Lab 03 Assignment

班级：202111
学号：19241027
姓名：胡峻诚

Question1

输出如下：

```
initialize A1
initialize A2
initialize A3
initialize A4
initialize A5
initialize A6
copy from A6
initialize B1
initialize A8
main begins
initialize A9
initialize A6
copy from A6
initialize B2
initialize A8
main ends
```

Question2

这段代码能够证明“在属性定义处初始化的属性，比在方法中初始化的属性先被初始化”吗？

- 可以，从

```
public B(int i) {
    System.out.println("initialize B"+i);
    a8 = new A(8);
}
A a7 = new A(a6);
```

的输出是

```
copy from A6
initialize B1
initialize A8
```

可以看出，在先初始化了定义的属性 a7 后，才初始化的方法中的属性 a8。

这段代码能够证明“在属性定义处初始化的属性，初始化顺序等同于他们在类定义中出现的顺序”吗?

- 相同类型的可以，从

```
static A a1 = new A(1);
static A a2 = new A(2);
```

的输出是

```
initialize A1
initialize A2
```

可以看出，是按照顺序，先定义 a1，再定义 a2。

Question3

顺序如下：

- 类属性（静态变量）定义时的初始化
- static 块中的初始化代码
- 对象属性（非静态变量）定义时的初始化
- 构造方法（函数）中的初始化代码

Question4

仅通过所提供的代码不能证明。

将第 36 行 `static B b1 = new B(1);` 此行代码注释掉，如下所示：

```
class A {
    int value;
    static A a1 = new A(1);
    public A(int i) {
        System.out.println("initialize A"+i);
        value = i;
    }

    public A(A a) {
```

```
10          System.out.println("copy from A"+a.value);
11          value = a.value;
12      }
13      static A a2 = new A(2);
14
15  }
16
17  class B {
18      A a8;
19      // A a7 = new A(a6);
20      A a6 = new A(6);
21      static A a3 = new A(3);
22      static A a4;
23      static {
24          a4 = new A(4);
25      }
26      static A a5 = new A(5);
27
28      public B(int i) {
29          System.out.println("initialize B"+i);
30          a8 = new A(8);
31      }
32      A a7 = new A(a6);
33  }
34
35  public class Initialization {
36  //    static B b1 = new B(1); // 将此行注释掉
37      static B b2;
38      public static void main(String[] args) {
39          System.out.println("main begins");
40          A a9 = new A(9);
41          b2 = new B(2);
42          System.out.println("main ends");
43      }
44  }
```

输出变成：

```
main begins
initialize A1
initialize A2
initialize A9
initialize A3
initialize A4
initialize A5
initialize A6
copy from A6
initialize B2
initialize A8
main ends
```

只有在程序第一次使用到某个类时才去尝试读取其 `.class` 文件。类加载只会进行一次，这一次类加载会完成所有的静态初始化工作。

Question5

不能。因为构造方法是 `private` 的，其他的外部类无权访问。

Question6

因为构造方法是 `private` 的，只能在类内部访问，而类内部仅仅 `new` 了一个实例，那么这个类就只可能有一个实例。因为唯一的实例是静态变量，因此在类加载时被构造。

Question7

```
Singleton.getInstance().foo();
```