# Solution to the Sleeping Barber Problem

CS-329 Operating Systems Complex Engineering Project

Author: Muhammed Omer BinMujeeb (CS-19095)

# My Solution Briefly Explained

I took on the problem by using Python's own threading library. I used it to create barber and customer semaphores. The mutex semaphore required also required for the solution was instead implemented using Python's threading library's own implementation of mutexes which they call "locks".

The problem was tackled by invoking a barber routine on a single thread which waits for customer threads to service by posting the barber semaphore whenever a customer thread arrives.

Meanwhile, customer threads created as per the user's inputs call the customer function which wait in the waiting room if possible, otherwise concluding with no servicing by the barber thread/function. The customer threads which are waiting continuously try to acquire the barber semaphore which is analogous to customers checking the barber room window to see if the room is free. If the acquiring is successful which is only when the barber is free, the customer thread is serviced and is taken out of waiting i.e. its thread is shut.

# Python code:

```
from multiprocessing import Semaphore
import threading    # Import the threading module
import time         # Import the time module
import random       # Import the random module
import concurrent.futures


# Create a global variable to store the number of customers waiting
num_waiting = 0
# Create a global variable to store the number of chairs in the waiting room
num_chairs = 2
barbers=Semaphore(0)
customers=Semaphore(0)
mutex=threading.Lock()

def cutHair():
    print("Barber cutting hair")
    time.sleep(10)
    print("Barber finished cutting hair")

def getHaircut():
    print("customer getting haircut")

def balk():
    print("customer is leaving")

def barber():
    # global customers
    # global barbers
    global num_waiting
    # global mutex

    while True:


        print("Barber waiting for customer")
        customers.acquire()
        print("Barber awaken by customer")

        mutex.acquire()
        num_waiting-=1
        barbers.release()
```

```
        mutex.release()
        cutHair()



def customer():
    # global customers
    # global barbers
    global num_waiting

    mutex.acquire()
    if num_waiting < num_chairs:

        num_waiting+=1
        print("Customer arrived")
        customers.release()
        mutex.release()

        barbers.acquire()
        getHaircut()
    else:
        mutex.release()
        balk()

def main():
    # global customers
    # global barbers
    global num_waiting
    global num_chairs
    # global mutex

    # Create a thread for the barber
    barber_thread = threading.Thread(target=barber)
    barber_thread.start()

    #Create a thread for each customer

    while True:

        time.sleep(5)
        customer_threads = []
        num_chairs = int(input("Enter the number of chairs in waiting room:
"))
        n_cust=int(input(("Enter the number of customers that enter the
barbershop: ")))
```

```
        cust_times=[]
        for i in range(n_cust):
            cust_time=input(f"Enter the time in units at which customer {i+1}
arrives: ")
            cust_times.append(int(cust_time))

        for tim in cust_times:
            customer_thread = threading.Thread(target=customer)
            customer_threads.append(customer_thread)
        prev=0
        n=0
        for tim in cust_times:
            time.sleep(tim-prev)
            customer_threads[n].start()
            prev=tim
            n+=1

        for customer_thread in customer_threads:
            customer_thread.join()



    barber_thread.join()

main()
```

# Test Cases:

| Number of customers | Times customers arrive | Chairs in waiting room | Expected Output | Observed Output |
|---|---|---|---|---|
| 3 | 1,2,3 | 2 | Barber waiting for customer<br>Customer arrived<br>Barber awaken by customer<br>Barber cutting hair<br>customer getting haircut<br>Customer arrived<br>Customer arrived<br>Barber finished cutting hair<br>Barber waiting for customer<br>Barber awaken by customer<br>Barber cutting hair<br>customer getting haircut<br>Barber finished cutting hair<br>Barber waiting for customer<br>Barber awaken by customer<br>Barber cutting hair<br>customer getting haircut<br>Barber finished cutting hair<br>Barber waiting for customer | Barber waiting for customer<br>Customer arrived<br>Barber awaken by customer<br>Barber cutting hair<br>customer getting haircut<br>Customer arrived<br>Customer arrived<br>Barber finished cutting hair<br>Barber waiting for customer<br>Barber awaken by customer<br>Barber cutting hair<br>customer getting haircut<br>Barber finished cutting hair<br>Barber waiting for customer<br>Barber awaken by customer<br>Barber cutting hair<br>customer getting haircut<br>Barber finished cutting hair<br>Barber waiting for customer |
| 6 | 1,2,3,4,5,6 | 2 | Barber waiting for customer<br>Customer arrived<br>Barber awaken by customer<br>Barber cutting hair<br>customer getting haircut<br>Customer arrived<br>Customer arrived<br>customer is leaving<br>customer is leaving<br>customer is leaving<br>Barber finished cutting hair<br>Barber waiting for customer<br>Barber awaken by | Barber waiting for customer<br>Customer arrived<br>Barber awaken by customer<br>Barber cutting hair<br>customer getting haircut<br>Customer arrived<br>Customer arrived<br>customer is leaving<br>customer is leaving<br>customer is leaving<br>Barber finished cutting hair<br>Barber waiting for customer<br>Barber awaken by |

