**计算机程序设计基础（C++)**

**实验报告**

专业班级：

学 号：

姓 名：

**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**

**3.1**

#include <iostream>

using namespace std;

int yin(int m,int n)

{

int result;

int i;

for(i=1;i<=m&&i<=n;i++)

{

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

{

result=i;

}

}

return result;

}

int bei(int m,int n)

{

int result;

int i,c;

if (m < n)

{

c = n;

}

else

{

c = m;

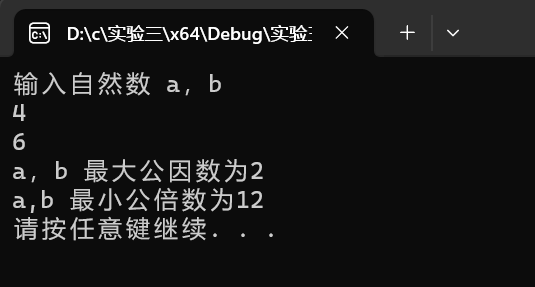
}

for(i=c;i<=m\*n;i++)

{

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

{

 result=i;break;

}

}

return result;

}

int main()

{

int a, b, c, d;

cout << "输入自然数 a，b" << endl;

cin >> a >> b;

c = yin(a, b);

d = bei(a, b);

cout << "a，b 最大公因数为" << c << endl << "a,b 最小公倍数为" << d << endl;

system("pause");

return 0;

}

3.2

#include <iostream>

using namespace std;

bool is\_prime(int num)

{

if (num == 2)

{

return true;

}

if (num % 2 == 0 || num < 2)

{

return false;

}

else

{

for (int i = 3; i \* i <= num; i += 2)

{

if (num % i == 0)

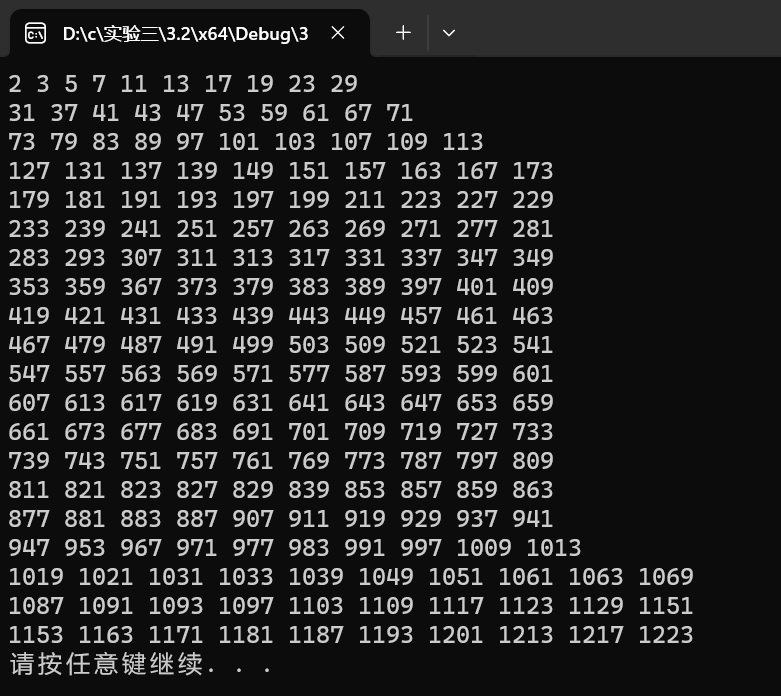
{

return false;

}

}

return true;

 }

}

int main()

{

int result, i, a, count;

count = 0;

a = 1;

while (count < 200)

{

a++;

if (is\_prime(a))

{

count++;

cout << a << " ";

if (count % 10 == 0)

{

cout << endl;

}

}

}

system("pause");

return 0;

}

3.3

3.3.h

#pragma once

#include <iostream>

using namespace std;

double celsius\_to\_fah(double cel)//摄氏至华氏

{

double fah;//fah 指华氏

fah = 9.0 / 5 \* cel + 32;

return fah;

}

double fahrenheit\_to\_cels(double fah)//华氏至摄氏

{

double cel;

cel = (5.0 / 9) \* (fah - 32);

return cel;

