**计算机程序设计基础（C++）**

**实验报告**

**专业班级：软件工程2204班**

**学号：8209220406**

**姓名：张明浩**

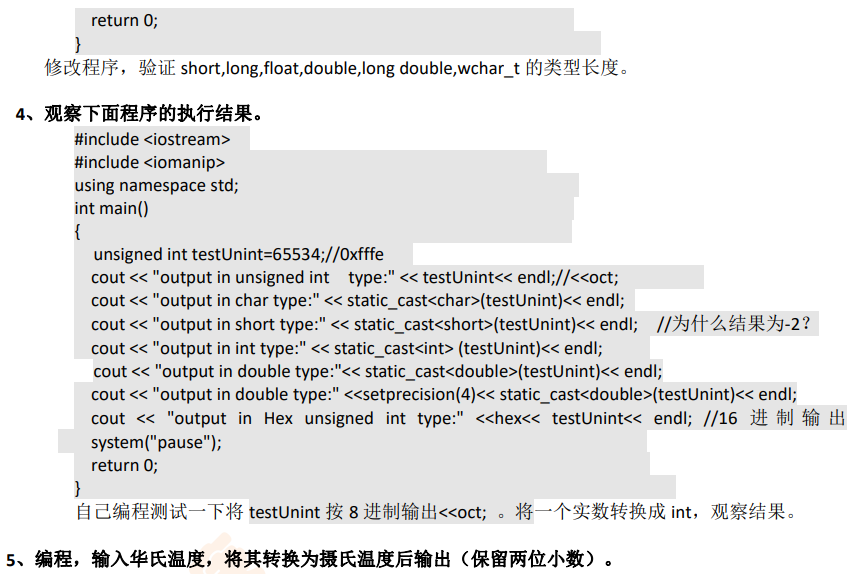
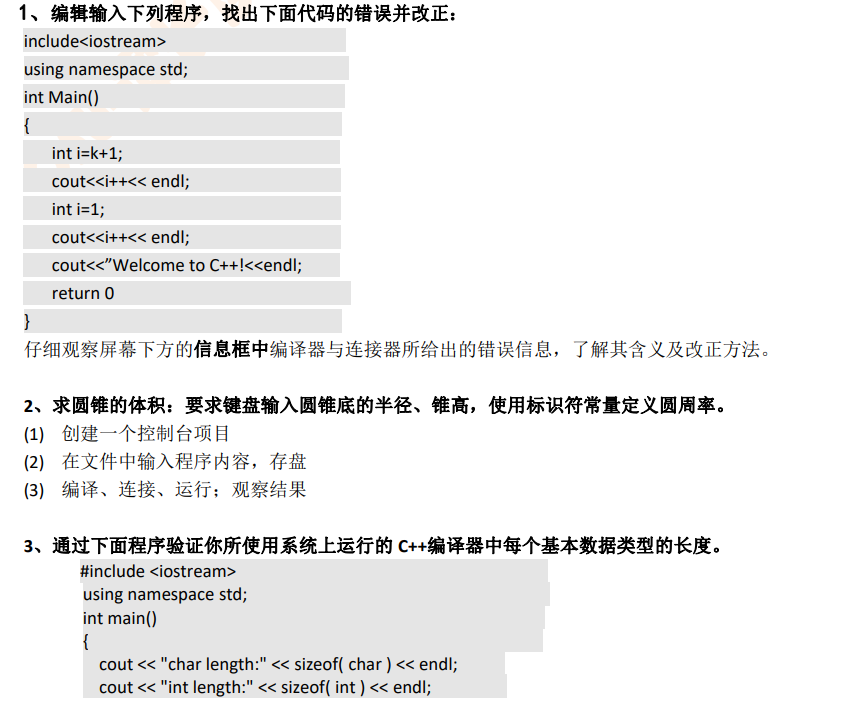
**实验报告成绩：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 实验 | 实验一 | 实验二 | 实验三 | 实验四 | 实验五 |
| 成绩 |  |  |  |  |  |

**批阅教师：**

**实验一、实验环境与简单程序设计**

1. **实验目的**
2. 掌握集成开发环境，掌握C++程序的基本要素以及完整的C++程序开发过程。
3. 掌握基本数据类型、运算符和表达式的使用。理解隐式转换和强制转换，理解数据超过该数据类型表示范围时的溢出。掌握不同数据类型之间混合运算中数据类型的转换。
4. 变量的定义与常量的使用。
5. 输入，输出的实现。
6. 编译信息的理解与错误的修改。
7. 简单程序的设计。
8. **实验内容**



1. **实验步骤，算法与结果分析**
2. 程序：

#include<iostream>

using namespace std;

int main()

{

int k;

int i=k+1;

cout << i++ << endl;

i=1;

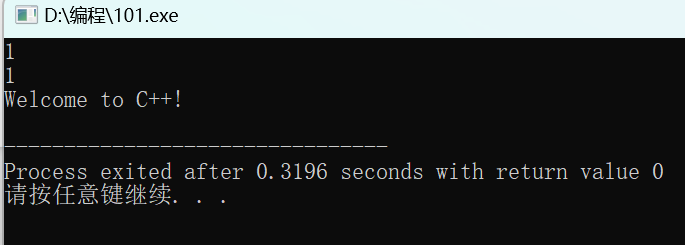
cout << i++ << endl;

cout << "Welcome to C++!" << endl;

return 0;

}

结果：



1. 程序：

#include<iostream>

using namespace std;

int main()

{

double V,r,h;

double a=3.14;

cin >> r;

cin >> h;

V=a\*r\*r\*h/3;

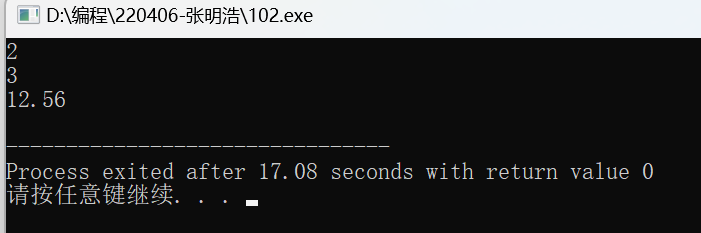
cout << V << endl;

return 0;

}

结果：

输入半径为2，高为3时观察输出结果



1. 程序：

#include<iostream>

using namespace std;

int main()

{

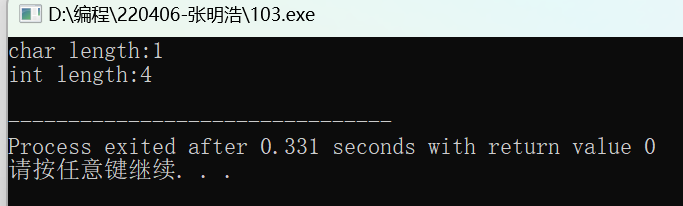
cout << "char length:" << sizeof(char) << endl;

cout << "int length:" << sizeof(int) << endl;

return 0;

}

结果：



1. 程序：

#include<iostream>

#include<iomanip>

using namespace std;

int main()

{

unsigned int testUnint=65534;//0xfffe

cout << "output in unsigned int type:" << testUnint<< endl;//<<oct;

cout << "output in char type:" << static\_cast<char>(testUnint)<< endl;

cout << "output in short type:" << static\_cast<short>(testUnint)<< endl; //为什么结果为-2？

cout << "output in int type:" << static\_cast<int> (testUnint)<< endl;

cout << "output in double type:"<< static\_cast<double>(testUnint)<< endl;

cout << "output in double type:" <<setprecision(4)<< static\_cast<double>(testUnint)<< endl;

