## Declaration on Plagiarism

*This form must be filled in and completed by the student(s) submitting an assignment*

|  |  |
| --- | --- |
| **Name:** | Xiaofo Geng |
| **Student Number:** | 19210733 |
| **Programme:** | MCM |
| **Module Code:** | CA670 |
| **Assignment Title:** | Concurrent Programming |
| **Submission Date:** | 16 March 2020 |
| **Module Coordinator:** | Dr. David Sinclair |

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. I understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. I have read and understood the Assignment Regulations. I have identified and included the source of all facts, ideas, opinions, and viewpoints of others in the assignment references. Direct quotations from books, journal articles, internet sources, module text, or any other source whatsoever are acknowledged and the source cited are identified in the assignment references. This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study.

I have read and understood the referencing guidelines found at <http://www.dcu.ie/info/regulations/plagiarism.shtml>, <https://www4.dcu.ie/students/az/plagiarism> and/or recommended in the assignment guidelines

Name:\_\_\_\_Xiaofo Geng\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_16 March 2020\_\_\_\_\_\_\_\_\_\_\_\_

## Project Description

1. Project overview

A small barber shop has a set of M barbers who spends all their lives serving customers, one at a time. Each barber has chair in which the customer sits when they are getting their hair cut. When there are no customers in the shop waiting for their hair to be cut, a barber sleeps in his chair. Customer arrive at random intervals. If a customer arrives and finds the a barber asleep, he awakens the barber, sits in the barber’s chair and sleeps while his hair is being cut. If a customer arrives and all the barbers are busy cutting hair, the customer goes asleep in waiting chairs. When the barber finishes cutting a customer’s hair, he awakens the customer and holds the exit door open for him. If there are any waiting customers, he awakens one and waits for the customer to sit in the barber's chair, otherwise he goes to sleep.

1. The Purpose of the Project

Implement the technology of thread concurrent operation. Using java technology to avoid deadlock, starvation and other non-ideal situations.

1. Design ideas

In the single barber problem, the barber should check the waiting queue firstly, when there is no customer in the queue, it stays the waiting state (sleeping). when the user enters the shop, wake up the barber and have a haircut. If there is no free barber, the customer enters the waiting queue(sit-in chairs). If the waiting queue is full, the customer leaves off, release the thread and counting. “A *multiple sleeping barbers problem* has the additional complexity of coordinating several barbers among the waiting customers(Wikipedia, 2020).” So, I decided to bind a customer to a barber. Each barber shares the waiting queue of customers, but each customer can only choose their own barber to cut their hair, avoiding the situation that multiple barbers are awakened by one customer at the same time. To do this, I add an additional attribute Id for the customer and barber then bind them by get(), set() method. The number of barbers, the number of customers and the number of waiting chairs are input by keyboard. The user's arrival interval and barber's working time are random by the random() function. Mainly realize functions are following:

1. Using BarThread thread to control the barber's behaviour.
2. Using CusThread thread to control the customer's behaviour.
3. visitShop() realizes that a customer wakes up a barber and binds his ID.
4. leaveshop () implements the customer's haircut and leaves.
5. hellocustomer () implements that barber cut hair for customers and sleep when without customers.
6. byecustomer () enables the barber to finish work and release the current customer.
7. Logic model

