组

{

cin >> list2[j];

}

merge(list1, size1, list2, size2, list3);

arrange(list3, size3);

cout << "The merged list is" << "\t";

for (int m = 0; m < size3; m++)

{

cout << list3[m];

}

delete[]list1;

delete[]list2;

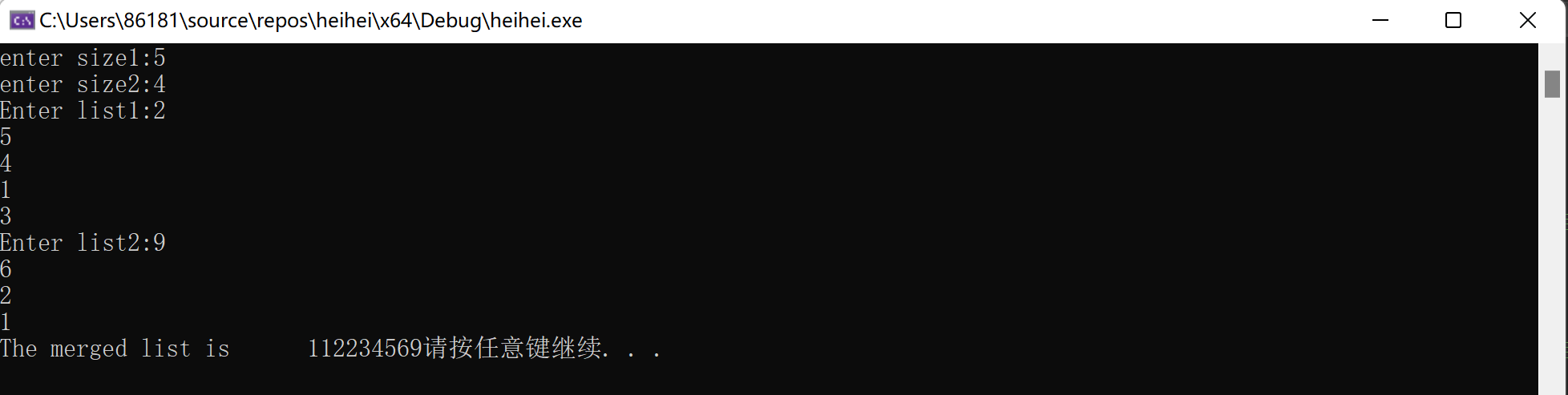
delete[]list3;

system("pause");

return 0;

}

**结果：**

****

1. **算法：**

开始

编写indexOf函数

输入数组s1，s2

调用indexOf函数

判断s1是否为s2的子串

**N Y**

输出s1首字母在s2中的下标

输出-1

结束

**程序：**

#include<iostream>

#include<cstring>//用到定义字符型数组要加

using namespace std;

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

{

int size1, size2, i = 0, flag = 0;

size1 = strlen(s1);//strlen（）是求字符型数组和字符串长度的。不会将/0之后的元素加入计算

size2 = strlen(s2);

for (int j = 0; j < size2; )

{

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

{

flag++;

i++;

if (s1[i] == 0)

return j - size1 + 1;

}

else if (flag)//进入这个条件是前一组s1与s2同，但是再下一个元素不同，此时让j值不变，i=0，避免we与wwwe无法正确输出。

{

i = 0;

continue;//continue是终止本次循环，直接进行下次循环

}

else i = 0;

j++;//无论如何每一遍只要flag不为1，都会j++

flag = 0;

}

return -1;

}

int main()

{

char s1[1000], s2[1000];

cout << "Enter the first string:";

cin.getline(s1, 1000);//cin.getline(字符型数组名，数组长度)来输入字符型数组

cout << "Enter the second string:";

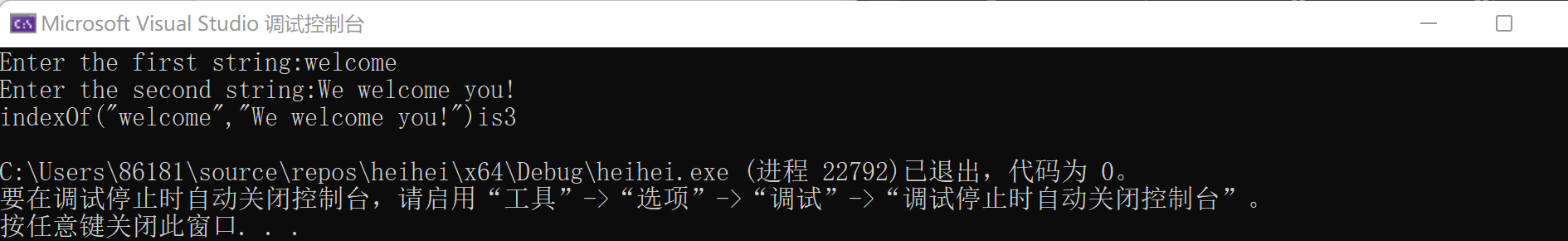
cin.getline(s2, 1000);

