**计算机程序设计基础（C++)**

**实验报告**

专业班级： 软工2402

学 号： 8209240224

姓 名： 范雪雯

**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？

**四、算法分析，程序结果**

**1.#include <iostream>**

**using namespace std;**

**int max(int x, int y)**

**{**

**if (x > y)**

**{**

**int c = 0;**

**while (x%y!=0)**

**{**

**c = x % y;**

**x = y;**

**y = c;**

**}**

**}**

**if (x < y)**

**{**

**int c = 0;**

**while (x % y != 0)**

**{**

**c = x % y;**

**x = y;**

**y = c;**

**}**

**}**

**return y;**

**}**

**int min(int y, int z)**

**{**

**int l = 0;**

**l = y \* z / max(y, z);**

**return l;**

**}**

**int main()**

**{**

**int m = 0, n = 0;**

**cout << "m=" << endl;**

**cout << "n=" << endl;**

**cin >> m;**

**cin >> n;**

**int a = 0, b = 0;**

**a = max(m, n);**

**b = min(m, n);**

**cout << "最大公因数为："<<a << endl;**

**cout << "最小公倍数为：" << b << endl;**

**}**

**2.**#include<iostream>

#include<cmath>

using namespace std;

bool is\_prime(int num)

{

if (num <= 1)return false;

if (num <= 3)return true;

if (num % 2 || num % 3 == 0)return false;

for (int i = 5; i < sqrt(num);i++)

{

if (num % i == 0)return false;

}

return true;

}

int main()

{

int count = 0;

int num = 2;

while (count <= 200)

{

if (is\_prime(num))

{

cout << num << " ";

count++;

if (count % 10 == 0)cout << endl;

}

num++;

}

}

**3.头文件：**

**#pragma once**

**double celsius\_to\_fah(double cel);**

**double fahrenheit\_to\_cels(double fah);**

**源文件：**

**1.#include<iostream>**

**#include<iomanip>**

**#include"a.h"**

**using namespace std;**

**int main()**

**{**

**double celsius[] = { 40.0,39.0,31.0 };**

**double fahrenheit[] = { 120.0,110.0,30.0 };**

**cout << "Celsius\tFahrenheit|Fahrenheit\tCelsius" << endl;**

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

**{**

**cout << fixed << setprecision(1) << celsius[i] << "\t" << celsius;**

**}**

**system("pause");**

**return 0;**

**}**

**2.#include<iostream>**

**using namespace std;**

**double celsius\_to\_fah(double cel)**

**{**

**double a = cel \* 1.8 + 32.0;**

**return a;**

**}**

**double fahrenheit\_to\_cels(double fah)**

**{**

**double b = (fah - 32.0) / 1.8;**

**return b;**

**}**

**5.#include <iostream>**

**using namespace std;**

**int number(int a, int depth = 0)**

**{**

**if (depth == 10)**

**{**

**return a;**

**}**

**int sum = a \* 2 + 1;**

**a = sum;**

**return number(sum, depth + 1);**

**}**

**int main()**

**{**

**cout << number(1) << endl;**

**system("pause");**

**return 0;**

**}**

**五、遇到的问题与解决方法**

**六、体会**

**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**

**1.#include<iostream>**

**using namespace std;**

**int main()**

**{**

**int numbers[10];**

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

**{**

**cin >> numbers[i];**

**}**

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

**{**

**bool isdistinct = true;**

**for (int j = 0; j < i; j++)**

**{**

**if (numbers[i] == numbers[j])**

**{**

**isdistinct = false;**

**break;**

**}**

**}**

**if (isdistinct)**

**{**

**cout << numbers[i] << "";**

**}**

**}**

**cout << endl;**

**return 0;**

**}**

**2.#include<iostream>**

**using namespace std;**

**double a[10];**

**int len = 10;**

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

**{**

**double d=\*a;**

**\*a = \*b;**

**\*b = d;**

**return 0;**

**}**

**void bubble(double a[])**

**{**

**bool charged = true;**

**while (charged)**

**{**

**charged = false;**

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

**{**

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

**{**

**swap(&a[j], &a[j + 1]);**

**charged = true;**

**}**

**}**

**}**

**}**

**int main()**

**{**

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

**{**

**cout << "请输入数字" << endl;**

**cin >> a[i];**

**}**

**bubble(a);**

**cout << "排序后的数组为：";**

**for (int m = 0; m < 10; m++)**

**{**

**cout << a[m] ;**

**}**

**}**

**3.#include<iostream>**

**using namespace std;**

**int main()**

**{**

**bool boolarray[100];**

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

**{**

**boolarray[i] = true;**

**}**

**for (int student = 1; student <=100; student++)**

**{**

**for (int drawer = student; drawer <= 100; drawer += student+1)**

**{**

**if (boolarray[drawer] == true)**

**{**

**boolarray[drawer] =false;**

**}**

**else**

**{**

**boolarray[drawer] = true;**

**}**

**}**

**}**

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

**{**

**if (boolarray[m] == true)**

**{**

**cout << m + 1 << " ";**

**}**

**}**

**system ("pause");**

**return 0;**

**}**

**4.**

#include <iostream>

using namespace std;

