Operating system 2 Project – Cover sheet

Project Title Multiple Sleeping Barbers Problem Group#…….

Discussion time :- 11:20 AM Instructor ……

ID	Name(Arabic)	Bounce	Minus	Total Grade	Comment
202000042	احمد طارق عبدالعزيز عثمان				
202000135	اسماء عربي محمد عبدالمجيد				
202000218	بيشوي عصام فاروق				
202000236	چمان سمير صلاح الدين محمد خطاب				
202000549	عبدالله محمد حمدي عبدالعزيز				
202000587	عمر احمد عبدالرحيم محمد				
202000685	ماجد ممدوح محمد حلمي				
202001050	يارا محمد احمد عبداللطيف				

Critrial		Grade	Team Grade	Comment
Documentation	Solution pseudocode	1		
	Examples of Deadlock	1		
	How did solve deadlock	1		
	Examples of starvation	1		
	How did solve starvation	1		
	Explanation for real world application and how did apply the problem	1		
GitHub	Upload project files	2		
	Submitted before discussion time (shared GitHub project link with TA and Dr)	1		
	Only one contribution	-1		
Implementation	Run correctly (correct output)	5		
	Run but with incorrect output	-3		
	Not run at all (error and exceptions)	-8		
	Free from Deadlock	3		
	Free from deadlock in some cases and not free in other cases	-2		
	Free from Starvation	2		
	Free from Starvation in some cases and not free in other cases	-1		
	Apply problem to real world application	5		
Total	Total grade for Team	25		
	Total Team Grade(after adjustment)	25		
Bounce	Multithreading GUI Based Java Swing	+5		
	Multithreading GUI Based Java Swing(adjustment)			
	Multithreading GUI Based JavaFX	+10		
	Multithreading GUI Based JavaFX(adjustment)			
	Bounce Graphic and animation	+5		
Total with Bounce	Total Team Grade			
	Total Team Grade(after adjustment)			

Contents

1) Solution pseudocode

a. customers pseudocode:

1-If a customer enters the barbershop, and the one of the two barbers is sleeping and the other is unavailable:

if availableBarber=1

then

TheOtherBarberState := awake AND availableBarber :=0

2-If the customer enters the barbershop, and the two barbers are busy:

if available Barber=0 AND numOfWatingChairs=0

then

customerState := waiting AND WaitingChairsCounter--

3-If a customer enters the shop and there are no available chairs:

if availableBarber=0 AND numOfWatingChairs=0

then

customerState := leaving

4- if the two barbares ar done and ther are no more waiting customer :

if numOfCustomer=0 AND unWorkingBarbers = numOfBarbers

then

BarberState :=sleeping

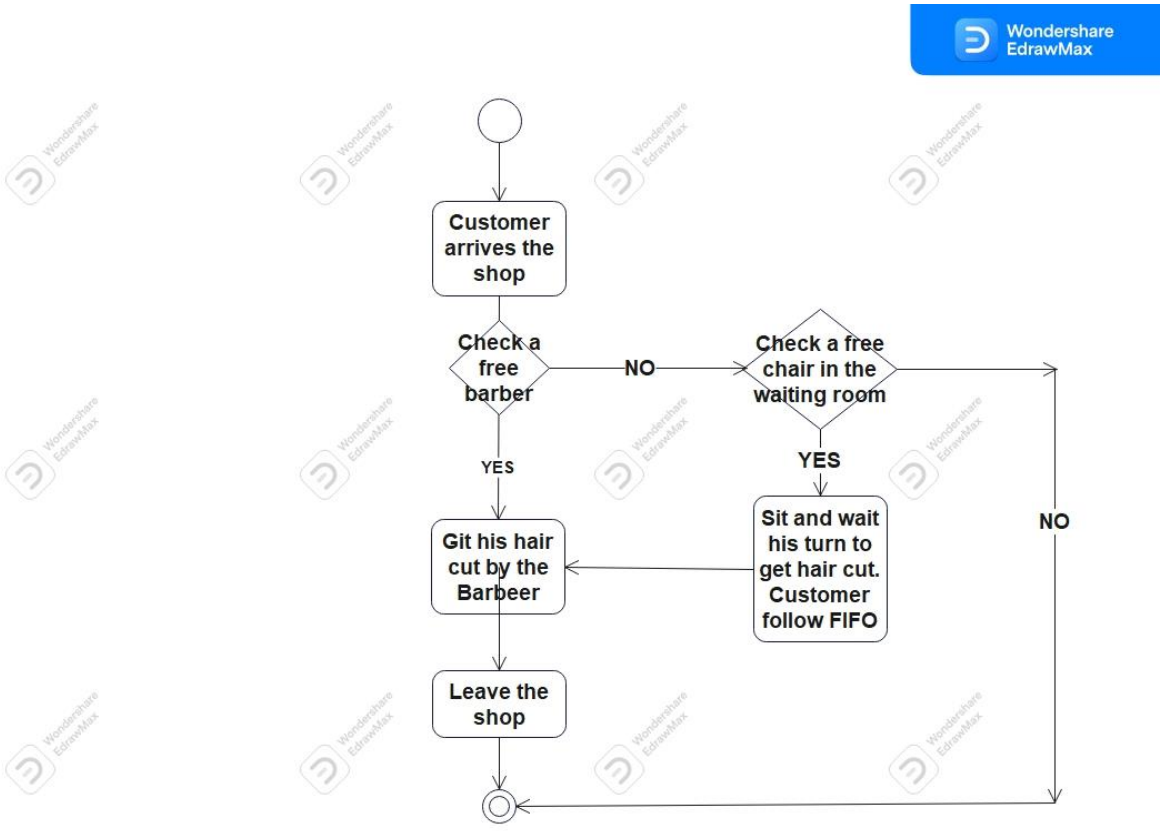
a. customers Language Processing :

- Customer should arrive at the shop.
- Go to the barber.
  - If there is a free barber.
  - The customer wakes him up and sits down to get his haircut.
  - If all barbers are busy
  - Go to the waiting room.
  - If there is an empty place, he should wait for his turn.