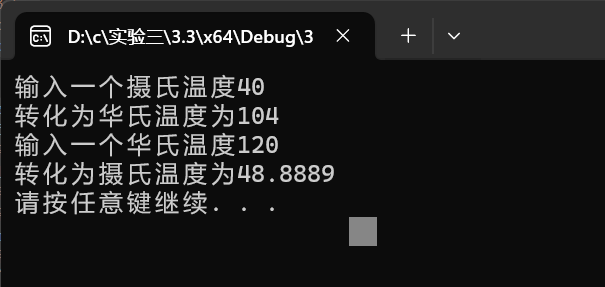
}

3.3.cpp

#include"3.3.h"

#include <iostream>

using namespace std;

int main()

{

double cel;

double fah;

cout<<"输入一个摄氏温度" ;

cin>>cel;

cout<<"转化为华氏温度为" << celsius\_to\_fah(cel)<<endl;

cout<<"输入一个华氏温度" ;

cin>>fah;

cout<<"转化为摄氏温度为" <<fahrenheit\_to\_cels(fah) <<endl;

system("pause");

return 0;

}

3.5

#include<iostream>

using namespace std;

int f(int i)

{

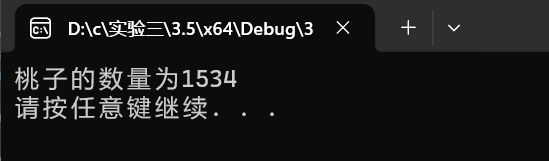
return (i + 1) \* 2;

}

int main()

{

int num = 1;

 for (int day = 10; day > 1; day--)

{

num = f(num);

}

cout << "桃子的数量为" << num<<endl;

system("pause");

return 0;

}

1. **遇到的问题与解决方法**

在做第二个实验的时候，起初我把每十个换一行的if句子写错循环了，导致中间出现了很多个空行，后面有检查了一下代码，发现了自己的错误。

在做第一个实验的时候，不得不承认，我忘记了素数该怎么计算，因为我转专业来的，大一学的是python，跟c++也有所不同，后面查阅了一下csdn，用了比较简单的方式表示出来了。

还有就是华氏度和摄氏度的问题，发现自己写出的代码运行结果和题目中给的好像有点出入，最后还是查找的浏览器确定了一下自己写的并没有什么问题。

1. **体会**

虽然比同学们多了一年的代码经验，但是太久不敲基本的代码的话还是会有一些遗忘的部分，而且c++与python确实有很大的差异，不应该不在意基础的实验，反而应该稳扎稳打，把基础打牢。

**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\_\_\_\_\_\_\_\_

(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

#include <iostream>

using namespace std;

void sortFromSmallToLarge(int abc[],int c)

{

int a;

int b;

int k;

for(a = 0;a < c-1;a++)

{

for(b = 0;b < c-a-1;b++)

{

if(abc[b] > abc[b+1])

{

int temp;

temp = abc[b];

abc[b] = abc[b + 1];

abc[b + 1] = temp;

}

}

}

}

void swap(int num[],int abc[])

{

int a;

int i;

int b;

int c = 0;

bool y;

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

{

y = true;

for(a = i+1;a<10;a++)

{

if(num[i]==num[a])

y = false;

}

if(y)

{

abc[c] = num[i];

c++;

}

}

sortFromSmallToLarge(abc,c);

for(int d = 0;d < c;d++)cout << abc[d] << " ";

}

int main()

{

int num[10];

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

{

cin >> num[a];

}

int abc[10];

swap(num,abc);

system("pause");

return 0;

}

4.2

#include <iostream>

using namespace std;

void sortFromSmallToLarge(int abc[])

{

int a, b, k;

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

{

for (b = 0; b < 9 - a; b++)

{

if (abc[b] > abc[b + 1])

{

int temp;

temp = abc[b];

abc[b] = abc[b + 1];

abc[b + 1] = temp;

}

}

}

}

int main()

{

int abc[10];

cout << "请依次输入 10 个数字:" << endl;

