#### [1. 类的初步](https://www.educoder.net/tasks/e6s2fzuc3i8f?coursesId=xosaywii)

#### 任务描述

定义一个整数类，其功能包含：

1）无参构造函数，可以不赋值，或者赋固定值，并输出“无参构造函数被调用”；

2）有参构造函数，是实现对整数的初始化，并输出“有参构造函数被调用”；

3）拷贝构造函数，实现用整数类类对象为其初始化，并输出“拷贝构造函数被调用”；

4）析构函数，在析构函数里输出“析构函数被调用”；

5）分别定义成员函数，实现如下功能：给整数数赋值，读取整型数，完数判断、质数判断、回文数判断、计算位数。 编写主函数测试。unfun

**相关知识**

之前学的各种知识的组合。

**编程要求**

输入一个整数t（1,2,3,4,5），代表要执行功能的类型:

1）分别调用无参构造函数，有参构造函数，拷贝构造函数。

2）调用无参构造函数对一个类初始化，并用此类进行完数的判断。输出（Yes or No）

3）调用无参构造函数对一个类初始化，并用此类进行质数的判断。输出（Yes or No）

4）调用无参构造函数对一个类初始化，并用此类进行回文数的判断。输出（Yes or No）

5）调用无参构造函数对一个类初始化，并用此类进行位数的计算。输出一个数字。 无论t为多少，最后都要使用调用析构函数。

**测试说明**

平台会对你编写的代码进行测试，比对你输出的数值与实际正确数值，只有所有数据全部计算正确才能通过测试：

测试输入：1 预期输出： 无参构造函数被调用 有参构造函数被调用 拷贝构造函数被调用 析构函数被调用 析构函数被调用 析构函数被调用

测试输入：3 5(这里的3为t，5为要判断的质数) 预期输出： 无参构造函数被调用 Yes 析构函数被调用

测试输入：5 1234 预期输出： 无参构造函数被调用 4 析构函数被调用

### 第2关：复数

**任务描述**

定义一个复数类Complex，使得下面代码能够工作。

Complex c1(3,5); //用复数3+5i初始化c1

Complex c2(4.5); //用实数4.5初始化c2  
c1.add(c2); //将c1和c2相加保存在c1中  
c1.show(); //将c1输出

**编程要求**

按要求编写程序，输入共一行三个数字，前两个数字分别为c1的实部和虚部，第三个数字为实数c2。 输出一行，为c1与c2相加的结果。 结果保留一位小数。

注意：若虚部为1或-1，则应省略虚部；若实部为0，虚部不为0，应省略实部。 (如：1.0-i，1.0+i，i，-i)

**测试说明**

平台会对你编写的代码进行测试,输出正确的复数。

样例输入：3 5 4.5 样例输出：7.5+5.0i

开始你的任务吧，祝你成功！

#include<iostream>  
#include<cmath>  
#include <iomanip>  
using namespace std;  
  
class Complex

{  
double real; double imag;  
public:

 Complex(double a, double b);  
  Complex(double c);  
  void add(Complex c)

{  
  real += c.real; imag += c.imag;  
  }

void show() {   
  cout << setiosflags(ios::fixed);   
  if (real != 0) {  
  if (imag > 0 && imag != 1) cout << setprecision(1) << real << "+" << imag << "i" << endl;  
  if (imag == 1) cout << setprecision(1) << real << "+" << "i" << endl;  
  if (imag < 0 && imag != -1) cout << setprecision(1) << real<< imag << "i" << endl;  
  if (imag == -1) cout << setprecision(1) << real <<"-"<< "i" << endl;  
  if (imag == 0) cout << setprecision(1) << real << endl;  
  }  
  if (real == 0) {  
  if (imag > 0 && imag != 1) cout << setprecision(1)<< imag << "i" << endl;  
  if (imag == 1) cout << setprecision(1) << "i" << endl;  
  if (imag < 0 && imag != -1) cout << setprecision(1) << imag << "i" << endl;  
  if (imag == -1) cout << setprecision(1)<< "i" << endl;  
  if (imag == 0) cout << "0" << endl;  
  }  
  }  
};

