**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：**

（1）数组部分：

**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 ；**

**思路：**

创建两个数组A，B。A数组的元素为已输入的元素，之后利用for循环一一对比A与B中的元素，如有不同就将其加入B数组中；

**程序：**

#include<iostream>

using std::endl;

using std::cout;

using std::cin;

int main() {

int array1[10], array2[10];

cout << "Enter ten numbers:";

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

cin >> array1[i];

}

//输入十个数字；

cout << "The distinct numbers are:";

array2[0] = array1[0];

cout << array2[0] << " ";

//打印第一个不同的数字；

int num = 1;

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

bool is = true;

for (int z = j; z > 0; z--) {

if (array1[j] == array2[z]||array1[j]==array2[0]) {

is = false;

}

}

if (is == true) {

array2[num] = array1[j];

cout << array2[num] << " ";

num++;

}

}

system("pause");

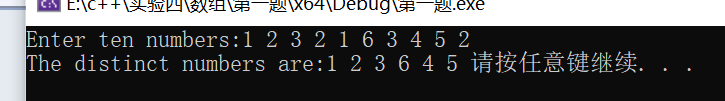
return 0;

}

**输出结果：**

* **（1）尝试输入一串数据：1 2 3 2 1 6 3 4 5 2，观察输出结果；**

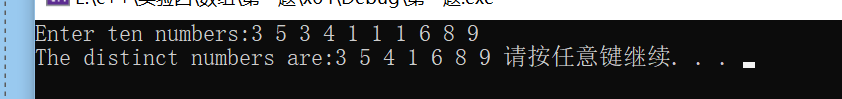
**预计输出结果为： 1 2 3 6 4 5 ；**



**实验数据与预期结果相符；**

* **（2）尝试输入一串数据：3 5 3 4 1 1 1 6 8 9 ；观察输出结果；**

**预计输出结果为：3 5 4 1 6 8 9；**



**实验数据与预期结果相符；**

1. **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。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；**

**思路：**

利用嵌套for循环，两两比较大小然后排序；其中比较大小并交换顺序的过程由函数

void swap(double\* a, double\* b)来实现；注意此时函数形参应使用**指针或者引用形式**，

保证**地址传递**而不是值传递；

**程序：**

#include<iostream>

using std::cout;

using std::endl;

using std::cin;

void swap(double\* a, double\* b) {

if (\*a > \*b) {

double temp;

temp = \*a;

\*a = \*b;

\*b = temp;

}

}

//值传递的函数swap；

int main() {

const int listSize = 10;

double array[listSize];

cout << "Please enter ten double precision numbers:" << endl;

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

cin >> array[i];

}

//输入十个双精度的数据；

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

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

swap(&array[j], &array[j + 1]);

}

}

//交换这十个数据的顺序；

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

cout << array[i] << " ";

}

system("pause");

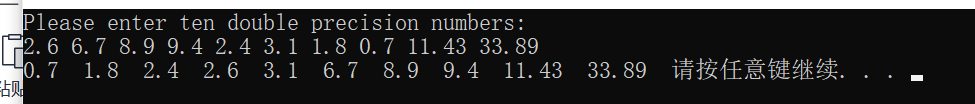
return 0;

}

**输出：**

* **尝试输入十个双精度的数据（2.6、6.7、8.9、9.4、2.4、3.1、1.8、0.7、11.43、33.89），**

**观察输出结果；**



**符合预期输出结果；**

1. **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；**

**思路:**

这题需要将过程理清楚：第一个学生打开所有的存物柜，表现为程序即为true变为false；在之后的第二个到第一百个学生同样是这样的变化，但是需要确定间隔数量。第n个学生需要隔n+1个学生才到下一个学生，把握好间隔数量即可完成本题；

**程序:**

#include<iostream>

using std::cout;

using std::endl;

int main() {

bool array[100];

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

array[i] = false;

}

//存物柜开始全都是关闭的（false）；

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

array[i] = true;

}

//第1个学生操作后存物柜全部开启（true）；

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

int a = i;

do {

if (array[a] == true) {

array[a] = false;

}

else if (array[a] == false) {

array[a] = true;

}

a += i + 2;

} while (a < 100);

}

//第2~100个学生操作后柜子的情况；

cout << "开着的箱子的编号为：" << endl;

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

if (array[i] == true) {

cout << i + 1 << " ";

