**计算机程序设计基础（C++)**

**实验报告**

专业班级： 软工2301

学 号： 8209230121

姓 名： 邓煜涛

**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？

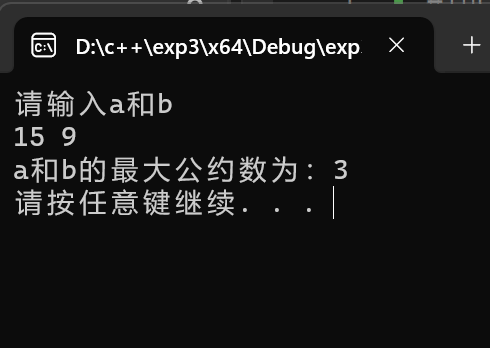
答：函数体内必须保证返回值类型的一致性，返回值必须符合函数的类型，否则会出错。

2. 本实验中主函数调用函数时采用的是何种传递方式？

答：实验1是引用传递，其他的都是值传递。

**四、算法分析，程序结果**

1. （1）

****

#include<iostream>

using namespace std;

int maxy(int &a, int &b) {

int temp = 0;

temp = a > b ? b : a;

for (;; temp--) {

if (a % temp == 0 && b % temp == 0) {

return temp;

}

}

}

int main() {

int a = 0, b = 0;

cout << "请输入a和b" << endl;

cin >> a;

cin >> b;

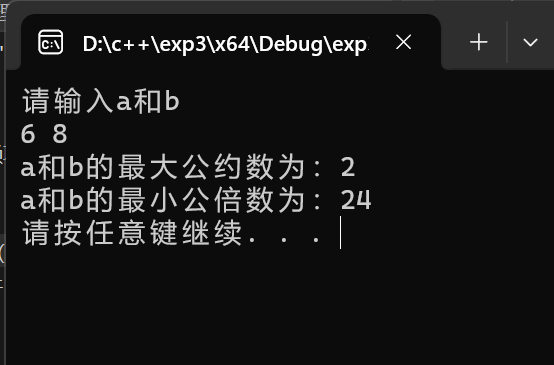
cout << "a和b的最大公约数为：" << maxy(a, b) << endl;

system("pause");

return 0;

}

1.（2）



#include<iostream>

using namespace std;

int maxy(int& a, int& b) {

int temp = 0;

temp = a > b ? b : a;

for (;; temp--) {

if (a % temp == 0 && b % temp == 0) {

return temp;

}

}

}

int minb(int& a, int& b) {

int temp = 0;

temp = a > b ? a : b;

for (;; temp++) {

if (temp % a == 0 && temp % b == 0) {

return temp;

}

}

}

int main() {

int a = 0, b = 0;

cout << "请输入a和b" << endl;

cin >> a;

cin >> b;

cout << "a和b的最大公约数为：" << maxy(a, b) << endl;

