[EOPSY] - Task nr 5 - Synchronization - The barber problem

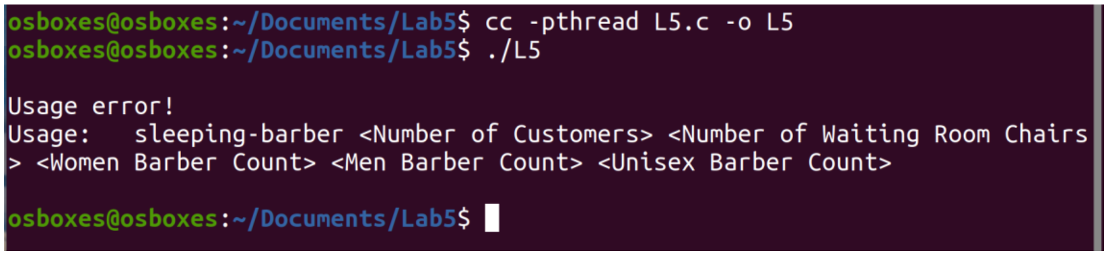
Seden Bayar - 295308

Description of the solution

In this task after including required files, constants (maximum customer number and haircut time for barbers) are defined. First, to run this program we need to enter parameters in order.

<Number of Customers> <Number of Waiting Room Chairs> <Women Barber Count> <Men Barber Count> <Unisex Barber Count>

- If the user enters more or less than those variables it prints the usage error and gives information about the usage rule.



```
osboxes@osboxes:~/Documents/Lab5$ cc -pthread L5.c -o L5
osboxes@osboxes:~/Documents/Lab5$ ./L5

Usage error!
Usage:   sleeping-barber <Number of Customers> <Number of Waiting Room Chairs
> <Women Barber Count> <Men Barber Count> <Unisex Barber Count>

osboxes@osboxes:~/Documents/Lab5$ 
```

In barber shop problem, there are N1 barbers serving only women, N2 barbers serving only men and N3 barbers serving both women and men. In the beginning we have only barbers in the shop and each barber has one barber's chair in a cutting room.

Before the shop opens all the queues that are defined and all the chairs in the shop are empty. When the program starts it shows general informations about the number of customers are going to be served, the number of chairs in the waiting room and the total number of barbers.

For each gender of customers have different queues. By means of this female or male barbers can easily take the waiting customer from their queues and unisex barbers can take customers from both queues.

Manage concurrent processes

Semaphore is a significant technique to manage concurrent processes by using a simple integer value. It is simply a variable which is non-negative and shared between threads. This variable is used to solve the critical section problem and to achieve process synchronization in the multiprocessing environment.

Counting Semaphores are initialized for all types of barbers and Binary Semaphores for a mutex (mutex lock).

Binary Semaphores – Only two states 0 & 1, i.e., locked/unlocked or available/unavailable.

Counting Semaphores – Semaphores which allow arbitrary resource count are called counting semaphores.

```
/* semaphores */

sem_t f_barbers;             /* semaphore for barbers */

sem_t m_barbers;

sem_t u_barbers;

sem_t mutex;                 /* provides mutual exclusive access to the barber chair */


/* creating semaphores */

   sem_init(&f_barbers, 0, 0);

   sem_init(&m_barbers, 0, 0);

   sem_init(&u_barbers, 0, 0);

   sem_init(&mutex, 0, 1);
```

POSIX Semaphores

sem_init() initializes the unnamed semaphore at the address pointed to by sem. The value argument specifies the initial value for the semaphore.

sem_wait() function locks the semaphore referenced by sem by performing a semaphore lock operation on that semaphore. If the semaphore value is currently zero, then the calling thread will not return from the call to sem_wait() until it either locks the semaphore or the call is interrupted by a signal.

sem_post() The sem_post() function unlocks the semaphore referenced by sem by performing a semaphore unlock operation on that semaphore.

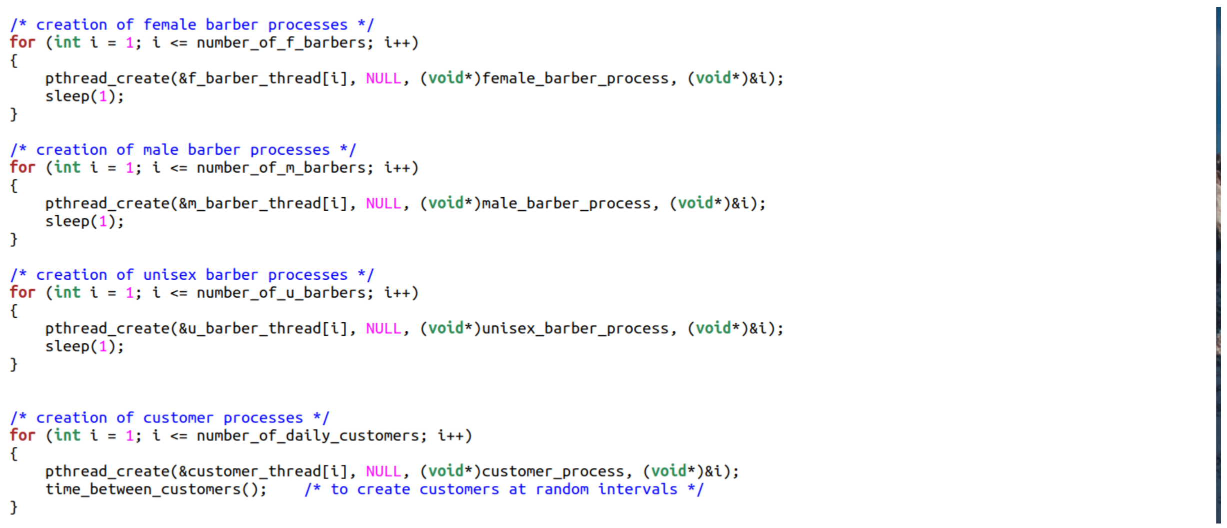
If the semaphore value resulting from this operation is positive, then no threads were blocked waiting for the semaphore to become unlocked; the semaphore value is simply incremented.

If the value of the semaphore resulting from this operation is zero, then one of the threads blocked waiting for the semaphore will be allowed to return successfully from its call to sem_wait().

sem_getvalue() places the current value of the semaphore pointed to sem into the integer pointed to by sval.

