**计算机程序设计基础（C++）**

**实验报告**

专业班级 软件工程2206班

学 号 8209220603

姓 名 陈 凯

**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:\_\_\_\_\_\_\_\_**

**实验四 指针与数组**

1. **实验目的与要求**

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。

1. **实验内容**

(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）。

Sample in 1 2 3 2 1 6 3 4 5 2

Sample out 1 2 3 6 4 5

2.起泡排序：

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

参见 ：<https://oi-wiki.org/basic/bubble-sort/>

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。

假定数组大小不超过 80。

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

Sample in:

Enter list1: 5 1 5 16 61 111

Enter list1: 4 2 4 5 6

Sample out:

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[])

Sample in:

welcome

We welcome you!

Sample out:

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

Sample in :

Welcome

We invite you!

Sample out:

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。

Sample in:

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

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）。

1. **实验步骤、算法与结果分析**
2. **数组**

第一题：

Code：

#include<iostream>

using namespace std;

void main()

{

const int Max = 10; //定义输入个数常量

int arr[Max];

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

{

cin >> arr[i]; //存入arr

}

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

{

bool flag = true; //是否输出标志 置为true

for (int j = i + 1; j < Max; j++) //遍历当前数之后的每一个数

{

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

{

flag = false; //若有重复 置输出标志为false

}

}

if (flag)

{

cout << arr[i]<<" "; //输出

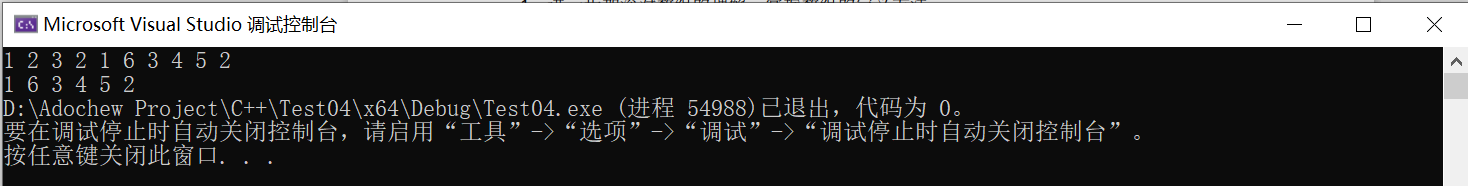
}

} //通过此法 若有重复 则会输出重复数字中的最后的一个

cout << arr[Max-1]; //最后数字无论是否重复都要输出

}

Result：



第二题：

Code：

#include <iostream>

using namespace std;

const int listsize = 10;

void swap(double& a, double& b)

{

double temp;

temp = a;

a = b;

b = temp;

}

void bubbleSort(double list[10])

{

double temp;

bool changed = true;

do

{

changed = false;

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

{

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

{

swap(list[j], list[j + 1]);

changed = true;

}

}

} while (changed );

}

void main()

{

double list[listsize];

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

{

cin >> list[i];

}

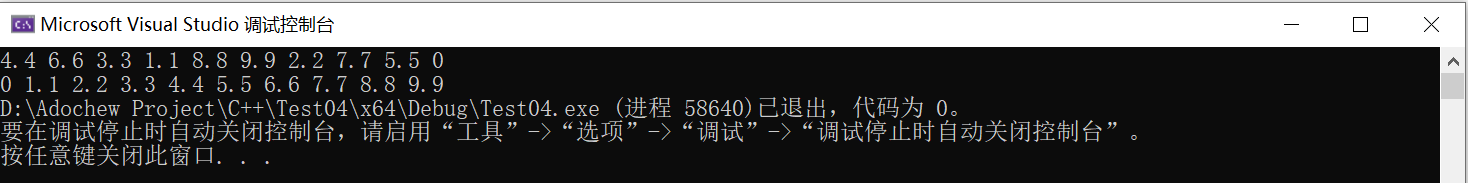
bubbleSort(list);

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

cout << list[i] << " ";

}

Result：



第三题：

Code：

#include <iostream>

using namespace std;

const int num = 100;

void main()

