1197: 继承与派生1

Time Limit: 1 Sec  Memory Limit: 128 MB  
Submit: 36  Solved: 27  
[[Submit](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/submitpage.php?id=1197)][[Status](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/problemstatus.php?id=1197)][[Web Board](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/bbs.php?pid=1197)]

Description

请以点类Point为基类派生出一个圆类Circle。圆类Circle的数据成员为r（私有属性，存储圆的半径，圆心的点坐标通过继承点类Point加以实现），成员函数有构造函数Circle、计算圆的面积函数Area、计算圆的周长函数Perimeter和输出函数Display，其中构造函数实现基类和圆类的数据成员的初始化，Display函数实现圆心坐标（利用基类Point的Display实现）、圆的半径、圆的面积（利用Area函数实现）和圆的周长（利用Perimeter函数实现）的输出。请编写圆类的定义及成员函数实现，并在主函数中定义圆类对象，验证各个函数的正确性。

说明：圆周率PI的取值为3.14

已知Point类的定义及main代码如下：（不允许改动）

class Point

{

public:

       Point(double xx,double yy); //constructor

       void Display();        //display point

private:

       double x,y;    //平面的点坐标x，y

};

int main()

{

       double x,y,r;

       cin>>x>>y>>r; //圆心的点坐标及圆的半径

       Circle C(x,y,r);

       C.Display(); //输出圆心点坐标，圆的半径，圆的面积，圆的周长

       return 0;

}

Input

Output

Sample Input

1.5 2.6 1.8

Sample Output

Center:Point(1.5,2.6)

Radius:1.8

Area:10.1736

Perimeter:11.304

#include<iostream>

using namespace std;

const double PI=3.14;

class Point

{

private:

double x,y;

public:

Point(double xx,double yy);

void Display();

};

void Point::Display()

{

cout<<”Center:Point”<<"("<<x<<","<<y<<")"<<endl;

}

Point::Point(double xx,double yy)

{

x=xx;y=yy;

}

class Circle:public Point

{

public:

Circle (double x,double y,double r1=0):Point(x,y)

{

r=r1;

}

double Area();

double Perimeter();

void Display();

private:

double r;

};

double Circle::Area()

{

return r\*r\*PI;

}

double Circle::Perimeter()

{

return 2\*r\*PI;

}

void Circle::Display()

{ Point::Display();

cout<<endl;

cout<<"Radius:"<<r<<endl;

cout<<"Area:"<<Area()<<endl;

cout<<"Perimeter:"<<Perimeter()<<endl;

}

main()

{

double x,y,r;

cin>>x>>y>>r;

Circle C(x,y,r);

C.Display();

return 0;

}

1217: 继承与派生2

Time Limit: 1 Sec  Memory Limit: 128 MB  
Submit: 25  Solved: 21  
[[Submit](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/submitpage.php?id=1217)][[Status](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/problemstatus.php?id=1217)][[Web Board](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/bbs.php?pid=1217)]

Description

1、    Person类派生大学生CollegeStu类(1)。设计一个Person类，其属性包括姓名name和身份证号id，其中name为指针类型，id为整型，编写成员函数：构造函数Person、Display函数（显示数据成员信息）和析构函数；由Person类派生出大学生类CollegeStu，其属性有专业subject（指针类型），C++程序设计课程成绩score（double型），编写构造函数（实现数据初始化）、输出函数Display（包括name，id，subject，score）。main的代码如下：（不允许改动）

int main()

{

       char name[81],subject[81];

       int id;

       double score;

       cin>>name>>id>>subject>>score;

       CollegeStu cs(name,id,subject,score);

       cs.Display();

       return 0;

}

Input

Output

Sample Input

Zhangsan 2 Computer 89.5

Sample Output

Name:Zhangsan

ID:2

Subject:Computer

C++ Score:89.5

#include <iostream>

using namespace std;

#include<cstring>

class Person

{

private:

char \*name;

int id;

public:

Person(char \*name1,int id1);

void Display();

~Person();

};

Person::~Person()

{

delete []name;

}

Person::Person(char \*name1,int id1)

{ int len;

len=strlen(name1);

name=new char[len+1];

strcpy(name,name1);

id=id1;

