**计算机程序设计基础（C++)**

**实验报告**

专业班级： 软件工程2303

学 号： 8209230326

姓 名： 于欣淇

**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？

**1.语法规定，怎么定义就返回什么类型**

**2.地址传递**

**四、算法分析，程序结果**

**1.**

**#include<iostream>**

**using namespace std;**

**#define ll long long**

**ll exgcd(ll a,ll b){**

**if(a%b==0){**

**return b;**

**}**

**return exgcd(b,a%b);**

**}**

**ll a,b;**

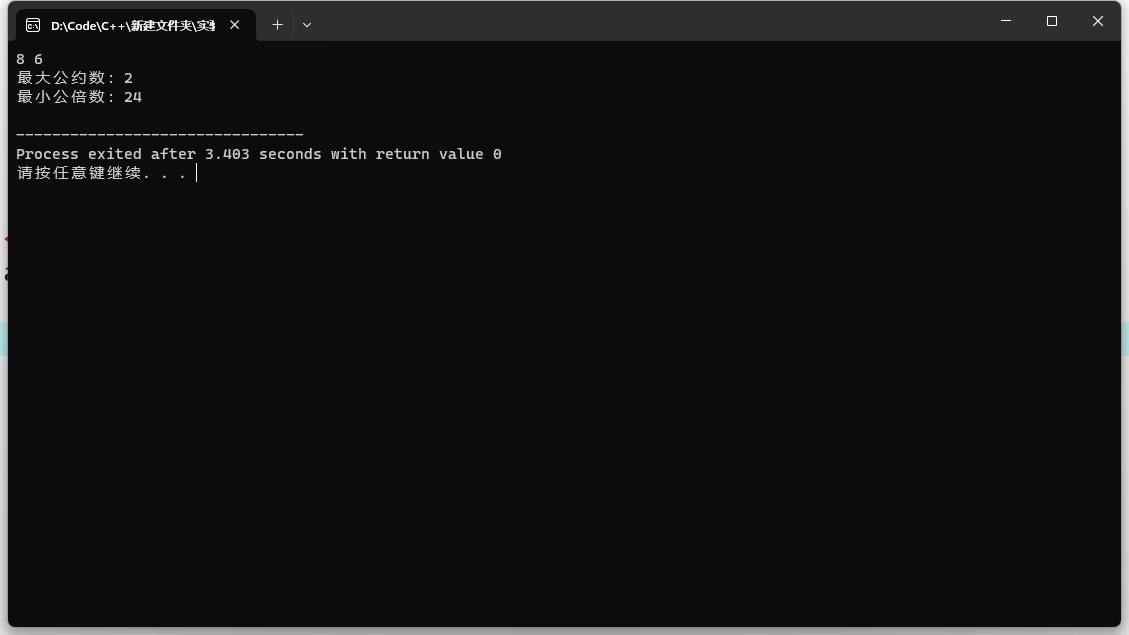
**int main(){**

**cin>>a>>b;**

**cout<<"最大公约数："<<exgcd(a,b)<<endl;**

**cout<<"最小公倍数："<<a\*b/exgcd(a,b)<<endl;**

**return 0;**

**}**

**2．**

**#include<iostream>**

**#include<string.h>**

**using namespace std;**

**int p[100000001],tot=1;**

**bool ip[100000001];**

**void phi(){**

**for(int i=2;i<=2000;i++){**

**if(ip[i]){**

**p[tot++]=i;**

**}**

**for(int j=1;j<=tot&&i\*p[j]<=2000;j++){**

**ip[i\*p[j]]=0;**

**if(i%p[j]==0) break;**

**}**

**}**

**}**

**bool is\_prime(int num){**

**if(ip[num]){**

**return true;**

**}**

**else return