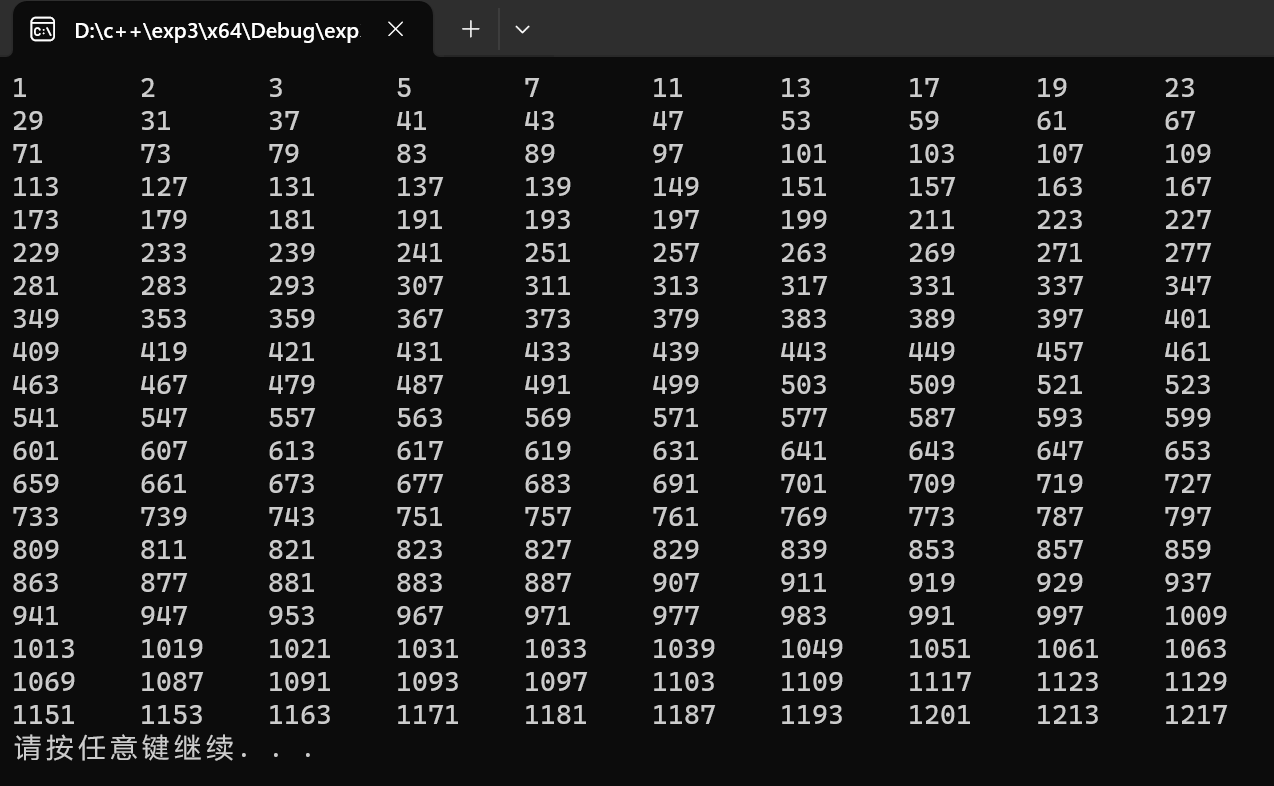
cout << "a和b的最小公倍数为：" << minb(a, b) << endl;

system("pause");

return 0;

}

2.



#include<iostream>

using namespace std;

bool isprime(int num) {

for (int i = num - 1; i > 1; i--) {

if (num % i == 0){

return false;

}

}

return true;

}

int main() {

bool isprime(int num);

int i = 0, sum = 0;

for (i=1;sum<200;i++) {

if (isprime(i)) {

cout << i << "\t" ;

sum++;

if (sum % 10 == 0) {

cout << endl;

}

}

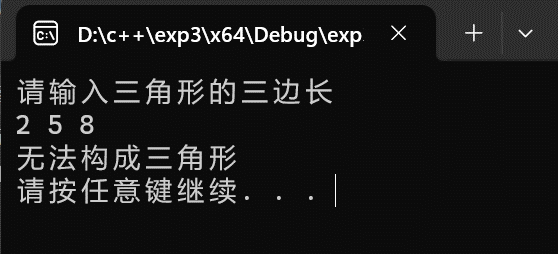
}

system("pause");

return 0;

}

4.



头文件：3.4mytriangle.h

#pragma once

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

double area(double side1, double side2, double side3);

源文件：3.4mytriangle.cpp

#include<iostream>

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

if (side1 + side2 <= side3 || side1 + side3 <= side2 || side2 + side3 <= side1) {

return false;

}

return true;

}

double area(double side1, double side2, double side3) {

double s = 0;

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

return sqrt(s\*(s - side1)\*(s - side2)\*(s - side3));

}

源文件（主函数）：

#include"3.4mytriangle.h"

#include<iostream>

using namespace std;

int main() {

int a = 0, b = 0, c = 0;

cout << "请输入三角形的三边长" << endl;

cin >> a >> b >> c;

if (is\_valid(a, b, c) == 1) {

cout << "三角形的面积为：" << area(a, b, c) << endl;

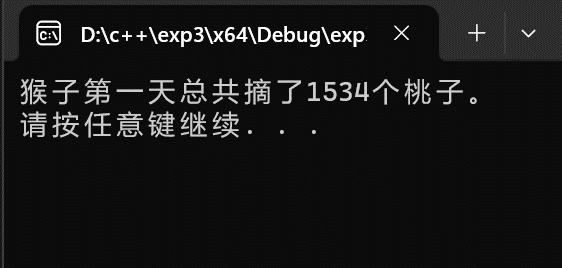
}

else cout << "无法构成三角形" << endl;

system("pause");

return 0;}

5.