}

void Person::Display()

{

cout<<"Name:"<<name<<endl<<"ID:"<<id<<endl;

}

class CollegeStu:public Person

{

private:

double score;

char \*subject;

public:

CollegeStu(char \*name1,int id1,char \*subject1,double score1);

void Display();

};

CollegeStu::CollegeStu(char \*name1,int id1,char \*subject1,double score1):Person(name1,id1)

{

score=score1;

subject=new char[strlen(subject1)+1];

strcpy(subject,subject1);

}

void CollegeStu::Display()

{ Person::Display();

cout<<"Subject:"<<subject<<endl<<"C++ Score:"<<score<<endl;

}

int main()

{

char name[81],subject[81];

int id;

double score;

cin>>name>>id>>subject>>score;

CollegeStu cs(name,id,subject,score);

cs.Display();

return 0;

}

1218: 继承与派生3

Time Limit: 1 Sec  Memory Limit: 128 MB  
Submit: 22  Solved: 19  
[[Submit](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/submitpage.php?id=1218)][[Status](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/problemstatus.php?id=1218)][[Web Board](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/bbs.php?pid=1218)]

Description

 Person类派生大学生CollegeStu类(2)。设计一个Person类，其属性包括姓名name和身份证号id，其中name为指针类型，id为整型，编写成员函数：构造函数Person、Display函数（显示数据成员信息）和析构函数；由Person类派生出大学生类CollegeStu，其属性有专业subject（指针类型），C++程序设计课程成绩score（double型），编写构造函数（实现数据初始化）、输出函数Display（只输出subject，score）。main的代码如下：（不允许改动）

int main()

{

char name[81],subject[81];

int id;

double score;

cin>>name>>id>>subject>>score; //输入学生的姓名、id号、专业、成绩

CollegeStu cs(name,id,subject,score);

cs.Person::Display(); //输出姓名，id

cs.Display(); //输出专业、成绩

return 0;

}

Input

Output

Sample Input

Lixu 5 Software 87.5

Sample Output

Name:Lixu

ID:5

Subject:Software

C++ Score:87.5

#include <iostream>

#include<cstring>

using namespace std;

class Person

{

private:

char \*name;

int id;

public:

Person(char \*name1,int id1);

void Display();

~Person();

};

Person::Person(char \*name1,int id1)

{id=id1;

name=new char[strlen(name1)+1];

strcpy(name,name1);

}

void Person::Display()

{

cout<<"Name:"<<name<<endl<<"ID:"<<id<<endl;

}

Person::~Person()

{

delete []name;

}

class CollegeStu:public Person

{

private:

char \*subject;

double score;

public:

CollegeStu(char \*name1,int id1,char \*subject1,double score1);

void Display();

~CollegeStu();

};

CollegeStu::CollegeStu(char \*name1,int id1,char \*subject1,double score1):Person(name1,id1)

{

subject=new char[strlen(subject1)+1];

strcpy(subject,subject1);

score=score1;

}

void CollegeStu::Display()

{

cout<<"Subject:"<<subject<<endl<<"C++ Score:"<<score<<endl;

}

CollegeStu::~CollegeStu()

{

delete []subject;

}

int main()

{

char name[81],subject[81];

int id;

double score;

cin>>name>>id>>subject>>score;

CollegeStu cs(name,id,subject,score);

cs.Person::Display();

cs.Display();

return 0;

}

1219: 继承与派生4

Time Limit: 1 Sec  Memory Limit: 128 MB  
Submit: 23  Solved: 22  
[[Submit](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/submitpage.php?id=1219)][[Status](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/problemstatus.php?id=1219)][[Web Board](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/bbs.php?pid=1219)]

Description

已知Base为基类，派生出Derived类，两个类的定义及main的代码如下（不允许改动），请完成Base类和Derived类的构造函数和析构函数，能够根据输入获取相应的输出。

class Base

{

private:

int b;

public:

Base(int);

~Base();

};

class Derived:public Base

{

private:

int d;

public:

Derived(int,int);

~Derived();

};

int main()

{

int a,b;

cin>>a>>b;

Derived dr(a,b);

return 0;