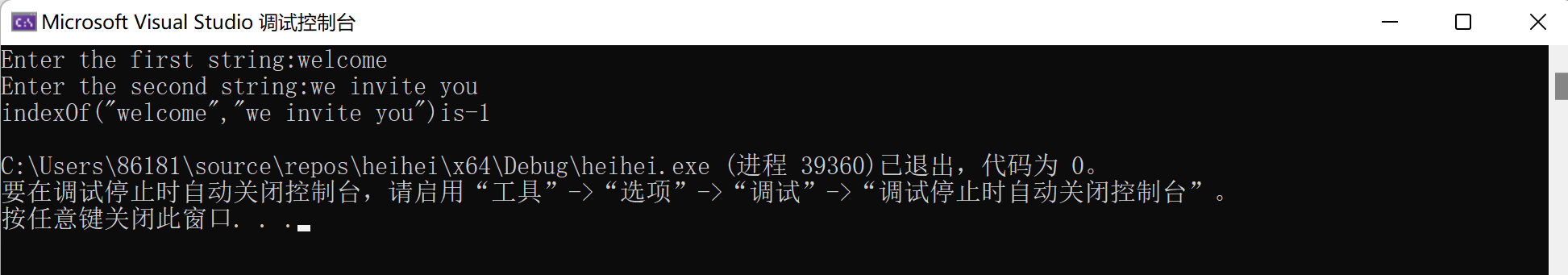
cout << "indexOf(\"" << s1 << "\",\"" << s2 << "\")is" << indexOf(s1, s2) << endl;

return 0;

}

**结果：**

****

****

1. **算法：**

开始

写入计算字母的函数、输出字母出现次数的函数

输入字符串

调用函数

输出结果

结果

**程序：**

#include<iostream>

using namespace std;

#include<cstring>

void count(const char s[1000], int counts[26])//完善counts数组，对字母进行计算

{

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

counts[j] = 0;

for (int i = 0; i < 1000; i++)//遍历字符串，每读入一个字母，都相应counts数组元素加1

{

counts[s[i] - 'a']++;

counts[s[i] - 'A']++;//ascll码，char强制转化为int

}

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

{

if (counts[m] != 0)//输出出现过的字母的次数

{

char t = m+97;//ascll码，int强制转化为char（a的int为97）

cout <<t<< ":" << counts[m] << "times" << endl;

}

}

}

int main()

{

char s[1000]="";

int counts[26];

cin.getline(s, 1000);//传入字符串

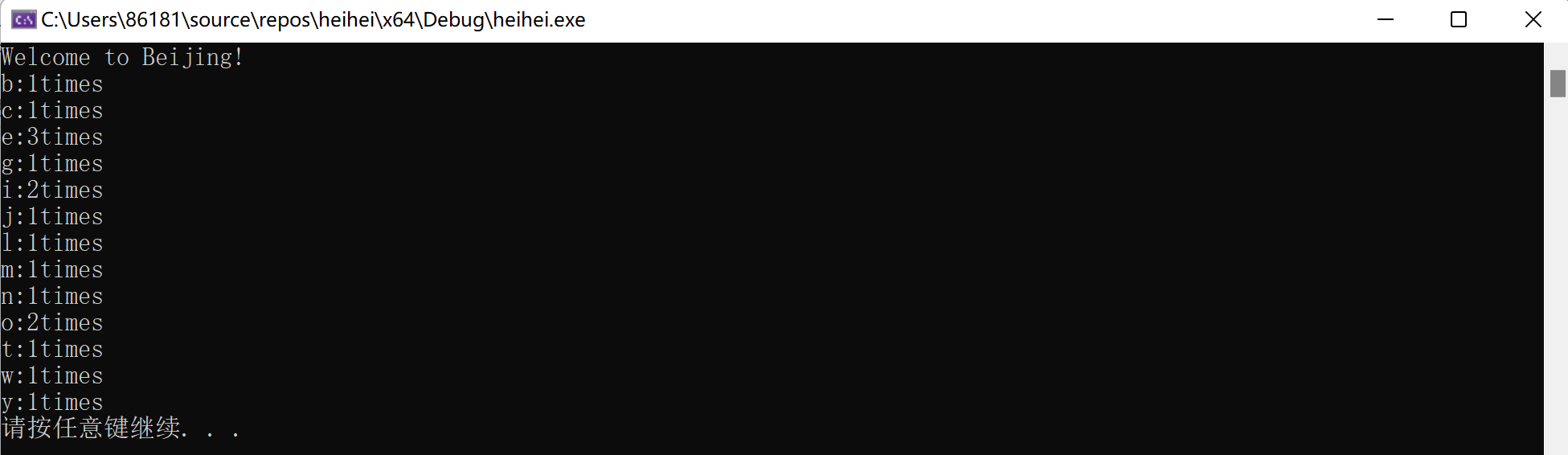
count(s,counts);//无法将char转换为const char？

system("pause");

return 0;