}

}

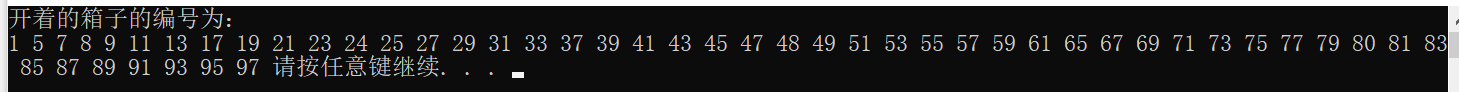
//打印最后仍为开启状态的柜子的编号；

system("pause");

return 0;

}

**输出：**



* 开着的箱子的编号为：

1 5 7 8 9 11 13 17 19 21 23 24 25 27 29 31 33 37 39 41 43 45 47 48 49 51 53 55 57 59 61 65 67 69 71 73 75 77 79 80 81 83 85 87 89 91 93 95 97；

1. **合并两个排列好的数组：**

**编写如下函数merge，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[])**

**思路：**

本题分为两个步骤：第一就是将list1和list2中的元素赋值给list3，第二就是利用函数swap(int list3[],int size1,int size2) 将list3的元素按从大到小的顺序排列输出；

**程序：**

#include<iostream>

using std::cout;

using std::cin;

using std::endl;

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

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

while (i < size1) {

list3[i] = list1[i];

break;

}

while (i >= size1) {

list3[i] = list2[i - size1];

break;

}

}

}

//将list1和list2中的元素赋值给list3;

void swap(int list3[],int size1,int size2)

{

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

}

}

}

}

//将list3中的元素按大小排列；

int main() {

cout << "Enter list1:";

int size1 = 0;

int list1[40];

cin >> size1;

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

cin >> list1[i];

}

//输入数组list1;

cout << "Enter list2:";

int size2 = 0;

int list2[40];

cin >> size2;

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

cin >> list2[i];

}

//输入数组List2；

int list3[80];

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

cout << "The merge list:";

swap(list3,size1,size2);

//交换list3的次序；

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

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

}

//打印List3；

system("pause");

return 0;

}

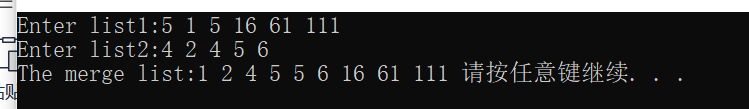
**输出：**

* **尝试输入Sample input:**

**Enter list1: 5 1 5 16 61 111;**

**Enter list2: 4 2 4 5 6;**

**The merged list is: 1 2 4 5 5 6 16 61 111;**



1. **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；**

**思路：**

本题的思路是首先确定s1的首字母是否在s2中存在。若存在确定它的位置并往后逐一比对，若完全相同则证明s1是s2的子串，返回true；否则均返回false，包括s1不为s2字串的情况，s1的长度大于s2的情况。其中要注意cin.getline输入空格的方法的使用；

**程序：**

#include<iostream>

using std::endl;

using std::cout;

using std::cin;

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

{

bool is = false;

int i;

for (i = 0; i < 50; i++)

{

if (strlen(s1) == 1 && s1[0] == s2[i])

{

return i;

}

//考虑到s1只由一个字符组成的情况；

if (s1[0] == s2[i])

{

int j = 1;

do {

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

{

j++;

is = true;

}

else

{

is = false;

break;

}

} while (j < strlen(s1));

if (is == true)

{

return i;

break;

}

}

}

//s1不止一个字符；

if (is == false)

return -1;

}

int main() {

cout << "Enter first string:";

char s1[50];

cin.getline(s1, 50);

cout << "Enter second string:";

char s2[50];

cin.getline(s2, 50);

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

indexOf(s1, s2) << endl;

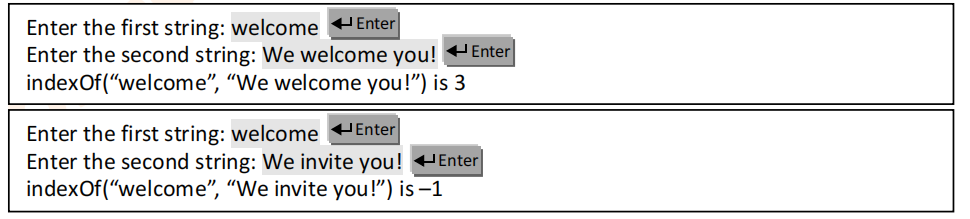
system("pause");