}

Input

Output

Sample Input

1 3

Sample Output

Base 1 says hello

Derived 3 says hi

Derived 3 says bye

Base 1 says goodbye

#include <iostream>

using namespace std;

#include <cstring>

class Base

{

private:

int b;

public:

Base(int );

~Base();

};

Base::Base(int b1)

{ b=b1;

cout<<"Base "<<b<<" says hello"<<endl;

}

Base::~Base()

{

cout<<"Base "<<b<<" says goodbye"<<endl;

}

class Derived:public Base

{

private:

int d;

public:

Derived(int ,int);

~Derived();

};

Derived::Derived(int b1,int d1):Base(b1)

{

d=d1;

cout<<"Derived "<<d<<" says hi"<<endl;

}

Derived::~Derived()

{

cout<<"Derived "<<d<<" says bye"<<endl;

}

int main()

{

int a,b;

cin>>a>>b;

Derived dr(a,b);

return 0;

}

1220: 继承与派生5

Time Limit: 1 Sec  Memory Limit: 128 MB  
Submit: 38  Solved: 13  
[[Submit](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/submitpage.php?id=1220)][[Status](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/problemstatus.php?id=1220)][[Web Board](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/bbs.php?pid=1220)]

Description

由Array类派生出有序数组SortArray类，SortArray类中实现有序数组的插入。

已知Array类的定义如下（不允许增加成员函数）：

class Array

{

public:

       Array();    //构造函数，初始化为空数组（length置为0）

       int Length();   //获取数组的实际长度

       double Get(int pos);     //获取data中下标为pos的元素的值

       void Insert(int pos, double x); //在下标pos处插入x

       void Display();       //输出线性表

private:

       double data[MaxSize];  //存储元素(MaxSize为常量)

       int length;              //数组的实际长度

};

SortArray类定义如下（不允许增加成员函数）：

class SortArray:private Array

{

public:

       SortArray();

       int Length();   //获取数组的实际长度

       double Get(int pos);     //获取data中下标为pos的元素的值

       void Display();       //输出线性表

       void Insert(double x); //递增有序数组中插入x，使序列仍有序

};

请实现Array类和SortArray类的成员函数， main中输入若干个实数，以0结束，利用SortArray类中的Insert函数将它们插入data中，得到有序序列，再利用Display函数输出有序序列。代码如下（不允许修改）：

int main()

{

       SortArray sa;

       double num;

       while(1)

       {

              cin>>num;

              if(fabs(num)<=1e-6) break;

              try

              {

                     sa.Insert(num); //

              }

              catch(char\* message)

              {

                     cout <<message<<endl;    //如失败提示失败信息

              }

       }

       sa.Display();

    return 0;

}

Input

Output

Sample Input

2.5 6.7 8.3 2.8 6.53 6.82 7.33 0

Sample Output

The length:7

The elements:2.5 2.8 6.53 6.7 6.82 7.33 8.3

#include <iostream>

#include <cmath>

using namespace std;

const int MaxSize=100;

class Array

{

public:

Array();

int Length();

double Get(int pos);

void Insert(int pos,double x);

void Display();

private:

double data[MaxSize];

int length;

};

Array::Array()

{

length=0;

}

int Array::Length()

{

return length;

}

double Array::Get(int pos)

{ if(pos>=0&&pos<length)

return data[pos];

else throw "illegal";

}

void Array::Insert(int pos,double x)

{ int i;

for(i=length;i>pos;i--)

data[i]=data[i-1];

data[pos]=x;

length++;

}

void Array::Display()

{ int i;

cout<<"The length:"<<length<<endl<<"The elements:";

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

cout<<data[i]<<" ";

cout<<endl;

}

class SortArray:private Array

{

public:

SortArray();

int Length();

double Get(int pos);

void Display();

void Insert(double x);

};

SortArray::SortArray()

{

Array();

}

int SortArray::Length()

{

return Array::Length();

}

void SortArray::Insert(double x)

{

int i;

for(i=0;i<Length()&&Get(i)<=x;i++);

Array::Insert(i,x);

}

double SortArray::Get(int pos)

{

return Array::Get(pos);

}

void SortArray::Display()

{

Array::Display();

}

int main()

{

SortArray sa;

double num;

while(1)