#include<iostream>

using namespace std;

int peach(int day) {

if (day == 10) {

return 1;

}

else {

return (peach(day + 1) + 1) \* 2;

}

}

int main() {

int day = 1;

cout << "猴子第一天总共摘了" << peach(day) << "个桃子。" << endl;

system("pause");

return 0;

}

**五、遇到的问题与解决方法**

|  |  |
| --- | --- |
| **遇到的问题** | **解决方法** |
| 1.不知道头文件与源文件之间要如何引用 | 在头文件.h输入函数声明，再创建一个同名的源文件.cpp，进行函数定义，在主函数源文件中加上#include"头文件名" |
| 2.不知道如何表示平方根 | 在表达式前加上sqrt() |
| 3.不知道递归函数要怎么使用 | 一般形式为：函数类型 递归函数名f(参数x）{if(满足结束条件）{结果=初值；} else {结果=含（x-1)表达式；} 返回结果；} |

**六、体会**

在实验过程中，自学能力十分重要，如平方根不知道如何实现，就在csdn上自主学习学会了，再如不懂头文件如何使用，于是自己找了网课进行学习，大学学习中自学是十分重要的一个环节，不仅体现在c++学习中 ，在各科都十分重要。

很多语法只是看懂了完全不够，我发现当自己敲代码的时候还是磕磕绊绊，需不停的重新对照，敲代码需要不断训练，熟能生巧，才能真正掌握语法，在自己写代码的时候才能得心应手。

**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;

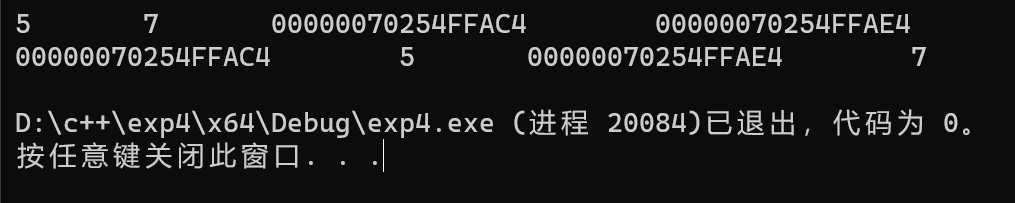
i=5;j=7;

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\_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

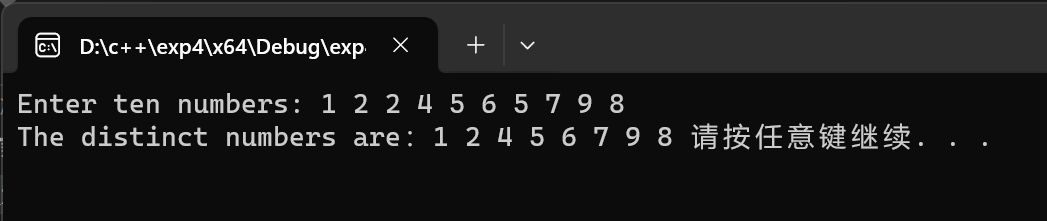
1. 主程序中建立一动态数组（使用new），数组元素及元素个数由键盘输入，动态调试观察指针及指针指向的内容；设计一个函数对数组由小到大排序；主程序中用指针方式输出数组元素；最后释放数组内存（delete）。

【完成实验报告】

**实验报告只要求写程序设计部分**

**三、算法分析，程序结果**

**（一）**

1.

