**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 8209220622

姓名 王昱凯

**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;

int main()

{

const int size = 10;

int list[size];

cout << "enter ten numbers : " << endl;

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

cin >> list[i];

}

cout << "The distinct numbers are:";

cout << list[0] << " ";

for (int m = 1; m < size; m++)

{

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

{

if (list[m] != list[j])

{

if (j == m - 1)

cout << list[m] << " ";

else

continue;

}

else

break;

}

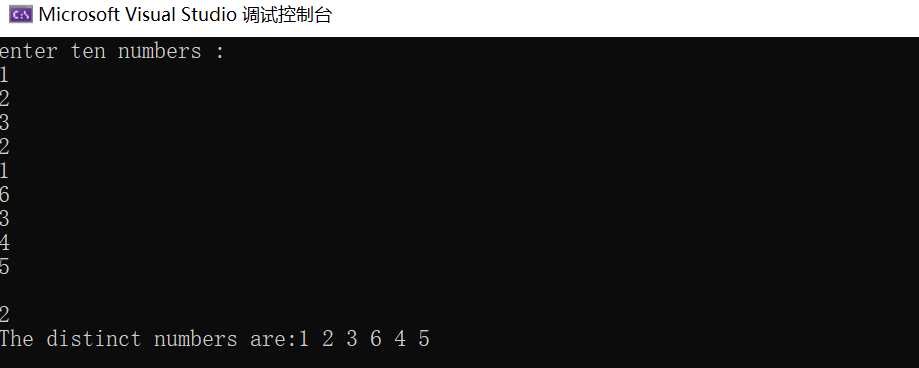
}

cout << endl;

return 0;

}

**Result：**

****

**第二题：**

**Code：**

#include <iostream>

#include<iomanip>

using namespace std;

void bubblesort(double\* a)

{

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

{

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

{

double temp;

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

{

temp = a[j];

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

a[j + 1] = temp;

}

}

}

}

void f(double\*a)

{

cout << endl;

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

{

cout << a[i] <<setw(5);

}

}

int main()

{

cout << "输入十个数" << endl;

double a[10];

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

{

cin >> a[i];

}

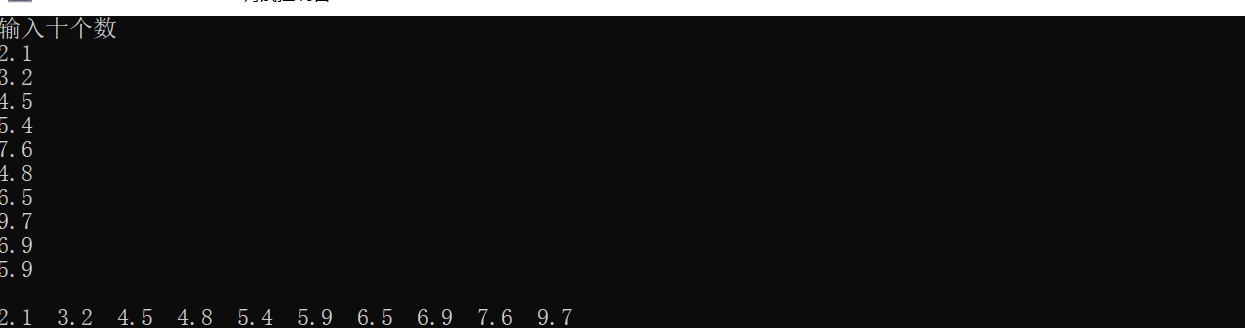
bubblesort(a);

f(a);

return 0;

}

**Result：**

****

**第三题：**

**Code：**

#include <iostream>

using namespace std;

int main()

{

bool a[100];

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

{

a[i] = false;

}

for (int j = 2; j < 100; j++)

{

for (int k = j - 1; k< 100; k =k+j+1)

{

if (a[k])

{

a[k] = false;

}

else

{

a[k] = true;

}

}

}

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

{

if (a[m])

{

cout << m+ 1 << " ";

}

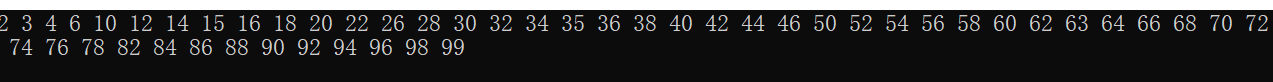
}

cout << endl;

return 0;

}

**Result：**

****

**第四题：**

**Code：**

#include <iostream>

using namespace std;

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

int main()

{

int list1[80];

int list2[80];

int list3[80];

int size1, size2;

cout << "输入较长的数组的元素个数：";

cin >> size1;

cout << "输入较短的数组的元素个数：";

cin >> size2;

cout << "输入俩个数组中较长的数组" << endl;

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

{

cin >> list1[i];

}cout << "输入俩个数组中较短的数组" << endl;

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

{

cin >> list2[i];

}

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

system("pause");

return 0;

}

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

{

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

{

list3[i] = list1[i];

}

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

{

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

}

//冒泡排序

for (int i = 0; i < size1+size2; i++)