false;**

**}**

**int main(){**

**memset(ip,1,sizeof(ip));**

**phi();**

**int cnt=0;**

**for(int i=1;i<=200;i++){**

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

**cnt++;**

**if(cnt==10){**

**cout<<endl;**

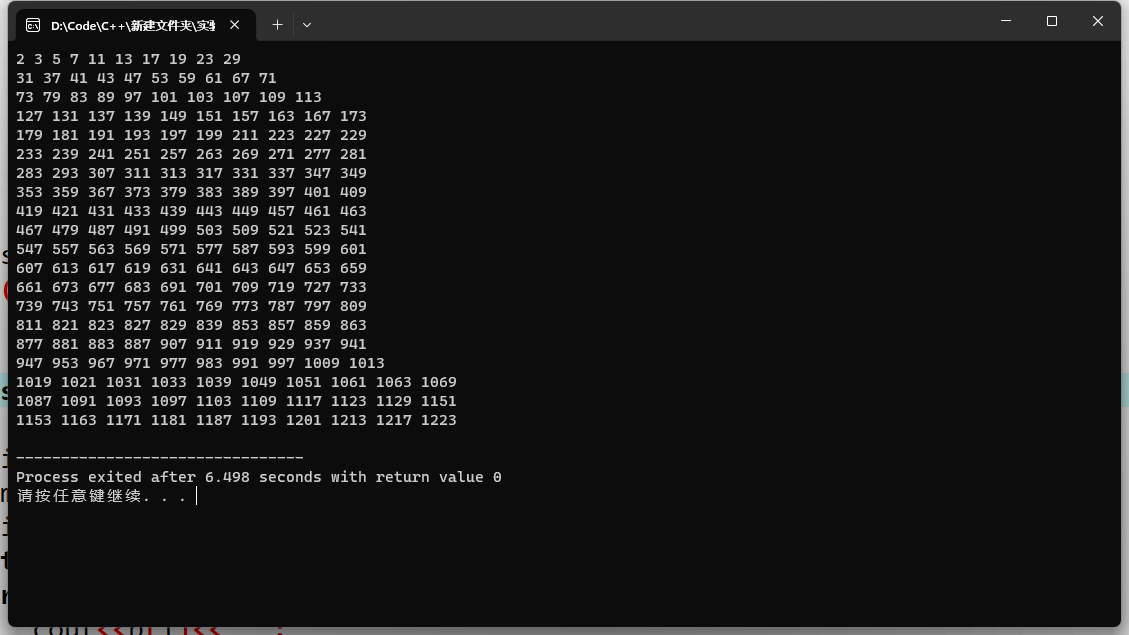
**cnt=0;**

**}**

**}**

**return 0;**

**}**



**3.**

#pragma once

#ifndef CIRCLE\_H

#define CIRCLE\_H

double celsius\_to\_fah(double cel);

double fahrenheit\_to\_cels(double fah);

#endif//头文件

double celsius\_to\_fah(double cel) {

double fah = 9.0 \* cel / 5.0 + 32.0;

return fah;

}

double fahrenheit\_to\_cels(double fah) {

double cel = 5.0\* (fah - 32.0) / 9.0;

return cel;

}//mytemperature.cpp

#include <iostream>

#include"mytemperature.h"

double x;

using namespace std;

int main(){

cout << "如果输入为摄氏度，请输入1" << endl;;

cout << "如果输入为华氏度，请输入2"<<endl;

int ch;

cin >> ch>>x;

if (ch == 1) {

cout << celsius\_to\_fah(x) << endl;

}

else cout << fahrenheit\_to\_cels(x) << endl;

return 0;