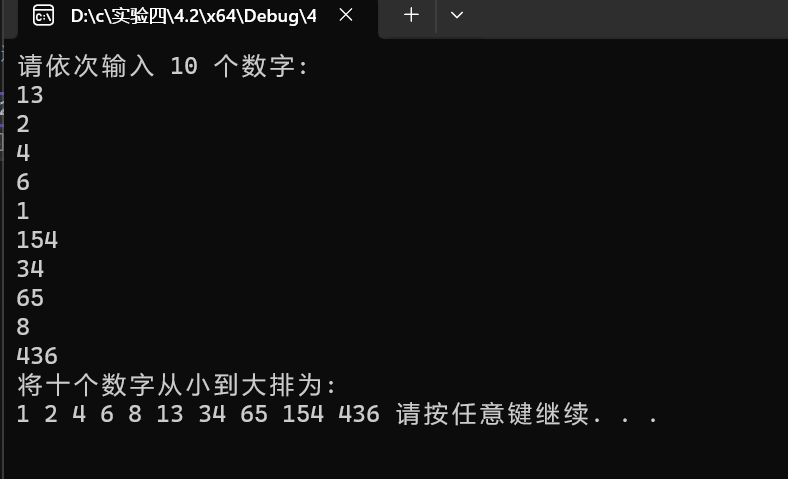
for (int i = 0; i < 10; i++)

{

cin >> abc[i];

}

sortFromSmallToLarge(abc);

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

{

cout << abc[a] << " ";

}

system("pause");

return 0;

}

4.3

#include <iostream>

using namespace std;

int main()

{

int b, c, i;

int box[100];

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

{

box[a] = 0;

}

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

{

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

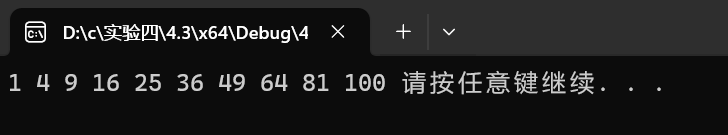
{

box[b]++;

}

}

for (c = 0; c < 100; c++)

 {

if (box[c] % 2 == 1)

cout << c + 1 << " ";

}

system("pause");

return 0;

}

4.4

#include <iostream>

using namespace std;

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

{

bool y = true;

int size3 = 0;

int c = 0, d = 0;

for (; c < size1; c++)

{

y = true;

d = 0;

for (; d < size2; d++)

{

if (list1[c] == list2[d])

{

y = false;

break;

}

}

if (y)

{

list3[size3] = list1[c];

size3++;

}

}

d = 0;

for (; d < size2; d++)

{

list3[size3] = list2[d];

size3++;

}

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

{

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

{

int temp;

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

{

temp = list3[g];

list3[g] = list3[g + 1];

list3[g + 1] = temp;

}

}

}

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

{

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

}

}

int main()

{

int size1, size2;

cout << "size 1" << endl;

cin >> size1 ;

int\* list1 = new int[size1];

cout << "size 2" << endl;

cin >>size2;

int\* list2 = new int[size2];

int\* list3 = new int[size2+size1];

cout << "list1" << endl;

int a = 0;

int b = 0;

for (; a < size1; a++)

{

 cin >> list1[a];

}

cout << "list2" << endl;

for (; b < size2; b++)

{

cin >> list2[b];

}

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

system("pause");

return 0;

}

4.5

#include <iostream>

#include <cstring>

#include <string>

using namespace std;

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

{

int length1 = strlen(s1);//s1 是子串

int length2 = strlen(s2);

bool isSon = true;

int d = 0;

int i = 0;

int c = 0;

int a;

for (a = 0; a < length2; a++)

{

if (\*(s1 + i) == \*(s2 + a))

{

i++;

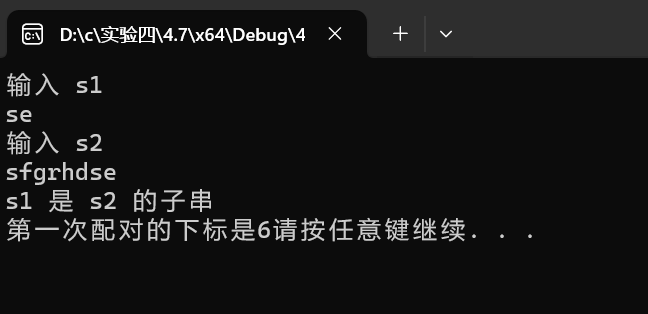
}

else i = 0;

if (i == length1)