{

bool box[num];

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

{

box[i] = false;

}

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

{

for (int j = i; j < num; j += (i + 1))

{

if (box[j] == false)

{

box[j] = true;

}

else if(box[j] == true)

{

box[j] = false;

}

}

}

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

{

if (box[i])

{

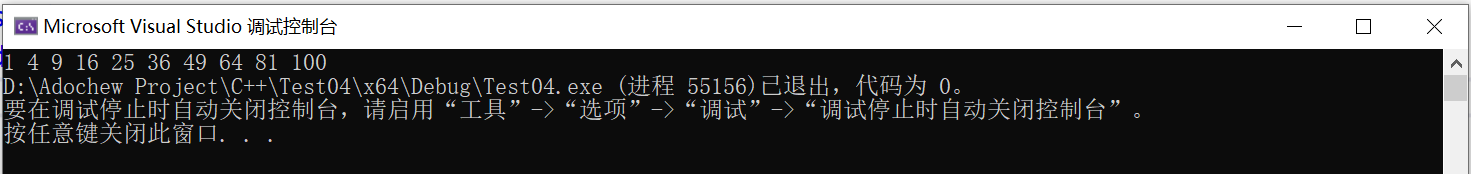
cout << i + 1 << " ";

}

}

}

Result：



第四题：

Code：

#include<iostream>

using namespace std;

void merge(int\* a, int num1, int\* b, int num2, int\* c)

{

int temp;

for (int i = 0; i < num1; i++)

{

c[i] = a[i];

}

for (int i = 0; i < num2; i++)

{

c[num1 + i] = b[i];

}

for (int i = 0; i < (num1+num2) - 1; i++)

{

for (int j = 0; j < (num1 + num2) - i - 1; j++)

{

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

{

temp = c[j];

c[j] = c[j + 1];

c[j + 1] = temp;

}

}

}

for (int i = 0; i < (num1 + num2); i++)

{

cout << c[i] << " ";

}

}

void main()

{

int num1, num2, num3;

cout << "输入数组1的大小: ";

cin >> num1;

cout << "输入数组1 : ";

int\*arr1 = new int [num1];

for (int i = 0; i < num1; i++)

{

cin >> arr1[i];

}

cout << "输入数组2的大小: ";

cin >> num2;

cout << "输入数组2 : ";

int\* arr2 = new int[num2];

for (int i = 0; i < num2; i++)

{

cin >> arr2[i];

}

num3 = num1 + num2;

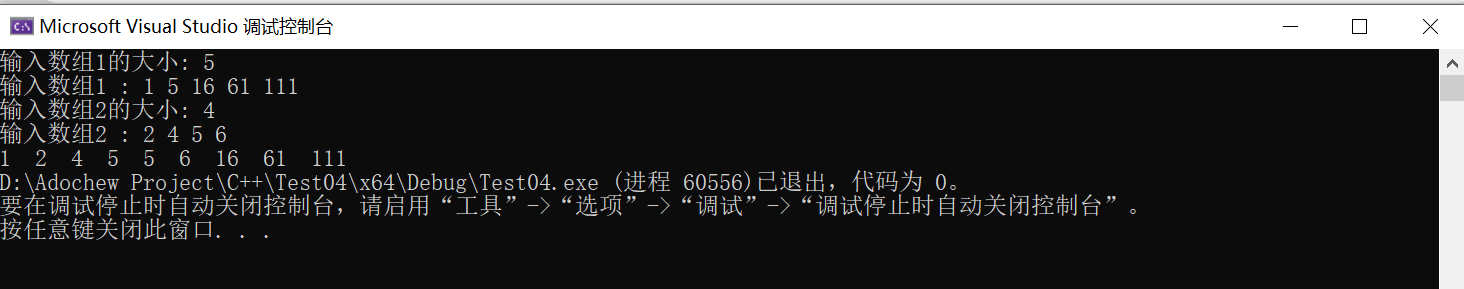
int\* arr3 = new int[num3];

merge(arr1, num1, arr2, num2, arr3);

delete[] arr1,arr2,arr3;

