OPERATING SYSTEMS
(CS-317)

REPORT

Complex Engineering Activity

THE BARBERSHOP PROBLEM

PROGRAM CODE

```
from threading import Semaphore, Thread
import time

#Variables
no = 0                                  # Customer no who gets hair cut
chairs = 2                              # Waiting Room chairs(n)
cust_count = 0
total_cust = 0

#Semaphores
semWakeup = Semaphore(0)               # For Waking the Barber when first customer arrives
semOnlyOne = Semaphore(1)              # For allowing only 1 Customer to have hair cut at a time
semstartcut= Semaphore(0)              # For Telling Barber to start the hair cut



def getHairCut():
    global cust_count,total_cust, no
    print("Customer",no," wants to get a haircut ans sits on the Barber chair")



def cutHair():
        print("Barber is cutting Customer",no ,"hair")


def balk():
        global cust_count,total_cust, no
        print("customer",total_cust," leaves due to non-empty chairs.")


#Barber
def barber():
    global chairs,cust_count, no
    while True:
        if cust_count == 0:                                     # No customers in shop
            print("Barber is sleeping")                         # Barber is sleeping

            semWakeup.acquire()                                 # First Customer arrives
            print("Barber is awake")                            # Barber wakes up

            while semstartcut.acquire():
                cutHair()                                       # Barber starts cutting the hair when a
                                                                  customer asks for hair cut
        break


#Customer
def customer():
    global chairs, cust_count, total_cust, no
    while True:
        cust_count = cust_count + 1                             # No of Customers
        total_cust = total_cust + 1
        #cust_queue.append(total_cust)
        if chairs!=0:                                           # If Waiting Chairs are empty
            chairs=chairs -1                                    # Customer occupies one chair
            if cust_count == 1:                                 # First Customer arrives
                print("The first customer is here!")
                semWakeup.release()                             # Customer wakes the Barber
                time.sleep(10)
```

```
            else:                                               # Second and onward Customers arrives
                print("Customer",total_cust,"arrives and sits on waiting chair")
                #break


            semOnlyOne.acquire()                                # Only 1 Customer can ask Barber for Hairc
                                                                  ut at a time
            no = no + 1                                         # Customer no to get haircut
            getHairCut()                                        # Asks the Barber for HairCut
            chairs = chairs + 1                                 # Customer sits on the Barber chair for
                                                                  Hair cut and one waiting chair
                                                                  gets emptied
            semstartcut.release()                               # Tells the Barber to start the haircut
            time.sleep(10)
            print("Customer",no,"leaves shop after haircut.")   # Customer leaves after the Hair Cut
            semOnlyOne.release()                                # Now any other customer waiting can ask f
                                                                  or the hair cut


            cust_count = cust_count-1                          # 1 Customer leaves the shop after hair cut
            if cust_count == 0:                                 # No Customers in the shop
                print("All Customers leave the shop")
                print("Barber goes back")
                barber()

        else:
            balk()                                              # If Waiting Chairs are full

        break



t1 = Thread(target = barber)                                    # Barber
t1.start()
t2 = Thread(target = customer)                                  # Customer 1
t2.start()
t3= Thread(target = customer)                                   # Customer 2
t3.start()
t4= Thread(target = customer)                                   # Customer 3
t4.start()
t5=Thread(target = customer)                                    # Customer 4
t5.start()
```

TEST CASES

Test Case 01: All the waiting room seats are occupied by the customers.

Input:
Waiting Room Chairs = n = 2
Barber Chair = 1
No of Customers arrived = 4

Output:
- Barber is sleeping.
- Customer 1 arrives, occupies a waiting chair and wakes the barber.
- Customer 2 arrives and occupies a waiting chair.
- Waiting Chairs are full.
- Customer 1 asks barber for a cut
- Customer 1 leaves the waiting chair and sits on the Barber Chair.
- 1 chair empty in waiting room.
- Barber cuts Customer 1 hair.
- Customer 3 arrives.
- Both the waiting chairs are full
- Customer 4 arrives and leaves the shop.
- Customer 1 leaves after the haircut.
- Customer 2 asks barber for a cut
- Customer 2 leaves the waiting chair and sits on the Barber Chair.
- Barber cuts Customer 2 hair.
- Customer 2 leaves.
- Customer 3 asks barber for a cut
- Customer 3 leaves the waiting chair and sits on the Barber Chair.
- All the chairs in waiting room are empty now.
- Barber cuts Customer 3 hair.
- Customer 3 leaves.

Test Case 02: Waiting room seats are available and the customers leave after their haircut.