}//主函数

**5.**

**#include<iostream>**

**using namespace std;**

**int t=10,ans=1;**

**int f(){**

**if(t==1){**

**return ans;**

**}**

**ans=(ans+1)\*2;**

**t--;**

**return f();**

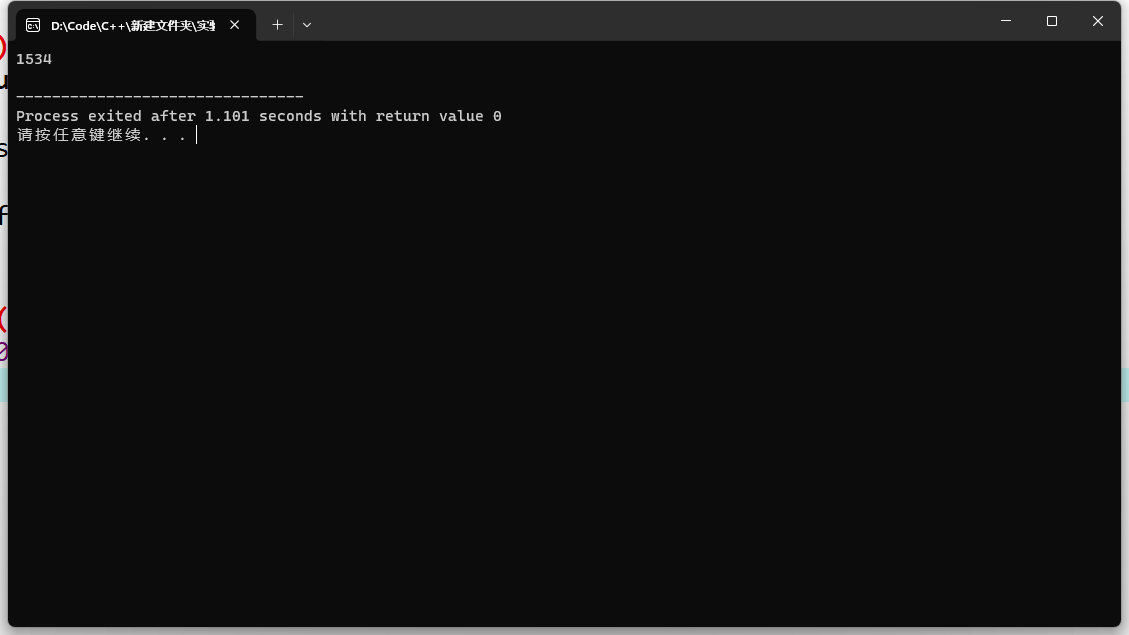
**}**

**int main(){**

**cout<<f()<<endl;**

**return 0;**

**}**



**五、遇到的问题与解决方法**

**不会写头文件**

**办法：查阅资料学会了**

**六、体会**

自学能力很重要

**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);

#include<cstdio>  
#include<cstring>  
#include<iostream>  
using namespace std;  
int len1, len2;  
int indexOf(const char s1[], const char s2[])  
{  
for (int i = 0;i <= len1 - len2 + 1;i++)  
{  
for (int j = 0;j < len2 && s2[j] == s1[i + j];j++)  
{  
if (j == len2 - 1 && s2[j] == s1[i + j])  
return i;  
               
  
}  
If(i==len1-len2)  
Return -1;  
  
}  
  
  
}  
int main()  
{  
char s1[50];  
char s2[50];  
cin.getline(s1, 50);  
cin.getline(s2, 50);  
len1 = strlen(s1);  
len2 = strlen(s2);  
int place = indexOf(s1, s2);  
cout << place+1<< endl;  
#include<cstdio>  
#include<cstring>  
#include<iostream>  
using namespace std;  
int len1, len2;  
int indexOf(const char s1[], const char s2[])  
{  
for (int i = 0;i <= len1 - len2 + 1;i++)  
{  
for (int j = 0;j < len2 && s2[j] == s1[i + j];j++)  
{  
if (j == len2 - 1 && s2[j] == s1[i + j])  
return i;  
               
  
}  
If(i==len1-len2)  
Return -1;  
  
}  
  
  
}  
int main()  
{  
char s1[50];  
char s2[50];  
cin.getline(s1, 50);  
cin.getline(s2, 50);  
len1 = strlen(s1);  
len2 = strlen(s2);  
int place = indexOf(s1, s2);  
cout << place+1<< endl;

(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

#include<cstdio>  
#include<cstring>  
using namespace std;  
int length;  
const char alphal[16] = { &apos;0&apos;,&apos;1&apos;,&apos;2&apos;,&apos;3&apos;,&apos;4&apos;,&apos;5&apos;,&apos;6&apos;,&apos;7&apos;,&apos;8&apos;,&apos;9&apos;,&apos;a&apos;,&apos;b&apos;,&apos;c&apos;,&apos;d&apos;,&apos;e&apos;,&apos;d&apos; };  
int parseHex(const char\* const hexString)  
{  
int hhex = 0;  
int num = 0;  
for (int i = 0; i < length; i++)  
{  
for (int j = 0; j < 16; j++)  
{  
  
if (hexString[i] == alphal[j])  
{  
num = j;  
}  
  
  
  