{

cout << "s1 是 s2 的子串" << endl << "第一次配对的下标是" << a - length1 + 1;

 break;

}

}

}

int main()

{

char s1[100], s2[100];

cout << "输入 s1" << endl;

cin.getline(s1, 100);

cout << "输入 s2" << endl;

cin.getline(s2, 100);

indexof(s1, s2);

system("pause");

return 0;

}

4.6

#include <iostream>

#include <cstring>

using namespace std;

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

{

int len = strlen(s);

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

{

counts[b] = 0;

}

int a = 0;

for (; a < len; a++)

{

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

{

if (s[a] == 'a' + i || s[a] == 'A' + i)

counts[i]++;

}

}

}

void Cout(int counts[])

{

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

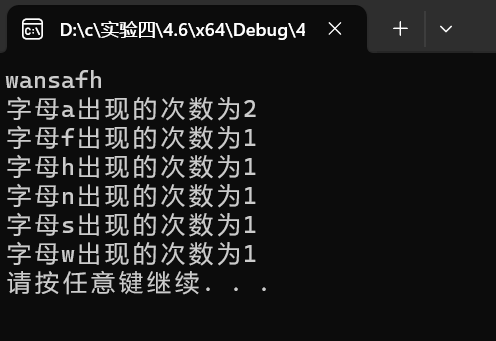
{

if (counts[i] != 0)

{

cout << "字母" << d << "出现的次数为" << counts[i] << endl;

}

 }

}

int main()

{

int counts[26];

char s[2];

cin >> s;

count(s, counts);

Cout(counts);

system("pause");

return 0;

}

4.7

#include <iostream>

#include <cstring>

#include <string>

using namespace std;

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

{

int length1 = strlen(s1);//s1 是子串

int length2 = strlen(s2);

bool isSon = true;

int d = 0;

int i = 0;

int c = 0;

int a;

for (a = 0; a < length2; a++)

{

if (\*(s1 + i) == \*(s2 + a))

{

i++;

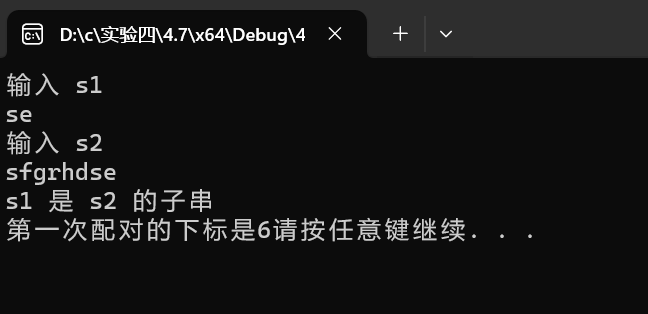
}

else i = 0;

if (i == length1)

{

cout << "s1 是 s2 的子串" << endl << "第一次配对的下标是" << a - length1 + 1;

 break;

}

}

}

int main()

{

char s1[100], s2[100];

cout << "输入 s1" << endl;

cin.getline(s1, 100);

cout << "输入 s2" << endl;

cin.getline(s2, 100);

indexof(s1, s2);

system("pause");

return 0;

}

4.8

#include <iostream>

#include <string>

#include <cmath>

using namespace std;

void eswitch()

{

int length;

cout << "请输入一个 16 进制的数字" << endl;

string hex;

cin >> hex;

length = hex.length();

int d = length - 1;

for (int c = 0; c < d; c++, d--)

{

int temp;

temp = hex[c];

hex[c] = hex[d];

hex[d] = temp;

}

int \*num=new int [length];

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

{

switch (hex[i])