| | | | | |
|---|---|---|---|---|
| | | | customer<br>Barber cutting hair<br>customer getting haircut<br>Barber finished cutting hair<br>Barber waiting for customer<br>Barber awaken by customer<br>Barber cutting hair<br>customer getting haircut<br>Barber finished cutting hair<br>Barber waiting for customer | customer<br>Barber cutting hair<br>customer getting haircut<br>Barber finished cutting hair<br>Barber waiting for customer<br>Barber awaken by customer<br>Barber cutting hair<br>customer getting haircut<br>Barber finished cutting hair<br>Barber waiting for customer |
| 0 | - | 2 | Barber waiting for customer | Barber waiting for customer |
| 3 | 1,2,3 | 0 | Barber waiting for customer<br>Customer arrived<br>Barber awaken by customer<br>Barber cutting hair<br>customer getting haircut<br>customer is leaving<br>customer is leaving<br>Barber finished cutting hair<br>Barber waiting for customer | Barber waiting for customer<br>customer is leaving<br>customer is leaving<br>customer is leaving |

Note: The difference between the final expected and observed outputs is due to the fact that the problem assumes that a customer can only enter the barber room after sitting on a waiting room seat first.

# Testing Screenshots

```
Barber waiting for customer
Enter the number of chairs in waiting room: 2
Enter the number of customers that enter the barbershop: 3
Enter the time in units at which customer 1 arrives: 1
Enter the time in units at which customer 2 arrives: 2
Enter the time in units at which customer 3 arrives: 3
Customer arrived
Barber awaken by customer
Barber cutting hair
customer getting haircut
Customer arrived
Customer arrived
Barber finished cutting hair
Barber waiting for customer
Barber awaken by customer
Barber cutting hair
customer getting haircut
Barber finished cutting hair
Barber waiting for customer
Barber awaken by customer
Barber cutting hair
customer getting haircut
Enter the number of chairs in waiting room: Barber finished cutting hair
Barber waiting for customer
```

```
Enter the number of chairs in waiting room: Barber finished cutting hair
Barber waiting for customer
2
Enter the number of customers that enter the barbershop: 6
Enter the time in units at which customer 1 arrives: 1
Enter the time in units at which customer 2 arrives: 2
Enter the time in units at which customer 3 arrives: 3
Enter the time in units at which customer 4 arrives: 4
Enter the time in units at which customer 5 arrives: 5
Enter the time in units at which customer 6 arrives: 6
Customer arrived
Barber awaken by customer
Barber cutting hair
customer getting haircut
Customer arrived
Customer arrived
customer is leaving
customer is leaving
customer is leaving
Barber finished cutting hair
Barber waiting for customer
Barber awaken by customer
Barber cutting hair
customer getting haircut
Barber finished cutting hair
Barber waiting for customer
Barber awaken by customer
Barber cutting hair
customer getting haircut
Enter the number of chairs in waiting room: Barber finished cutting hair
```

```
Enter the number of chairs in waiting room: Barber finished cutting hair
Barber waiting for customer
2
Enter the number of customers that enter the barbershop: 0
Enter the number of chairs in waiting room: 0
Enter the number of customers that enter the barbershop: 3
Enter the time in units at which customer 1 arrives: 1
Enter the time in units at which customer 2 arrives: 2
Enter the time in units at which customer 3 arrives: 3
customer is leaving
customer is leaving
customer is leaving
Enter the number of chairs in waiting room:
```