}  
hhex = hhex \* 16 + num;  
     
}  
return hhex;  
  
}  
int main()  
{  
char str[50];  
while (1)  
{  
memset(str, 0, sizeof(str));  
scanf\_s("%s", str, sizeof(str));  
int x = 0;  
length = strlen(str);  
x = parseHex(str);  
printf("%d", x);  
  
  
    
}  
  
  
}

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）。

#include<cstdio>  
#include<algorithm>  
using namespace std;  
void ssort(int\* p,int a)  
{  
sort(p, p + a );  
  
  
}  
int main()  
{  
int a=0;  
int\* p = new int[50];  
while (scanf\_s("%d", &a) == 1 && a != 0)  
{  
for (int i = 0; i < a; i++)  
{  
scanf\_s("%d", &p[i]);  
  
  
}  
ssort(p, a);  
  
for (int i = 0; i < a; i++)  
{  
      printf("%d ", p[i]);  
  
  
}  
  
  
}  
delete p;

【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<bits/stdc++.h>**

**using namespace std;**

**int s[11],a[800080],cnt=0;**

**int main(){**

**//memset(a,0,sizeof(a));**

**for(int i=1;i<=10;i++){**

**int x;**

**cin>>x;**

**a[x]++;**

**if(a[x]==1){**

**cnt++;**

**s[cnt]=x;**

**}**

**}**

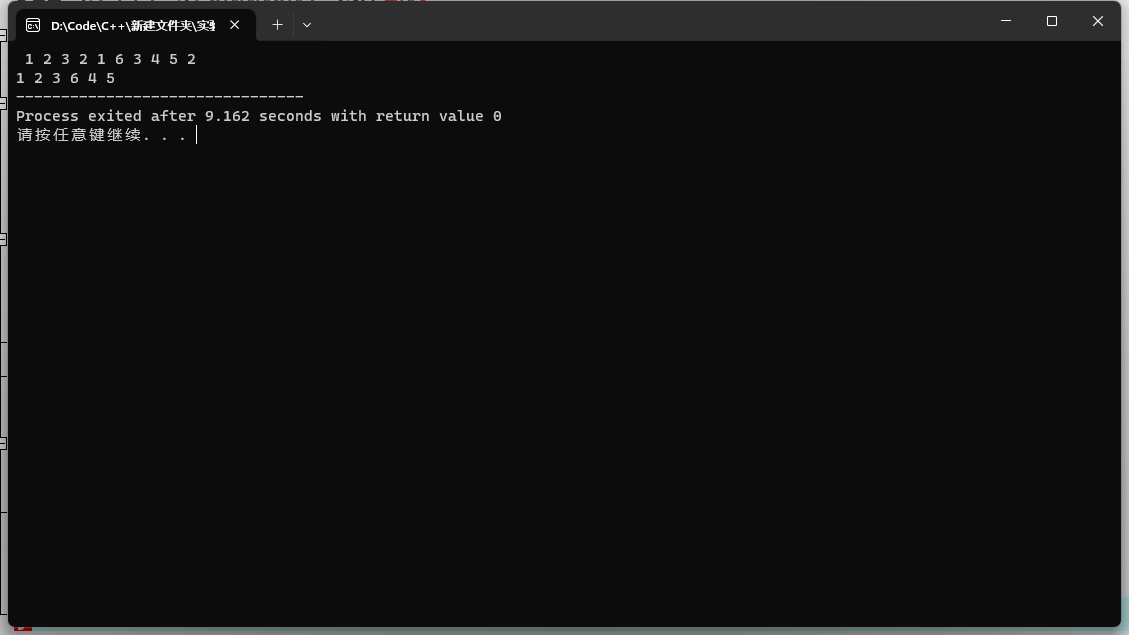