A multithreaded program contains two or more parts that can run concurrently. Each part of such a program is called a **thread**, and each thread defines a separate path of execution. In the main function there are 4 for loops for creation of all barbers and one for customer processes. Those processes have their own functions.

pthread_create() function starts a new thread in the calling process.



```c
/* creation of female barber processes */
for (int i = 1; i <= number_of_f_barbers; i++)
{
    pthread_create(&f_barber_thread[i], NULL, (void*)female_barber_process, (void*)&i);
    sleep(1);
}

/* creation of male barber processes */
for (int i = 1; i <= number_of_m_barbers; i++)
{
    pthread_create(&m_barber_thread[i], NULL, (void*)male_barber_process, (void*)&i);
    sleep(1);
}

/* creation of unisex barber processes */
for (int i = 1; i <= number_of_u_barbers; i++)
{
    pthread_create(&u_barber_thread[i], NULL, (void*)unisex_barber_process, (void*)&i);
    sleep(1);
}


/* creation of customer processes */
for (int i = 1; i <= number_of_daily_customers; i++)
{
    pthread_create(&customer_thread[i], NULL, (void*)customer_process, (void*)&i);
    time_between_customers();    /* to create customers at random intervals */
}
```

Functions

void female_barber_process(void* number);

void male_barber_process(void* number);

void unisex_barber_process(void* number);

void customer_process(void* number);

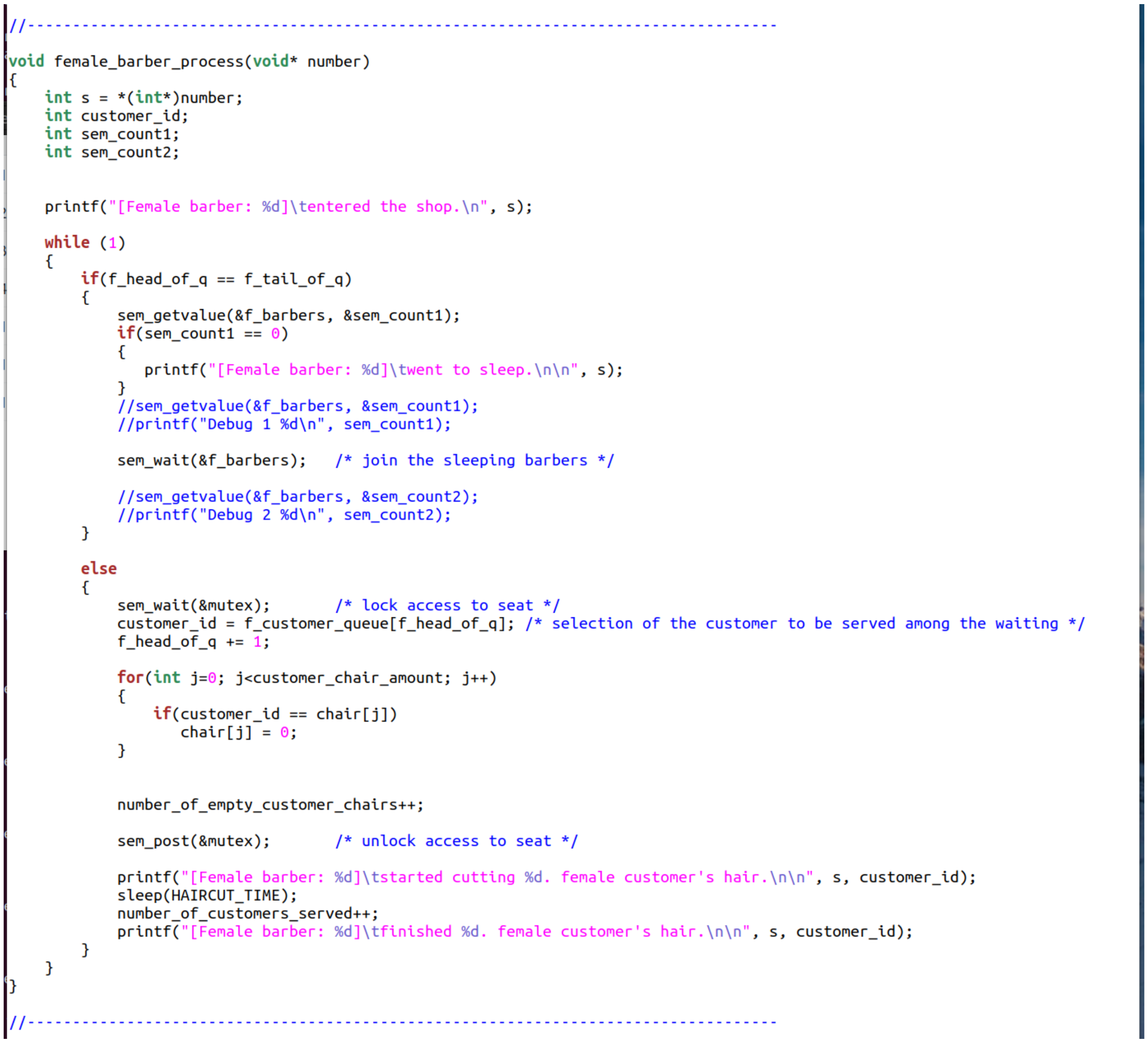
void time_between_customers();

Barber process functions

In Female/Male barber function before the first customer arrives to the shop all queues, chairs in the waiting room and all barber chairs are empty. When barber enters the shop, it checks the queue if it is empty, they return to the chair and sleeps in it.

Customers enter to the shop in random times. Random times between customers are provided with time_between_customers function. If there are customers, sem_wait function locks access to seat, he takes the customer from the queue (but only if he is able to cut hair, i.e. the barber brings a man into the cutting room only if he can cut his hair, etc.) back to the chair, unlocks access to seat and cuts hair. When the barber finishes cutting a customer's hair, he dismisses the customer and goes to the waiting room to see if there are others waiting. If there are, same process continues. If there are none, he returns to the chair and sleeps in it. For the female and the male barbers' process functions are the same but the unisex barbers can check both queues and takes customers from them. If both queues are empty unisex barbers go to sleep else, first they check the female queue if there are no female customers in the queue then check the male queue and follows the same steps in male and female functions.