{

cin>>num;

if(num==0) break;//fabs(num)<=1e-6

try{

sa.Insert(num);

}

catch(char \*message)

{

cout<<message<<endl;

}

}

sa.Display();

return 0;

}

1221: 继承与派生6

Time Limit: 1 Sec  Memory Limit: 128 MB  
Submit: 7  Solved: 5  
[[Submit](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/submitpage.php?id=1221)][[Status](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/problemstatus.php?id=1221)][[Web Board](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/bbs.php?pid=1221)]

Description

已知Array类的定义如下（不允许增加成员函数）：

class Array

{

public:

     Array(int size);  //构造函数，初始化数据成员（为data分配内存，MaxSize置为size，length置为0）

       int Length();   //获取顺序表实际长度

       double Get(int pos); //获取下标为pos的元素的值

       void Insert(int pos, double x); //在下标pos处插入x

       void Display();//输出线性表

private:

       double \*data; //存储元素

    int MaxSize;

       int length;   //数组的实际长度

};

SortArray类定义如下（不允许增加其它成员函数）：

class SortArray:private Array

{

public:

    SortArray(int size);

    int Length();   //获取顺序表实际长度

    double Get(int pos);  //获取下标为pos的元素的值

    void Display();   //输出线性表

    void Insert(double x); //递增有序数组中插入x，使序列仍有序

};

main中的代码如下（不允许改动）：

int main()

{

       int size;

       cin>>size;

       SortArray sa(size);

       double num;

       while(1)

       {

        cin>>num;

        if(fabs(num)<=1e-6) break;

        try

        {

         sa.Insert(num);

         }

        catch(char\* wrong)

        {

        cout <<wrong<<endl; //如失败提示失败信息

              }

       }

       sa.Display();

    return 0;

}

请实现Array类和SortArray类的成员函数。

Input

Output

Sample Input

20 2.5 6.7 8.3 2.8 6.53 6.82 7.33 0

Sample Output

The length:7

The elements:2.5 2.8 6.53 6.7 6.82 7.33 8.3

#include <iostream>

#include <cmath>

using namespace std;

class Array

{

public:

Array(int size);

int Length();

double Get(int pos);

void Insert(int pos,double x);

void Display();

private:

double \*data;

int MaxSize;

int length;

};

Array::Array(int size )

{

MaxSize=size;

length=0;

data=new double[MaxSize];

}

int Array::Length()

{

return length;

}

double Array::Get(int pos)

{

return data[pos];

}

void Array::Insert(int pos,double x)

{

int i;

if(length==MaxSize) throw"wrong";

else

{for(i=length;i>pos;i--)

data[i]=data[i-1];

data[pos]=x;

length++;}

}

void Array::Display()

{ int i;

cout<<"The length:"<<length<<endl<<"The elements:";

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

cout<<data[i]<<" ";

cout<<endl;

}

class SortArray:private Array

{

public:

SortArray(int size);

int Length();

double Get(int pos);

void Display();

void Insert(double );

};

SortArray::SortArray(int size):Array::Array(size)

{

}

int SortArray::Length()

{

return Array::Length();

}

double SortArray::Get(int pos)

{

return Array::Get(pos);

}

void SortArray::Display()

{

Array::Display();

}

void SortArray::Insert(double x)

{ int i;

if(Length()==0) Array::Insert(0,x);

else for(i=0;i<Length();i++)

{if(x<=Get(i)) {Array::Insert(i,x);break;}

else if(x>Get(Length()-1))

{Array::Insert(Length(),x);break;}

}

}

int main()

{

int size;

cin>>size;

SortArray sa(size);

double num;

while(1)

{

cin>>num;

if(fabs(num)<=1e-6) break;

try{sa.Insert(num);

}

catch(char\* wrong){

cout<<wrong<<endl;

}

}

sa.Display();

return 0;

}

1223: 继承与派生7

Time Limit: 1 Sec  Memory Limit: 128 MB  
Submit: 8  Solved: 8  
[[Submit](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/submitpage.php?id=1223)][[Status](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/problemstatus.php?id=1223)][[Web Board](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/bbs.php?pid=1223)]