**// cout<<a[1]<<" "<<s[1]<<endl;**

**for(int i=1;i<=cnt;i++){**

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

**}**

**return 0;**

**}**

**2.**

**#include<iostream>**

**using namespace std;**

**double a[11];**

**void pop\_sort(){**

**for(int i=1;i<=10;i++){**

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

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

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

**}**

**}**

**}**

**}**

**int main(){**

**for(int i=1;i<=10;i++){**

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

**}**

**pop\_sort();**

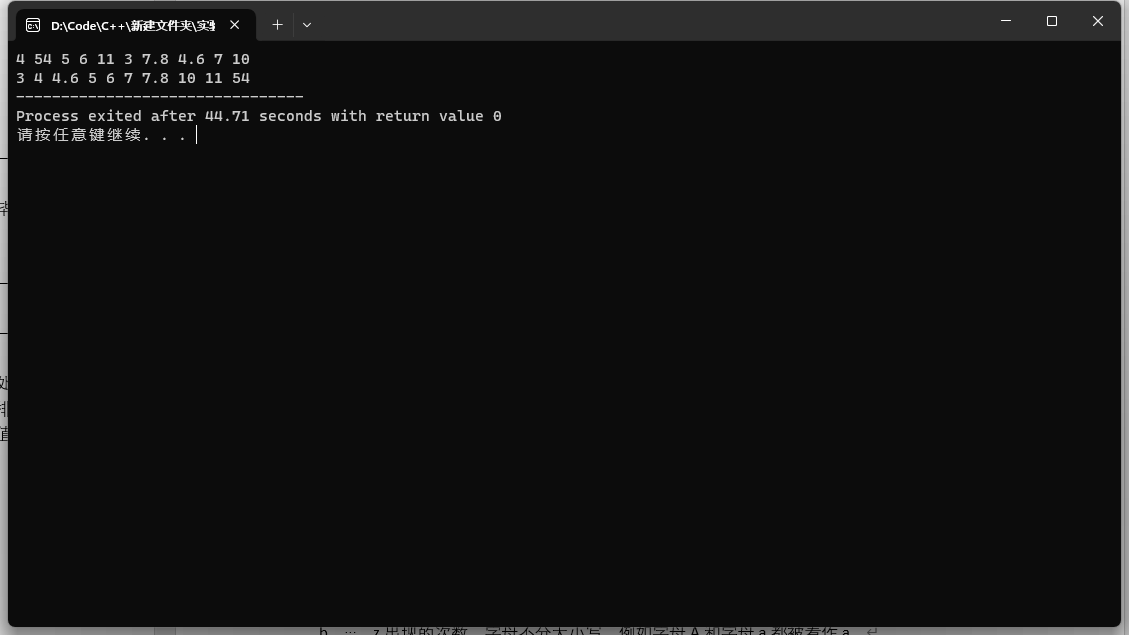
**for(int i=1;i<=10;i++){**

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

**}**

**return 0;**

**}**



**3.**

**#include<iostream>**

**using namespace std;**

**bool flag[105]={1};**

**int main(){**

**for(int i=2;i<=50;i++){**

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