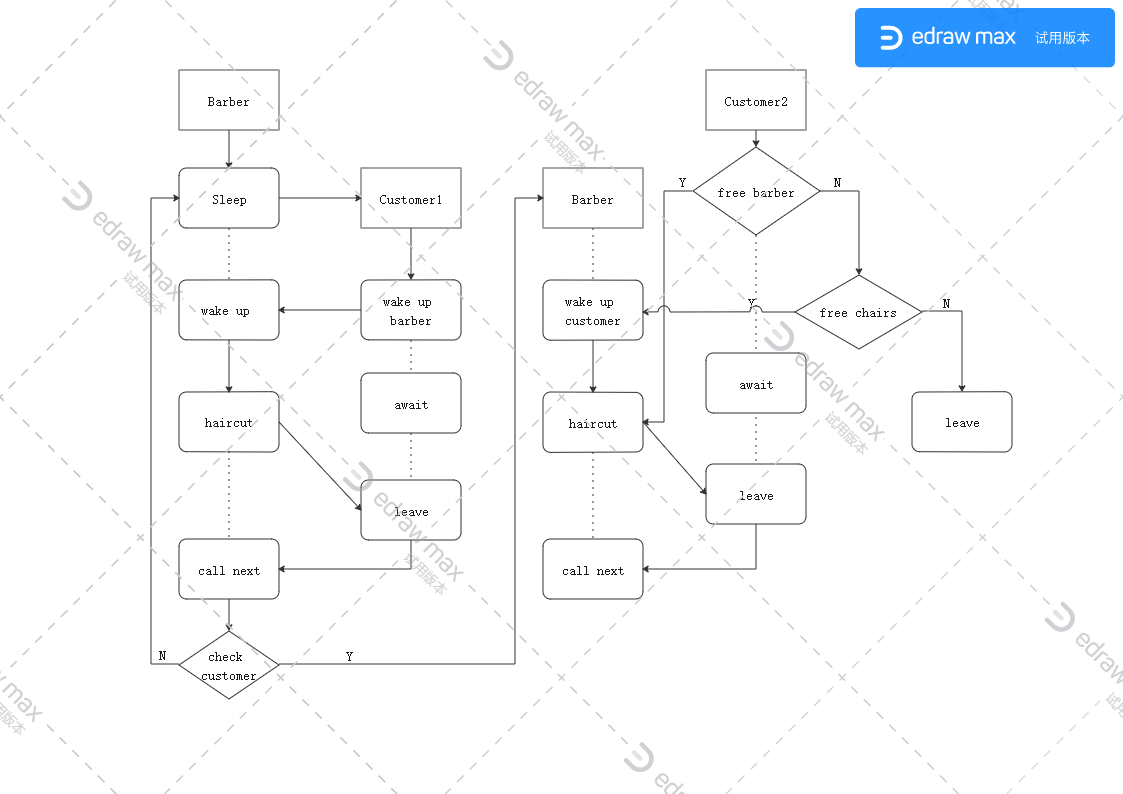


image 1 the logic model of sleeping barber

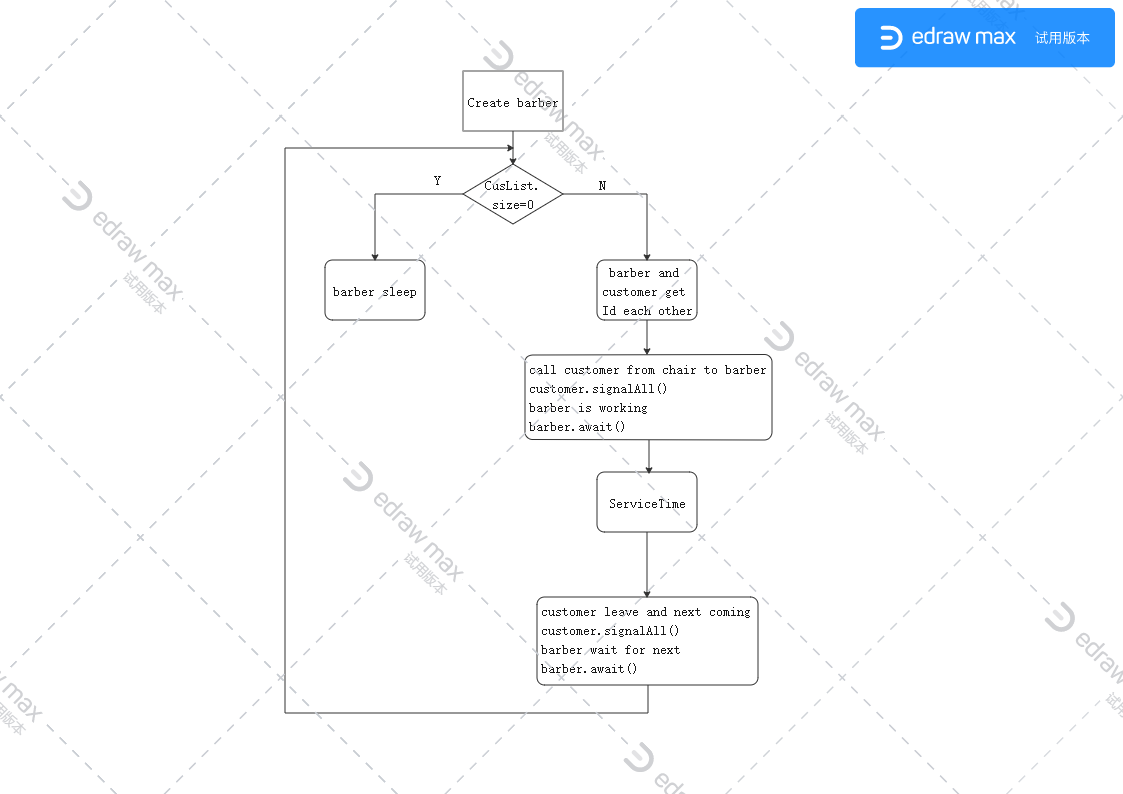
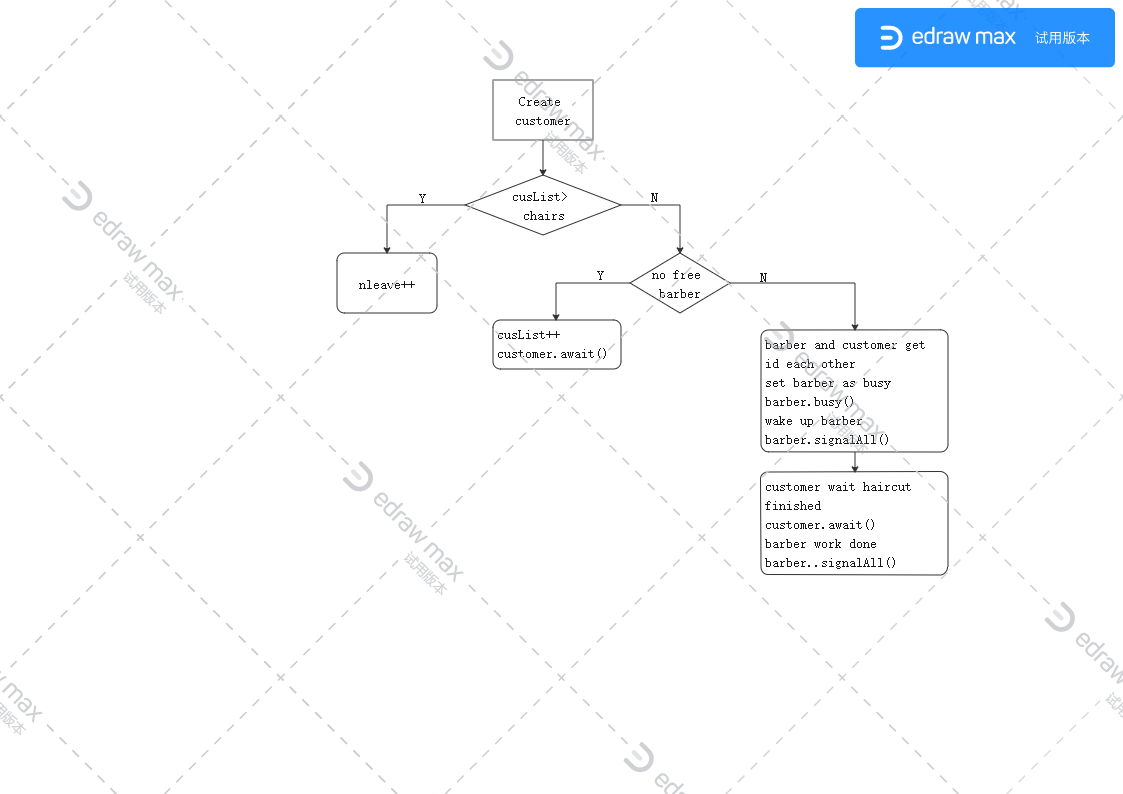
1. Flow chart

image 3 the flow chart of customer thread

image 2 the flow chart of barber thread

## Correctness and Fairness

1. Safety properties

Create a barber and customer list include barId and cusId attributes, respectively. When a customer wakes up a barber, he will get the Id of the barber who is currently waked up. At the same time, the barber will also get the Id of the current customer. Avoiding the situation that the customer wakes up multiple barbers and one barber cuts for multiple customers at the same time. Secondly, the main thread is placed in the try {} catch {} block to ensure that the exception handling mechanism when the thread cannot run.

