CS 61B Spring 2018

Scope, Pass-by-Value, Static

Exam Prep 1: January 22, 2018

difficulty:
static/class attribute
instance attribute

1 Static Shock

Write what the `main` method will print out once it is executed. It might be helpful to draw box and pointer diagrams to keep track of variables.

```
public class Shock {
    public static int bang;
    public static Shock baby;
    public Shock() {
        this.bang = 100;
    }
    public Shock (int num) {
        this.bang = num;
        baby = starter();
        this.bang += num;
    }
    public static Shock starter() {
        Shock gear = new Shock();
        return gear;
    }
    public static void shrink(Shock statik) {
        statik.bang -= 1;
    }
    public static void main(String[] args) {
        Shock gear = new Shock(200);
300     System.out.println(gear.bang);        __400__300
299     shrink(gear);
100->99 shrink(starter());
        System.out.println(gear.bang);        __398__99
    }
}
```

the trap is in starter()!
starter() calls the default constructor, which will set Shock.bang back to 100

2 Horse-o-Scope

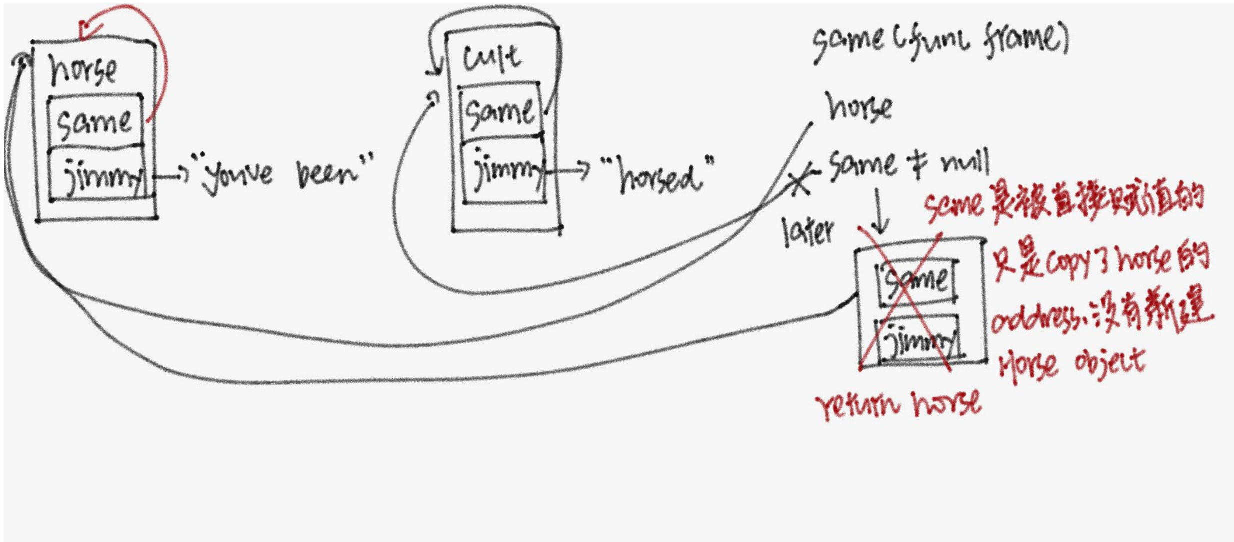
Given the following program, draw out the box and pointer diagram that results from executing the `inputArray` method. What is the output printed by the program? (Summer '16, MT1)

```java
public class Horse {
    Horse same;
    String jimmy;

    public Horse(String lee) {
        jimmy = lee;
    }

    public Horse same(Horse horse) {
        if (same != null) {
            Horse same = horse;
            same.same = horse;
            same = horse.same;
        }
        return same.same;
    }

    public static void main(String[] args) {
        Horse horse = new Horse("youve been");
        Horse cult = new Horse("horsed");
        cult.same = cult;
        cult = cult.same(horse);
        System.out.println(cult.jimmy);
        System.out.println(horse.jimmy);
    }
}
```

the key or trap of this question is: class attribute vs. instance attribute
class attribute is distinguished by "static"
otherwise the attribute varies by instance; positions do not matter

Program Output:

```
horsed
horsedyouve been
```



same 是被直接賦值的 只是copy了horse的 address，沒有新建 Horse object

3 Give em the 'Ol Switcheroo

For each function call in the `main` method, write out the x and y values of both foobar and baz after executing that line. (Spring '15, MT1)

```
public class Foo {
    public int x, y;

    public Foo (int x, int y) {
        this.x = x;
        this.y = y;
    }

    public static void switcheroo (Foo a, Foo b) {
        Foo temp = a;
        a = b;
        b = temp;
    }

    public static void fliperoo (Foo a, Foo b) {
        Foo temp = new Foo(a.x, a.y);
        a.x = b.x;
        a.y = b.y;
        b.x = temp.x;
        b.y = temp.y;
    }

    public static void swaperoo (Foo a, Foo b) {
        Foo temp = a;
        a.x = b.x;
        a.y = b.y;
        b.x = temp.x;
        b.y = temp.y;
    }

    public static void main (String[] args) {
        Foo foobar = new Foo(10, 20);
        Foo baz = new Foo(30, 40);
        switcheroo(foobar, baz);    foobar.x: 10 foobar.y: 20 baz.x: 30 baz.y: 40
        fliperoo(foobar, baz);      foobar.x: 30 foobar.y: 40 baz.x: 10 baz.y: 20
        swaperoo(foobar, baz);      foobar.x: 10 foobar.y: 20 baz.x: 10 baz.y: 20
    }
}
```