# Sleeping Barbers problem – observations

In the given task we were tasked with implementing a solution to the sleeping barber problem with multiple barbers and different client types. Clients could be male or female and were treated as a resource for the barbers who were implemented as 3 types of threads: female barbers -only cuts female customers, male barbers – only cut male clients and unisex barbers – they cut both male and female clients.

Instructions on how to compile and run are provided in README in the github folder. Parameters are defined by macros at the beginning of the file, commented with their description. Parameters include:

* M - number of seats in the waiting room.
* N1, N2, N3 – number of female, male and unisex barbers.

Multipliers:  
Invoke wait\_rand to wait random length of time between 2 and 5 seconds, multiplied by multiplier used on call.

* BarberSleepLen – how long barbers will sleep.
* CustomerSpawnCooldown -how long until next Customer is spawned.
* CutMultiplier – For all types of barbers, multiplier for cut time.

**Observations:**Observations and variations of parameters was rather rudimentary as there is a lot of possible combinations and not a lot of time.   
For these parameters:  
SleepLen = CutMultiplier = 3 and SpawnCoolDown = 1

With 4 barbers (2F, 1M, 1U) number of waiting spaces didn’t impact the simulation a lot. With simulation running into ~40 customers it was observed that the waiting room rarely filled up and not once did it run out of space. Those numbers were chosen as good representation of a small barber shop.  
Decreasing the waiting space and number of barbers event further – 1U and 1M + 2 seats in the waiting room made it that the waiting room filled very quickly and mostly with female customers (as there was no dedicated barber for them and only one that could serve them). So quickly customers were left without a haircut.

This simulation able to be run in multitude of configurations and I suspect that it was not the goal of the task to see how the parameters impact a simulated hair salon and its customers. So, I’m talk about my observations about the synchronization.

**Synchronization:**While it proved a bit tricky at first using semaphores for synchronization between thread of Barbers yielded good (in my opinion) results. While the barbers cannot talk between each other in my implementation the synchronization between barbers and the main thread that handles the creation of customers (resources) works rather well. By acting on just one cell in an array the correct thread can pick up the change and act on it. Unfortunately, in my version each thread that is sleeping need to constantly check its semaphore, as I was unable to implement a way to let it sleep and make it sleep externally.