Female barbers process function



```c
//-----------------------------------------------------------------------------------

void female_barber_process(void* number)
{
    int s = *(int*)number;
    int customer_id;
    int sem_count1;
    int sem_count2;


    printf("[Female barber: %d]\tentered the shop.\n", s);

    while (1)
    {
        if(f_head_of_q == f_tail_of_q)
        {
            sem_getvalue(&f_barbers, &sem_count1);
            if(sem_count1 == 0)
            {
               printf("[Female barber: %d]\twent to sleep.\n\n", s);
            }
            //sem_getvalue(&f_barbers, &sem_count1);
            //printf("Debug 1 %d\n", sem_count1);

            sem_wait(&f_barbers);   /* join the sleeping barbers */

            //sem_getvalue(&f_barbers, &sem_count2);
            //printf("Debug 2 %d\n", sem_count2);
        }

        else
        {
            sem_wait(&mutex);       /* lock access to seat */
            customer_id = f_customer_queue[f_head_of_q]; /* selection of the customer to be served among the waiting */
            f_head_of_q += 1;

            for(int j=0; j<customer_chair_amount; j++)
            {
                if(customer_id == chair[j])
                   chair[j] = 0;
            }


            number_of_empty_customer_chairs++;

            sem_post(&mutex);       /* unlock access to seat */

            printf("[Female barber: %d]\tstarted cutting %d. female customer's hair.\n\n", s, customer_id);
            sleep(HAIRCUT_TIME);
            number_of_customers_served++;
            printf("[Female barber: %d]\tfinished %d. female customer's hair.\n\n", s, customer_id);
        }
    }
}

//-----------------------------------------------------------------------------------
```

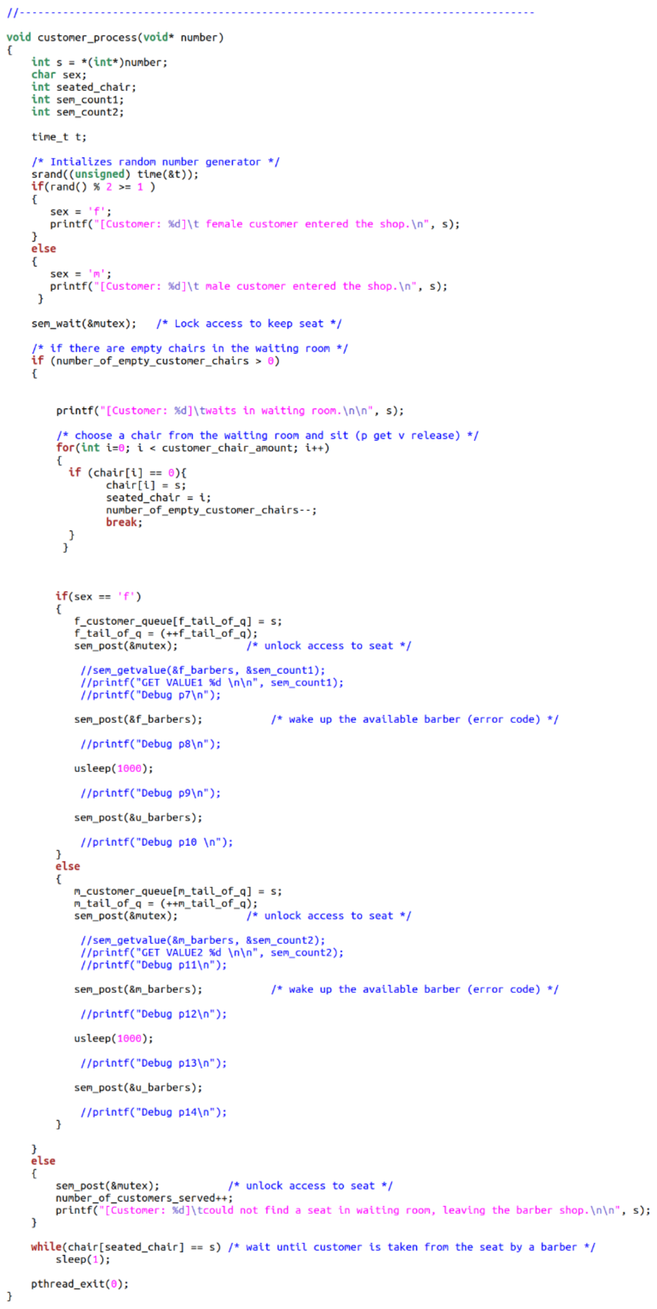
Unisex barbers process function

```
//------------------------------------------------------------------------------------

void unisex_barber_process(void* number)
{
    int s = *(int*)number;
    int customer_id;
    int sem_count1;
    int sem_count2;


    printf("[Unisex barber: %d]\tentered the shop.\n", s);

    while (1)
    {
        if ((f_head_of_q == f_tail_of_q) && (m_head_of_q == m_tail_of_q))
        {
        sem_getvalue(&u_barbers, &sem_count1);
        if(sem_count1 == 0)
        {
            printf("[Unisex barber: %d]\twent to sleep.\n\n", s);
        }
            //sem_getvalue(&u_barbers, &sem_count1);
            //printf("Debug 5 %d\n", sem_count1);

            sem_wait(&u_barbers);   /* join the sleeping barbers */

            //sem_getvalue(&u_barbers, &sem_count2);
            //printf("Debug 6 %d\n", sem_count2);
        }

        else
        {
            if(f_head_of_q != f_tail_of_q)
            {
               sem_wait(&mutex);       /* lock access to seat */
               customer_id = f_customer_queue[f_head_of_q]; /* selection of the customer to be served among the waiting */
               f_head_of_q += 1;

               for(int j=0; j<customer_chair_amount; j++)
               {
                   if(customer_id == chair[j])
                      chair[j] = 0;
               }

               number_of_empty_customer_chairs++;

               sem_post(&mutex);       /* unlock access to seat */

               printf("[Unisex barber: %d]\tstarted cutting %d. female customer's hair.\n\n", s, customer_id);
               sleep(HAIRCUT_TIME);
               number_of_customers_served++;
               printf("[Unisex barber: %d]\tfinished %d. female customer's hair.\n\n", s, customer_id);
            }
            else
            {
               sem_wait(&mutex);       /* lock access to seat */
               customer_id = m_customer_queue[m_head_of_q]; /* selection of the customer to be served among the waiting */
               m_head_of_q += 1;

               for(int j=0; j<customer_chair_amount; j++)
               {
                   if(customer_id == chair[j])
                      chair[j] = 0;
               }


               number_of_empty_customer_chairs++;

               sem_post(&mutex);       /* unlock access to seat */

               printf("[Unisex barber: %d]\tstarted cutting %d. male customer's hair.\n\n", s, customer_id);
               sleep(HAIRCUT_TIME);
               number_of_customers_served++;
               printf("[Unisex barber: %d]\tfinished %d. male customer's hair.\n\n", s, customer_id);
            }
        }
    }
}
```

