# Sleeping Barber

**CSCI 201**

**Principles of Software Development**

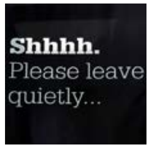
Jeffrey Miller, Ph.D.
*jeffrey.miller@usc.edu*

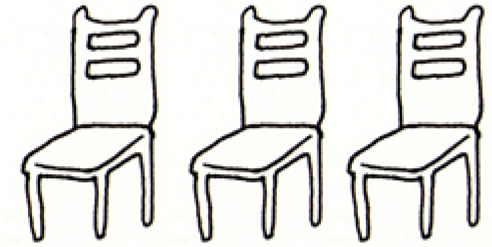
## Outline

- Sleeping Barber

# Sleeping Barber Overview

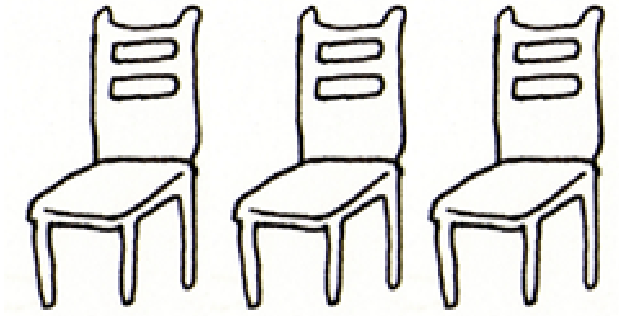
- The Sleeping Barber problem contains one barber and a number of customers
- There are a certain number of waiting seats
  - If a customer enters and there is a seat available, he will wait
  - If a customer enters and there is no seat available, he will leave
- When the barber isn't cutting someone's hair, he sits in his barber chair and sleeps
  - If the barber is sleeping when a customer enters, he must wake the barber up


Shhhh.
Please leave quietly...



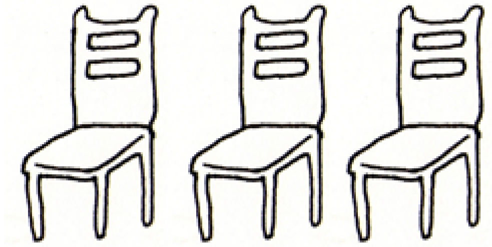
# Program

- Write a solution to the Sleeping Barber problem. You will need to utilize synchronization and conditions.



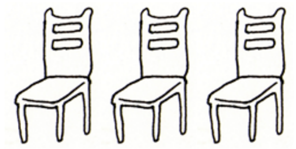
# Program Steps – `main` Method

- The `SleepingBarber` will be the main class, so create the `SleepingBarber` class with a main method
  - The `main` method will instantiate the `SleepingBarber`
  - Then create a certain number of `Customer` threads that arrive at random times
    - Have the program sleep for a random amount of time between customer arrivals
  - Have the main method wait to finish executing until all of the `Customer` threads have finished
  - Print out that there are no more customers, then wake up the barber if he is sleeping so he can go home for the day



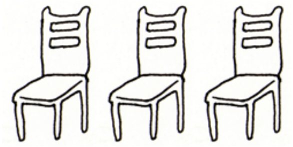
# Program Steps – `SleepingBarber`

- The `SleepingBarber` constructor will initialize some member varia
  - Total number of seats in the waiting room
  - Total number of customers who will be coming into the barber shop
  - A `boolean` variable indicating whether more customers will be coming into the barber shop
  - An `ArrayList` of customers who are currently waiting to get their hair cut
  - A lock on the barber with a condition to signify whether he is sleeping
- Start the `SleepingBarber` thread at the end of the constructor
- The `run()` method of the `SleepingBarber` will perform the hair cutting as follows
  - Continue looping as long as more customer will be coming into the barber shop
  - As long as there are any customers waiting
    - Get the customer who has been waiting the longest and cut his hair for a certain amount of time
    - Let the customer know when you have started and when you have finished cutting his hair
  - If there are no customers waiting, go to sleep (i.e. get the lock and await() on the sleeping condition)
    - Don't forget to release the lock in a `finally` block
- Create a method to add a customer to the waiting list
  - If the number of waiting customers equals the number of seats in the waiting room, have the customer leave
  - Otherwise, add the customer to the waiting list
  - Print out all of the customers currently waiting
  - Make sure to synchronize this method so that customers cannot be added to the list while iterating through it
- Create a method customers can call to wake up the barber
  - Acquire the lock, signal on the sleeping condition, then release the lock



# Program Steps – `Customer`

- The `Customer` constructor will initialize some member variables
  - A variable representing the customer name
  - An instance of the `SleepingBarber`
  - A lock on the customer with a condition to signify whether he is getting a haircut
- Create a method that allows the barber to get the customer's name
- Create a method the barber will call when he starts cutting the customer's hair
  - This method should just print out that the customer is getting his hair cut
- Create a method the barber will call when he is done cutting the customer's hair
  - This method will print out that the customer is done getting his hair cut
  - It also should signal the condition signifying that he is getting his hair cut
- The `run()` method of the `Customer` will perform the following tasks
  - Add the customer to the barber's waiting list
    - If there are no seats available in the waiting area, print out a message letting the user know the customer is leaving without getting his hair cut
  - Wake up the barber if he is sleeping
    - Note: There is nothing wrong with signaling a condition even if no thread is waiting on it
  - Get the customer lock and wait on the haircut condition
    - Even though the customer may not be getting a haircut yet, the barber will call the method above when he starts cutting his hair, then the barber will call the other method above when he is done
    - Don't forget to release the lock in the `finally` block
  - Print out a message that the customer is leaving



# Program Steps – `Util`

- The `Util` class will be used for printing
- Create a static method called `printMessage` that takes a `String` as a parameter
  - This method will print out a formatted date/time followed by a dash then the message
  - To format a date/time, use the `java.util.Calendar` class
  - Get an instance of the `Calendar`, then get the data out of the `Calendar` so that it can be formatted as follows

    ```
    yyyy-MM-dd HH:mm:ss.SSS
    ```
  - For example:

    ```
    2016-10-31 11:30:21.534
    ```
  - This method should be called from the `SleepingBarber` and `Customer` classes whenever something needs to be printed