**if(flag[j]){**

**flag[j]=false;**

**}**

**else flag[j]=true;**

**}**

**}**

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

**if(flag[i]){**

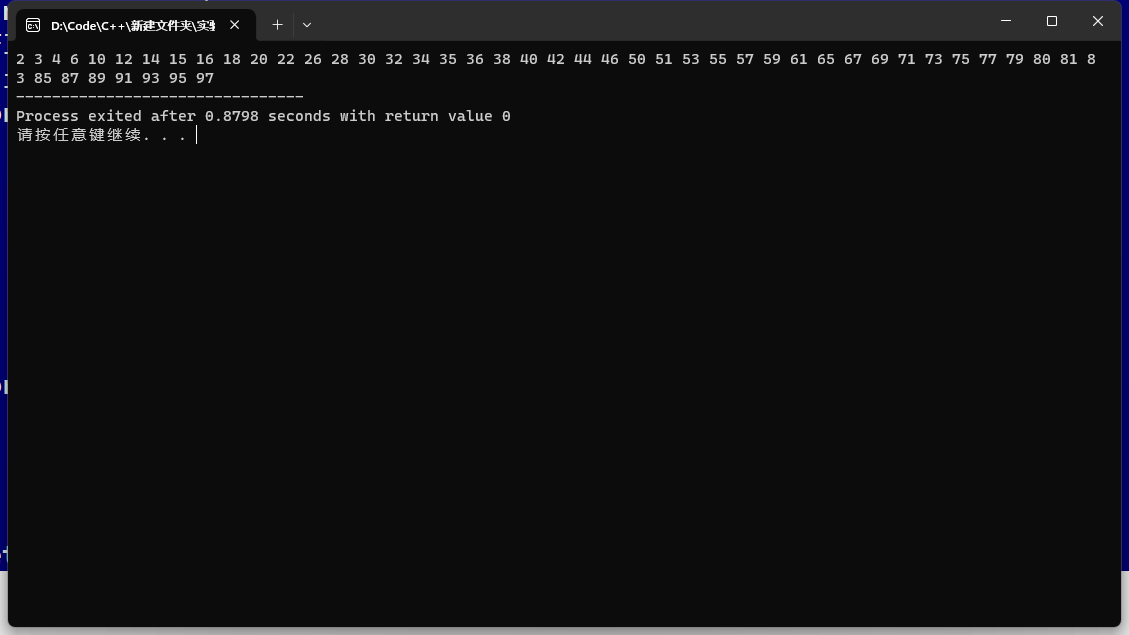
**cout<<i<<" ";**

**}**

**}**

**return 0;**

**}**



**4.**

**#include<iostream>**

**#include<algorithm>**

**using namespace std;**

**int list1[88],list2[88],list3[188];**

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

**for(int i=1;i<=size1;i++){**

**list3[i]=list1[i];**

**}**

**for(int i=size1+1;i<=size1+size2;i++){**

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

**}**

**sort(list3+1,list3+size1+size2+1);**

**}**

**int main(){**

**int n,m;**

**cout<<"Enter list1:";**

**cin>>n;**

**for(int i=1;i<=n;i++){**

**cin>>list1[i];**

**}**

**cout<<endl;**

**cout<<"Enter list2:";**

**cin>>m;**

**for(int i=1;i<=m;i++){**

**cin>>list2[i];**

**}**

**merge(list1,n,list2,m,list3);**

**cout<<"The merged list is";**

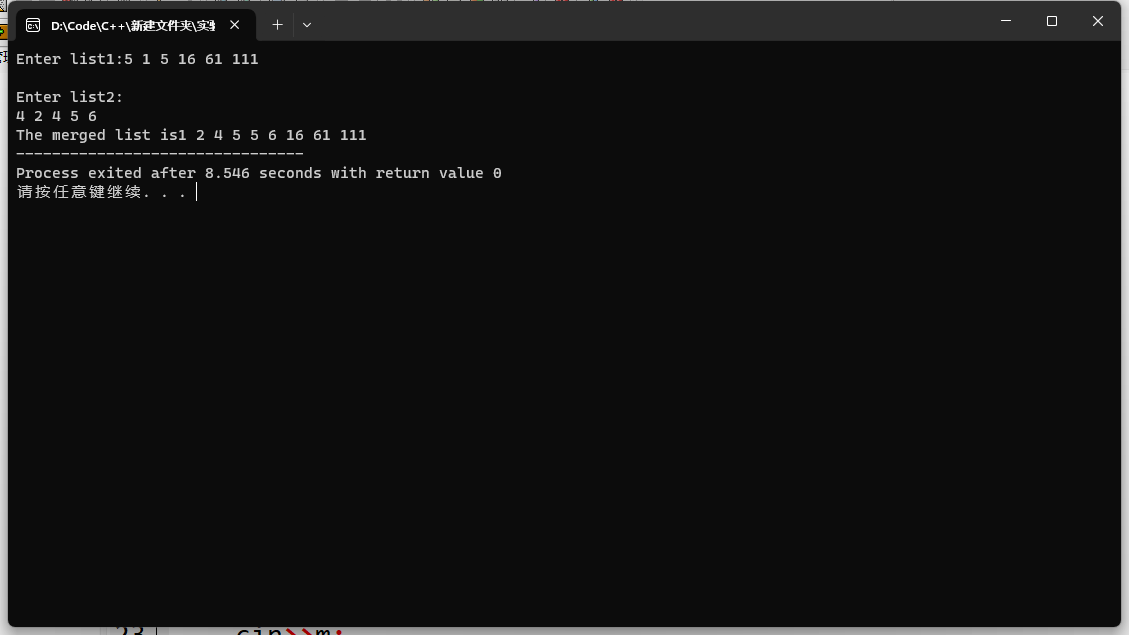