Customer process function

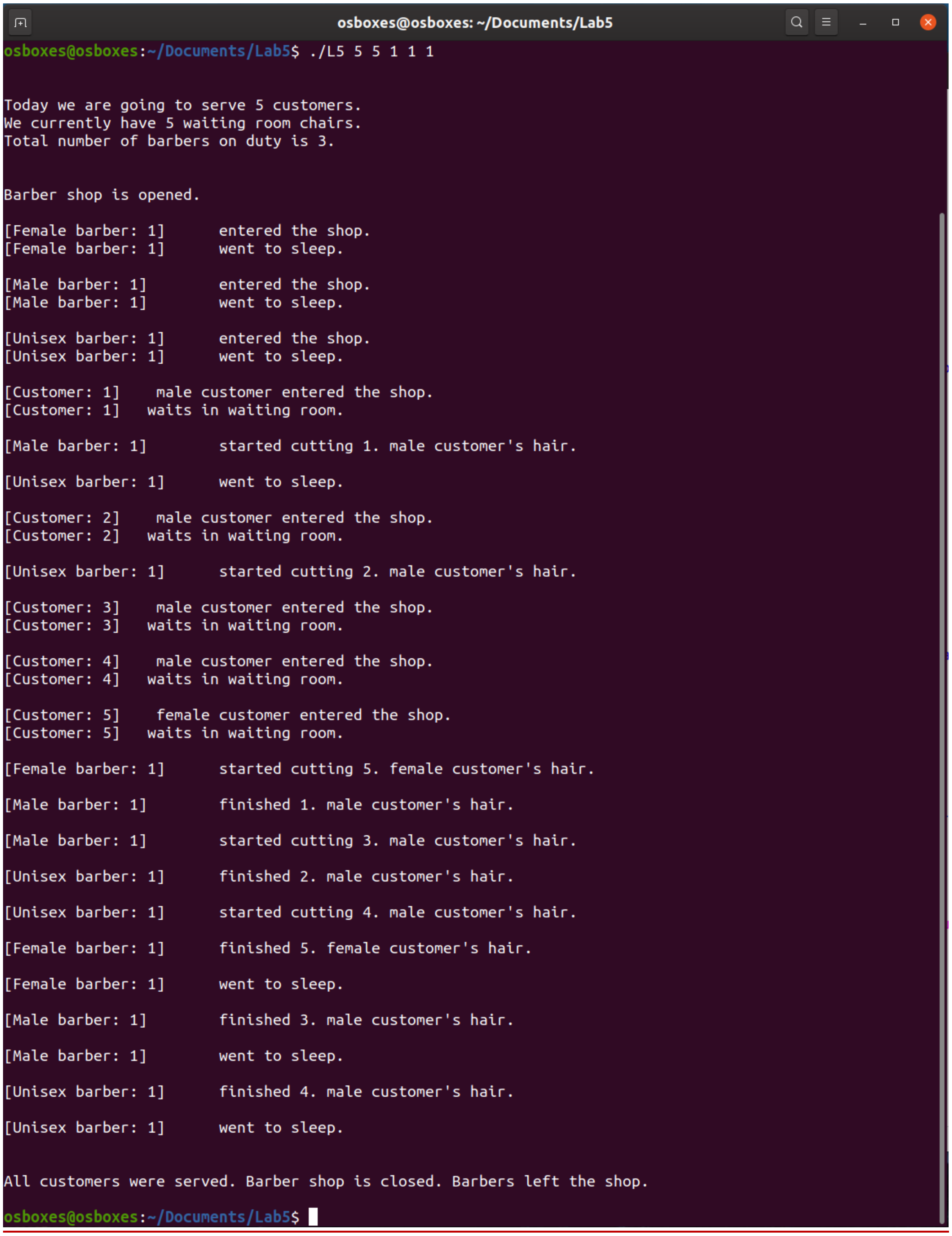
Each customer can be either a male client or a female client. In the customer process function the customers genders are initializing randomly. When a client arrives, he/she checks what barbers are doing. If there is any barber sleeping (who is able to serve the client), the customer wakes the barbers who can able to serve that customer with sem_post fuction and sits in the cutting room chair. If all barbers (able to serve the client) are cutting hair, the customer stays in the waiting room. If there is a free chair in the waiting room, the customer sits in it and waits their turn. If there is no free chair sem_post unlocks access to seat, the customer leaves.

After seeing all the customers that enters the barber shop were served the shop closes.