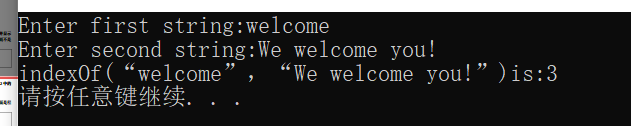
return 0;

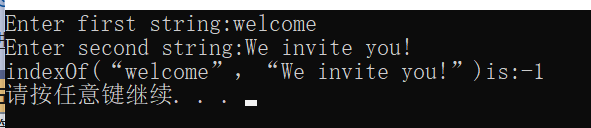
}

**输出：**

* **尝试按如下样例输入字符串，观察是否能得到一样的结果；**

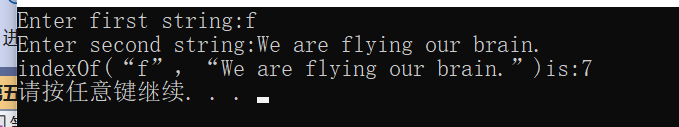






**可以看到输出结果符合实验预期；**

* **下面尝试当s1仅一个字符的情况；**



1. **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[])**

**字母不分大小写，例如字母 A 和字母 a 都被看作 a；**

**思路：**

创建一个包含整数1~26的counts数组，const[0]，const[1]，…，const[25]分别记录 a，b，…，

z 出现的次数，最后输入counts中不为零的数对应的字母。需要注意字母大小写之间ASCII码以及数据类型之间的的灵活转换；

**程序：**

#include<iostream>

#include<string>

using std::cin;

using std::endl;

using std::cout;

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]++;

}

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

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

}

}

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

* //为什么不能用sizeof(counts)/sizeof(counts[0])? 若被除数是指针，通常会出现错误的结果；

if (counts[i] != 0) {

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

}

}

}

int main() {

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

char s[50];

int counts[26];

cin.getline(s, 20);

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

counts[i] = 0;

}

count(s, counts);

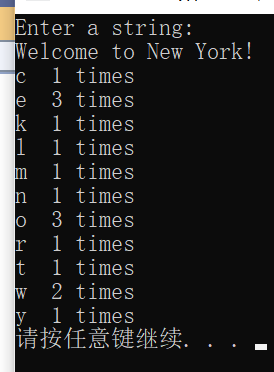
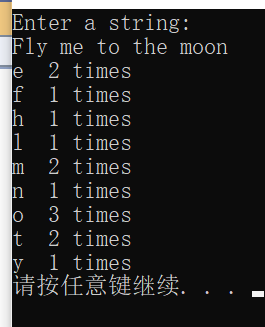
system("pause");

return 0;

}

**输出：**

* **尝试输入字符“Welcome to New York!”和“Fly me to the moon”，观察各自的输出结果；**

1. 指针部分：
2. **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);**

**思路：**

本题跟数组部分的第五题类似,不同点在于函数indexof的形参不同,一个是数组一个是数组指针,使用时要注意二者的区别;

**程序：**

#include<iostream>

using std::endl;

using std::cout;

using std::cin;

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

{

bool is = false;

int i;

for (i = 0; i < 50; i++)

{

if (strlen(s1) == 1 && s1[0] == s2[i])

{

return i;

}

//考虑到s1只由一个字符组成的情况；

if (s1[0] == s2[i])

{

int j = 1;

do {

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

{

j++;

is = true;

}

else

{

is = false;

break;

}

} while (j < strlen(s1));

if (is == true)

{

return i;

break;

}

//s1由多个字符组成；

}

}

if (is == false)

return -1;

}

int main() {

cout << "Enter first string:";

char s1[50];

cin.getline(s1, 50);

cout << "Enter second string:";

char s2[50];

cin.getline(s2, 50);

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

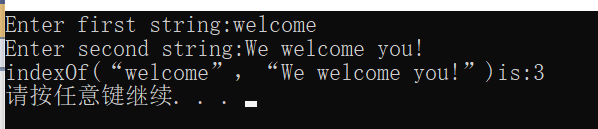
indexOf(s1, s2) << endl;

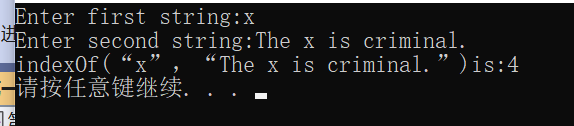
system("pause");

return 0;

}

**输出：**





* **输出结果与数组部分的第五题相同,说明二者用了两种不同方法实现了题目要求;**

1. **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);**

**思路：**

本题依然强调的是ASCII码的灵活运用，具体体现在如何将输入的字符串转化为计算得出的整数值以及字母大小写之间的转化。同时要注意数组中元素的位置，以免造成错误结果。