**for(int i=1;i<=m+n;i++){**

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

**}**

**return 0;**

**}**



**5.**

**#include<iostream>**

**#include<string>**

**using namespace std;**

**string a,b;**

**char s1[800080],s2[800080];**

**int cnt1,cnt2,u=0,p[1000010];**

**bool flag;**

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

**for(int i=1;i<cnt1;i++){**

**while(u>0&&s1[i]!=s1[u]){**

**u=p[u-1];**

**}**

**if(s1[u]==s1[i]){**

**u++;**

**}**

**p[i]=u;**

**}**

**u=0;**

**for(int i=0;i<cnt2;i++){**

**while(u>0&&s1[u]!=s2[i]){**

**u=p[u-1];**

**}**

**if(s2[i]==s1[u]){**

**u++;**

**}**

**if(u==cnt1){**

**printf("%d\n",i-cnt1+2);**

**flag=true;**

**u=p[u-1];**

**}**

**}**

**}**

**int main(){**

**cout<<"Enter the first string: ";**

**getline(cin,a);**

**cout<<endl;**

**cout<<"Enter the second string: ";**

**getline(cin,b);**

**int size1=a.length();**

**int size2=b.length();**

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

**if(a[i]!=' '){**

**s1[cnt1++]=a[i];**

**}**

**}**

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