{

for (int j = 0; j < size1+size2 - i-1; j++)

{

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

{

int temp = 0;

temp = list3[j];

list3[j] = list3[j + 1];

list3[j + 1] = temp;

}

}

}

cout << "排序后：";

for (int k = 0; k <( size1 + size2); k++)

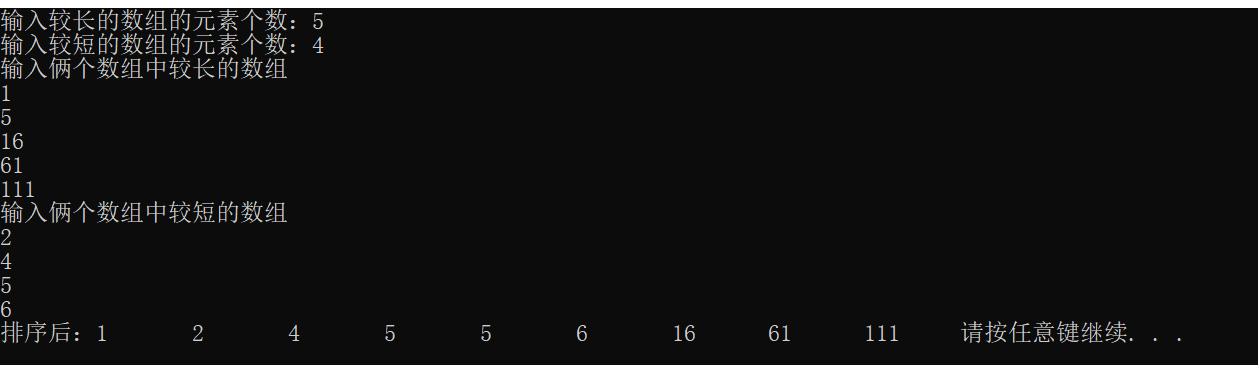
{

cout << list3[k] << '\t';

}

}

**Result：**

****

**第五题：**

**Code：**

#include <iostream>

#include <string>

using namespace std;

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

int main()

{

char s1[40];

char s2[40];

cout << "输入字符串1" << endl;

cin >> s1;

cout << "输入字符串2" << endl;

cin >> s2;

cout << indexof(s1, s2) << endl;

}

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

{

// 空串是任何字符串的子串

if (s1[0] == 0)

{

return 0;

}

else

{

for (int i = 0; s2[i]; i++)

{

int k = i, j = 0;

// k表示s2当前遍历的位置，j表示s1当前遍历的位置

for (; s1[j]; ++k, ++j)

{

// 一旦有不相同，跳出循环

if (s2[k] != s1[j])

{

break;

}

}

// 如果已经遍历完了s1才跳出的循环

if (s1[j] == 0) {

// 返回最开始的s2的起始位置

return i;

}

}

return -1;

}

}

**Result：**

****

****

**第六题：**

**Code：**

#include <iostream>

using namespace std;

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

int main()

{

char s[40];

int counts[26]={0};

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

cin >> s;

count(s, counts);

/\* char str[40] = "welcometonewyork";

cout << str[1] << endl;

cout << strlen(str) << endl;

cout << str[1]-97 << endl;\*/

system("pause");

return 0;

}

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

int i = 0;

for (i; i < strlen(s); i++)

{

if (s[i] >= 65 && s[i] <= 90)

{

counts[s[i] - 65] += 1;

}

if (s[i] >= 97 && s[i] <= 122)

{

counts[s[i] - 97] += 1;

}

}

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

{

if (counts[j] != 0) {

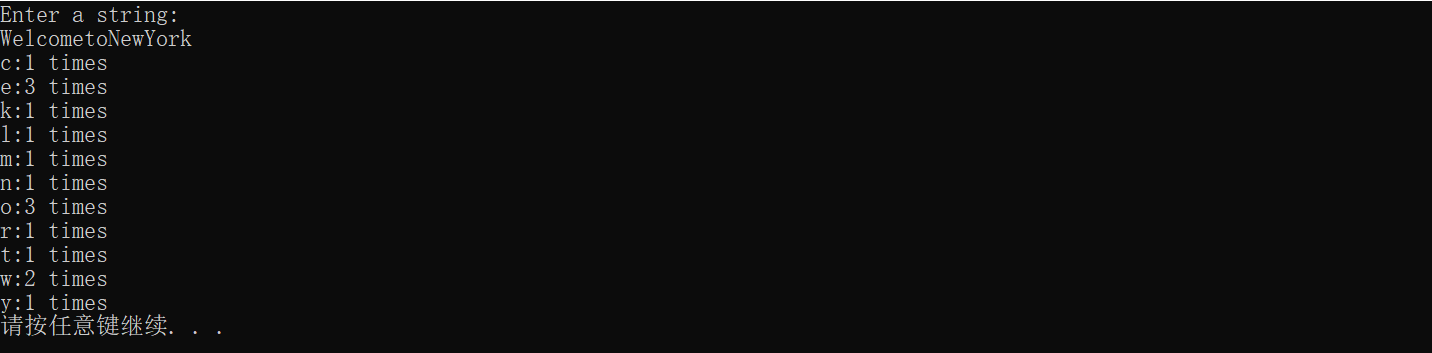
cout << (char)(j+97) << ":" << counts[j]<<" times" << endl;

