

# Assignment 13

---

by. 俞贤皓 21301114

2022.6.9

## 格式相关

- 文档提供markdown、pdf和html格式
  - 推荐使用html格式
- 文件夹README下 存放 **运行结果截图**
- 其他文件夹存放 **代码**

## Mod4 Ex1

- 

## Mod4 Ex3

- 关于wait/notify的使用：
  - 在addJob()中
    - 若队列已满，则该producer线程进入wait池；
    - 若成功加入job，则notifyAll()，让printer继续getJob()；
  - 在getJob()中
    - 若队列为空，则printer线程进入wait池；
    - 若成功读出job，则notifyAll()，让producer继续addJob()；
  - 我没有让两个方法抛出异常，我直接在方法内对异常进行处理。
- 停止程序的判断方法
  - 若 三个producer线程均执行完毕 且 printer队列为空，则说明程序执行完毕，可以终止printer；
  - 这个方法和标准程序的方法可能不一样，因为我没有利用到halt()方法中synchronized的性质。

```
C: Print manager is starting up.
C: Waiting on a job to print.
P: Adding job 'Kisaragi Tomi # 1' to the queue
P: Adding job 'Ogata Nenji # 1' to the queue
P: Adding job 'Shinonome Ryōko # 1' to the queue
C: Starting job 'Kisaragi Tomi # 1'
P: Adding job 'Kisaragi Tomi # 2' to the queue
C: Completed job 'Kisaragi Tomi # 1'
C: Starting job 'Ogata Nenji # 1'
P: Adding job 'Kisaragi Tomi # 3' to the queue
P: Adding job 'Ogata Nenji # 2' to the queue
P: Adding job 'Shinonome Ryōko # 2' to the queue
C: Completed job 'Ogata Nenji # 1'
C: Starting job 'Shinonome Ryōko # 1'
C: Completed job 'Shinonome Ryōko # 1'
C: Starting job 'Kisaragi Tomi # 2'
C: Completed job 'Kisaragi Tomi # 2'
C: Starting job 'Kisaragi Tomi # 3'
C: Completed job 'Kisaragi Tomi # 3'
C: Starting job 'Ogata Nenji # 2'
C: Completed job 'Ogata Nenji # 2'
C: Starting job 'Shinonome Ryōko # 2'
C: Completed job 'Shinonome Ryōko # 2'
C: Print manager is halted.
```

## Thread

- 我在编写这道题目的代码时遇到了一个问题，单例模式被多个线程同时初始化时，会产生多个对象。
  - 解决方法1：修改getInstance方法，使得其线程安全（可以被多个线程同时访问）。
    - [参考文章](#)
  - 解决方法2：在main方法里，先调用一次getInstance方法，使得单例模式初始化完毕。

入栈：a	入栈：k	
入栈：b	入栈：l	
入栈：c	入栈：m	
入栈：d	入栈：n	
入栈：e	入栈：o	
出栈：e	出栈：o	
出栈：d	出栈：n	
出栈：c	出栈：m	
出栈：b	出栈：l	入栈：u
出栈：a	出栈：k	入栈：v
入栈：f	入栈：p	入栈：w
入栈：g	入栈：q	入栈：x
入栈：h	入栈：r	入栈：y
入栈：i	入栈：s	出栈：y
入栈：j	入栈：t	出栈：x
出栈：j	出栈：t	出栈：w
出栈：i	出栈：s	出栈：v
出栈：h	出栈：r	出栈：u
出栈：g	出栈：q	入栈：z
出栈：f	出栈：p	出栈：z

.