**程序：**

#include<iostream>

#include<cmath>

using std::cin;

using std::cout;

using std::endl;

int parseHex(const char\* const hexString)

{

int sum = 0;

for (int i = 0; i < strlen(hexString); i++)

{

if (hexString[strlen(hexString) - i - 1] >= '0' && hexString[strlen(hexString) - i - 1] <= '9')

{

sum += (int)(hexString[strlen(hexString) - i - 1] - 48) \* pow(16, i);

}

//十六进制从0到9；

if (hexString[strlen(hexString) - i - 1] > '9')

{

sum += (int)(hexString[strlen(hexString) - i - 1] - 55) \* pow(16, i);

}

//十六进制从A到F；

}

return sum;

}

int main() {

cout << "请输入一个十六进制的数字：" << endl;

char hexString[40];

cin >> hexString;

cout << "该十六进制数字转化为十进制数字为："<<parseHex(hexString) << endl;

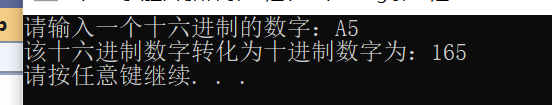
system("pause");

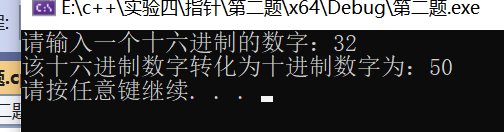
return 0;

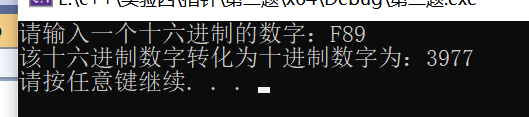
}

**输出：**

* **尝试输入十六进制数字“A5”“32”“F89”，分别观察它们的输出结果；**







* **计算可得输出结果均为正确值；**

**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）;**

**思路：**

本题的思路较为清晰，需要注意的是“用指针方式输出数组元素”的格式；

array[a][b] —> \*( \* ( array+a ) +b );

**程序：**

#include<iostream>

using std::cin;

using std::cout;

using std::endl;

void swap(int array[], int size)

{

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

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

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

{

int temp;

temp = array[j];

array[j] = array[j + 1];

array[j + 1] = temp;

}

}

}

}

//使用函数swap来给数组元素排序；

int main() {

cout << "请输入动态数组的大小:";

int size = 0;

cin >> size;

int\* p = new int[size];

int\* array = p;

cout << "请输入动态数组中的元素：";

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

{

cin >> array[i];

}

swap(array, size);

cout << "从小到大输出数组元素：";

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

{

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

}

cout << endl;

//利用指针方式输出数组元素；

cout << "观察指针以及指针指向的内容：" << endl;

cout << "动态数组array中的第三个元素指针为：" << (array + 2) << endl;

cout << "指针指向的内容：" << \*(array + 2) << endl;

delete [] p;

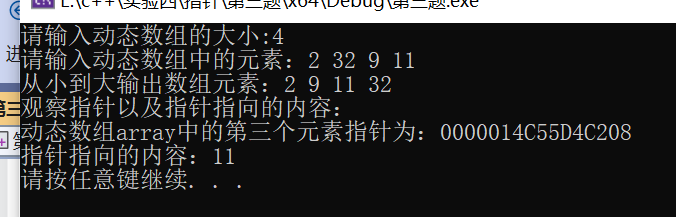
//delete释放数组内存空间；

system("pause");

return 0;

}

**输出：**



1. **遇到的问题和解决方法：**

本次实验较前三次的难度有较大提升，对于数组和指针的各个内容都有较为全面的考察。譬如数组4、5题难度较大，第四题需要我们创建第三个数组并对一二个数组逐一对比，将不同的数组元素引入到第三数组中；第五题考察检验子串，同样是对数组的深入考察。值得一提的是，本次实验有4次直接或间接地考察了利用函数对元素进行起泡排序，这说明这一内容我们应重点掌握并熟练使用。在完成实验的过程中，我除了自

己的思考，还参阅了教材查找不熟悉的知识点，以及在CSDN上搜索未学的知识，譬如cin.getline对于输入字符串的用法等；

1. **心得与体会：**

在完成本次实验内容的过程中，我深刻体会到了自主查询资料的重要性。对于不懂的问题我们也应及时向老师或同学请教，趁早解决。希望在坚持不懈的练习中我们能做到更好。