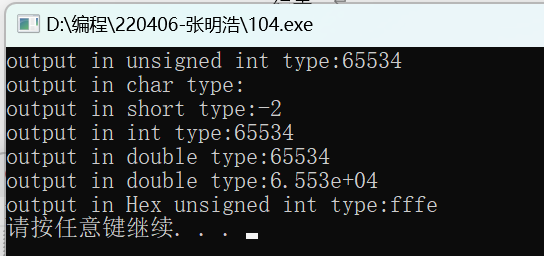
cout << "output in Hex unsigned int type:" <<hex<< testUnint<< endl; //16 进 制 输 出

system("pause");

return 0;

}

结果：



1. 程序：

#include<iostream>

#include<iomanip>

using namespace std;

int main()

{

float C,F;

cin >> F;

C=5\*(F-32)/9;

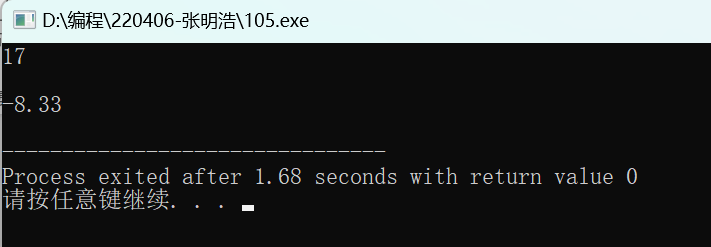
cout << setprecision(3) << endl;

cout << C << endl;

return 0;

}

结果：输入17摄氏度，观察输出结果。



1. **遇到的问题与解决办法**

在程序一中因未给变量k赋值而直接运行程序（使用Dev—C++进行编译）最终得到结果为1，在网上查阅资料后得知，若未给变量进行赋值，编译器会自动给变量赋值，且赋的值会因编译器不同而变化。不记得怎么使输出数据保留两位小数，上网查阅资料后得知可使用setpiecision（）函数进行设置。

1. **体会**

将近三个星期的军训过后，感觉一些不常用的C++知识差不多都要忘记了，只记得一些基础的，回归课堂后应该多看书，把基础知识捡起来，查漏补缺，为未来的学习打基础。

实验二、控制结构

1. **实验目的与要求**
2. 学习与掌握逻辑运算与逻辑表达式。
3. 熟练掌握if、switch、do while、for语句的语法结构与执行过程。
4. 掌握选择、循环程序的设计方法。
5. **实验内容**
6. 输入一个字符，如果为小写，转换为大写输出，否则，输出其后继字符的 ASCII 码值。
7. 输入 x 计算表达式的值

Y=3-2x 0<x<1

Y=2/4x+1 1<=x<5

Y=x^2 5<=x<10

分别输入0.2，1，5，0，观察输出结果。

3，输入三角形的三条边，求周长，并判断该三角形是否为等腰三角形（提示：要三边是否可以构成三角形）。

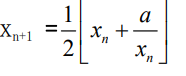
4、 完成计算器程序，实现（+ - \* / %）运算。考虑除数为 0 与运算符非法的情况

5、输入一行字符，分别统计出其中英文字母、空格、数字字符和其它字符的个数。 提示：从键盘上读入一个字符给变量 c，判断 c 是属于哪种字符并计数，循环读入下个字符，直到回 车换行字符'\n'为止。 cin，scanf（）都不能读入空格以及‘\n’字符，查找资料解决输入这两个字符的方法。 （这个题训练大家自主学习能力以及如何获取新知识、探索解决未知问题的能力。）

6、编写一个程序：从键盘上输入两个正整数，求 a 和 b 的最大公约数与最小公倍数。

7、使用循环结构输出下列图形： \* \*\* \*\*\* \*\*\*\* \*\*\*\*\*

8、从键盘输入 a，用迭代法求 a 的平方根 x= a 。求平方根的迭代公式



要求精确到|xn+1 - xn|，提示：迭代法是把 xn代入迭代公式右边，计算出 xn+1来，然后把 xn+1 作为新的 xn ，计算出新的 xn+1， 如此重复，直到|xn+1 - xn|。

xn+1 为所求的平方根。可以把 a作为xn的初始值。思考：（1）如果输入 a 为负，在运行时会出现什么情况? 修改程序使之能处理任何的 a 值。 (2）能否|xn+1 - xn|。

9、苹果每个 0.8 元，第一天买 2 个，第二天开始，每天买前天的 2 倍，直到购买的苹果数不超过 100 的最大值，求每天平均花多少钱。

三、**算法分析、程序与结果**

一、算法分析、程序与结果

1. 算法：

程序：

#include<iostream>

using namespace std*;*

int main()

{

char alph;

cin >> alph;

if(alph>=97&&alph<=122)

{

alph=toupper(alph);

cout << int（alph） << endl;

}

else

{

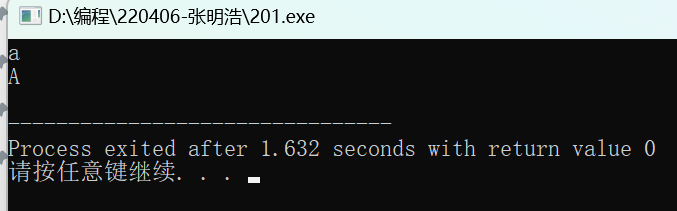
cout << alph << endl;

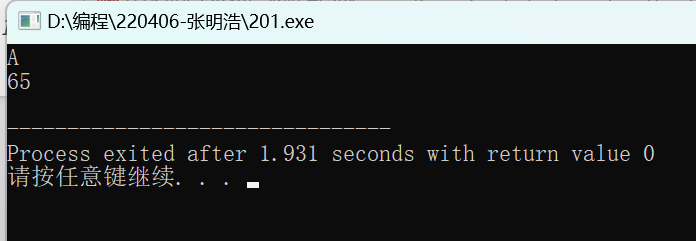
}

return 0;

}

结果：





2、程序：  
 #include<iostream>

using namespace std;

int main()

{

double x;

double y;

cin >> x;

if(x<1&&x>0)

{

y=3-2\*x;

}

else if(x<5&&x>=1)

{

y=2/(4\*x)+1;

}

else if(x>=5&&x<10)

{

y=x\*x;

}

else

{

cout << "Error." << endl;

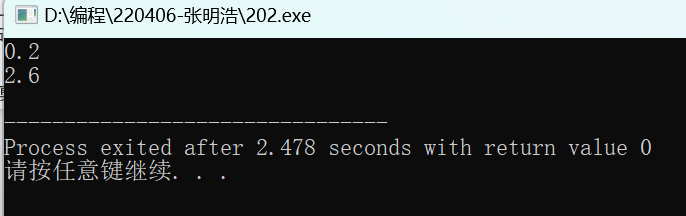
}

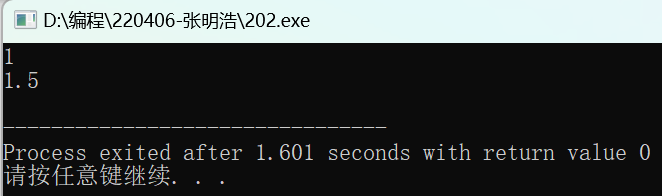
cout << y << endl;

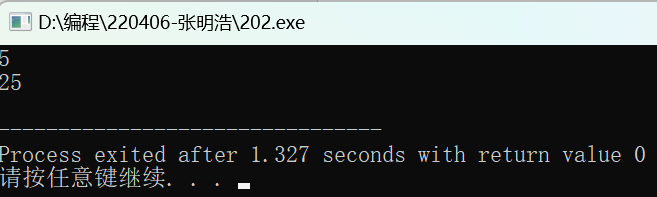
return 0;

}

结果：分别输入0.2，1，5，观察输出结果。







3、程序：

#include<iostream>

using namespace std;

int main()

{

float a,b,c;

float C;

cin >> a;

cin >> b;

cin >> c;

if(a+b<=c||a+c<=b||b+c<=a)//最好不要从正面判断

{

cout << "Error." << endl;

