In this report I will explain my solution.

The customer and barber threads were synchronized in my solution with the help of Pthreads, a mutex, and condition variables. The program uses the shared variable " customer_finish" to keep track of the number of customers who have finished their haircuts and the number of customers who are waiting for a haircut.

The barber thread runs in a loop and calls pthread_cond_wait(&barber_sleeping, &lock) to wait for customers to arrive. The barber thread wakes up when a customer comes in, cuts the customer's hair, and calls pthread_cond_signal(&customer_wait) to invite the next customer in.By using pthread_cond_wait(&customer_wait, &lock), the customer threads attempt to get a haircut in a loop. The client leaves if there are no chairs available. The customer calls pthread_cond_signal(&barber_sleeping) to wake the barber if there are chairs available and the barber is sleeping.I added a new variable to the customer thread called "start_time" to measure each customer's waiting time. This variable tracks the time a customer starts waiting for a haircut. I subtract the "start_time" from the current time to determine the waiting time after the customer has received a haircut. This waiting time can be recorded for later review or printed out. This solution ensures that customers will not have to wait indefinitely for a haircut and that the barber will always have someone to cut their hair. When the specified number of customers have received haircuts, the program will end.

So in my code I already measure to duration of waiting time for each customer.

This is a sample output for my code and duration of waiting times for each customers.

To run my code write

```
gcc -o barber barber.c

./barber 5 2 2 3 4
```

```
Barber is sleeping...

Customer 2 is waiting for a haircut...

Customer 1 is waiting for a haircut...

Barber is cutting hair...

Barber finished cutting hair.

Barber is cutting hair...

Barber finished cutting hair.

Barber is sleeping...

Customer 2 got a haircut. Waiting time: 4 seconds. Haircuts: 0/4

Customer 1 got a haircut. Waiting time: 4 seconds. Haircuts: 0/4

Customer 2 is waiting for a haircut...
```

```
Barber is cutting hair...
Barber finished cutting hair.
Barber is sleeping...
Customer 1 is waiting for a haircut...
Barber is cutting hair...
Barber finished cutting hair.
Barber is sleeping...
Customer 1 got a haircut. Waiting time: 3 seconds. Haircuts: 1/4
Customer 2 got a haircut. Waiting time: 5 seconds. Haircuts: 1/4
Customer 2 is waiting for a haircut...
Customer 1 is waiting for a haircut...
Barber is cutting hair...
Barber finished cutting hair.
Barber is cutting hair...
Barber finished cutting hair.
Barber is sleeping...
Customer 1 got a haircut. Waiting time: 5 seconds. Haircuts: 2/4
Customer 2 got a haircut. Waiting time: 5 seconds. Haircuts: 2/4
Customer 1 is waiting for a haircut...
Barber is cutting hair...
Barber finished cutting hair.
Barber is sleeping...
Customer 1 got a haircut. Waiting time: 1 seconds. Haircuts: 3/4
Customer 2 is waiting for a haircut...
Barber is cutting hair...
Barber finished cutting hair.
Barber is sleeping...
Customer 2 got a haircut. Waiting time: 3 seconds. Haircuts: 3/4
```