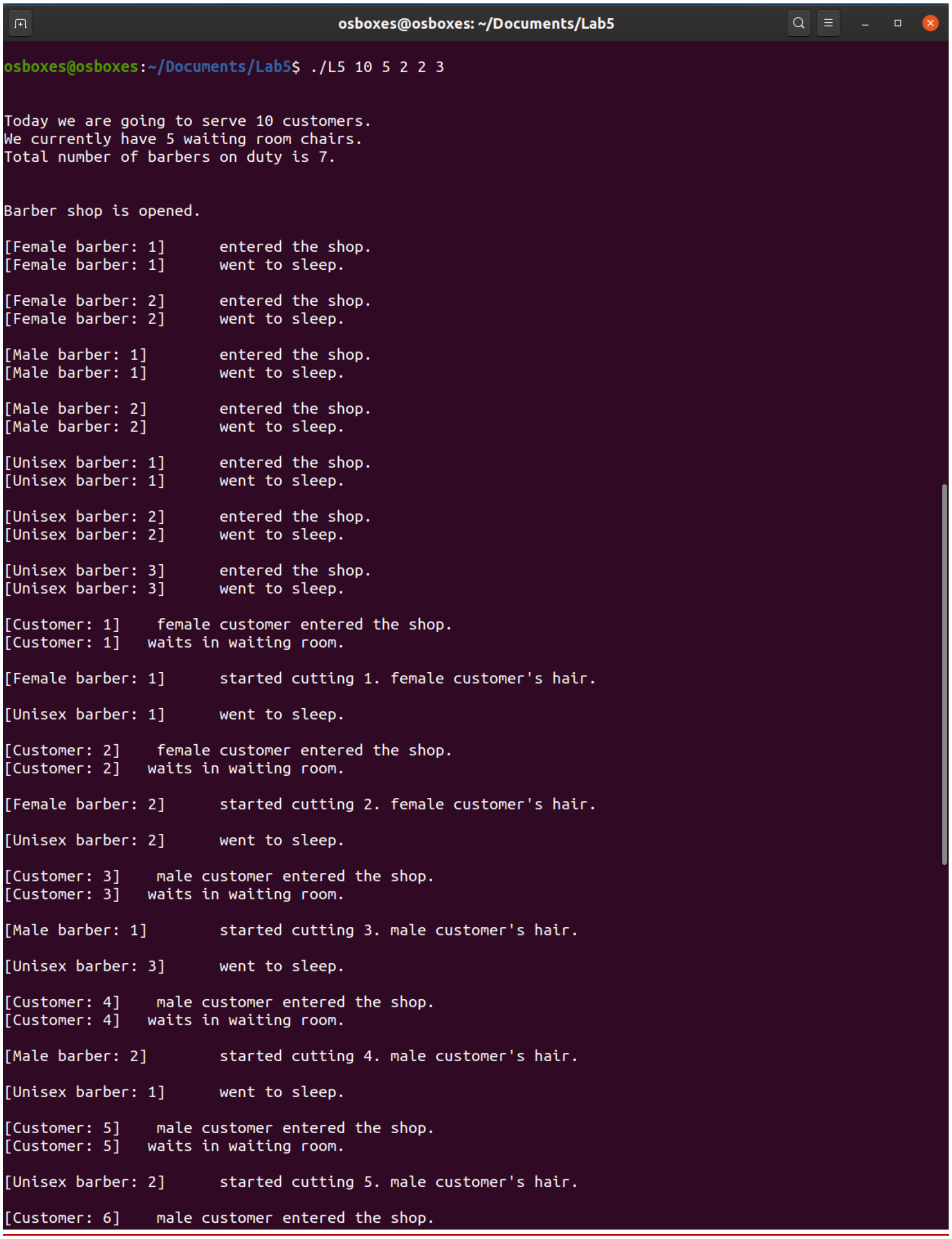
```c
//--------------------------------------------------------------------------------

void customer_process(void* number)
{
    int s = *(int*)number;
    char sex;
    int seated_chair;
    int sem_count1;
    int sem_count2;

    time_t t;

    /* Intializes random number generator */
    srand((unsigned) time(&t));
    if(rand() % 2 >= 1 )
    {
       sex = 'f';
       printf("[Customer: %d]\t female customer entered the shop.\n", s);
    }
    else
    {
       sex = 'm';
       printf("[Customer: %d]\t male customer entered the shop.\n", s);
     }

    sem_wait(&mutex);   /* Lock access to keep seat */

    /* if there are empty chairs in the waiting room */
    if (number_of_empty_customer_chairs > 0)
    {


        printf("[Customer: %d]\twaits in waiting room.\n\n", s);

        /* choose a chair from the waiting room and sit (p get v release) */
        for(int i=0; i < customer_chair_amount; i++)
        {
          if (chair[i] == 0){
                chair[i] = s;
                seated_chair = i;
                number_of_empty_customer_chairs--;
                break;
          }
         }


        if(sex == 'f')
        {
           f_customer_queue[f_tail_of_q] = s;
           f_tail_of_q = (++f_tail_of_q);
           sem_post(&mutex);           /* unlock access to seat */

            //sem_getvalue(&f_barbers, &sem_count1);
            //printf("GET VALUE1 %d \n\n", sem_count1);
            //printf("Debug p7\n");

           sem_post(&f_barbers);           /* wake up the available barber (error code) */

            //printf("Debug p8\n");

           usleep(1000);

            //printf("Debug p9\n");

           sem_post(&u_barbers);

            //printf("Debug p10 \n");
        }
        else
        {
           m_customer_queue[m_tail_of_q] = s;
           m_tail_of_q = (++m_tail_of_q);
           sem_post(&mutex);           /* unlock access to seat */

            //sem_getvalue(&m_barbers, &sem_count2);
            //printf("GET VALUE2 %d \n\n", sem_count2);
            //printf("Debug p11\n");

           sem_post(&m_barbers);           /* wake up the available barber (error code) */

            //printf("Debug p12\n");

           usleep(1000);

            //printf("Debug p13\n");

           sem_post(&u_barbers);

            //printf("Debug p14\n");
        }

    }
    else
    {
        sem_post(&mutex);           /* unlock access to seat */
        number_of_customers_served++;
        printf("[Customer: %d]\tcould not find a seat in waiting room, leaving the barber shop.\n\n", s);
    }

    while(chair[seated_chair] == s) /* wait until customer is taken from the seat by a barber */
        sleep(1);

    pthread_exit(0);
}
```

Test case 1


osboxes@osboxes: ~/Documents/Lab5

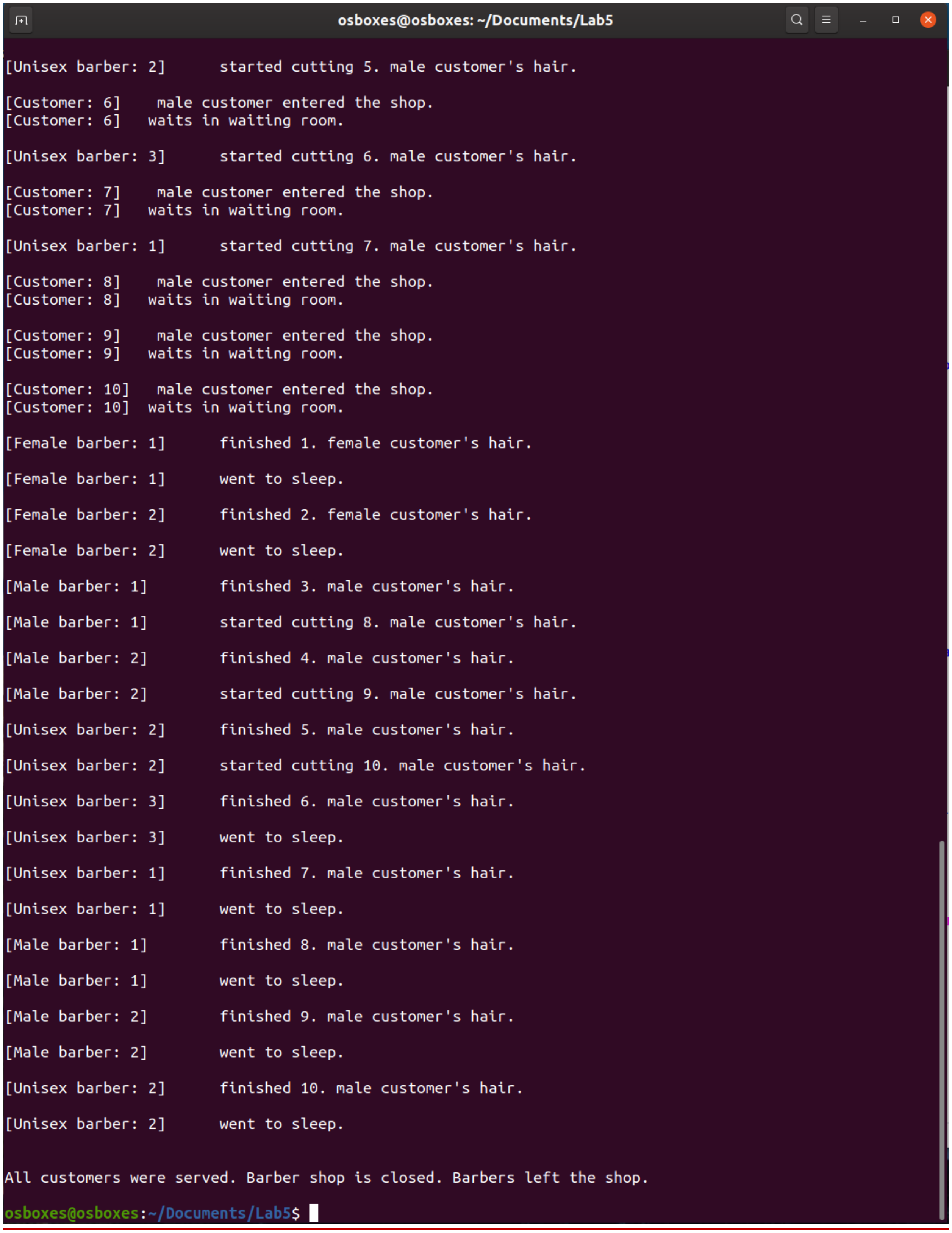
```
osboxes@osboxes:~/Documents/Lab5$ ./L5 5 5 1 1 1


Today we are going to serve 5 customers.
We currently have 5 waiting room chairs.
Total number of barbers on duty is 3.


Barber shop is opened.

[Female barber: 1]      entered the shop.
[Female barber: 1]      went to sleep.

[Male barber: 1]        entered the shop.
[Male barber: 1]        went to sleep.

[Unisex barber: 1]      entered the shop.
[Unisex barber: 1]      went to sleep.

[Customer: 1]    male customer entered the shop.
[Customer: 1]   waits in waiting room.

[Male barber: 1]        started cutting 1. male customer's hair.

[Unisex barber: 1]      went to sleep.

[Customer: 2]    male customer entered the shop.
[Customer: 2]   waits in waiting room.

[Unisex barber: 1]      started cutting 2. male customer's hair.

[Customer: 3]    male customer entered the shop.
[Customer: 3]   waits in waiting room.

[Customer: 4]    male customer entered the shop.
[Customer: 4]   waits in waiting room.

[Customer: 5]    female customer entered the shop.
[Customer: 5]   waits in waiting room.

[Female barber: 1]      started cutting 5. female customer's hair.

[Male barber: 1]        finished 1. male customer's hair.

[Male barber: 1]        started cutting 3. male customer's hair.

[Unisex barber: 1]      finished 2. male customer's hair.

[Unisex barber: 1]      started cutting 4. male customer's hair.

[Female barber: 1]      finished 5. female customer's hair.

[Female barber: 1]      went to sleep.

[Male barber: 1]        finished 3. male customer's hair.

[Male barber: 1]        went to sleep.

[Unisex barber: 1]      finished 4. male customer's hair.

[Unisex barber: 1]      went to sleep.


All customers were served. Barber shop is closed. Barbers left the shop.

osboxes@osboxes:~/Documents/Lab5$
```