Input:
Waiting Room Chairs = n = 2
Barber Chair = 1
No of Customers arrived = 2

Output:
- Barber is sleeping.
- Customer 1 arrives, occupies a waiting chair and wakes the barber.
- Customer 2 arrives and occupies a waiting chair.
- Waiting Chairs are full.
- Customer 1 asks barber for a cut

- Customer 1 leaves the waiting chair and sits on the Barber Chair.
- 1 chair empty in waiting room.
- Barber cuts Customer 1 hair.
- Customer 1 leaves after the haircut.
- Customer 2 asks barber for a cut
- Customer 2 leaves the waiting chair and sits on the Barber Chair.
- Barber cuts Customer 2 hair.
- Customer 2 leaves.
- All the chairs in waiting room are empty now.
- All the customers have left.
- Barber goes back to sleep.

<u>Test Case 03:</u> Waiting room seats are available, Customers waits and get their hair cut by the Barber.

<u>Input:</u>
Waiting Room Chairs = n = 5
Barber Chair = 1
No of Customers arrived = 5

<u>Output:</u>
- Barber is sleeping.
- Customer 1 arrives, occupies a waiting chair and wakes the barber.
- Customer 2 arrives and occupies a waiting chair.
- Customer 1 asks barber for a cut
- Customer 1 leaves the waiting chair and sits on the Barber Chair.
- 3 chair empty in waiting room.
- Barber cuts Customer 1 hair.
- Customer 3 arrives and sits on the waiting chair.
- Customer 4 arrives and sits on the waiting chair.
- Customer 5 arrives and sits on the waiting chair.
- Customer 1 leaves after the haircut.
- Customer 2 asks barber for a cut
- Customer 2 leaves the waiting chair and sits on the Barber Chair.
- Barber cuts Customer 2 hair.
- Customer 2 leaves.
- Customer 3 asks barber for a cut
- Customer 3 leaves the waiting chair and sits on the Barber Chair.
- Customer 3 leaves.
- Customer 4 asks barber for a cut
- Customer 4 leaves the waiting chair and sits on the Barber Chair.
- Customer 4 leaves.
- Customer 5 asks barber for a cut
- Customer 5 leaves the waiting chair and sits on the Barber Chair.
- Customer 5 leaves.

EXECUTION SNAPSHOTS

Test Case 01:

```
#Variables
no = 0                              # Customer no who gets hair cut
chairs = 2                          # Waiting Room chairs(n)
cust_count = 0
total_cust = 0

#Semaphores
semWakeup = Semaphore(0)            # For Waking the Barber when first customer arrives
semOnlyOne = Semaphore(1)           # For allowing only 1 Customer to have hair cut at a time
semstartcut= Semaphore(0)           # For Telling Barber to start the hair cut
```

```
t1 = Thread(target = barber)                                    # Barber
t1.start()
t2 = Thread(target = customer)                                  # Customer 1
t2.start()
t3= Thread(target = customer)                                   # Customer 2
t3.start()
t4= Thread(target = customer)                                   # Customer 3
t4.start()
t5=Thread(target = customer)                                    # Customer 4
t5.start()
```

Test Case 02:

```
t1 = Thread(target = barber)                                    # Barber
t1.start()
t2 = Thread(target = customer)                                  # Customer 1
t2.start()
t3= Thread(target = customer)                                   # Customer 2
t3.start()
```

```
#Variables
no = 0                              # Customer no who gets hair cut
chairs = 2                          # Waiting Room chairs(n)
cust_count = 0
total_cust = 0

#Semaphores
semWakeup = Semaphore(0)            # For Waking the Barber when first customer arrives
semOnlyOne = Semaphore(1)           # For allowing only 1 Customer to have hair cut at a time
semstartcut= Semaphore(0)           # For Telling Barber to start the hair cut
```

Test Case 03:

```
#Variables
no = 0                                  # Customer no who gets hair cut
chairs = 5                              # Waiting Room chairs(n)
cust_count = 0
total_cust = 0

#Semaphores
semWakeup = Semaphore(0)                # For Waking the Barber when first customer arrives
semOnlyOne = Semaphore(1)               # For allowing only 1 Customer to have hair cut at a time
semstartcut= Semaphore(0)               # For Telling Barber to start the hair cut


def getHairCut():
    global cust_count,total_cust, no
```

```
t1 = Thread(target = barber)                                    # Barber
t1.start()
t2 = Thread(target = customer)                                  # Customer 1
t2.start()
t3= Thread(target = customer)                                   # Customer 2
t3.start()
t4 = Thread(target = customer)                                  # Customer 3
t4.start()
t5= Thread(target = customer)                                   # Customer 4
t5.start()
t6 = Thread(target = customer)                                  # Customer 5
t6.start()
```