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

int current1 = 0;

int current2 = 0;

int current3 = 0;

while (current1 < size1 && current2 < size2) {

if (list1[current1] < list2[current2]) {

list3[current3++] = list1[current1++];

}

else {

list3[current3++] = list2[current2++];

}

}

while (current1 < size1) {

list3[current3++] = list1[current1++];

}

while (current2 < size2) {

list3[current3++] = list2[current2++];

}

}

int main() {

int list1[80], list2[80], list3[160];

int size1, size2;

cout << "Enter the size of list1: ";

cin >> size1;

cout << "Enter list1: ";

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

cin >> list1[i];

}

cout << "Enter the size of list2: ";

cin >> size2;

cout << "Enter list2: ";

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

cin >> list2[i];

}

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

cout << "The merged list is: ";

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

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

}

cout << endl;

return 0;

}

**5.**#include <iostream>

#include <string>

using namespace std;

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

string str1(s1);

string str2(s2);

int pos = str2.find(str1);

if (pos != string::npos) {

return pos;

}

else {

return -1;

}

}

int main() {

char s1[200];

char s2[200];

cout << "请输入第一个字符串：";

cin >> s1;

cout << "请输入第二个字符串：";

cin >> s2;

int result = indexOf(s1, s2);

if (result == -1) {

cout << s1 << " 不是 " << s2 << " 的子串" << endl;

}

else {

cout << s1 << " 是 " << s2 << " 的子串，下标为 " << result << endl;

}

return 0;

}

6.#include <iostream>

#include <cctype>

using namespace std;

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

{

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

{

counts[i] = 0;

}

for (int i = 0; s[i] != '\0'; i++)

{

char ch = tolower(s[i]);

if (ch >= 'a' && ch <= 'z')

{

counts[ch - 'a']++;

}

}

}

int main() {

const int MAX\_LENGTH = 100;

char s[MAX\_LENGTH];

int counts[26];

cout << "请输入一个字符串: ";

cin.getline(s, MAX\_LENGTH);

count(s, counts);

cout << "每个字母出现的次数如下: " << endl;

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

{

if (counts[i] != 0)

{

char ch = 'a' + i;

cout << ch << ": " << counts[i] << endl;

}

}

return 0;

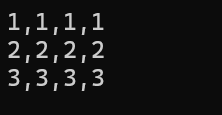
}

指针

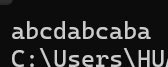
1. （1）



（2）



（3）



(4)

#include<iostream>

using namespace std;

int\* f()

{

int \*list=new int[4];

for (int i = 0; i < 4; 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>

#include<string>

using namespace std;

int indexof(const char\* S1, const char\* S2)

{

int lenth1 = 0, lenth2 = 0;

const char\* temp1 = S1;

const char\* temp2 = S2;

while (\*(temp1 + lenth1) != '\0')lenth1++;

while (\*(temp2 + lenth2) != '\0')lenth2++;

if (lenth1 > lenth2)

return -1;

for (int i = 0; i < lenth2 - lenth1; i++)

{

const char\* p1 = S1;

const char\* p2 = S2 + i;

int j;

for ( j = 0; j < lenth1; j++)

{

if (\*p1 != \*p2)

break;

p1++;

p2++;

}

if (j == lenth1)

return i;

}

return -1;

}

int main()

{

const int longg = 1000;

char s1[longg];

char s2[longg];

cout << "请输入字符串s1" << endl;

cin.getline(s1, longg);

cout << "请输入字符串s2" << endl;

cin.getline(s2, longg);

int result=indexof(s1, s2);

if (result != -1)

cout << "s1是s2的子串，第一次匹配的下表为：" <<result<< endl;

else

cout << "s1不是s2的子串" << endl;

}

（2）#include<iostream>

#include<cctype>

using namespace std;

int parseHex(const char\* const hexString)

{

const char\* p = hexString;

int result = 0;

while (\*p)

{

int value = \*p;

if (isdigit(value))

{

value = value - '0';

}

else if (isxdigit(value))

{

if (value >= 'A' && value <= 'F')

{

value

= value - 'A' + 10;

}

else if (value >= 'a' && value <= 'f')

{

value = value - 'a' + 10;

}

}

else

{

return 0;

}

result = result \* 16 + value;

p++;

}

return result;

}

int main()

{

int a = parseHex("A5");

cout << "A5的十进制表示方式为：" << a << endl;

}

（3）#include <iostream>

using namespace std;

int main()

{

int n = 0;

cout << "数组中元素个数为：" << endl;

cin >> n;

int\* arr = new int[n];

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

{

cout << "请输入元素：" << endl;

cin >> arr[i];

}

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

{

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

{

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

{

int t = arr[j];

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

arr[j + 1] = t;

}

}

}

cout << "排序后的数组为：" << endl;

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

{

cout << arr[q];

}

delete arr;

return 0;

}

**四、遇到的问题与解决方法**

**五、体会**