# JAVA编程进阶上机报告

****

**学 院 智能与计算学部**

**专 业 软件工程**

**班 级 1**

**学 号 3018216031**

**姓 名 白文杰**

1. **实验要求**

Singleton pattern. In software engineering, the singleton pattern is a software design pattern that restricts the instantiation of a class to one object. This is useful when exactly one object is needed to coordinate actions across the system. The concept is sometimes generalized to systems that operate more efficiently when only one object exists, or that restrict the instantiation to a certain number of objects.

The singleton design pattern describes how to solve the above problems:

(1)Hide the constructor of the class.

(2)Define a public static operation that returns the sole instance of the class.

Please implement the singleton pattern in Java by yourself.

Deadline: 23:59:59, 13rd  March. 2020

1. **源代码**

Singleton.java文件

**package** Singleton;

**public** **class** Singleton {

**private** **static** Singleton *singleton* = **new** Singleton();

**private** Singleton() {

System.***out***.println("This is always me! ");

}

**public** **static** Singleton getInstance() {

**return** *singleton*;

}

}

Demo.java文件

**package** Singleton;

**public** **class** Demo {

**public** **static** **void** main(String[] args) {

Singleton object1 = Singleton.*getInstance*();

Singleton object2 = Singleton.*getInstance*();

System.***out***.println("Is object1 same as object2? ");

System.***out***.println(object1 == object2) ;

}

}

1. **实验结果**

代码运行结果：

This is always me!

Is object1 same as object2?

true