Experiment – 6

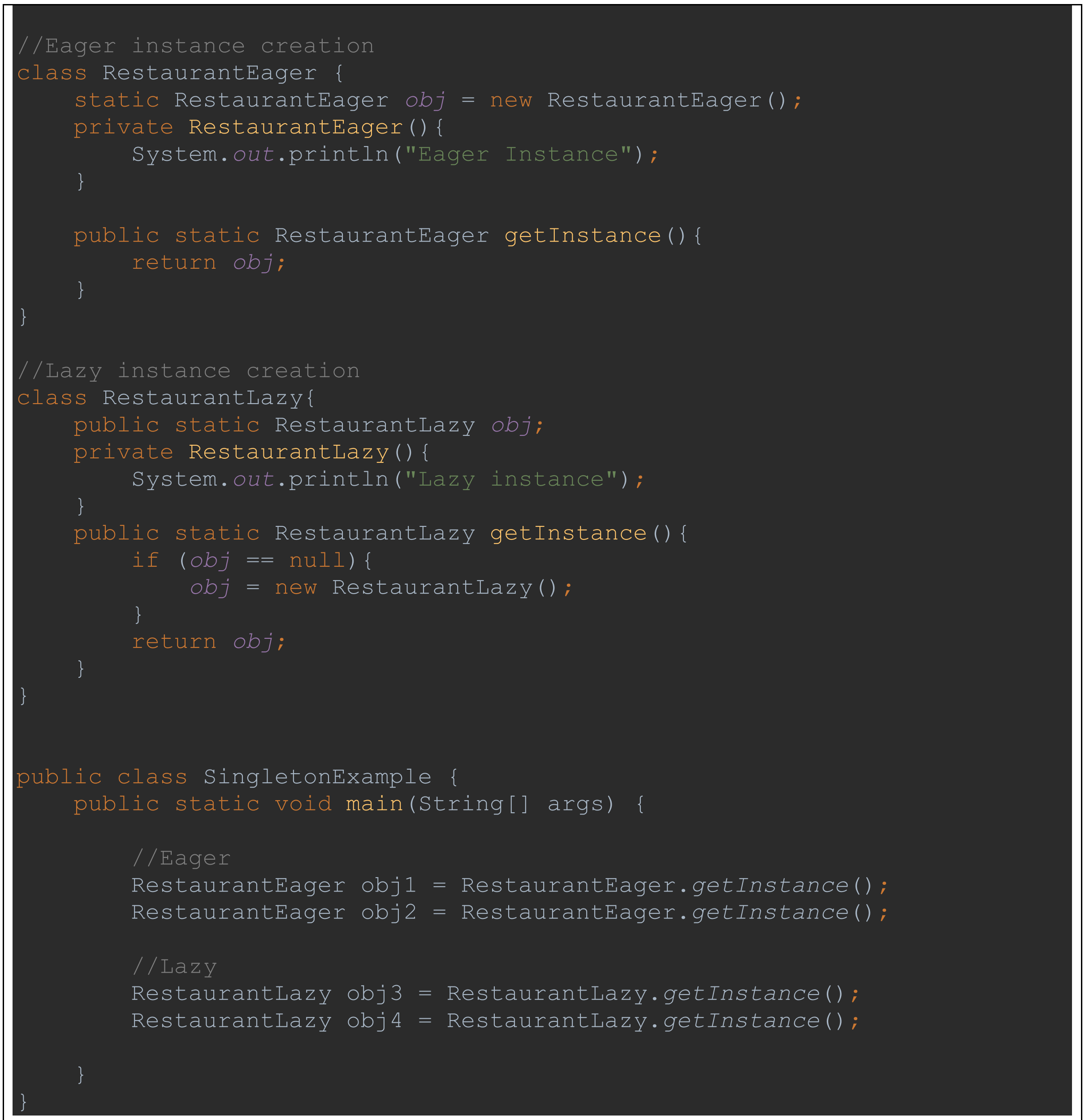
Aim – Singleton design pattern

Name – Shivraj Nakum , 21BCP125

Concept - With lazy initialization you crate instance only when its needed and not when the class is loaded. So you escape the unnecessary object creation.

///

SingletonExample.java



```java
//Eager instance creation
class RestaurantEager {
    static RestaurantEager obj = new RestaurantEager();
    private RestaurantEager(){
        System.out.println("Eager Instance");
    }

    public static RestaurantEager getInstance(){
        return obj;
    }
}

//Lazy instance creation
class RestaurantLazy{
    public static RestaurantLazy obj;
    private RestaurantLazy(){
        System.out.println("Lazy instance");
    }
    public static RestaurantLazy getInstance(){
        if (obj == null){
            obj = new RestaurantLazy();
        }
        return obj;
    }
}


public class SingletonExample {
    public static void main(String[] args) {

        //Eager
        RestaurantEager obj1 = RestaurantEager.getInstance();
        RestaurantEager obj2 = RestaurantEager.getInstance();

        //Lazy
        RestaurantLazy obj3 = RestaurantLazy.getInstance();
        RestaurantLazy obj4 = RestaurantLazy.getInstance();

    }
}
```

Output –

```
Eager Instance
Lazy instance
```