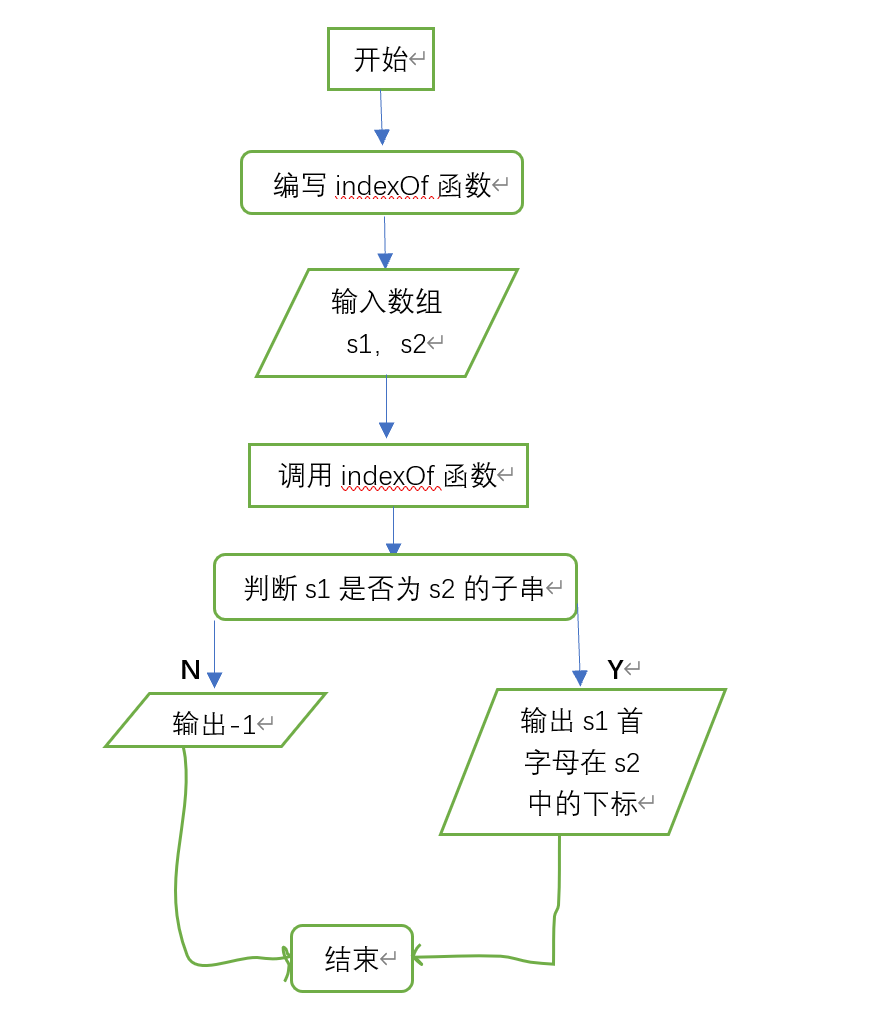
}

**结果：**

****

**（指针）**

1. **算法：**

****

**程序：**

#include<iostream>

#include<cstring>

using namespace std;

int indexof(const char\* s1, const char\* s2)

{

int size1 = strlen(s1);

int size2 = strlen(s2);

int i = 0;

int flag = 0;

for (int j = 0; j < size2;)

{

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

{

i++;

flag++;

if (s1[i] == 0)

return j - size1 + 1;

}

else if (flag)

{

i = 0;

continue;

}

else i = 0;

j++;

flag = 0;

}

return -1;

}

int main()

{

char s1[1000];

char s2[1000];

cout << "string1:";

cin.getline(s1, 1000);

cout << "string2:";

cin.getline(s2, 1000);

cout << "indexOf(\"" << s1 << "\",\"" << s2 << "\")is" << indexof(s1, s2) << endl;

system("pause");

return 0;

}

**结果：**

****

****

1. **算法：**

开始

编写parseHex函数

逐个读入字母，并转化，再加和

输入数组

结束

**程序：**

#include<iostream>

#include<cstring>

#include<string>

using namespace std;