}

else

{

C=a+b+c;

cout << C << endl;

if(a==b||a==c||b==c)//一个等号是赋值 ；两个等号才是相等

{

cout << "This is a isosceles triangle." << endl;

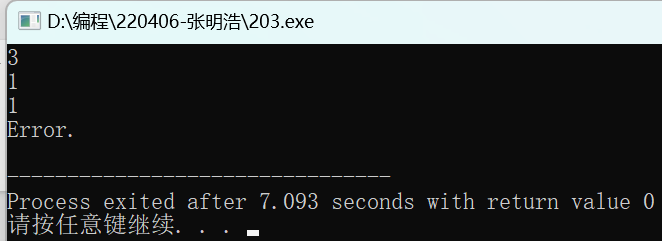
}

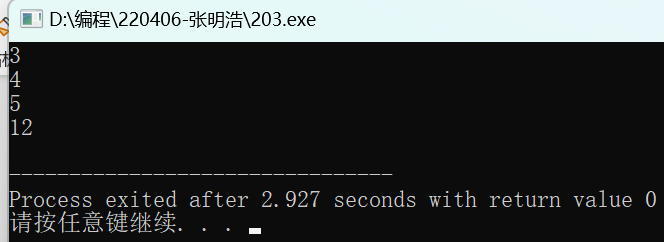
}

return 0;

}

结果：





4、程序：

#include<iostream>

using namespace std;

int main()

{

int a,b,c;

char d;

cin >> a;

cin >> d;

cin >> b;

if(d=='+')

{

c=a+b;

cout << c << endl;

}

else if(d=='-')

{

c=a-b;

cout << c << endl;

}

else if(d=='\*')

{

c=a\*b;

cout << c << endl;

}

else if(d=='/')

{

if(b==0)

{

cout << "Error." << endl;

}

else

{

c=a/b;

cout << c << endl;

}

}

else if(d=='%')

{

if(b==0)

{

cout << "Error." << endl;

}

else

{

c=a%b;

cout << c << endl;

}

}

else

{

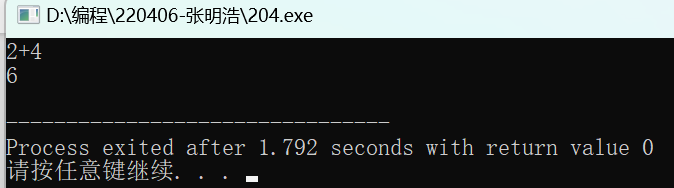
cout << "Error." << endl;

}

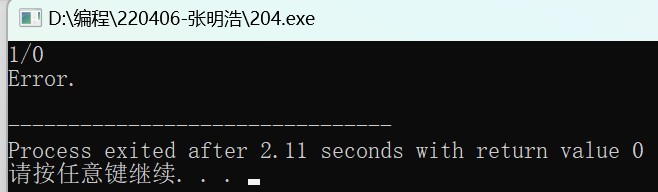
return 0;

}

结果：输入2+4



输入1/0



5、程序：

#include<iostream>

#include<string>

#include<istream>

using namespace std;

int main()

{

string str;

getline(cin,str);

int a=0,b=0,c=0,d=0;

int i=0;

while(str[i]!=0)

{

if((str[i]>='a'&&str[i]<='z')||(str[i]>='A'&&str[i]<='Z'))

{

++a;

}

else if(str[i]==' ')

{

++b;

}

else if((str[i]>='0')&&(str[i]<='9'))

{

++c;

}

else

{

++d;

}

i++;

}

cout << a<< endl;

cout << b<< endl;

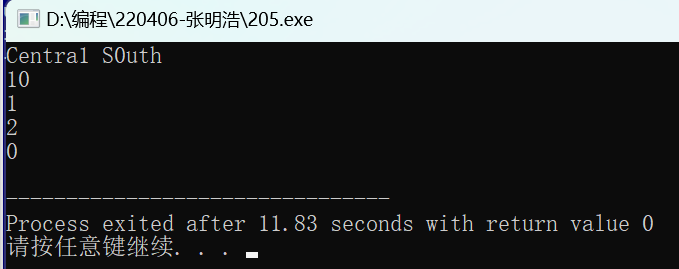
cout << c << endl;

cout << d<< endl;

return 0;

}

结果：输入Centra1 S0uth，观察输出结果:



6.程序：

#include<iostream>

using namespace std;

int main()

{

int a,b;

cin >> a;

cin >> b;

if(a<b)

{

int temp;

temp=a;

a=b;

b=temp;

}

if(a%b==0)

{

cout << "greatest common divisor:" << b << endl;//最大公约数

cout << "Maximum common multiple:" << a << endl;//最大公倍数

}

else

{

cout << "greatest common divisor:" << 1 << endl;//最大公约数

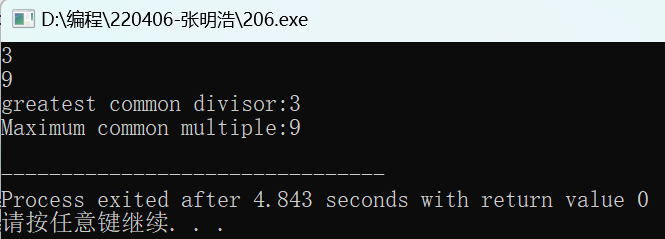
cout << "Maximum common multiple:" << a\*b << endl;//最大公倍数

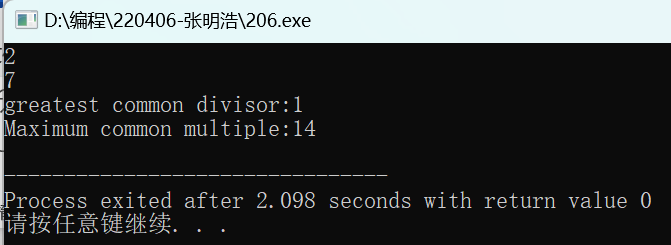
}

return 0;

}

结果：输入3，9和2，7，观察输出结果：





7.程序：

#include<iostream>

using namespace std;

int main()

{

int n=5;

int i=0;

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

{

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

{

cout << " ";

}

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

{

cout << "\*";

}

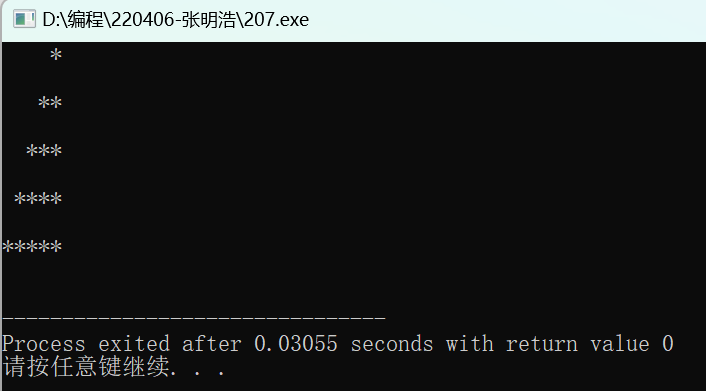
cout <<"\n"<< endl;

}

return 0;

}

结果：



8.程序：

#include <iostream>

#include <cmath>

using namespace std;

int main()

{

int a;

double x1,x2;

cin >> a;

x1=a/2;

x2=(x1+a/x1)/2;

while(fabs(x1-x2)>=1e-5)

{

x1=x2;

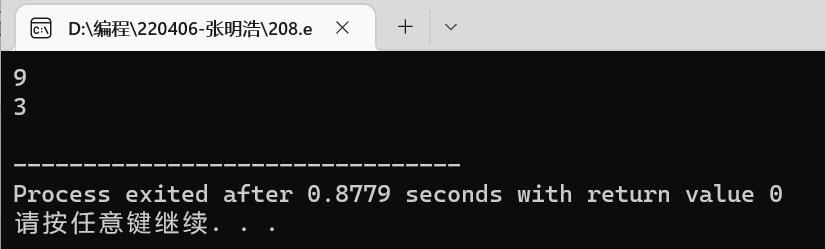
x2=(x1+a/x1)/2;

