**计算机程序设计基础（C++)**

**实验报告**

专业班级：软件工程2402

学 号：8209240220

姓 名：李淇涵

**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？

函数返回的值是由函数类型决定的。

2. 本实验中主函数调用函数时采用的是何种传递方式？

1是引用传递，245为值传递

**四、算法分析，程序结果**

**1.**#include <iostream>

using namespace std;

int gcd(int a, int b)

{

while ( b!= 0)

{

int temp = b;

b = a % b;

a = temp;

}

return a;

}

int lcm(int a, int b, int& c)

{

return (a \* b / c);

}

int main()

{

int m, n;

cout << "请输入自然数m和n:" << endl;

cin >> m >> n;

int c = gcd(m, n);

int d = lcm(m, n, c);

cout << "m和n的最大公约数为:" << c << endl;

cout << "m和n的最小公倍数为:" << d << endl;

}

2.

#include <iostream>

#include <cmath>

using namespace std;

bool is\_prime(int num);

int main()

{

const int total = 200;

int count = 0; // 已找到的素数数量

int num = 2; // 从2开始检查每个数是否为素数

while (count < total)

{

if (is\_prime(num))

{

// 输出找到的素数

cout << num << "\t";

++count; // 增加找到的素数计数

if (count % 10 == 0)

{

// 每输出10个素数换行

cout << endl;

}

}

++num; // 检查下一个数

}

return 0;

}

bool is\_prime(int num)

{

if (num ==1)

{

return false;

}

if (num == 2)

{

return true;

}

if (num % 2 == 0)

{

return false;

}

for (int i = 3; i <= sqrt(num); i += 2)

{

if (num % i == 0)

{

return false;

}

}

return true;

}

**4.mytriangle.h:**

#include<iostream>

using namespace std;

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

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

**mytriangle.cpp:**

#include"mytriangle.h"

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

{

if (side1 + side2 > side3 && side1 + side3 > side2 && side2 + side3 > side1)

{

return true;

}

else

{

return false;

}

}

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

{

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

double square = sqrt(s \* (s - side1) \* (s - side2) \* (s - side3));

return square;

}

**test3.4.cpp:**

#include"mytriangle.h"

int main()

{

int side1, side2, side3;

cout << "请分别输入三边边长" << endl;

cin >> side1 >> side2 >> side3;

int val = is\_valid(side1, side2, side3);

if (val == true)

{

double square = area(side1, side2, side3);/

cout << "定义的三角形面积为:" << square << endl;

}

else

{

cout << "定义的三边长不合法" << endl;

}

return 0;

}

5.#include <iostream>

using namespace std;

int add(int n)

{

int m = (n + 1) \* 2;

return m;

}

int main()

{

int a = 1;

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

{

int b = add(a);

int temp = b;

b = a;

a = temp;

i++;

}

cout << "第一天猴子吃了" << a << "个桃子" << endl;

}

**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 <iostream>

using namespace std;

//#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**

**4.1.1**

#include <iostream>

using namespace std;

void sortFromSmallToLarge(int abc[], int c)

{

int a;

int b;

int k;

for (a = 0; a < c - 1; a++)

{

for (b = 0; b < c - a - 1; b++)

{

if (abc[b] > abc[b + 1])

{

int temp;

temp = abc[b];

abc[b] = abc[b + 1];

abc[b + 1] = temp;

}

}

}

}

void swap(int num[], int abc[])

{

int a;

int i;

int b;

int c = 0;

bool y;

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

{

y = true;

for (a = i + 1; a < 10; a++)

{

if (num[i] == num[a])

y = false;

}

if (y)

{

abc[c] = num[i];

c++;

}

}

sortFromSmallToLarge(abc, c);

for (int d = 0; d < c; d++)

cout << abc[d] << " ";

}

int main()

{

int num[10];

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

{

cin >> num[a];

}

int abc[10];

swap(num, abc);

}

4.1.2

#include <iostream>

using namespace std;

void sort(double a[10])

{

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

{

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

{

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

{

int temp = a[i + 1];

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

a[j] = temp;

}

}

}

}

int main()

{

cout << "请输入数组的值：" << endl;

double m[10];

// 使用循环读取数组元素

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

{

cin >> m[i];

}

// 调用排序函数

sort(m);

// 输出排序后的数组

cout << "冒泡排序之后的数组为：" << endl;

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

cout << m[i] << " ";

}

cout << endl;

return 0;

}

4.1.3

#include <iostream>

using namespace std;

int main()

{

bool a[100] = { false };

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

{

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

{

a[j] = !a[j];

}

}

cout << "Open lockers: ";

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

if (a[i])//如果a[i]的值是真的，那么if语句才可以执行，本身a[i]自带条件。

{

cout << i+1 << " ";

}

}

cout << endl;

return 0;

}

4.1.4

#include <iostream>

using namespace std;

void sort(int add[], int c)

{

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

{

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

{

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

{

int temp;

temp = add[j];

add[j] = add[j + 1];

add[j + 1] = temp;

}

}

}

}

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

{

int size3 = size1 + size2;

for (int i = 0; i < size3; i++)

{

if (i < size1)

{

list3[i] = list1[i];

}

else

{

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

}

}

}

int main()

{

int size1, size2, size3;

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

cout << "请输入数组1的长度：" << endl;

cin >> size1;

cout << "请输入数组1：" << endl;

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

{

cin >> list1[i];

}

cout << "请输入数组2的长度：" << endl;

cin >> size2;

cout << "请输入数组2：" << endl;

