Java 程序设计 LAB03

JDK 版本	jdk-8u221-windows-64bit jdk1.8.0_221	代码文件编码方式	UTF-8
IDE 种类（如果有）	Eclipse	IDE 版本（如果有）	jdk-8u131-windows-x64_8.0.1310.11.exe

1. 初始化 I

阅读这段代码，请问程序的输出是什么？

```
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
```

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？

能证明，A6、A7都在A8之前输出

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？

不能证明，静态属性会比非静态属性先初始化，在输出中可以发现，a6 出现早于 a3、a4、a5，但是 A3、A4、A5 的输出早于 A6，即 a6 初始化晚于 a3、a4、a5，初始化顺序不等同于他们在类定义中出现的顺序。

描述静态属性的初始化方式和实际初始化时的顺序。

当这个类第一次被使用时，会对其进行加载（即初始化这个类），这个类中的静态属性也会被初始化，即随着类的加载而加载，并且只加载一次。但是实际初始化时，会按照 静态属性在这个类中出现的顺序对其进行初始化。

这段代码能够证明 “在类的实例第一次被构造、或类的静态属性和静态方法第一次被访问

时，JVM 会执行类加载”吗？如果不能，请尝试修改代码并证明。

不能证明

修改代码如下：

```
public class Initialization
    { static A aa=new A(10);
    static B b1;
    static A aaa=new A(20);
    static A ab=b1.a3;
    static B b2;
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println("main  begins");
        A a9 = new A(9);
        b1 = new B(1);
        b2        =        new        B(2);
        System.out.println("main ends");
    }
}
```

输 出 ：

```
initialize A1
initialize A2
initialize A10
initialize A20
initialize A3
initialize A4
initialize A5
main begins
initialize A9
initialize A6
copy from A6
initialize B1
initialize A8
initialize A6
copy from A6
initialize B2
initialize A8
main ends
```

解释：

输出中的

```
initialize A1
initialize A2
```

1. 类的实例第一次被构造时，JVM 会执行类加载

是在 aa 第一次被构造时，JVM 加载了A 类时输出的，紧接着按照 `public A(int i)`构造方法输出 `initialize A10`。

2．类的静态属性和静态方法第一次被访问时，JVM 会执行类加载
紧接着定义对象 b1，在 b1 之后定义 aaa，其后 b1.a3 调用了类 B 的静态成员变量，输出了
initialize A3
initialize A4
initialize A5
此时对类 B 进行了加载，依次输出 A3、A4、A5，并且在 main begins 之前；但是在此之前输出了 initialize A20 这也说明了只是定义实例，不调用相应的类的变量和和方法 JVM 是不会加载该类的。
该代码路径：\18373540_陈欣渝_LAB03\oo\01\ Initialization.java

2. 单例模式

其他的外部类可以通过`new Singleton()`来构造一个新的 Singleton 变量吗？

不能

为什么只可能有 1 个 Singleton 类的实例同时存在？这个唯一的实例在什么时候被构造？

因为其他的外部类不能通过“new Singleton()”来构造一个新的 Singleton 变量，而在类 Singleton()的内部，定义了一个这个类的 private 对象，以及这个 private 构造方法，这些都是外部类不能访问的，所以只可能有 1 个 Singleton 类的实例同时存在。
这个唯一的实例在加载该类时被构造。

请写出任意一种外部类调用 Singleton 类的 foo()的方法。

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