Have three semaphores

They can be accessed using only two functions - wait and signal/post

* one semaphore for counting the number of customers present in the waiting room (not including the customer in the chair)
* one semaphore for indicating the status of barber: working or sleeping
* one mutex for giving the mutual exclusion to let the processes execute

If number of customers == number of chairs in the waiting rooms --> If new customers show up then they have to leave the shop.

If barber is ready, customer will go and get a haircut else the customer will wait in the waiting room.

Number of available seats in the waiting room - will decrease if more and more customers wait.

What is the initial state?

* Barber is sleeping.
* No customer is there.
* All seats are available in the waiting room.

Show Pseudocode from Wikipedia and Baeldung.

Barber

Infinite loop:

wait for customers to come in (wait on semaphore customer\_ready -- decrement)

wait for getting access to modify the number of available seats in waiting room

(wait on semaphore modify\_seats)

increment the count of available seats in waiting room by 1.

post semphore for getting access to modify the number of available seats in waiting room

(unlock semaphore modify\_seats)

post semaphore -- barber\_ready - make barber ready to serve

cut hair:

Lock the mutex?? - for putting critical section of work which barber thread does.

Customer

wait for getting access to modify the number of available seats in waiting room

(wait on semaphore modify\_seats)

if one or more seats are available:

decrement available seats count

post semaphore - customer\_ready (unlocked semaphore: means the customer is waiting and ready to be served by the barber)

post semphore for getting access to modify the number of available seats in waiting room

wait semaphore - barber\_ready

(customer is waiting for the barber to be ready)

get hair cut

else:

post semphore for getting access to modify the number of available seats in waiting room