}

cout << x2 << endl;

return 0;

}结果：



9.程序：

#include<iostream>

using namespace std;

int main()

{

float price1=0;

float average=0;

int a=2,n=0;

for(int a=2;a<=100;a=a\*2)

{

price1=(a\*0.8)+price1;

n++;

}

cout << n << endl;

cout << price1 << endl;

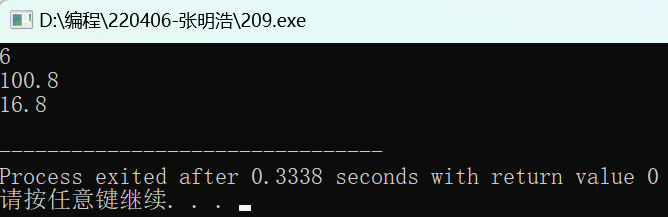
average=price1/(n);

cout << average << endl;

return 0;

}

结果：



四、遇到的问题与解决办法：

在编写实验二第五题时因为粗心把&&打成了||而导致输出结果不对，也忘记了if else结构，导致程序不会统计其他字符，最后在老师的帮助下解决了问题。第7题不知道怎么输出的图形，最后上网查资料后知道用两个for循环可以实现。

五、体会：

虽然学了将近两个月的C++，但是还有很多东西不会，有些学过的东西也不能学以致用，在编程过程中转化成代码，在今后的学习中应该巩固基础，多练习，争取能够把书本上的知识转化成代码，能够流利使用。

**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。

（2）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）3，4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、实验步骤、算法与结果分析**

1、程序：

#include <iostream>

using namespace std;

int func(int &m,int &n)

{

if(m>n)//将m，n排序，使n总是大的那个；

{

int temp;

temp=m;

m=n;

n=temp;

}

if(n%m==0)

{

cout << m << endl;

cout << n << endl;

}

else

{

cout << 1 << endl;

cout << m\*n << endl;

}

return 0;

}

int main()

{

int m,n;

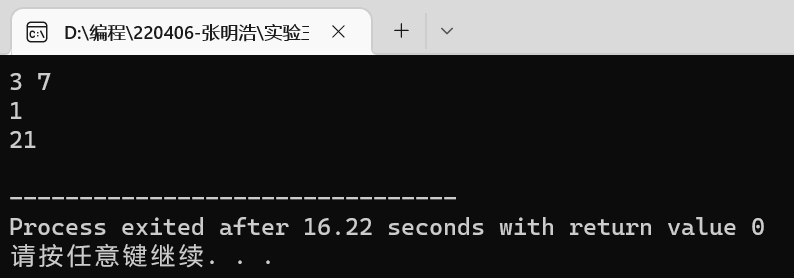
cin >> m >> n;

func(m,n);

return 0;

}

结果：



2、程序：

#include<iostream>

#include<cmath>

#include<iomanip>

using namespace std;

bool is\_prime(int num)

{

double stop=sqrt(num);

if(num==1)

{

return 0;

}

for(int k=2;k<=stop;k++)

{

if(num%k==0)

{

return 0;

}

}

return 1;

}

int prime()

{

int size=0;

for(int i=1;size<200;i++)

{

if(is\_prime(i))

{

cout << i << "\t";

if((size+1)%10==0)

cout << endl;

size++;

}

}

return 0;

}

int main()

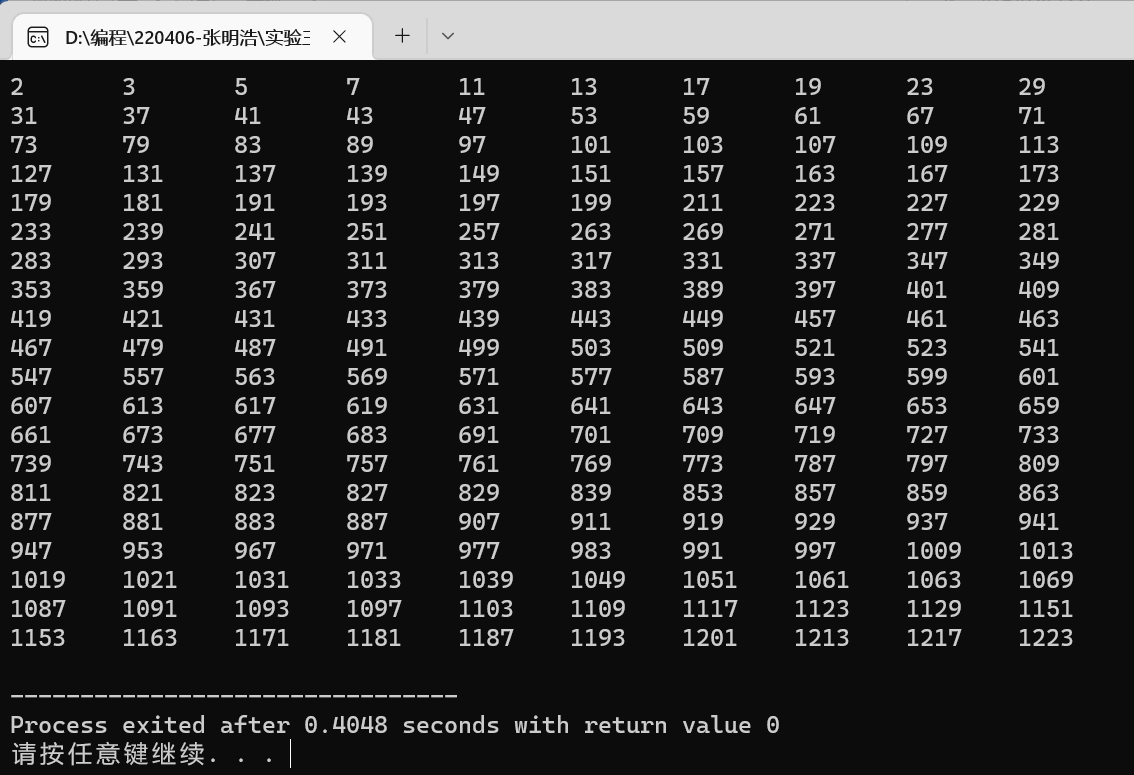
{

prime();

return 0;

}

结果：



3或4、程序：

#ifndef SOMETHING\_H

#define SOMETHING\_H

#include<cmath>

#include<iostream>

using namespace std;

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

{

if(side1>side2)

{

double temp;

temp=side1;

side1=side2;

side2=temp;//按从小到大排列；

}

if(side1>side3)

{

double temp;

temp=side1;

side1=side3;

side3=temp;

}

if(side2>side3)

{

double temp;

temp=side2;

side2=side3;

side3=temp;

}

if(side1+side2<side3)

{

cout << “Error.”；

return 0;

}

return 1;

}

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

{

double s,M;

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

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

return M;

}

#endif

#include<iostream>

#include "mytriangle.h"

using namespace std;

int main()

{

double side1,side2,side3;

cin >> side1 >> side2 >> side3;

if(is\_valid(side1,side2,side3))

{

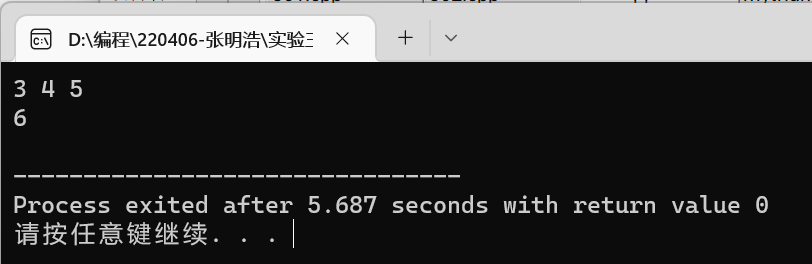
cout << area(side1,side2,side3)<< endl;