Complex::Complex(double a, double b) { real = a; imag = b; }

Complex::Complex(double c) { real = c; imag = 0; }

int main() {  
double a, b, c; cin >> a >> b >> c;  
Complex c1(a, b);  
Complex c2 = c;  
c1.add(c2);  
c1.show();  
}

### 第3关：时钟

#### 任务描述

设计一个时钟类Clock 数据成员:时(int hour),分(int minute),秒(int second);成员函数：构造函数(Clock(int hour=0, int minute = 0, int second =0))；设置函数(void settime(int h, int m, int s))；显示时间函数(void showtime())，计算本对象时刻与一给定时刻的间隔秒数函数 (int interval(int h, int m, int s)) 形参也可以是clock。编写主函数测试Clock 类。

#### 编程要求

编写程序完成要求，输入两行，第一行表示需要设定的时间，第二行表示需要与类对象对比的时间。输出两行，第一行表示显示的时间（如22:04:51），第二行表示计算出来的间隔秒数（正数）。(两个时刻为同一天内的两个时刻)

#### 测试说明

平台会对你编写的代码进行测试，比对你输出的数值与实际正确数值，只有所有数据全部计算正确才能通过测试：

测试输入： 22 4 51 1 59 59 预期输出： 22:04:51 72292

#### 第3关：时钟

#### 任务描述

设计一个时钟类Clock 数据成员:时(int hour),分(int minute),秒(int second);成员函数：构造函数(Clock(int hour=0, int minute = 0, int second =0))；设置函数(void settime(int h, int m, int s))；显示时间函数(void showtime())，计算本对象时刻与一给定时刻的间隔秒数函数 (int interval(int h, int m, int s)) 形参也可以是clock。编写主函数测试Clock 类。

#### 编程要求

编写程序完成要求，输入两行，第一行表示需要设定的时间，第二行表示需要与类对象对比的时间。输出两行，第一行表示显示的时间（如22:04:51），第二行表示计算出来的间隔秒数（正数）。(两个时刻为同一天内的两个时刻)

#### 测试说明

平台会对你编写的代码进行测试，比对你输出的数值与实际正确数值，只有所有数据全部计算正确才能通过测试：

测试输入： 22 4 51 1 59 59 预期输出： 22:04:51 72292

### 第4关：圆

#### 任务描述

定义一个

circle类 数据成员：半径radius；位置position， position是point类的类对象（函数）；静态变量记录圆的个数。

point类：设置有参构造函数，设置位置函数、显示位置函数。

circle类：有参构造函数，并实现对point有参构造函数传递参数； 计算面积函数；设置半径和位置函数，显示半径和位置函数，静态函数显示圆的个数。

注意：为了安全，哪些函数可以设置为常函数

#### 相关知识

#### 静态：

<https://www.runoob.com/cplusplus/cpp-static-members.html> <https://my.oschina.net/u/1537391/blog/219432>

#### 常函数：

类的成员函数后面加 const，表明这个函数不会对这个类对象的数据成员（准确地说是非静态数据成员）作任何改变。 在设计类的时候，一个原则就是对于不改变数据成员的成员函数都要在后面加 const，而对于改变数据成员的成员函数不能加 const。

#### 编程要求

输入一行3个数，表示设定的圆的半径，位置坐标（x,y)。

你控制的输出共2行，第一行输出圆的半径，位置坐标（x,y)和面积。

第二行输出圆的个数。

有一行输出已帮你写出，无需在意，仅做为一项测试。（样例中不体现）

半径和总面积均保留两位小数。 π近似为3.14 坐标需要用括号括起的形式表示（注意看样例）

#### 测试说明

平台会对你编写的代码进行测试，比对你输出的数值与实际正确数值，只有所有数据全部计算正确才能通过测试：

样例: 测试输入：1 1 1 预期输出： 1.00 (1,1) 3.14 1

#include<cstdio>

using namespace std;

int main(){

    /\* 请勿删除，否则无法通过测试点\*/

    {

       Point p1;/\*如果你的点的类名与此不同，请修改\*/

       circle c1(p1,r);/\*如果你的圆的类名与此不同，请修改\*/

       circle::showCount(); /\*如果你的圆的类名或显示静态成员变量的静态函数与此不同，请修改\*/

     }

    /\*请勿删除，否则无法通过测试点\*/

    return 0;

}