}

Result：



第五题：

Code：

#include<iostream>

using namespace std;

int indexOf(char ch1[], char ch2[])

{

int num1, num2;

num1 = strlen(ch1);

num2 = strlen(ch2);

int i = 0;

int j = 0;

while (i < num2)

{

if (j == (num1 - 1) && ch1[j] == ch2[i])

{

//子串

return i - num1 + 1;

}

if (ch1[j] == ch2[i]) //

{

i++;

j++;

}

if (ch1[j] != ch2[i])

{

i++;

j = 0;

}

}

return -1;

}

void main()

{

char ch1[80], ch2[80];

cout << "Enter the first string : ";

cin.getline(ch1, 80);

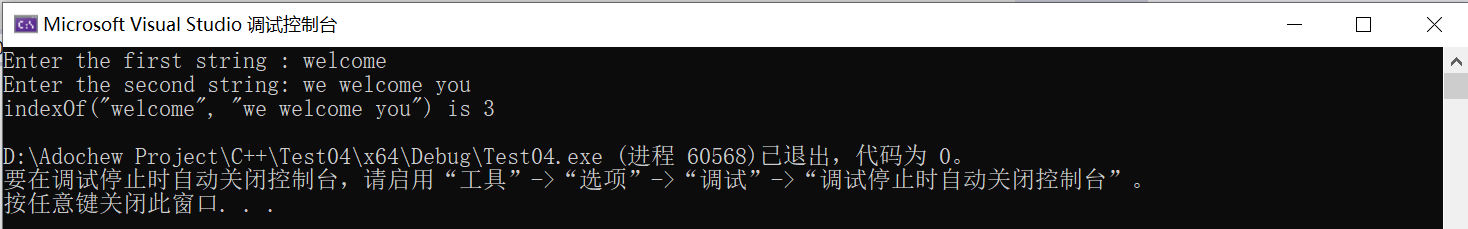
cout << "Enter the second string: ";

cin.getline(ch2, 80);

cout << "indexOf(\"" << ch1 << "\", \"" << ch2 << "\") is " << indexOf(ch1, ch2) << endl;

}

Result：



第六题：

Code：

#include<iostream>

using namespace std;

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

{

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

{

counts[i] = 0;

}

for (int i = 0; i < 80; i++)

{

if ((s[i] - 'a') >= 0)

{

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

}

if ((s[i] - 'A') <= 25)

{

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

}

}

}

void display(int counts[])

{

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

{

if (counts[i] != 0)

{

cout << char(i + 'a') << " : " << counts[i] << " times" << endl;

}

}

}

void main()

{

int counts[26];

char s[80];

cout << "Enter a string : ";

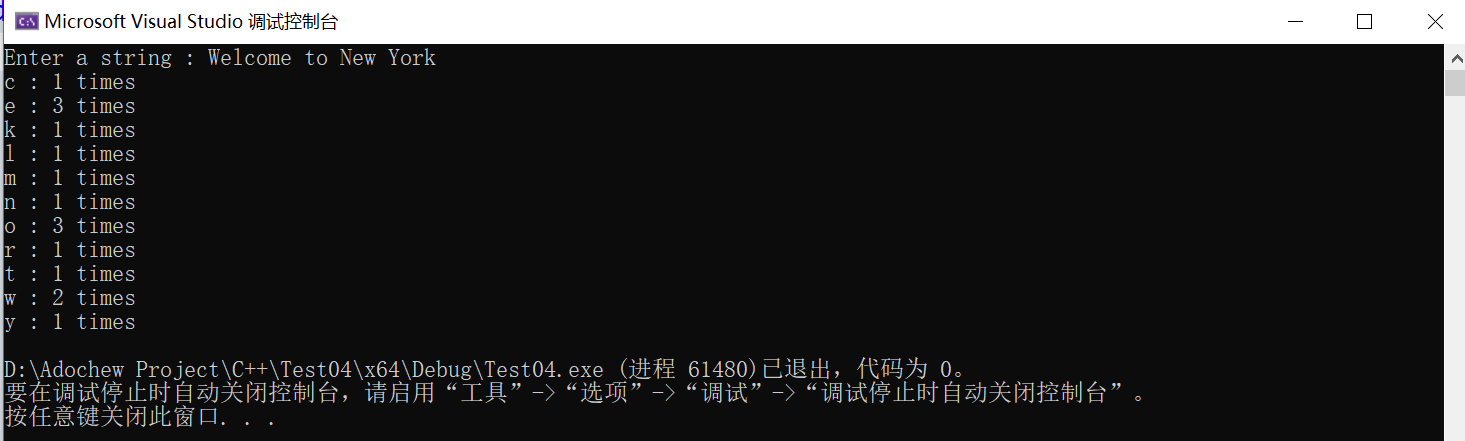
cin.getline(s, 80);