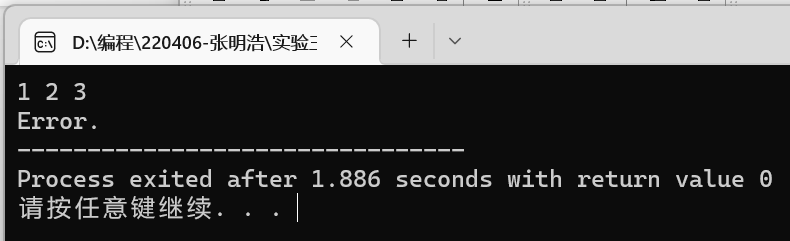
}

return 0;

}

结果：





5、程序：

#include <iostream>

using namespace std;

int main()

{

int x=1;

for(int k=1;k<10;k++)

{

x=(1+x)\*2;

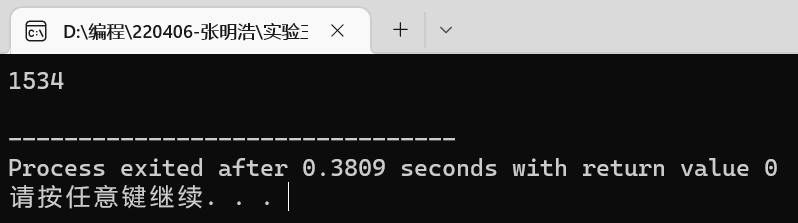
}

cout << x <<endl;

return 0;

}

结果：



**四、遇到的问题与解决办法：**

在实验304中，刚开始编写了两个cpp文件，在头文件中声明一个cpp文件中的两个函数，再在另一个cpp文件中调用第一个cpp文件中的函数，结果在编译时，编译器报错：

“Id returned 1 exit status”,刚开始由于不知道这个报错是什么原因，后来上网查询资料才知道是因为不能有两个cpp文件，最终就把第一个cpp文件放在头文件里了。

在实验302中，完全没有头绪，不知道怎么使求到的素数自增，从而实现求出前200个素数，最终在苦思冥想下有了一点思路，但用我的思路写出来的程序不能每行输出10个数据，对此在求助大佬后，使用取模运算限制输出个数从而解决了问题。

五、体会

编程虽然有时候碰到难的题目会让人没有学下去的欲望，但当自己做出来之后就会有一种满足感，就会有想要继续学下去的欲望，在这过程中学到的东西的印象也会比较深刻，动手实践还是比只看书学得更快。

**实验四、数组与指针**

**一、【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。

**二、【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 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 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 the first string: welcome

Enter the second string: We welcome you!

indexOf(“welcome”, “We welcome you!”) is 3

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 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）。

**三、【实验步骤、算法与结果分析】**

1、程序：

#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 << list[0] <<"\t";

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]<<"\t";

}

else

{

continue;

}

}

else

{

break;

}

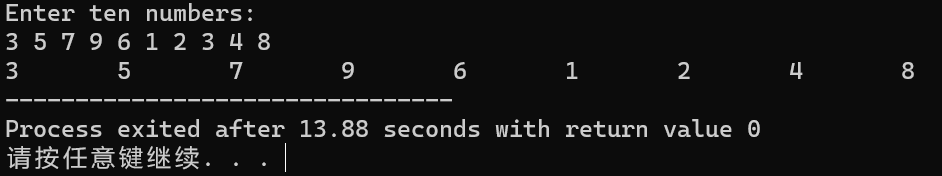
}

}

return 0；

}

结果：



2、程序：

#include<iostream>

using namespace std;

const int size=10;

void bubblesort(double list[10])

{

double temp;

bool changed=true;

do

{

changed=false;

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

{

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

{

temp=list[j];

list[j]=list[j+1];

list[j+1]=temp;

changed=true;

}

}

}while(changed);

}

int main()

{

double list[10];

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

{

cin >> list[i];

}

bubblesort(list);

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

{

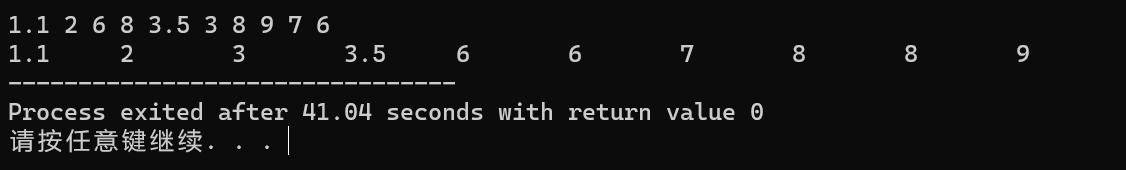
cout << list[k] << "\t";

}

return 0;

}

结果：



3、程序：

#include<iostream>

using namespace std;

const int q=1;

int main()

{

bool box[100];

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

{

box[k]=false;

}

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

{

for(int n=m;n<100;n+=m+1)

{

if(box[n]==true)

{

box[n]=false;

}

else

{

box[n]=true;

}

}

}

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

{

if(box[p]==true)

{

cout << p+1 <<"\t";

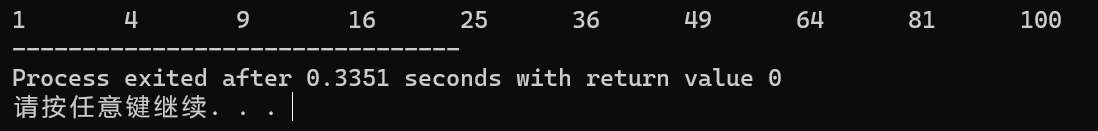
}

}

return 0;

}

结果：



4、程序：

#include<iostream>

#include<string>

using namespace std;

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

int main()

{

int size1,size2,size3;

cout<<"Enter size1: ";

cin>>size1;

cout<<"Enter list1: ";

int list1[size1];

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

cin>>list1[i];

}

cout<<"Enter size2: ";

cin>>size2;

cout<<"Enter list2: ";

int list2[size2];

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

cin>>list2[i];

}

size3=size1+size2;

int list3[size3];

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

cout<<endl;

return 0;

}

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

{

int temp;

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

list3[j]=list1[j];

}

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

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

}

cout<<"The merged list is : ";

for (int m=0;m<(size1+size2)-1;m++){

for(int b=0;b<(size1+size2)-1;b++){

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

{

temp=list3[b];

list3[b]=list3[b+1];

list3[b+1]=temp;

}

}

}

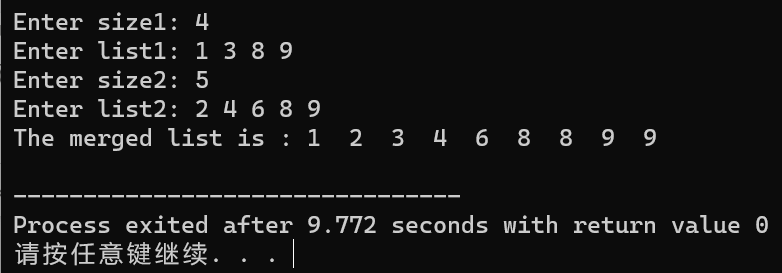
for (int a=0;a<(size1+size2);a++){

cout<<list3[a]<<" ";

}

}

结果：



5、程序：  
#include<iostream>

#include<cstring>

using namespace std;

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

int main()