- Then he should go to the barber at his turn following FIFO.
- If there isn't an empty place the customer should leave the shop.

- When the customer finishes the cut he should leave the shop.

a. Flow Chart :



b. Barbers pseudocode:

1- If there is no customer to be served, the barbers are sleeping:

if numCustomer=0

then

barberState := sleeping

2- if the customer enters the barbershop, and the barber is sleeping:

if numCustomer=1 AND barberState=sleeping

then

barberState:= awake AND availableBarber=1

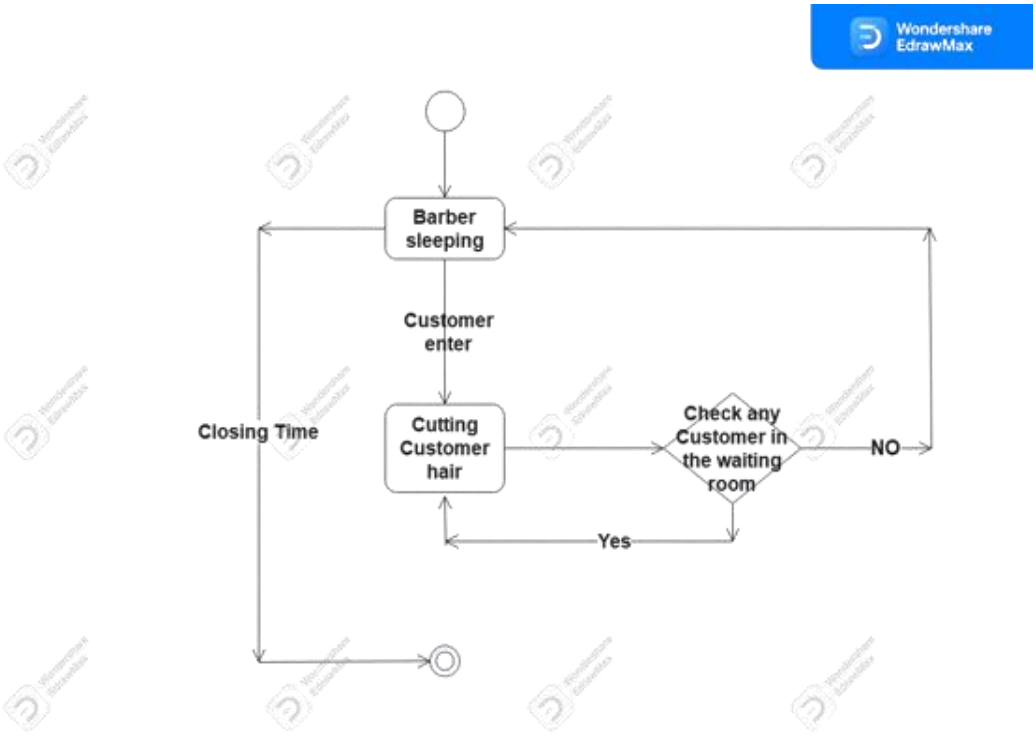
a. Barbers Language Processing :

- The barber is sleeping.
- If the customer enters the shop, he will wake the barber up.
- The barber will make the haircut for the customer who woke him up.
- Check if there is a customer in the waiting room.

- If yes: the barber will make him a haircut.
- Get ready for the next customer.
- If there isn't a customer in the waiting room: the barber will go to sleep.

– Repeat steps from step no 2 til the end of the program.

b. Flow Chart :



2) The problem of deadlock:

In general deadlock is Situation where two (or more) operations need overlapping sets of resources, and neither can complete because they cannot obtain all locks necessary to complete an operation and release their locks.
Deadlock is a situation where multiple operations are waiting for the same resource(s).

In our problem the deadlock problem will happen when a customer may arrive to find the barber cutting hair, so he returns to the waiting room to take a seat but while walking back to the waiting room the barber finishes the haircut and goes to the waiting room, (which he finds chair. That's the customer walks slowly or went to the restroom) and thus goes to sleep in the barber chair. That's why the customer will still be waiting, and the barber will still be sleeping.

3) The solution of the deadlock:

Making a customer list to put in any customer who enters the shop following FIFO strategy.

---

4) The problem of starvation:

In general, it is a problem encountered in concurrent computing where a process is perpetually denied necessary resources to process its work.

In our problem it happens when there is a customer came to the waiting room and the barber didn't notice so when anew customer came and checked whether there is a free barber he had his haircut and the customer in the waiting room is still waiting (who came first).

5) The solution of the starvation:

It is the same solution as the deadlock problem which is making a customer list to put in any customer who enters the shop following FIFO strategy.

---

6) Real-world application of the design

This design is the best analogy for a customer care call center. Initially when there is no customer on-call all call executives just relax and wait for the call. The moment the first customer dials the number he/she is connected to any call-executive and in a scenario when all call-executives are busy the customer will have to wait in a queue till they are assigned to a call-executive. If all executives are busy and the

waiting line is full, the customers are disconnected with a message that executives are busy and customers will be contacted later by the company. This best relates to this design as the customers are picked from the queue on a first come first serve basis and call-executives are utilized in such a way that everyone executive gets at least one call.
This idea is the same as the sleeping barber problem, as the employee who is the barber the barber is still sleeping till, he notices that there is a customer while in the call center problem the employee relaxes and wait till, he has the call which end his relaxation.
And the client who calls the employee and ends the employee's relaxation is the customer who came to the barber shop and wake the barber to have his haircut.