Description

已知由Automobille类派生出Car类和Wagon类，而后两者共同派生出StationWagon类，各类的定义及main中的代码（不允许改动）如下，请实现各个类的成员函数，完成相应的输出：

class Automobile //汽车类

{

private:

int power; //马力

public:

Automobile(int p);

void Display();

};

class Car:virtual public Automobile //小客车类

{

private:

int seat; //座位

public:

Car(int p,int s);

void Display();

};

class Wagon:virtual  public Automobile //小货车类  
{  
private:  
   int load;     //装载量  
public:  
    Wagon(int p,int l);  
 void Display();  
};

class StationWagon :public Car, public Wagon //客货两用车类

{

public:

StationWagon(int p, int s,int l);

void Display();

};

int main()

{

int power,load,seat;

cin>>power>>seat>>load;

StationWagon sw(power,seat,load);

sw.Display();

return 0;

}

Input

Output

Sample Input

108 3 10

Sample Output

StationWagon:

Power:108

Seat:3

Load:10

#include <iostream>

using namespace std;

class Automobile

{

private:

int power;

public:

Automobile(int p);

void Display();

};

Automobile::Automobile(int p)

{

power=p;

}

void Automobile::Display()

{

cout<<"Power:"<<power<<endl;

}

class Wagon:virtual public Automobile

{

private:

int load;

public:

Wagon(int p,int l);

void Display();

};

Wagon::Wagon(int p,int l):Automobile(p)

{

load=l;

}

void Wagon::Display()

{

cout<<"Load:"<<load<<endl;

}

class Car:virtual public Automobile

{

private:

int seat;

public:

Car(int p,int s);

void Display();

};

Car::Car(int p,int s):Automobile(p)

{

seat=s;

}

void Car::Display()

{

cout<<"Seat:"<<seat<<endl;

}

class StationWagon:public Car,public Wagon

{

public:

StationWagon(int p,int s,int l):Car(p,s),Wagon(p,l),Automobile(p)

{

cout<<"StationWagon:"<<endl;

}

void Display();

};

void StationWagon::Display()

{

Automobile::Display();

Car::Display();

Wagon::Display();

}

int main()

{

int power,load,seat;

cin>>power>>seat>>load;

StationWagon sw(power,seat,load);

sw.Display();

return 0;

}

1224: 多态-虚函数问题

Time Limit: 1 Sec  Memory Limit: 128 MB  
Submit: 22  Solved: 18  
[[Submit](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/submitpage.php?id=1224)][[Status](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/problemstatus.php?id=1224)][[Web Board](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/bbs.php?pid=1224)]

Description

请以点类Point为基类派生出一个圆类Circle。Point类的数据成员为x、y（私有属性，存储点的横纵坐标），成员函数有构造函数Point（实现对数据成员x、y的初始化）、输出函数Display（输出点坐标）；圆类Circle的数据成员为r（私有属性，存储圆的半径，圆心的点坐标通过继承点类Point加以实现），成员函数有构造函数Circle、计算圆的面积函数Area、计算圆的周长函数Perimeter和输出函数Display，其中构造函数实现基类和圆类的数据成员的初始化，Display函数实现圆心坐标、圆的半径、圆的面积（利用Area函数实现）和圆的周长（利用Perimeter函数实现）的输出。实现Point类和Circle类的定义及成员函数。主函数的代码（不允许改动）如下：

说明：圆周率PI的取值为3.14

提示：Display应设计为虚函数

Input

Output

Sample Input

0 0 1

Sample Output

Point(0,0)

Circle's center:Point(0,0)

Radius:1

Area:3.14

Perimeter:6.28

#include <iostream>

const double PI=3.14;

using namespace std;

class Point

{

private:

double x,y;

public:

Point(double x=0,double y=0);

virtual void Display();

};

Point::Point(double x1,double y1)

{

x=x1;y=y1;

}

void Point::Display()

{

cout<<"Point("<<x<<","<<y<<")"<<endl;

}

class Circle:public Point

{

private:

double r;

public:

Circle(double x1,double y1,double r1):Point(x1,y1)

{

r=r1;

}

virtual void Display();

double Area();

double Perimeter();

};

double Circle::Area()