Test case 2



osboxes@osboxes: ~/Documents/Lab5

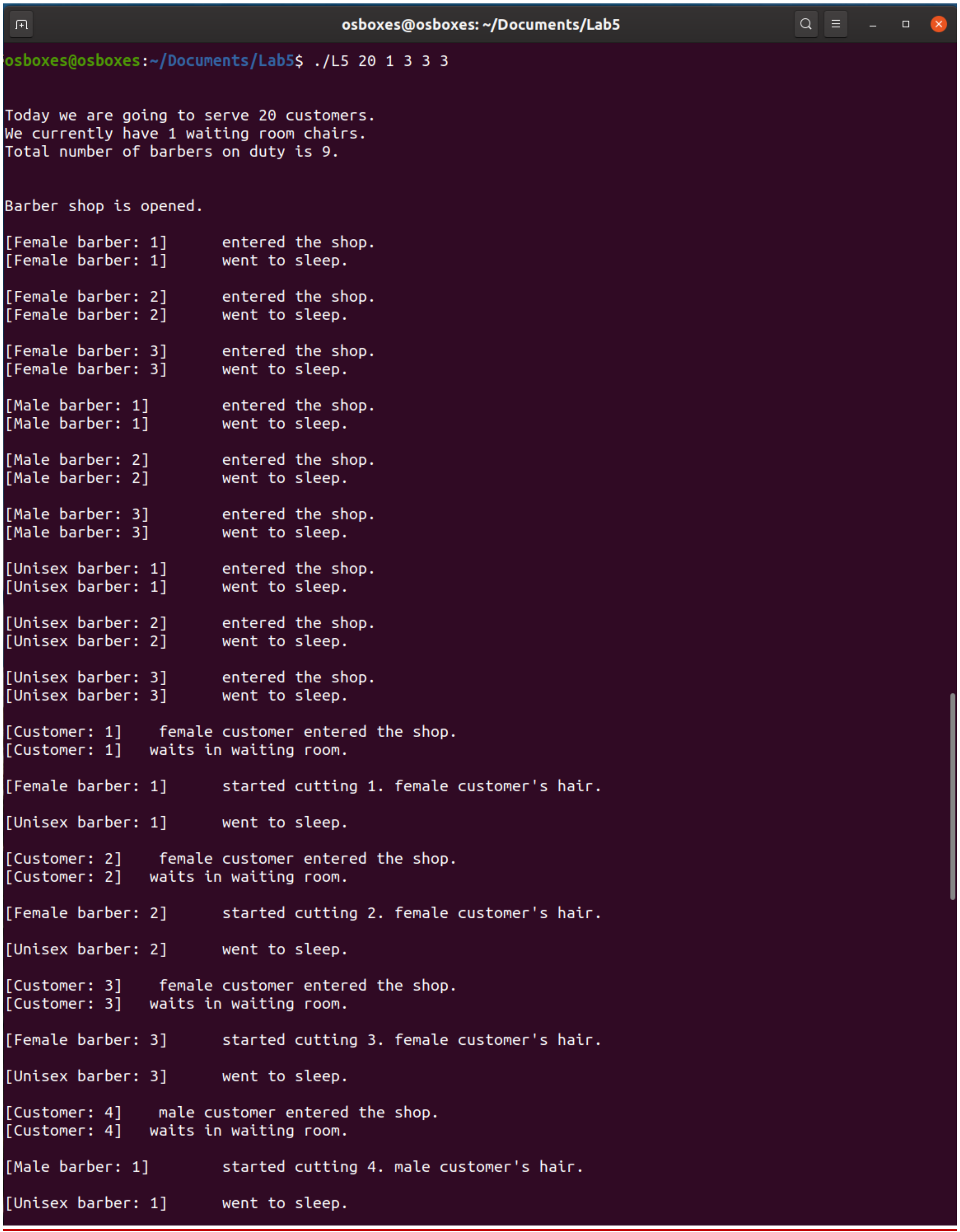
```
osboxes@osboxes:~/Documents/Lab5$ ./L5 10 5 2 2 3


Today we are going to serve 10 customers.
We currently have 5 waiting room chairs.
Total number of barbers on duty is 7.


Barber shop is opened.

[Female barber: 1]      entered the shop.
[Female barber: 1]      went to sleep.

[Female barber: 2]      entered the shop.
[Female barber: 2]      went to sleep.

[Male barber: 1]        entered the shop.
[Male barber: 1]        went to sleep.

[Male barber: 2]        entered the shop.
[Male barber: 2]        went to sleep.

[Unisex barber: 1]      entered the shop.
[Unisex barber: 1]      went to sleep.

[Unisex barber: 2]      entered the shop.
[Unisex barber: 2]      went to sleep.

[Unisex barber: 3]      entered the shop.
[Unisex barber: 3]      went to sleep.

[Customer: 1]    female customer entered the shop.
[Customer: 1]   waits in waiting room.

[Female barber: 1]      started cutting 1. female customer's hair.

[Unisex barber: 1]      went to sleep.

[Customer: 2]    female customer entered the shop.
[Customer: 2]   waits in waiting room.

[Female barber: 2]      started cutting 2. female customer's hair.

[Unisex barber: 2]      went to sleep.

[Customer: 3]    male customer entered the shop.
[Customer: 3]   waits in waiting room.

[Male barber: 1]        started cutting 3. male customer's hair.

[Unisex barber: 3]      went to sleep.

[Customer: 4]    male customer entered the shop.
[Customer: 4]   waits in waiting room.

[Male barber: 2]        started cutting 4. male customer's hair.

[Unisex barber: 1]      went to sleep.

[Customer: 5]    male customer entered the shop.
[Customer: 5]   waits in waiting room.

[Unisex barber: 2]      started cutting 5. male customer's hair.

[Customer: 6]    male customer entered the shop.
```