#include<iostream>

using namespace std;

int main() {

int a[10];

int con = 0;

cout << "Enter ten numbers: ";

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

cin >> a[i];

}

cout << "The distinct numbers are：";

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

con = 0;

for (int j = i - 1; j >= 0; j--) {

if (a[i] == a[j]) {

con = 1;

break;

}

}

if (!con) {

cout << a[i] << " ";

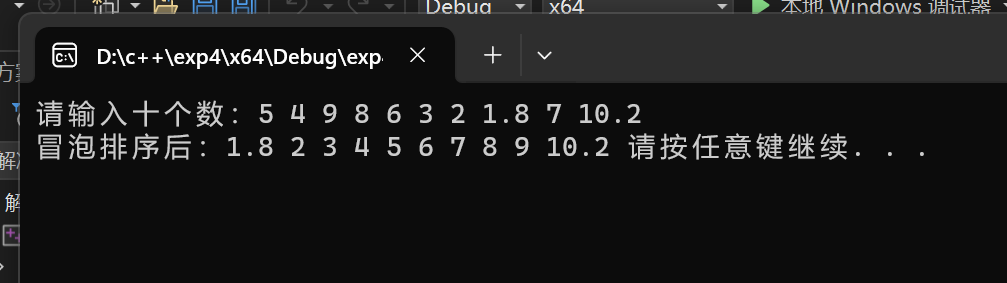
}

}

system("pause");

return 0;

}

2.

#include<iostream>

using namespace std;

void swap(float\* a,int len) {

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

for (int j = 0; j < 10 - i - 1; j++) {

if (a[j] > a[j + 1]) {

int temp = a[j];

a[j] = a[j + 1];

a[j + 1] = temp;

}

}

}

}

int main() {

float a[10];

cout << "请输入十个数：" ;

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

cin >> a[i];

}

swap(a, 10);

cout << "冒泡排序后：";

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

cout << a[i] << " ";

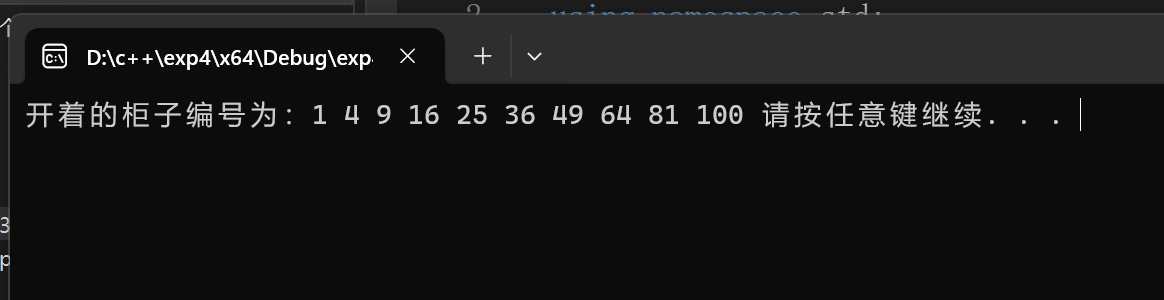
}

system("pause");

return 0;

}

3.

****

#include<iostream>

using namespace std;

int main() {

bool isopen[100] = {0};

//i代表学生编号 j代表乘数

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

for (int j = 1; i \* j <= 100; j++) {

if (i \* j > 100) {

break;

}

isopen[i \* j] = !isopen[i \* j];

}

}

cout << "开着的柜子编号为：";

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

if (isopen[i] == 1) {

cout << i << " ";

}

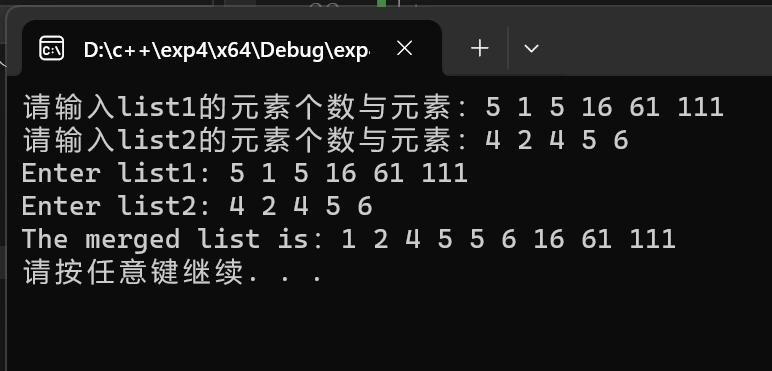
}

system("pause");

return 0;

}

4.