{ cout<<"Area:";

return PI\*r\*r;

}

double Circle::Perimeter()

{ cout<<"Perimeter:";

return PI\*r\*2;

}

void Circle::Display()

{

cout<<"Circle's center:";

Point::Display();

cout<<"Radius:"<<r<<endl;

cout<<Area()<<endl;

cout<<Perimeter()<<endl;

}

int main()

{

double x,y,r;

cin>>x>>y>>r; //圆心的点坐标及圆的半径

Point \*p;

p=new Point(x,y);

p->Display();

p=new Circle(x,y,r);

p->Display();

return 0;

}

1229: 多态-抽象类1

Time Limit: 1 Sec  Memory Limit: 128 MB  
Submit: 7  Solved: 7  
[[Submit](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/submitpage.php?id=1229)][[Status](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/problemstatus.php?id=1229)][[Web Board](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/bbs.php?pid=1229)]

Description

定义一个抽象类Shape，应用抽象类Shape派生圆类circle、圆的内接正方形类In\_Square和圆的外切正方形类Ex\_Square。main中利用指向抽象类的指针实现求圆、圆的内接正方形和圆的外切正方形的面积和周长。请实现各个类的成员函数，已知各类的定义及main中的代码如下（不允许修改）：

class Shape

{

public:

       Shape();

       virtual double Area()=0;

       virtual double Perimeter() = 0;

};

class Circle:public Shape

{

public:

       Circle(double rr);

       double Radius();

       double Area();

       double Perimeter();

private:

       double r; //圆的半径

};

class In\_Square:public Circle

{

public:

       In\_Square(double rr);

       double Area();

       double Perimeter();

};

class Ex\_Square:public Circle

{

public:

       Ex\_Square(double x);

       double Area();

       double Perimeter();

};

int main()

{

      Shape \*ptr;

       double r;

       cin>>r;

       ptr=new Circle(r);

       cout<<"Circle's area:"<<ptr->Area()<<endl;

       cout<<"Circle's perimeter:"<<ptr->Perimeter()<<endl;

       ptr=new In\_Square(r);

    cout<<"In\_Square's area:"<<ptr->Area()<<endl;

       cout<<"In\_Square's perimeter:"<<ptr->Perimeter()<<endl;

       ptr=new Ex\_Square(r);

    cout<<"Ex\_Square's area:"<<ptr->Area()<<endl;

       cout<<"Ex\_Square's perimeter:"<<ptr->Perimeter()<<endl;

       return 0;

}

圆周率PI值取3.14。

Input

Output

Sample Input

1.5

Sample Output

Circle's area:7.065

Circle's perimeter:9.42

In\_Square's area:4.5

In\_Square's perimeter:8.48528

Ex\_Square's area:9

Ex\_Square's perimeter:12

#include<iostream>

#include<cmath>

using namespace std;

const double PI=3.14;

class Shape

{

public:

virtual double Area()=0;

virtual double Perimeter()=0;

};

class Circle:public Shape

{

public:

Circle(double rr);

double Radius();

double Area();

double Perimeter();

private:

double r; //圆的半径

};

Circle::Circle(double rr)

{

r=rr;

}

double Circle::Area()

{

return r\*r\*PI;

}

double Circle::Perimeter()

{

return 2\*PI\*r;

}

double Circle::Radius()

{

return r;

}

class In\_Square:public Circle

{

public:

In\_Square(double rr):Circle(rr)

{

}

double Area();

double Perimeter();

};

double In\_Square::Area()

{

return 2\*Radius()\*Radius();

}

double In\_Square::Perimeter()

{

return 4\*sqrt(2)\*Radius();

}

class Ex\_Square:public Circle

{

public:

Ex\_Square(double xx):Circle(xx)

{

}

double Area();

double Perimeter();

};

double Ex\_Square::Area()

{

return 4\*Radius()\*Radius();

}

double Ex\_Square::Perimeter()

{

return 8\*Radius();

}

int main()