}

}

}

**Result：**

****

1. **指针**

**第一题：**

**（1）Code：**

#include <iostream>

using namespace std;

int 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;

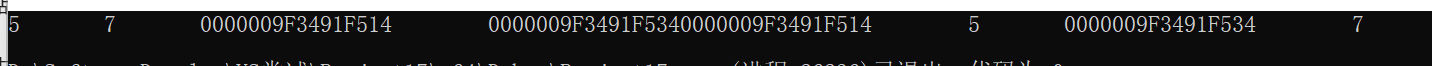
i = 5; j = 7;

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

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

}

**Result：**

****

**（2）Code：**

#include <iostream>

using namespace std;

int main()

{

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

}

**Result：**

****

**（3）Code：**

/#include <iostream>

using namespace std;

#include <string>

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);

}

**Result：**

****

**（4）Code：**

#include<iostream>

using namespace std;

int\* f()

{

int\* p = new int[4];

for (int i = 0; i < 4; i++)

{

p[i] = i + 1;

}

return p;

}

void main()

{

int\* p = f();

cout << p[0] << endl;

cout <<p[1] << endl;

delete []p;

}

**Result：**

****

**第二题：**

**（1）Code：**

#include <iostream>

#include <string>

using namespace std;

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

int main()

{

char s1[40];

char s2[40];

cout << "输入字符串1" << endl;

cin >> s1;

cout << "输入字符串2" << endl;

cin >> s2;

cout << indexof(s1, s2) << endl;

}

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

{

// 空串是任何字符串的子串

if (s1[0] == 0)

{

return 0;

}

else

{

for (int i = 0; s2[i]; i++)

{

int k = i, j = 0;

// k表示s2当前遍历的位置，j表示s1当前遍历的位置

for (; s1[j]; ++k, ++j)

{

// 一旦有不相同，跳出循环

if (s2[k] != s1[j])

{

break;

}

}

// 如果已经遍历完了s1才跳出的循环

if (s1[j] == 0) {

// 返回最开始的s2的起始位置

return i;

}

}

return -1;

}

}

**Result：**

****

****

**（2）Code：**

#include<iostream>

#include<string>

#include<cmath>

using namespace std;

int parseHex(const char\* const hexString);

int main()

{

int size = 999;

char s[40];

cout << "请输入一个以字符串形式表示的十六进制数"<<endl;

cin>>s;

cout << "转化为10进制数为" << parseHex(s) << endl;

return 0;

}

int parseHex(const char\* const hexString)

{

int a, sum = 0;

a = strlen(hexString);

int list[40];

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

{

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

list[i] = (static\_cast<int>(hexString[i]) - 55) \* (pow(16, a - 1 - i));

else

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

sum = sum + list[i];

}

return sum;

}

**Result：**

****

**（3）Code：**

#include <iostream>

using namespace std;

void f(int\* p, int size)

{

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

{

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

{

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

{

int temp = p[j];

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

p[j + 1] = temp;

}

}

}

}

void show(int\* p, int size)

{

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

{

cout << p[i] << '\t';

}

}

int main()

{

int size = 1;

cout << "输入元素个数" << endl;

cin >> size;

cout << "输入数组元素" << endl;

int\* p = new int[size];

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

{

cin >> p[i];

}

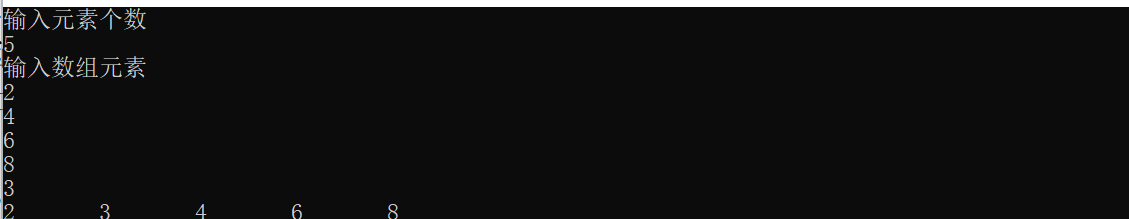
f(p, size);

show(p, size);

delete[]p;

}

**Result：**

****

1. **遇到的问题和解决方法**

1.存物柜问题问题没有思路，自己的思路过于麻烦以至于无法下手写代码

2.检查字串问题，不知道如何合理的使用循环

解决方法：

通过向同学求助，以及上网上，csdn上搜集资料得到解决办法，并自己动手将代码敲出来

1. **体会**

实验四的内容相较于前三次实验，实验难度明显增加，出现了许多我无从下手的问题，但是通过实验四，我对数组和指针的相关知识有了更深一步的了解，提高了自己动手写代码的能力，以及上网搜集资料，独立解决问题的能力。