#include<iostream>

using namespace std;

void swap(int\* a, int len) {

for (int k = 0; k < len; k++) {

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

if (a[g] > a[g + 1]) {

int temp = a[g];

a[g] = a[g + 1];

a[g + 1] = temp;

}

}

}

}

int main() {

int i = 0, j = 0, p = 0;

int a[] = { 0 };

int b[] = { 0 };

int c[] = { 0 };

//cout << "Enter list1: ";

cout << "请输入list1的元素个数与元素：";

cin >> i;

for (int q = 0; q < i; q++){

cin >> a[q];

}

//cout << "Enter list2: ";

cout << "请输入list2的元素个数与元素：";

cin >> j;

for (int q = 0; q < j; q++) {

cin >> b[q];

}

cout << "Enter list1: " << i << " ";

for (int q = 0; q < i; q++) {

cout << a[q]<<" ";

}

cout << endl;

cout << "Enter list2: "<<j<<" ";

for (int q = 0; q < j; q++) {

cout << b[q] << " ";

}

cout << endl;

for (int q = 0; q < i; q++) {

c[q] = a[q];

}

for (int q = i,m=0; q < i + j; q++,m++) {

c[q] = b[m];

}

swap(c,i+j);

cout << "The merged list is：" ;

for (int q = 0; q < i+j; q++) {

cout << c[q] << " ";

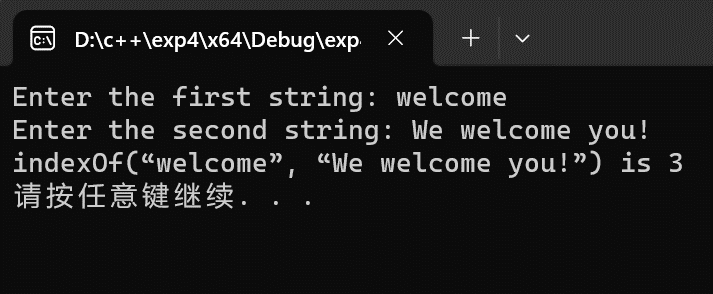
}

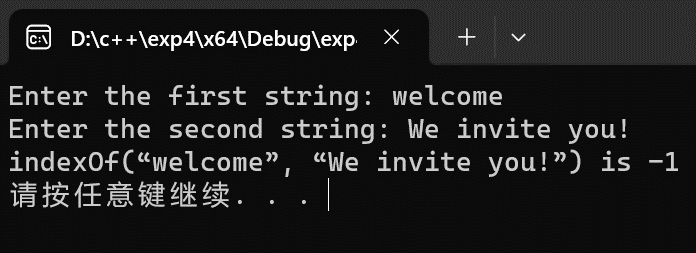
cout << endl;

system("pause");

return 0;

}

5. 



#include<iostream>

using namespace std;

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

if (s1[0] == 0) {

return 0;

}

for (int i = 0; i < strlen(s2); i++) {

int j = 0;

for (j = 0; j < sizeof(s1); j++) {

if (s2[i + j] != s1[j]) {

break;

}

}

if (s1[j] == 0) {

return i;

}

}

return -1;

}

int main() {

char s1[999];

char s2[999];

cout << "Enter the first string: ";

cin.getline(s1, 999);

cout << "Enter the second string: ";

cin.getline(s2, 999);

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

system("pause");

return 0;

}

6.



#include<iostream>

using namespace std;

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

for (int i = 0; i <strlen(s); i++) {

if (s[i] >= 'a' && s[i] <= 'z') {

counts[(int)(s[i] - 97)]++;

}

else if (s[i] >= 'A' && s[i] <= 'Z') {

counts[(int)(s[i] - 65)]++;

}

else continue;

}

for (int j = 0; j < 25; j++) {

if (counts[j] != 0) {

cout << (char)(j + 97) << "：" << counts[j] << "times";

cout << endl;

}

}

}

int main() {

int counts[26] = { 0 };

cout << "Enter a string: ";

char s[999] = { 0 };

cin.getline(s, 999);

count(s, counts);