osboxes@osboxes: ~/Documents/Lab5

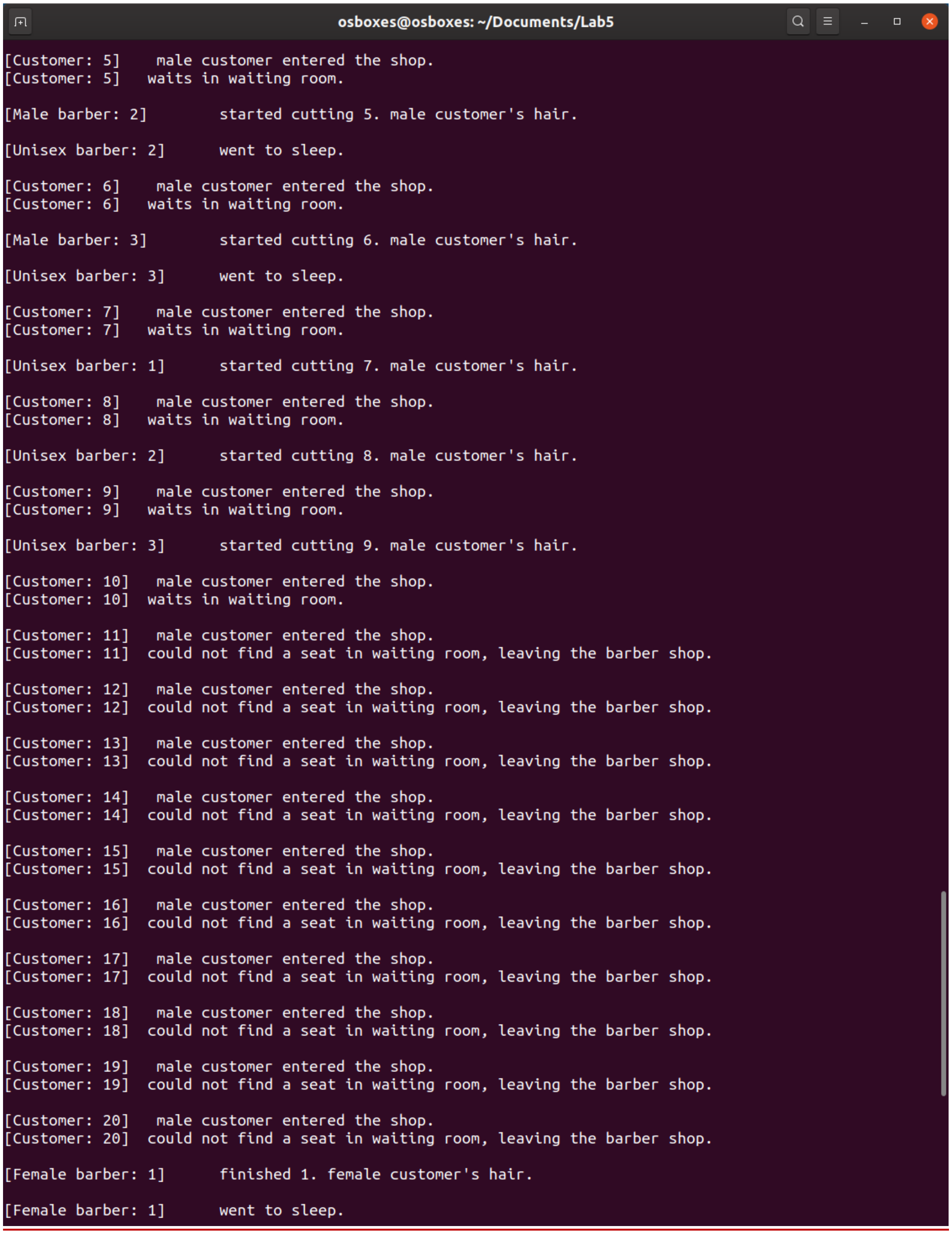
```
[Unisex barber: 2]      started cutting 5. male customer's hair.

[Customer: 6]    male customer entered the shop.
[Customer: 6]   waits in waiting room.

[Unisex barber: 3]      started cutting 6. male customer's hair.

[Customer: 7]    male customer entered the shop.
[Customer: 7]   waits in waiting room.

[Unisex barber: 1]      started cutting 7. male customer's hair.

[Customer: 8]    male customer entered the shop.
[Customer: 8]   waits in waiting room.

[Customer: 9]    male customer entered the shop.
[Customer: 9]   waits in waiting room.

[Customer: 10]   male customer entered the shop.
[Customer: 10]  waits in waiting room.

[Female barber: 1]      finished 1. female customer's hair.

[Female barber: 1]      went to sleep.

[Female barber: 2]      finished 2. female customer's hair.

[Female barber: 2]      went to sleep.

[Male barber: 1]        finished 3. male customer's hair.

[Male barber: 1]        started cutting 8. male customer's hair.

[Male barber: 2]        finished 4. male customer's hair.

[Male barber: 2]        started cutting 9. male customer's hair.

[Unisex barber: 2]      finished 5. male customer's hair.

[Unisex barber: 2]      started cutting 10. male customer's hair.

[Unisex barber: 3]      finished 6. male customer's hair.

[Unisex barber: 3]      went to sleep.

[Unisex barber: 1]      finished 7. male customer's hair.

[Unisex barber: 1]      went to sleep.

[Male barber: 1]        finished 8. male customer's hair.

[Male barber: 1]        went to sleep.

[Male barber: 2]        finished 9. male customer's hair.

[Male barber: 2]        went to sleep.

[Unisex barber: 2]      finished 10. male customer's hair.

[Unisex barber: 2]      went to sleep.


All customers were served. Barber shop is closed. Barbers left the shop.

osboxes@osboxes:~/Documents/Lab5$ 
```

