1

1：阅读程序并写出输出结果

```
public class ConstructorDemo {
    public static void main(String[] args) {
        new Faculty("hello");
    }
}
class Person {
    public Person() {
        System.out.println("Call Person's constructor
without parameter");
    }
}
class Employee extends Person {
    public Employee() {
        this("Call Employee's overloaded
constructor");
        System.out.println("Call Employee's
constructor without parameter");
    }
    public Employee(String s) {
        System.out.println(s);
        System.out.println("Call Employee's
constructor with parameter: " + s);
    }
}
class Faculty extends Employee {
    public Faculty() {
        System.out.println("Call Faculty's
constructor without parameter");
    }
    public Faculty(String s) {
        System.out.println("Call Faculty's
constructor with parameter: " + s);
    }
}
```

> Call Person's constructor without parameter
> Call Employee's overloaded constructor
> Call Employee's constructor with parameter: Call Employee's overloaded constructor
> Call Employee's constructor without parameter
> Call Faculty's constructor with parameter: hello

2

```
public class HideDemo {
    @SuppressWarnings("static-access")
    public static void main(String[] args) {
        A x = new B();
        System.out.println("(1)x.i is " + x.i);
        System.out.println("(2)x.j is " + x.j);
        System.out.println("(3)x.m1() is " + x.m1());
        System.out.println("(4)x.m2() is " + x.m2());
        System.out.println("(5)x.m3() is " + x.m3());
        B y = (B)x;
        System.out.println("(1)y.i is " + y.i);
        System.out.println("(2)y.j is " + y.j);
        System.out.println("(3)y.m1() is " + y.m1());
        System.out.println("(4)y.m2() is " + y.m2());
        System.out.println("(5)y.m3() is " + y.m3());
    }
}
class A {
    public int i = 1;
    public static int j = 11;
    public static String m1() {
        return "A's static m1";
    }
    public String m2() {
        return "A's instance m2";
    }
    public String m3() {
        return "A's instance m3";
    }
}
class B extends A {
    public int i = 2;
    public static int j = 22;
    public static String m1() {
        return "B's static m1";
    }
    public String m2() {
        return "B's instance m2";
    }
}
```

> (1)x.i is 1
> (2)x.j is 11
> (3)x.m1() is A's static m1
> (4)x.m2() is B's instance m2
> (5)x.m3() is A's instance m3
> (1)y.i is 2
> (2)y.j is 22
> (3)y.m1() is B's static m1
> (4)y.m2() is B's instance m2
> (5)y.m3() is A's instance m3

3

```
package hw3;
public class Circle extends GeometricObject
implements GeometricObjectArea{
    private double radius;
    public double getRadius() {
        return radius;
    }

    public void setRadius(double radius) {
        this.radius = radius;
    }

    @Override
    public double getArea() {
        return Math.PI * radius * radius;
    }
}

package hw3;
public interface GeometricObjectArea{
    double getArea();
}

package hw3;
public class GeometricObject {
    private String color;

    public String getColor() {
        return color;
    }

    public void setColor(String color) {
        this.color = color;
    }
}


package hw3;
public class GeometricObjectsManager {
    static double totalArea(GeometricObjectArea[]
geometricObjectAreas){
        double res = 0;
        for(GeometricObjectArea g :
geometricObjectAreas){
            res += g.getArea();
        }
        return res;
    }
}

package hw3;
public class Rectangle implements
GeometricObjectArea{
    private double width;
    private double length;

    public double getWidth() {
        return width;
    }

    public void setWidth(double width) {
        this.width = width;
    }

    public void setLength(double length) {
        this.length = length;
    }

    public double getLength() {
        return length;
    }

    @Override
    public double getArea() {
        return  width * length;
    }
}

package hw3;
public class Test {
    public static void main(String[] args) {
        GeometricObjectArea[] obs = new
GeometricObjectArea[20];
        for (int i = 0; i < 20; i++) {
            if(i < 10){
                obs[i] = new Circle();
                ((Circle)obs[i]).setRadius(1);
            }else{
                obs[i] = new Rectangle();
                ((Rectangle)obs[i]).setLength(1);
                ((Rectangle)obs[i]).setWidth(1);
            }
        }


System.out.println(GeometricObjectsManager.totalArea(
obs));
    }
}
```