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

{

cin >> list2[i];

}

size3 = size1 + size2;

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

sort(list3, size3);

cout << "合并之后的数组是:" << endl;

for (int a = 0; a < size3; a++)

{

cout << list3[a] << endl;

}

}

4.1.5

#include <iostream>

#include <cstring>

using namespace std;

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

int len1 = strlen(s1);

int len2 = strlen(s2);

if (len1 == 0 || len2 == 0 || len1 > len2) {

return -1;

}

for (int i = 0; i <= len2 - len1; ++i) {

bool found = true;

for (int j = 0; j < len1; ++j) {

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

found = false;

break;

}

}

if (found) {

return i;

}

}

return -1;

}

int main() {

char s1[100], s2[100];

cout << "Enter string s1: ";

cin.getline(s1, 100);

cout << "Enter string s2: ";

cin.getline(s2, 100);

int index = indexOf(s1, s2);

if (index != -1) {

cout << "s1 is a substring of s2 at index: " << index << endl; // 如果是子串，输出下标

}

else {

cout << "s1 is not a substring of s2" << endl;

}

return 0;

}

4.1.6

#include <iostream>

#include <string>

using namespace std;

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

{

int len1 = strlen(s);

for (int i = 0; i < len1; i++)

{

char ch = s[i];

if (ch >= 'A' && ch <= 'Z')

{

ch += 'a' - 'A';

}

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

{

counts[ch - 'a']++;

}

}

}

int main()

{

const int a = 26;

int counts[a] = { 0 };

char input[100];

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

cin.getline(input, 100);

count(input, counts);

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

{

if (counts[i] != 0)

{

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

}

}

return 0;

}

【指针】

1.（1）

#include <iostream>

using namespace std;

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<< endl;

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

}

(2)

#include <iostream>

using namespace std;

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

}

(3)#include<iostream>

using namespace std;

//#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);

}

（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;

}

4.2.2.1

#include<iostream>

using namespace std;

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

{

int len\_s1 = strlen(s1);//求s1长度

int len\_s2 = strlen(s2);//求s2长度

int equal = 0;//计数=0

for (int i = 0; i < len\_s2; i++)

{

equal = 0;//计数=0

if (s1[0] == s2[i] && len\_s1 <= (len\_s2 - i + 1))//如果子串的第一项与母串的某一项相等 且子串长度小于母串剩余长度

{

equal++;//计数+1

for (int j = 1; j < len\_s1; j++)//建立循环

{

i++;//i+1

if (s1[j] == s2[i])//如果子串下一项和母串下一项相等

{

equal++;//计数+1

}

else//有不相等

{

break;//跳出循环

}

}

if (equal == len\_s1)//如果相等数和s1长度相等

{

return (i - len\_s1 + 1);//返回子串下标

}

else

{

continue;//否则进行下次循环

}

}

}

return -1;//不是子串关系 返回-1

}

int main()

{

char s1[100], s2[100];//定义两个字符串

cout << "请输入string1(子串):";//输入字符串s1

cin.getline(s1, 100);

cout << "请输入string2(母串):";//输入字符串s2

cin.getline(s2, 100);

int index = indexof(s1, s2);//调用函数

cout << "indexof(" << s1 << " , " << s2 << ")is" << index << endl;

return 0;

}

4.2.2.2

#include<iostream>

#include<cmath>

using namespace std;

int parseHex(const char\* const hexString)//定义函数

{

int s\_len = strlen(hexString);//定义数组长度

int num = 0;//定义计数=0

for (int i = 0; i < s\_len; i++)//建立循环

{

if (hexString[i] >= '0' && hexString[i] <= '9')//对于0-9的数值

{

num = num + (((int(hexString[i])) - 48) \* pow(16, s\_len - i - 1));//转换成int类型的0-9 根据相应位数乘上16的x-1次方

}

else if (hexString[i] >= 'A' && hexString[i] <= 'F')//对于A-F的数值

{

num = num + (((int(hexString[i])) - 55) \* pow(16, s\_len - i - 1));//转换为int类型的10-15 根据相应位数乘上16的x-1次方

}

else if (hexString[i] >= 'a' && hexString[i] <= 'f')//对于a-f的小写数值

{

num = num + (((int(hexString[i])) - 87) \* pow(16, s\_len - i - 1));//转换为int类型的10-15 根据相应位数乘上16的x-1次方

}

else {//对于其他字符

cout << "数值不是16进制" << endl;//提示字符串不合法

return 0;//返回错误值

}

}

return num;

}

int main()

{

char s[50];//定义数组

cout << "请输入16进制数字:";//输入数组

cin.getline(s, 50);

int dec = parseHex(s);//调用函数

cout << "十进制数字为:" << dec << endl;//输出十进制数字

}

4.2.2.3

#include<iostream>

using namespace std;

void bubble\_sort(int\* arr, const int\* size){

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

{

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

{

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

{

int temp = arr[j];

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

arr[j + 1] = temp;

}

}

}

}

int main()

{

int size;

cout << "请输入数组大小:";

cin >> size;

int\* arr = new int[size];//用new定义动态数组

cout << "请输入数组数值:";

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

cin >> arr[i];

}

bubble\_sort(arr, &size);

cout << "排序后的数组为:";

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

{

cout << \*&arr[i] << ' ';

}

delete[]arr;//删除创建的动态数组

return 0;

}

**四、遇到的问题与解决方法**

**五、体会**