{

int size=999;

char s1[size],s2[size];

cout<<"Enter the first string : ";

cin.getline(s1,size);

cout<<"Enter the second string: ";

cin.getline(s2,size);

cout<<"indexOf(\""<<s1<<"\", \""<<s2<<"\") is "<<indexOf(s1,s2)<<endl;

return 0;

}

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

{

int a,b,index,j,k;

a=strlen(s1);

b=strlen(s2);

bool s3[a];

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

s3[m]=false;

for(int i=0;i<b-a+1;i++) //控制s2的首字母

{

for(j=0,k=i;(j<a)&&(k<i+a);j++,k++) //控制s2的总输出数

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

{

index=i;

s3[j]=true;

}

int h=0;

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

{

if(s3[n]==false&&i==b-a)

{

return -1;break;

}

if(s3[n])

h++;

}

if(h==a)

{

return index;

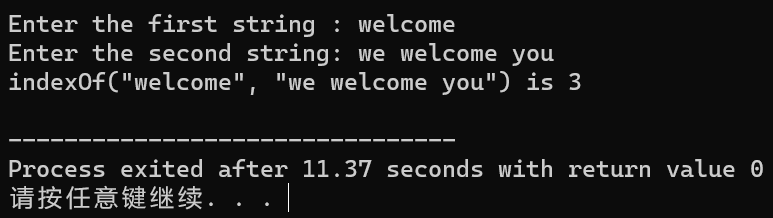
}

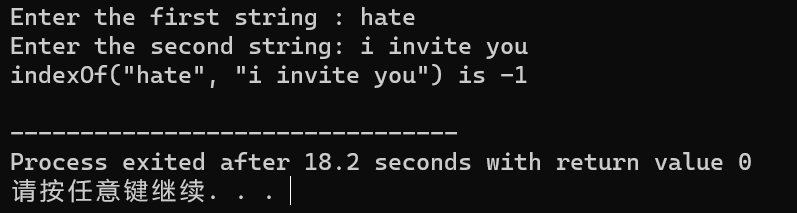
}

return 0;

}

结果：





6、程序：

#include <iostream>

using namespace std;

const int num1=26;

const int num2=99;

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

{

for(int i=0;i<num1;i++)

{

counts[i]=0;

}

for(int k=0;k<num2;k++)

{

counts[s[k]-'a']++;

counts[s[k]-'A']++;

}

for(int j=0;j<num1;j++)

{

if(counts[j]!=0)

{

cout << static\_cast<char>(j+'a') << ":" << counts[j] << "times" << endl;

}

}

}

int main()

{

char s[num2];

int counts[num1];

cout << "Enter a string:";

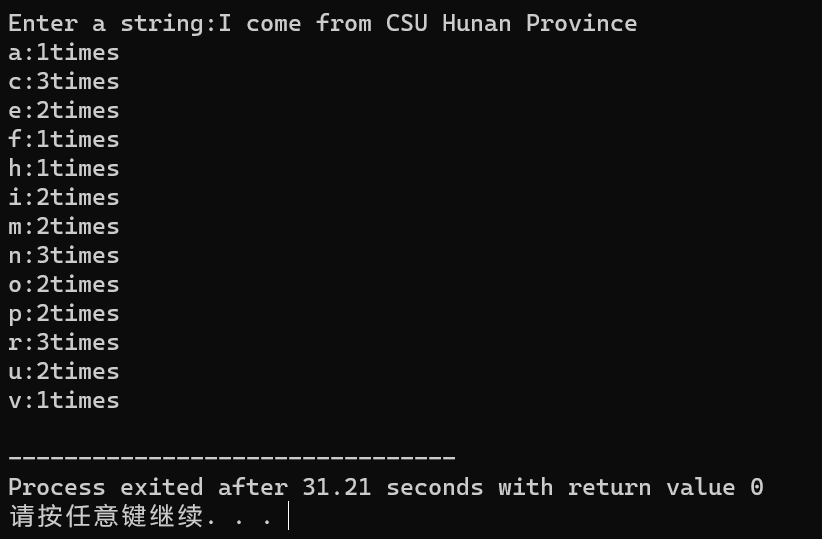
cin.getline(s,num2);

count(s,counts);

return 0;

}

结果：



7、程序：

#include<iostream>

#include<cstring>

using namespace std;

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

int main()

{

int size=999;

char s1[size],s2[size];

cout<<"Enter the first string : ";

cin.getline(s1,size);

cout<<"Enter the second string: ";

cin.getline(s2,size);

cout<<"indexOf(\""<<s1<<"\", \""<<s2<<"\") is "<<indexOf(s1,s2)<<endl;

return 0;

}

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

{

int a,b,index,j,k;

a=strlen(s1);

b=strlen(s2);

bool s3[a];

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

s3[m]=false;

for(int i=0;i<b-a+1;i++) //控制s2的首字母

{

for(j=0,k=i;(j<a)&&(k<i+a);j++,k++) //控制s2的总输出数

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

{

index=i;

s3[j]=true;

}

int h=0;

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

{

if(s3[n]==false&&i==b-a)

{

return -1;break;

}

if(s3[n])

h++;

}

if(h==a)

{

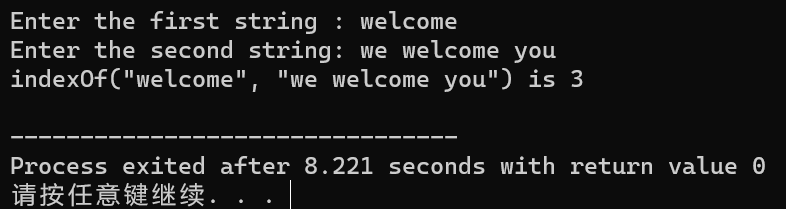
return index;

}

}

return 0;

}

结果：  


8、程序：

#include<iostream>

#include<string>

#include<cmath>

using namespace std;

int parseHex(const char\*const hexString)

{

int a,sum=0;

a=strlen(hexString);

int list[a];

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-1-i);

}

sum=sum+list[i];

}

return sum;

}

int main(){

char \* str;

str= new char[20];

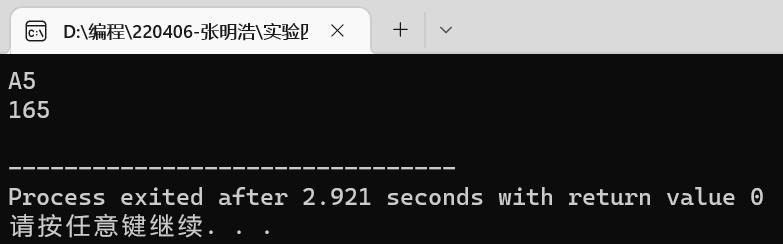
cin.get(str,20);

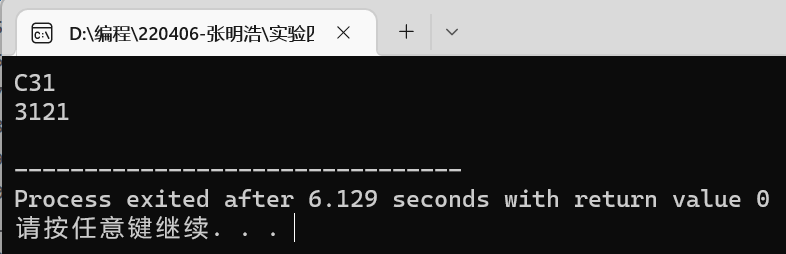
cout << parseHex(str) << endl;

return 0;

}

结果：





9、程序：

#include<iostream>

#include<string>

using namespace std;

int sort(int p[],int size)

{

int temp;

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

{

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

{

if(\*(p+1+i)<\*(p+i))

{

temp=\*(p+i+1);

\*(p+i+1)=\*(p+i);

\*(p+i)=temp;

}

}

}

return 0;

}

int main()

{

int size,l;

cin >> size;

int \*p;

p=new int[size];