count(s, counts);

display(counts);

}

Result：



1. **指针**

第一题：

Code：

#include<iostream>

using namespace std;

int indexOf(char\* ch1, char\* ch2)

{

int num1, num2;

num1 = strlen(ch1);

num2 = strlen(ch2);

int i = 0;

int j = 0;

while (i < num2)

{

if (j == (num1 - 1) && ch1[j] == ch2[i])

{

//子串

return i - num1 + 1;

}

if (ch1[j] == ch2[i])

{

i++;

j++;

}

if (ch1[j] != ch2[i])

{

i++;

j = 0;

}

}

return -1;

}

void main()

{

char ch1[80], ch2[80];

cout << "Enter the first string : ";

cin.getline(ch1, 80);

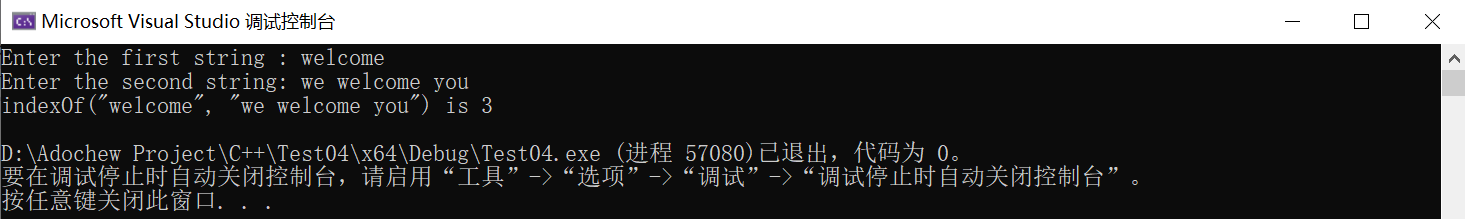
cout << "Enter the second string: ";

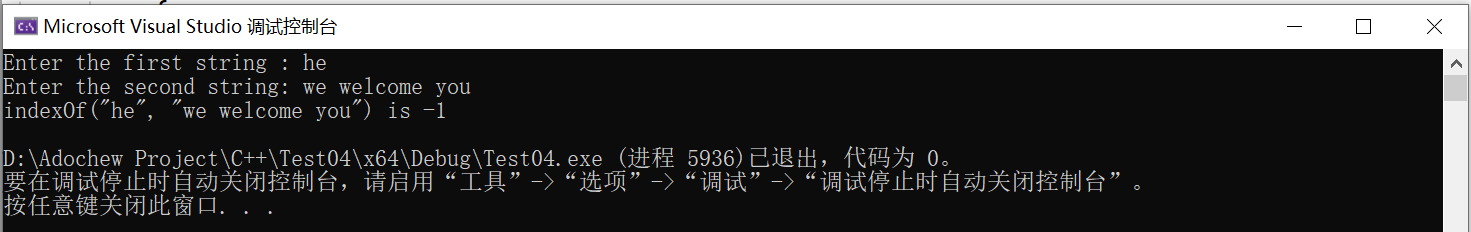
cin.getline(ch2, 80);

cout << "indexOf(\"" << ch1 << "\", \"" << ch2 << "\") is " << indexOf(ch1, ch2) << endl;