{

case '0':num[i] = 0 \* pow(16, i); break;

case '1':num[i] = 1 \* pow(16, i); break;

case '2':num[i] = 2 \* pow(16, i); break;

case '3':num[i] = 3 \* pow(16, i); break;

case '4':num[i] = 4 \* pow(16, i); break;

case '5':num[i] = 5 \* pow(16, i); break;

case '6':num[i] = 6 \* pow(16, i); break;

case '7':num[i] = 7 \* pow(16, i); break;

case '8':num[i] = 8 \* pow(16, i); break;

case '9':num[i] = 9 \* pow(16, i); break;

case 'a':num[i] = 10 \* pow(16, i); break;

case 'b':num[i] = 11 \* pow(16, i); break;

case 'c':num[i] = 12 \* pow(16, i); break;

case 'd':num[i] = 13 \* pow(16, i); break;

case 'e':num[i] = 14 \* pow(16, i); break;

case 'f':num[i] = 15 \* pow(16, i); break;

case 'A':num[i] = 10 \* pow(16, i); break;

case 'B':num[i] = 11 \* pow(16, i); break;

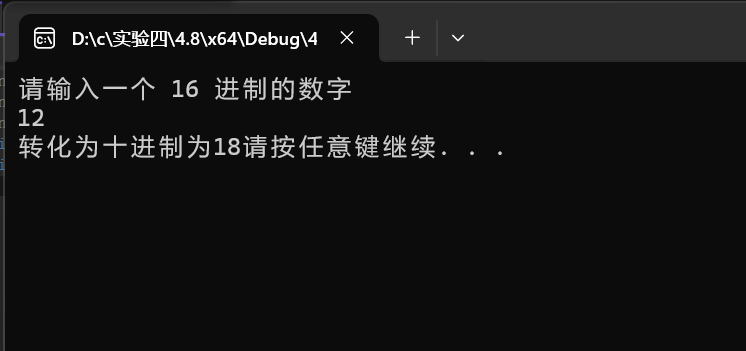
case 'C':num[i] = 12 \* pow(16, i); break;

case 'D':num[i] = 13 \* pow(16, i); break;

case 'E':num[i] = 14 \* pow(16, i); break;

case 'F':num[i] = 15 \* pow(16, i); break;

default:cout << "输入错误"; break;

 }

}

int sum = 0;

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

{

sum = sum + num[a];

}

cout << "转化为十进制为" << sum;

}

int main()

{

int a = 1;

eswitch();

system("pause");

return 0;

}

4.9

#include <iostream>

using namespace std;

void sort(int\* number,int size)

{

int a;

int b;

for (a = 0; a < size - 1; a++)

{

for (b = 0; b < size - 1 - a; b++)

{

if (number[b] > number[b + 1])

{

int temp;

temp = number[b];

number[b] = number[b + 1];

number[b + 1] = temp;

}

}

}

}

int main()

{

cout << "请输入数组的大小" << endl;

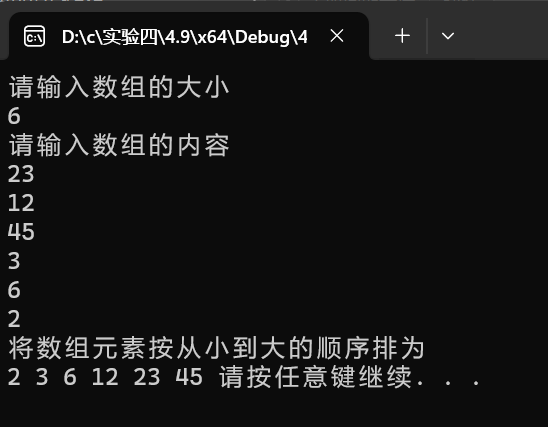
int n;

cin >> n;

int\* p = new int[n];

cout << "请输入数组的内容" << endl;

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

 {

cin >> p[i];

}

sort(p, n);

cout << "将数组元素按从小到大的顺序排为" << endl;

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

{

cout << p[i]<<" ";

delete p ;

system("pause");

return 0;

}

**四、遇到的问题与解决方法**

1.在这段代码中我遇到的最大的问题就是在指针问题里面的第四题，关于指针函数的问题，由于学c++对我来说也是有一点点时间距离的事情了，导致起初我有点没反应过来，后面又重新回顾了一下关于这方面的知识才写出来。

2.第二个问题就是关于数组排序的问题，由于在大二的时候在计算机算法设计与分析中学习了快速排序，差点下意识用快排，还好后面又看了一遍题目才及时改正。

**五、体会**

1.大二学习了快速排序，但是并不能忘记大一所学的内容，毕竟在python中，我使用最多的其实也是冒泡排序。

2.还是应该多桥一些代码，积累经验。