1. Deadlock check

Using JConsole to check the thread deadlock situation. “JConsole is a graphical monitoring tool to monitor Java Virtual Machine (JVM) and Java applications both on a local or remote machine(Wikipedia, 2019).”

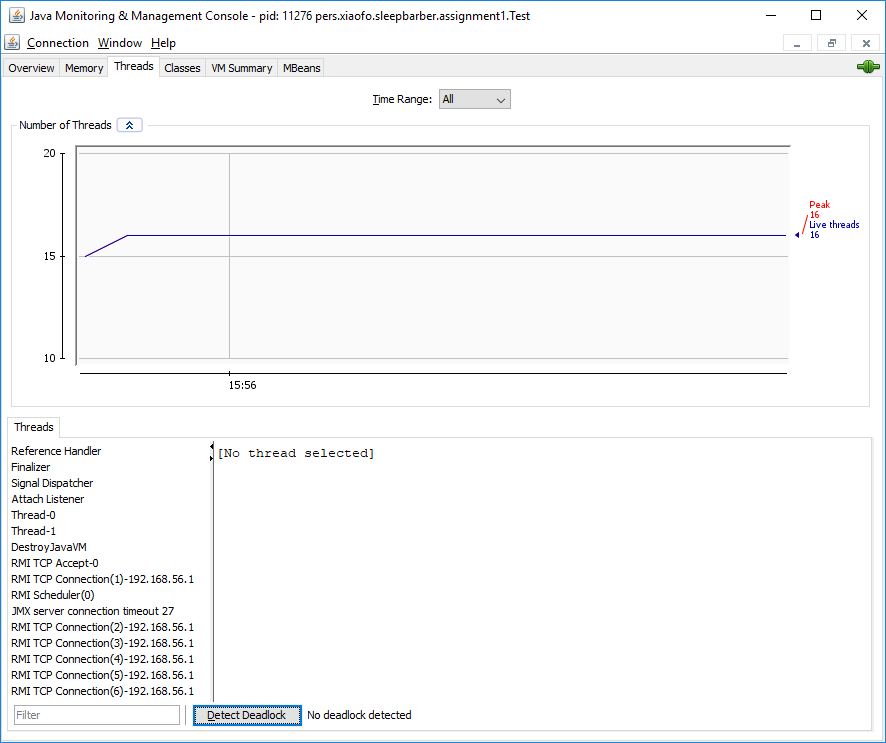


image 4 Deadlock check by JConsole

No deadlock is detected by JConsole.

## Problem solution

In this assignment, my solution is to use the condition() method and lock() method of ReentrantLock() mechanism in Java for interthread communication to solve the problem of sleeping barber. After the customer enters the barbershop, check whether the waiting queue is full. If it is full, the thread will be released. If not, check whether there are free barbers. If there are no free barbers, the customer calls customer. condition. await() and enters the waiting queue (chairs) to wait. If there are free barbers, customer wakes up him and asks to have a haircut. When there are no customers, the barber calls his barber. condition. await() to stay in wait.

When a customer comes, call barber.condition.signalall () to wake up a barber and call his own customer.condition.await () for haircut. At the same time, the barber calls his own barber.condition.await () for haircut; in this time, the customer and the barber are both stay in wait state. After the haircut time, the barber informs the customer that the haircut is finished and calls the customer. condition. signalall() of the customer to wake up the customer and calls his barber. condition. await() to fall back into waiting again. At this moment, the customer leaves and calls barber. condition. signalall() of the barber to end the barber's working status. Finally, barber check the waiting queue. If there is a customer in the waiting queue, it will call customer. condition. signalall() of the customer to wake up the customer and start haircutting. If there is no customer in the waiting queue, the barber will call his barber. condition. await() to start sleeping.

## Reference

Wikipedia (2019). JConsole. Available at: https://en.wikipedia.org/wiki/JConsole

Wikipedia (2020). Sleeping barber problem. Available at: <https://en.wikipedia.org/wiki/Sleeping_barber_problem>

Code reference :

The Sleeping-Barbers Problem <http://courses.washington.edu/css503/prog/prog2.pdf>