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

{

cin >> \*(p+m);

}

cout << "Enter the number of this arr:" << endl;

cin >> l;

cout << \*(p+l-1) << endl;;

sort(p,size);

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

{

cout << \*(p+m) << "\t";

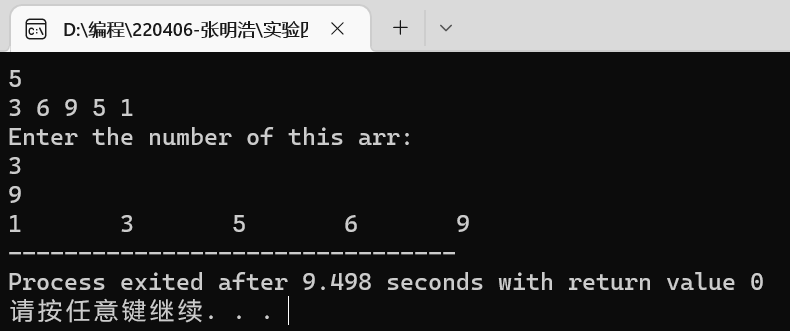
}

delete p;

return 0;

}

结果：



**四、遇到的问题与解决方法：**

这次实验几乎每道题都很难，每道题都至少写了两个小时，其中的问题也很多，最难的还是指针的运用。不会求取指针指向的动态整型数组的长度，查找资料后才知道这样的数组长度求不出来，只能使用全局变量事先给定。数组的排序看似简单，但稍微复杂一点的数组需要考虑的东西就很多，需要缜密的思维，需要将这些比较步骤转化成代码。有些细微的差错通过请教老师和同学解决。

**五、体会：**

数组和指针果然是C++中最难的部分之二，做完这个实验，发现还有很多数组和指针方面的知识没有掌握，没有理解，以至于做实验的时候不会灵活运用。应该马上把他们补起来，多花点时间看看书动手写代码帮助理解。

**实验五、类与对象**

**一、【实验目的与要求】**

1、掌握声明类的方法，类和类的成员的概念以及定义对象的方法。

2、初步掌握用类和对象编制基于对象的程序。

3、学习检查和调试基于对象的程序。

**二、【实验内容】**

1、有以下程序：

#include<iostream>

using namespace std；

class Time // 定义 Time 类

{

public： // 数据成员为公用的

int hour;

int minute;

int sec ;

};

int main()

{

Time tl; //定义 t1 为 Time 类对象

cin>>t1.hour; //输入设定的时间

cin>>t1.minute;

cin>>t1.sec;

cout<<t1.hourl<<”：”<<t1.minute<<”：”<<t1.sec<<endl ;

return 0;

}

改写程序，要求：

(1)将数据成员改为私有的；

(2)将输入和输出的功能改为由成员函数实现；

(3)在类体内定义成员函数。

然后编译和运行程序。请分析什么成员应指定为公用的?什么成员应指定为私有的? 什么函

数最好放在类中定义? 什么函数最好在类外定义?

2、分别给出如下的 3 个文件：

(1)含类定义的头文件 student.h，

//student.h (这是头文件，在此文件中进行类的声明)

class Student //类声明

{

public: //公用成员函数原型声明

void display();

private:

int num;

char name[20];

char sex;

};

(2)包含成员函数定义的源文件 student.cpp

//student.cpp 在此文件中进行函数的定义

#include <iostream>

#include”student.h” //不要漏写此行，否则编译通不过

void Student::display() //在类外定义 display 类函数

{

cout<<”num：”<<num<<endl;

cout<<”name：”<<name<<endl;

cout<<”sex：”<<sex<<endl;

}

(3)包含主函数的源文件 main.cpp。

为了组成一个完整的源程序，应当有包括主函数的源文件：

//main.cpp 主函数模块

#include <iostream> //将类声明头文件包含进来

#include “student.h”

int main()

{

Student stud; //定义对象

Student stud1(007,”tcg”,’m’);

stud.display(); //执行 stud 对象的 display 函数

return 0;

}

请完善该程序，在类中增加一个对数据成员赋初值的成员函数 set\_value。上机调试并运行。

3、需要求 3 个长方柱的体积，请编一个基于对象的程序。数据成员包括 length(长)、width(宽)、

height(高)。要求用成员函数实现以下功能：

(1)由键盘分别输入 3 个长方柱的长、宽、高；

(2)计算长方柱的体积；

(3)输出 3 个长方柱的体积。

请编程序，上机调试并运行。

4、编写程序：定义抽象基类 Shape，由它派生出五个派生类：Circle（圆形）、Square（正

方形）、Rectangle（ 长方形）、Trapezoid （梯形）和 Triangle （三角形），用虚函数分别计算

各种图形的面积，并求出它们的和。要求用基类指针数组。使它的每一个元素指向一个派生类的

对象。

注：主函数中定义如下对象

Circle circle(10.2);

Square square(3);

Rectangle rectangle(3,4);

Trapezoid trapezoid(2.0,4.5,3);

Triangle triangle(4,5,6);

5、设计一个函数，返回两个向量的交集，函数头如下：

template<typename T>

vector<T> intersection(const vector<T>& v1, const vector<T>& v2)

两个向量的交际为它们的共同包含的元素，编写一个测试程序，提示用户输入两个向量，每

个包含 5 个字符串，打印输出它们的交集。

**三、实验步骤、算法与结果分析**

1、程序：

#include <iostream>

using namespace std;

class Time

{

private:

int hour;

int minute;

int sec;

public:

Time(int hour=0,int minute=0,int sec=0)

{

this->hour=hour;

this->minute=minute;

this->sec=sec;

}

void Showtime()

{

cout << hour << ":" << minute << ":" << sec << endl;

}

};

int main()

{

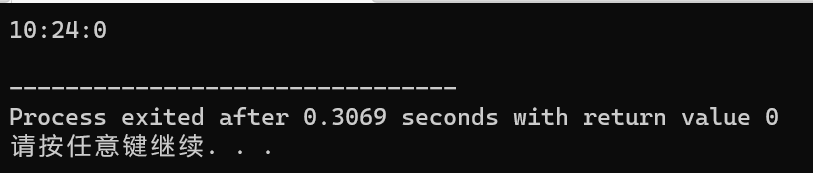
Time t1(10,24,00);

t1.Showtime();

return 0;

}

结果：



2、程序：

（1）头文件：

#ifndef SOMETHING\_H

#define SOMETHING\_H

class Student {

private:

int num;

char name[20];

char sex;

// Private section

public:

void display();

Student(int num=000,char name[20]="NULL",char sex='m')

{

this->num=num;

for(int i=0;i<20;i++){

this->name[i]=name[i];

}//写成this->name[20]=name[20]就只给第二十一个数组元素赋值了

this->sex=sex;

}// Public Declarations

protected:

// Protected Declarations

};

#endif

（2）源文件：

#include <iostream>

#include "student.h"

using namespace std;

void Student::display()

{

cout << "num:"<<num << endl;

cout << "name:" << name << endl;

cout << "sex:" << sex << endl;

}

int main()

{

Student stud;//未初始化，默认使用构造函数

Student stud1(007,"tcg",'m');//给定数据初始化

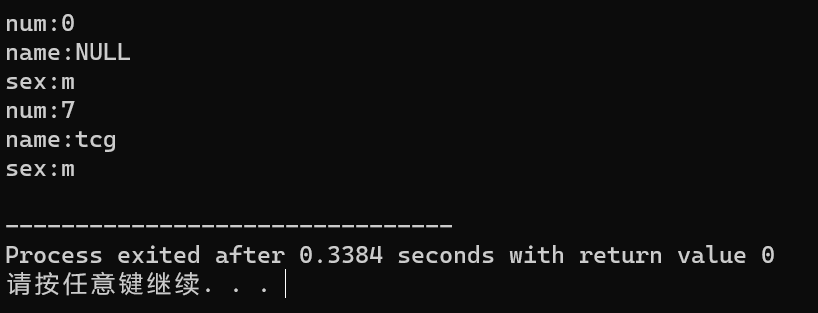
stud.display();

stud1.display();

return 0;