{

Shape \*ptr;

double r;

cin>>r;

ptr=new Circle(r);

cout<<"Circle's area:"<<ptr->Area()<<endl;

cout<<"Circle's perimeter:"<<ptr->Perimeter()<<endl;

ptr=new In\_Square(r);

cout<<"In\_Square's area:"<<ptr->Area()<<endl;

cout<<"In\_Square's perimeter:"<<ptr->Perimeter()<<endl;

ptr=new Ex\_Square(r);

cout<<"Ex\_Square's area:"<<ptr->Area()<<endl;

cout<<"Ex\_Square's perimeter:"<<ptr->Perimeter()<<endl;

return 0;

}

1237: 多态-抽象类2

Time Limit: 1 Sec  Memory Limit: 128 MB  
Submit: 9  Solved: 8  
[[Submit](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/submitpage.php?id=1237)][[Status](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/problemstatus.php?id=1237)][[Web Board](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/bbs.php?pid=1237)]

Description

定义一个抽象类Shape，应用抽象类Shape派生圆类Circle和长方形类Rectangle。请完成各个类的定义及成员函数实现。已知Shape类的定义及main中的代码如下（不允许修改）：

class Shape

{

public:

Shape();

virtual double Area()=0;

virtual double Perimeter() = 0;

};

int main()

{

double r,a,b;

char \*s[2]={"Circle:\n","Rectangle:\n"};

cin>>r>>a>>b;

Shape \*ptr[]={new Circle(r),new Rectangle(a,b)};

for(int i=0;i<2;i++)

{

cout<<s[i];

cout<<"Area:"<<ptr[i]->Area()<<endl;

cout<<"Perimeter:"<<ptr[i]->Perimeter()<<endl;

}

return 0;

}

Input

Output

Sample Input

1 2 3

Sample Output

Circle:

Area:3.14

Perimeter:6.28

Rectangle:

Area:6

Perimeter:10

#include<iostream>

#include<cmath>

using namespace std;

const double PI=3.14;

class Shape

{

public:

Shape();

virtual double Area()=0;

virtual double Perimeter() = 0;

};

Shape::Shape()

{

}

class Circle:public Shape

{

public:

Circle(double rr=0);

double Area();

double Perimeter();

private:

double r; //圆的半径

};

Circle::Circle(double rr)

{

r=rr;

}

double Circle::Area()

{

return r\*r\*PI;

}

double Circle::Perimeter()

{

return 2\*PI\*r;

}

class Rectangle:public Shape

{

private:

double x,y;

public:

Rectangle(double x1=0,double y1=0);

double Area();

double Perimeter();

};

Rectangle::Rectangle(double x1,double y1)

{

x=x1;

y=y1;

}

double Rectangle::Area()

{

return x\*y;

}

double Rectangle::Perimeter()

{

return 2\*(x+y);

}

int main()

{

double r,a,b;

char \*s[2]={"Circle:\n","Rectangle:\n"};

cin>>r>>a>>b;

Shape \*ptr[]={new Circle(r),new Rectangle(a,b)};

for(int i=0;i<2;i++)

{

cout<<s[i];

cout<<"Area:"<<ptr[i]->Area()<<endl;

cout<<"Perimeter:"<<ptr[i]->Perimeter()<<endl;

}

return 0;

}

1238: 多态-运算符重载1

Time Limit: 1 Sec  Memory Limit: 128 MB  
Submit: 21  Solved: 10  
[[Submit](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/submitpage.php?id=1238)][[Status](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/problemstatus.php?id=1238)][[Web Board](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/bbs.php?pid=1238)]

Description

实现复数的<< ，>>， +， -的运算符重载

已知Complex类的定义（不允许添加）及main中的代码（不允许改动）如下：

class Complex

{

public:

       Complex(double r=0,double i=0);

       Complex operator +(const Complex &c);

       Complex operator -(const Complex &c);

       friend ostream &operator<<(ostream &os,Complex &com);

       friend istream &operator>>(istream &is,Complex &com);

       void Display();

private:

       double real;

       double imag;

};

int main()

{

       Complex c1,c2,c3;

       cin>>c1>>c2;

       cout<<"("<<c1<<")+"<<"("<<c2<<")=";

       c3=c1+c2;

       cout<<c3<<endl;

       cout<<"("<<c1<<")-"<<"("<<c2<<")=";

       c3=c1-c2;

       cout<<c3<<endl;

       return 0;