}

Result：





第二题：

Code：

#include<iostream>

using namespace std;

int parseHex(const char\* const hexString)

{

int len, sum = 0;

len = strlen(hexString);

int\* arr=new int [len];

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

{

if (hexString[i] >= 'A' && hexString[i] <= 'F')

{

arr[i] = (int(hexString[i]) - 55) \* (pow(16, len - 1 - i));

}

else

{

arr[i] = (hexString[i] - 48) \* pow(16, len - i - 1);

}

sum = sum + arr[i];

}

delete[] arr;

return sum;

}

void main()

{

char s[80];

cout << "请输入一个16进制数字: " << endl;

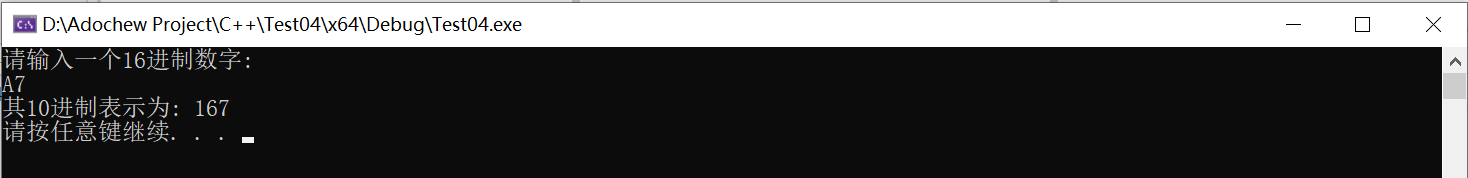
cin.getline(s, 80);

cout<<"其10进制表示为: "<<parseHex(s)<<endl;

system("pause");

}

Result：



第三题：

Code：

#include<iostream>

using namespace std;

void Sort(int\*arr ,int num)

{

int temp;

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

{

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

{

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

{

temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

}

void main()

{

int num;

cout << "请输入数组大小: " << endl;

cin >> num;

int\* arr = new int[num];

cout << "请输入数组: " << endl;

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

{

cin >> \*(arr + i);

}

Sort(arr, num);

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

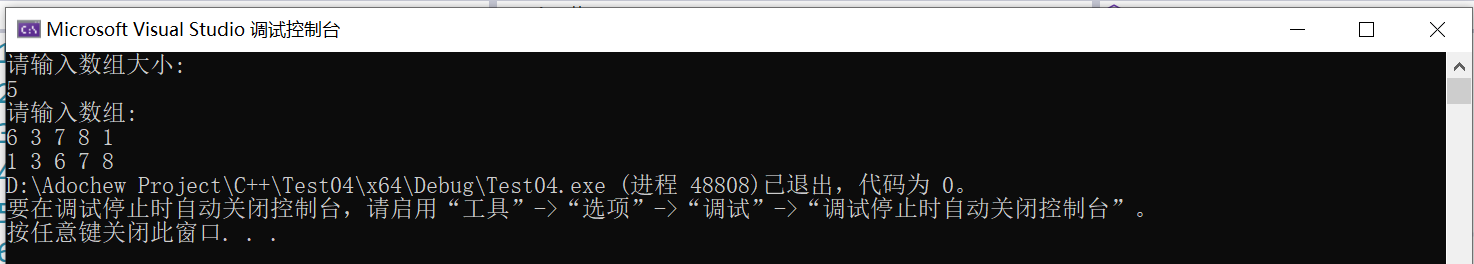
{

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

}

}

Result：



1. **遇到的问题和解决方法**

在数组第六题中, void count(const char s[], int counts[])函数一开始的for循环

for (int i = 0; i < 80; i++)

{

if ((s[i] - 'a') >= 0)

{

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

}

if ((s[i] - 'A') <= 25)

{

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

}

}

中没有写if判断 导致counts数组越界 初出结果为乱码

通过判断保证的大小写字母可以同时被计数而数组不越界

1. **体会**

通过这次实验,熟悉数组的排序和编程, 理解了内存动态分配的含义、掌握了内存动态分配方法, 明确了指针的运算, 了解了指针和数组的联系.