**if(b[i]!=' '){**

**s2[cnt2++]=b[i];**

**}**

**}**

**cout<<"indexOf(\""<<a<<"\", \""<<b<<"\") is ";**

**indexOf(s1,s2);**

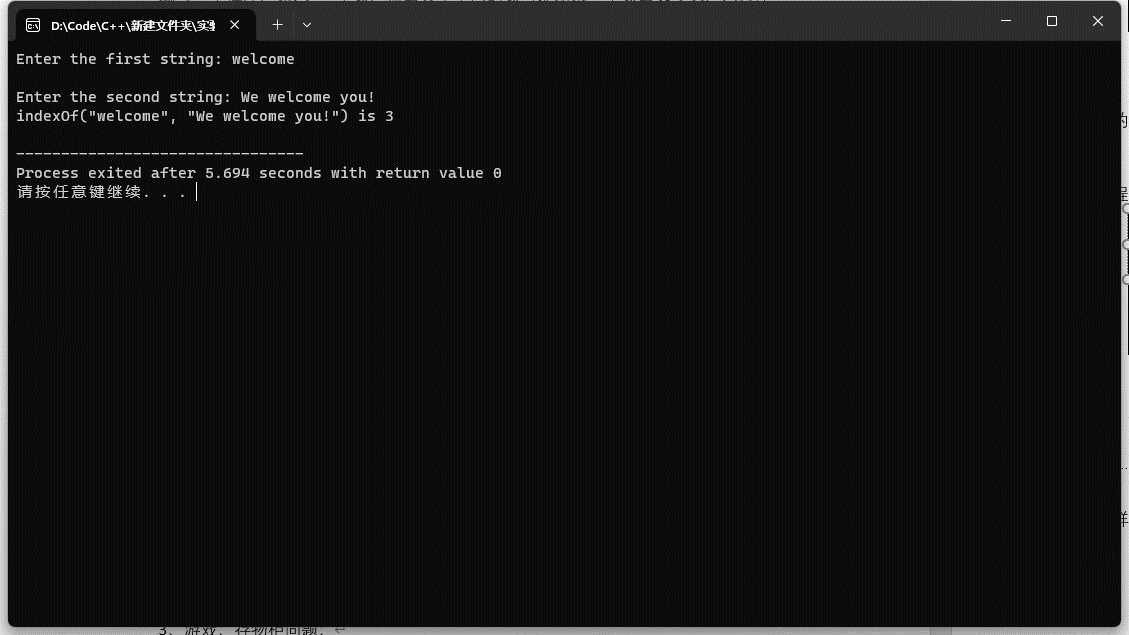
**if(!flag){**

**cout<<-1<<endl;**

**}**

**return 0;**

**}**



**6.**

**#include<iostream>**

**using namespace std;**

**int n;**

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

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

**if('a'<=s[i]&&s[i]<='z'){**

**int a=(int)s[i]-96;**

**counts[a]++;**

**}**

**}**

**}**

**char s[800080];**

**int counts[30];**

**int main(){**

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

**cin>>s;**

**cout<<endl;**

**n=sizeof(s);**

**count(s,counts);**

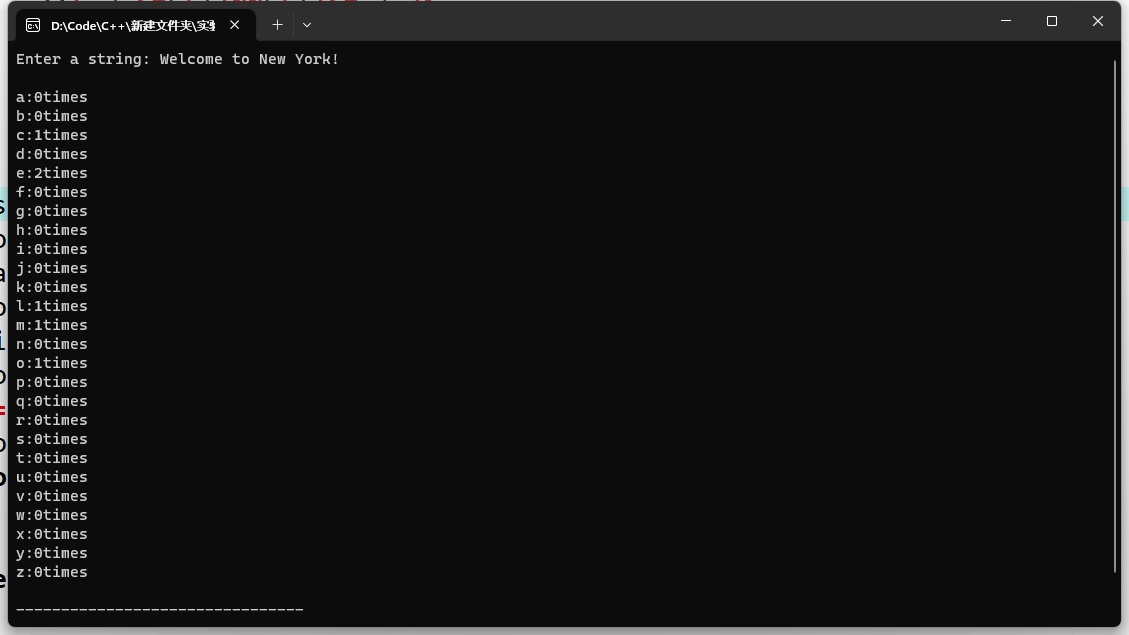
**for(int i=1;i<=26;i++){**

**cout<<char(i+96)<<":"<<counts[i]<<"times"<<endl;**

**}**

**return 0;**

**}**



**四、遇到的问题与解决方法**

**问题：字符串的空格的写入和处理，在遇到是卡住了**

**解决：读入用getline（cin）用去掉空格重新复制到另一组函数**

**五、体会**

**一定要实践，不实践只听课是搞不懂上课的讲的知识的。**