//stoi函数作用：将n进制字符串转化为十进制数

int parseHex(const char\* const hexString);

int main()

{

int size;

cout << "the size of your string is:";

cin >> size;

char\*hexString=new char[size];

cout << "please input your string:";

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

{

cin >> hexString[i];

}

cout<<parseHex(hexString);

delete[]hexString;//释放数组

system("pause");

return 0;

}

int parseHex(const char\* const hexString)

{

string name1;

int size = strlen(hexString);

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

{

name1 = name1 + hexString[i];

}//把字符数组转换为字符串

return std::stoi(name1,0,16);

}

**结果：**

****

****

1. **算法：**

建立new动态数组

开始

输入数组

动态检测数组输出

构建并调用排序函数

输出排序后的数组

结束

**程序：**

#include<iostream>

using namespace std;

void sort(int\*p,int size)

{

int temp;

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

{

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

{

if (\*(p+j) > \*(p+j + 1))

{

temp = \*(p+j);

\*(p+j) = \*(p+j + 1);

\*(p+j + 1) = temp;

}

}

}

}

int main()

{

int size;

cin >> size;

int\* p = new int[size];

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

{

cin >> \*(p+i);

}//建立动态数组

cout << p << '\t' << \*p<<'\t';//观察指针及指针指向的内容

sort(p,size);

for (int j = 0; j < size; j++)

{

cout << \*(p + j) << '\t';

}

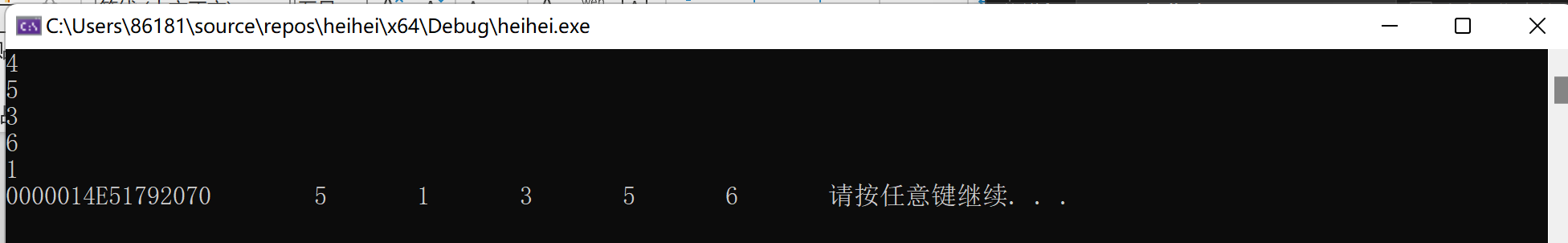
delete p;

system("pause");

return 0;

}

**结果：**

****

**四、遇到的问题与解决方法**

在写数组的第三个实验的时候，两个数组中元素相等的表示我只用了一个等号（=表示赋值，==用与比较），结果一直运行不正确，让其他人看了看，才发现这个问题，说出来我也知道是怎么回事，但是当时写的时候就没有注意到，甚至检查的时候也没检查出来，说明自己不够细心没有特别关注到这个问题。在数组的第四个实验中，最后一直在对我的数组的定义报错，后来才知道是数组定义时变量不能作下标，int arr[size]不可以，可以用动态数组解决问题，即利用new开辟堆区内存，int\*arr=new int[size]。写数组第五个实验，代码改了很多很多版本，因为每次的代码都能运行，但总能找到个反例来运行从而证明我这个代码是有问题的，第一个代码发现了一个问题，改完后第二个代码没有了第一个问题却出现了另外一个问题，以此类推，最终我发现了我思维的漏洞都有哪些，最后进行了修改。数组的第六个实验，我开始一直在想数字和字母要怎样相互转化，看过csdn后才明白ASCII码的作用，之前在学的时候ASCII码对我就很陌生，写完这个实验，对ASCII码又有了进一步更深刻的了解。

**五、体会**

等号与双等号的问题要细心对待，还有写数组第三个实验的时候，我将我的程序发给了一个学长，学长给了我一些建议，其中印象比较深刻的是一个if else语句，换成三目运算符来做，整个程序简介了不少，所以以后目标不仅仅是要把程序做出来，更要做的好。写数组第四个实验的时候，我一直觉得我的数组的定义没有问题，便问同学，问完才知道我的问题所在，并且以前对动态数组知之甚少，为了写好这个程序学懂了动态数组，我觉得写实验真的是提高代码水平的很重要的一步哇。数组的第五个实验真正让我体会到了，算法是写代码最重要的一步，算法写的不对，在一步步改正的过程中是很痛苦的，也可能bug越改越多，因此以后要多花一些时间思考我的算法是否正确可行全面。