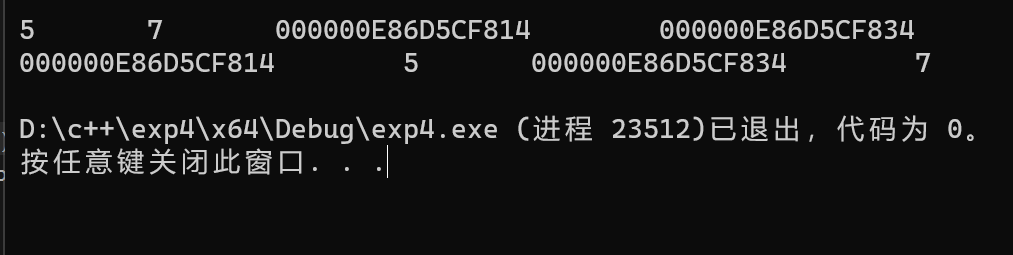
system("pause");

return 0;

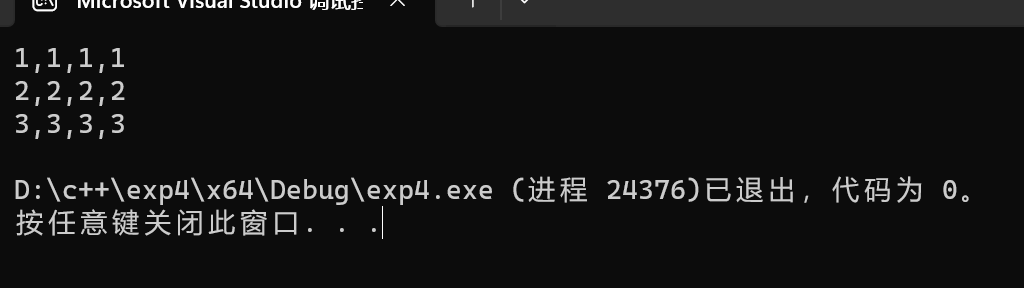
}

**(二)**

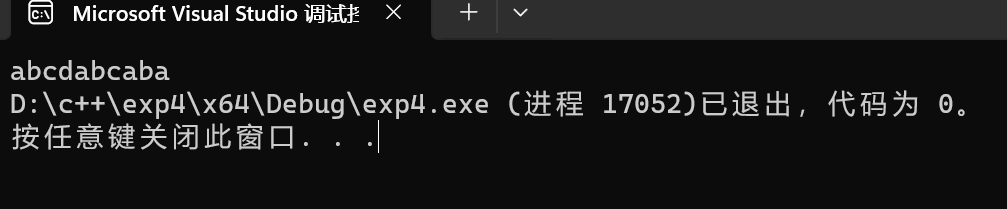
1.（1）



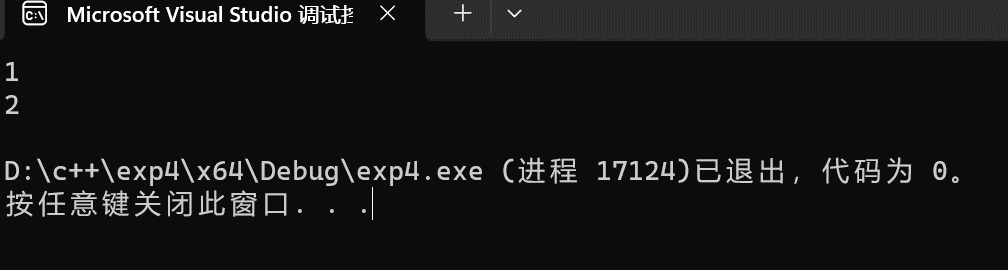
1.（2）



1.（3）、



1.（4）



#include<iostream>

using namespace std;

int\* f()

{

int \*list = new int[4];

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

list[i] = i + 1;

}

return list;

}

void main()

{

int\* p = f();

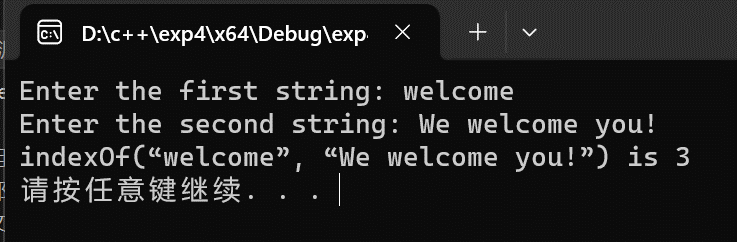
cout << p[0] << endl;