}

结果：



3、程序：

#include <iostream>

using namespace std;

class cuboid

{

private:

double height;

double length;

double width;

public:

cuboid(double height=0,double length=0,double width=0)

{

this->height=height;

this->length=length;

this->width=width;

}

void Setdata(){

cin >> height;

cin >> length;

cin >> width;

}

double calculateV(){

double V;

V=height\*length\*width;

cout << "V=" << V << endl;

return 0;

}

};

int main()

{

cuboid A1;

A1.Setdata();

cuboid A2;

A2.Setdata();

cuboid A3;

A3.Setdata();

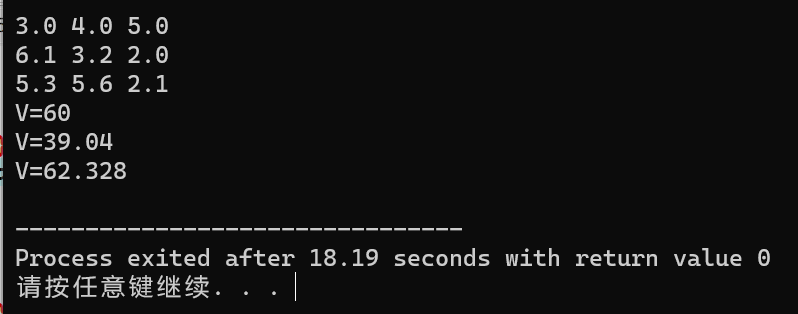
A1.calculateV();

A2.calculateV();

A3.calculateV();

return 0;

}

结果：  


4、程序：

#include <iostream>

#include <cmath>

const double PI=3.14;

using namespace std;

class Shape

{

public:

virtual double area()=0;

virtual ~Shape(){

};

};

class Circle:public Shape

{

private:

double r;

public:

Circle(double r)

{

this->r=r;

}

double area()

{

return PI\*r\*r;

}

~Circle(){

};

};

class Square:public Shape

{

private:

int d;

public:

Square(int d)

{

this->d=d;

}

double area()

{

return d\*d;

}

~Square(){

};

};

class Rectangle:public Shape

{

private:

int a;

int b;

public:

Rectangle(int a,int b)

{

this->a=a;

this->b=b;

}

double area()

{

return a\*b;

}

~Rectangle(){

};

};

class Trapezoid:public Shape

{

private:

double a,b,c;

public:

Trapezoid(double a,double b,double c)

{

this->a=a;

this->b=b;

this->c=c;

}

double area()

{

return (a+b)\*c/2;

}

~Trapezoid(){

};

};

class Triangle:public Shape

{

private:

int a,b,c;

public:

Triangle(int a,int b,int c)

{

this->a=a;

this->b=b;

this->c=c;

}

double area()//进行三角形面积计算时判定三角形是否成立

{

if(a+b>c&&a+c>b&&b+c>a)

{

double S;

S=sqrt((a+b+c)\*(a+b-c)\*(a+c-b)\*(b+c-a))/4;

return S;

}

else

{

cout << "error" << endl;

return 0;

}

}

~Triangle(){

};

};

int main()

{

Shape \* A[5];

A[0]=new Circle (10.2);//circle类对象进行初始化;

A[1]=new Square (3);//square类对象进行初始化

A[2]=new Rectangle (3,4);//rectangle类对象进行初始化

A[3]=new Trapezoid (2.0,4.5,3);//trapezoid类对象进行初始化

A[4]=new Triangle (4,5,6);//triangle类对象进行初始化

for(int i=0;i<5;i++)

{

cout << A[i]->area() << endl;

//delete A[i];如果在这里就释放了内存，后面面积求和无法返回结果

}

double sum=0;

double s[5];

for(int i=0;i<5;i++){

s[i]=A[i]->area();

sum=sum+s[i];

delete A[i];

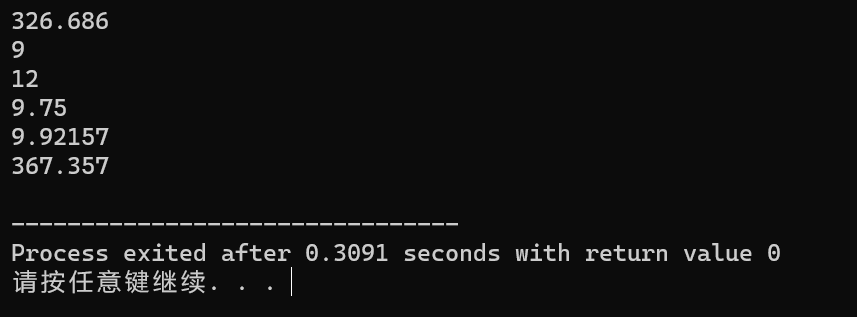
}

cout << sum << endl;

return 0;

}

结果：



5、程序：

#include <iostream>

#include <string>

#include <vector>

using namespace std;

template <class T>

//找两个vector容器中的交集，每个容器中有五个字符串

vector <T> intersection(vector<T>& v1,vector<T>& v2,vector<T> v3){

class vector<T>::iterator it;

class vector<T>::iterator it1;

for(it=v1.begin();it!=v1.end();it++){

for(it1=v2.begin();it1!=v2.end();it1++){

if(\*it==\*it1){

v3.push\_back(\*it1);//如果v2中元素与v1一致就放入v3容器中

};

};

};

for(int j=0;j<v3.size();j++){

cout << v3[j] << "\t";

}

return v3;

}

int main()

{

vector <string> v1,v2,v3;

string str1[5];

string str2[5];

for(int i=0;i<5;i++){

cin >> str1[i];

}

for(int j=0;j<5;j++){

cin >> str2[j];

}//注意string的输入方式：如果写成for(int i=0;i<5;i++){

// cin >> str1[i];

// cin >> str2[i];}则是交替将输入的数据分别存入两个数组中；

for(int j=0;j<5;j++){

v1.push\_back(str1[j]);

v2.push\_back(str2[j]);

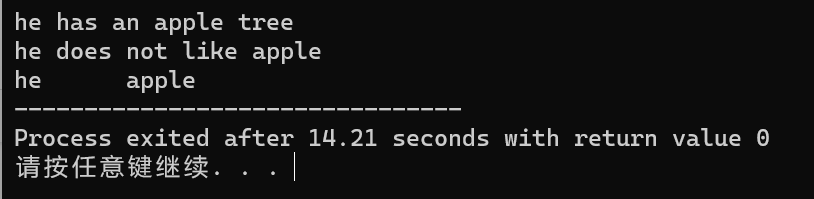
}

intersection(v1,v2,v3);

return 0;

}

结果：



**四、遇到的问题与解决方法**

这次实验遇到的问题集中在第五道实验中，首先是因为对STL的不熟悉，定义模板时使用typename <T>后程序报错 …is not a class or struct。之后将typename改成class后问题解决。然后又遇到了一个我用了很久才解决但实际上是个低级问题的问题，在输入string数组时因为在同一个for循环内同时输入，然后以为是先输入str1再输入str2，但是实际上是交替输入，导致最后的结果始终不对，最终在数个小时的纠错中终于发现了错误，最终程序可以正常运行。

**五、心得与体会**

刚学的知识就拿去做实验非常不熟练，还是要课后多看看书然后自己动手实践，加深理解，做到可以灵活使用，有些细节就要靠自己的积累和细心了，不应该想当然，可以把类似的问题简化后动手实践，验证一下是否正确，可以提高代码的正确率。