}

Input

Output

Sample Input

3 2 6 -3

Sample Output

(3+2i)+(6-3i)=9-1i

(3+2i)-(6-3i)=-3+5i

#include<iostream>

#include<cmath>

using namespace std;

class Complex

{

public:

Complex(double r=0,double i=0);

Complex operator +(const Complex &c);

Complex operator -(const Complex &c);

friend ostream &operator<<(ostream &os,Complex &com);

friend istream &operator>>(istream &is,Complex &com);

void Display();

private:

double real;

double imag;

};

Complex::Complex(double r,double i)

{

real=r;

imag=i;

}

Complex Complex::operator +(const Complex &c)

{

return Complex(real+c.real,imag+c.imag);

}

Complex Complex:: operator -(const Complex &c)

{

return Complex(real-c.real,imag-c.imag);

}

ostream &operator<<(ostream &os,Complex &com)

{ if(com.real!=0)

{if(com.imag>0)

os<<com.real<<"+"<<com.imag<<"i";

else if(com.imag==0)

os<<com.real;

else os<<com.real<<com.imag<<"i";}

else

{if(com.imag>0)

os<<"+"<<com.imag<<"i";

else if(com.imag==0)

os<<"0";

else os<<com.imag<<"i";}

return os;

}

istream &operator>>(istream &is,Complex &com)

{

is>>com.real>>com.imag;

return is;

}

void Complex::Display()

{

}

int main()

{

Complex c1,c2,c3;

cin>>c1>>c2;

cout<<"("<<c1<<")+"<<"("<<c2<<")=";

c3=c1+c2;

cout<<c3<<endl;

cout<<"("<<c1<<")-"<<"("<<c2<<")=";

c3=c1-c2;

cout<<c3<<endl;

return 0;

}

1255: 多态-运算符重载2

Time Limit: 1 Sec  Memory Limit: 128 MB  
Submit: 18  Solved: 5  
[[Submit](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/submitpage.php?id=1255)][[Status](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/problemstatus.php?id=1255)][[Web Board](http://221.203.21.203:8001/rwt/USTL/http/GV6T6N3UFZ5UGLSSGM3B/OnlineJudge/bbs.php?pid=1255)]

Description

实现Array类的>>、 <<的运算符重载。已知Array类的定义（不允许增加其它成员函数）及main中的代码如下（不允许修改）：

class Array{

public:

Array(); //初始化length=0

void Insert(int i, double x); //在下标i处插入x

friend ostream &operator<<(ostream &out,Array &arr); //<<运算符重载

friend istream &operator>>(istream &in,Array &arr); //>>运算符重载

private:

double data[MaxSize]; //存储元素，MaxSize为常量，需先定义

int length; //顺序表实际长度

};

int main(){

Array a;

cin>>a;

cout<<a;

return 0;

}

请完成Array类中的成员函数，使之能够通过运算符重载实现数组中数据的输入和输出。数据的输入格式为：第一个数是数组中的数据个数，之后为数组中的数据值。

Input

Output

Sample Input

5 2.3 5.6 7.98 6.5 6.32

Sample Output

Length:5

Data:2.3 5.6 7.98 6.5 6.32

#include<iostream>

const int MaxSize=1000;

using namespace std;

class Array{

public:

Array();

void Insert(int i, double x);

friend ostream &operator<<(ostream &out,Array &arr);

friend istream &operator>>(istream &in,Array &arr);

private:

double data[MaxSize];

int length;

};

Array::Array()

{

length=0;

}

void Array::Insert(int i,double x)

{ int z;

for(z=length;z>i;z--)

data[z]=data[z-1];

data[i]=x;

}

ostream &operator<<(ostream &out,Array &arr)

{ int i;

out<<"Length:"<<arr.length<<endl<<"Data:";

for(i=0;i<arr.length-1;i++)

out<<arr.data[i]<<" ";

out<<arr.data[i];

return out;

}

istream &operator>>(istream &in,Array &arr)

{ int i=0;

double x;

in>>arr.length;

for(i=0;i<arr.length;i++)

{ in>>x;

arr.Insert(i,x);

}

return in;

}

int main()

{

Array a;

cin>>a;

cout<<a;

return 0;

}