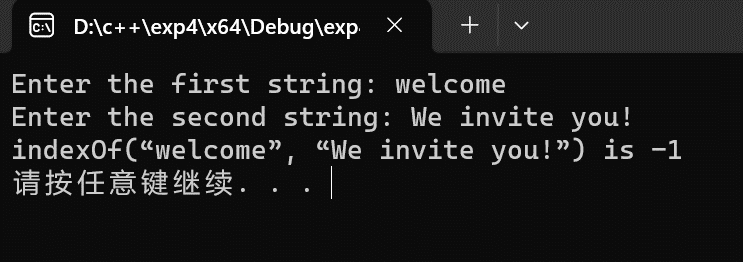
cout << p[1] << endl;

delete[]p;

}

2.（1）





#include<iostream>

using namespace std;

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

if (s1[0] == 0) {

return 0;

}

for (int i = 0; i < strlen(s2); i++) {

int j = 0;

for (j = 0; j < sizeof(s1); j++) {

if (s2[i + j] != s1[j]) {

break;

}

}

if (s1[j] == 0) {

return i;

}

}

return -1;

}

int main() {

char s1[999];

char s2[999];

cout << "Enter the first string: ";

cin.getline(s1, 999);

cout << "Enter the second string: ";

cin.getline(s2, 999);

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

system("pause");

return 0;

}

2.（2）



#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)

{

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 stoi(name1, 0, 16);

}

int main()

{

char hexString[] = { 0 };

cout << "please input your string:";

cin >> hexString;

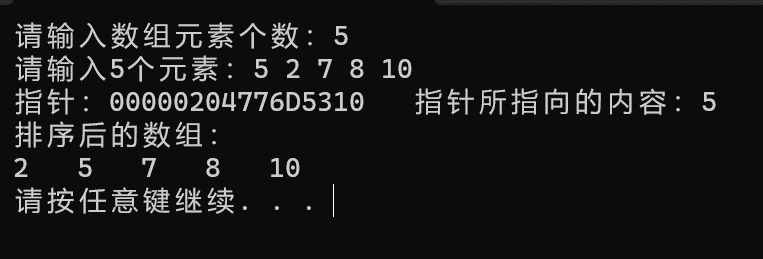
cout << "其十进制数为：" << parseHex(hexString) << endl;

system("pause");

return 0;

}

2.（3）



#include<iostream>

using namespace std;

void test(int \*p,int len) {

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

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

if (p[j] > p[j + 1]) {

int temp = p[j];

p[j] = p[j + 1];

p[j + 1] = temp;

}

}

}

}

int main(){

int size;

cout << "请输入数组元素个数：";

cin >> size;

int\* p = new int[size];

cout << "请输入" << size << "个元素：";

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

cin >> p[i];

}

cout << "指针：" << p << " " << "指针所指向的内容：" << \*p << endl;

test(p,size);

cout << "排序后的数组：" << endl;

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

cout << \*(p+i) << " ";

}

cout << endl;

delete[]p;

system("pause");

return 0;

}

**四、遇到的问题与解决方法**

|  |  |
| --- | --- |
| **遇到的问题** | **解决方法** |
| 1.定义一个int size，用size定义数组元素个数时报错 | 数组定义时，变量不能作下标，可用new操作符new int[size] |
| 2.参数为数组时，sizeof与strlen的区别 | sizeof可以表示数组元素的个数，当参数为一个静态数组时，返回全部数组长度，不能返回动态数组长度大小 strlen用来计算字符串的实际长度，只能用char\*做参数 |
| 3.cin一个字符数组时，到空格便停止输入 | 普通的cin字符数组，无法输入空格，可用cin.getline(数组名，数组长度)读入空格 |
| 4.不知道该如何将十六进制字符串转化为十进制数字，脑海里闪过switch case | stoi可将十六进制字符串转化为十进制数字（字符串名，NULL，16） |

**五、体会**

细节决定成败，很多时候程序运行不出来，不是算法没搞懂，而是细节没有处理好，可能是忘记包含头文件，可能是用了中文符号等等，一定要养成良好习惯，防止未来出现相同问题。不仅仅对c++适用，在生活中的方方面面都要细心。

实践对于编程来说十分重要，只有实践才能让自己熟练，光看懂别人的代码，算法是不够的，必须自己上级操作才知道自己到底懂没懂。