Test case 3

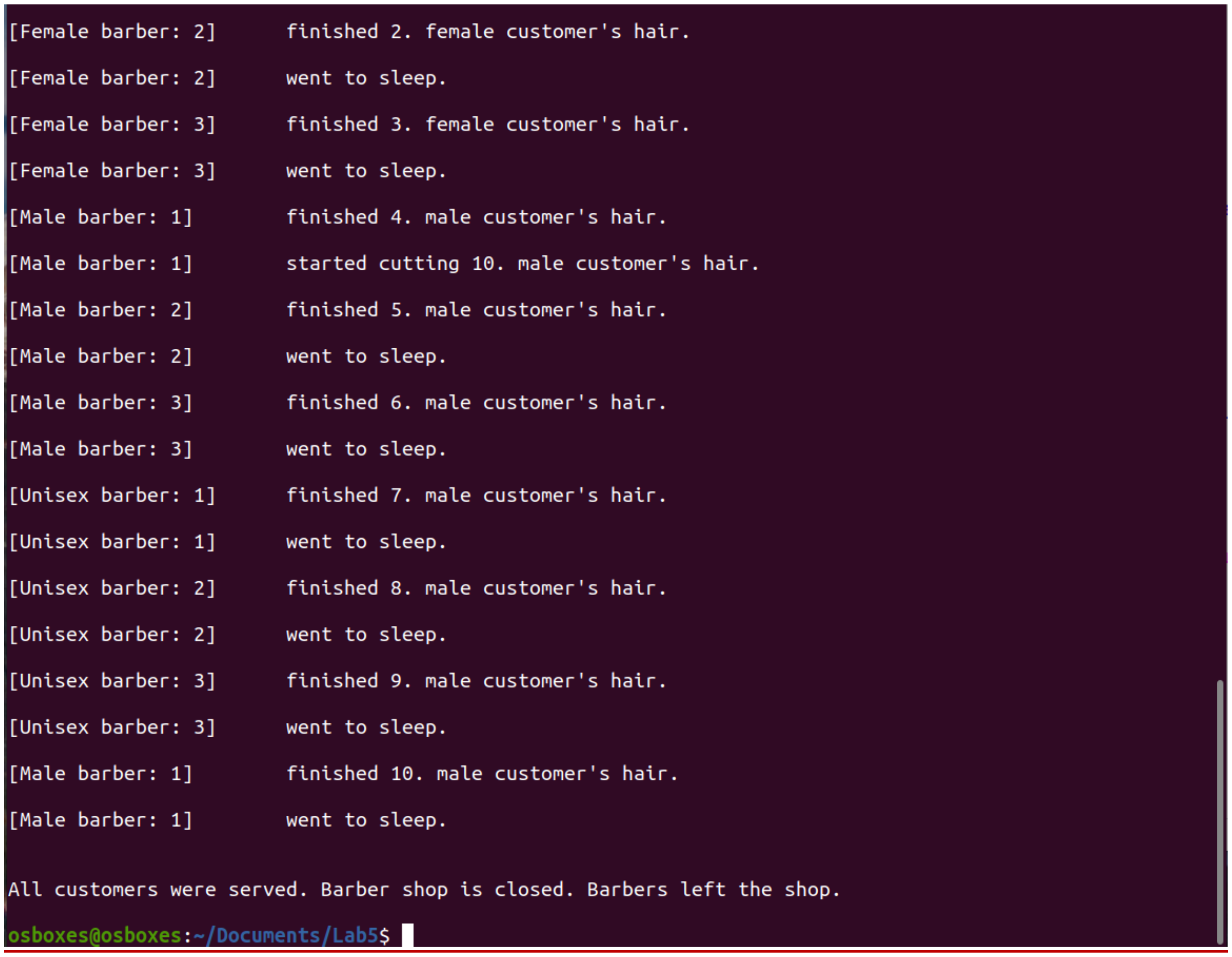


osboxes@osboxes: ~/Documents/Lab5

```
osboxes@osboxes:~/Documents/Lab5$ ./L5 20 1 3 3 3


Today we are going to serve 20 customers.
We currently have 1 waiting room chairs.
Total number of barbers on duty is 9.


Barber shop is opened.

[Female barber: 1]      entered the shop.
[Female barber: 1]      went to sleep.

[Female barber: 2]      entered the shop.
[Female barber: 2]      went to sleep.

[Female barber: 3]      entered the shop.
[Female barber: 3]      went to sleep.

[Male barber: 1]        entered the shop.
[Male barber: 1]        went to sleep.

[Male barber: 2]        entered the shop.
[Male barber: 2]        went to sleep.

[Male barber: 3]        entered the shop.
[Male barber: 3]        went to sleep.

[Unisex barber: 1]      entered the shop.
[Unisex barber: 1]      went to sleep.

[Unisex barber: 2]      entered the shop.
[Unisex barber: 2]      went to sleep.

[Unisex barber: 3]      entered the shop.
[Unisex barber: 3]      went to sleep.

[Customer: 1]    female customer entered the shop.
[Customer: 1]   waits in waiting room.

[Female barber: 1]      started cutting 1. female customer's hair.

[Unisex barber: 1]      went to sleep.

[Customer: 2]    female customer entered the shop.
[Customer: 2]   waits in waiting room.

[Female barber: 2]      started cutting 2. female customer's hair.

[Unisex barber: 2]      went to sleep.

[Customer: 3]    female customer entered the shop.
[Customer: 3]   waits in waiting room.

[Female barber: 3]      started cutting 3. female customer's hair.

[Unisex barber: 3]      went to sleep.

[Customer: 4]    male customer entered the shop.
[Customer: 4]   waits in waiting room.

[Male barber: 1]        started cutting 4. male customer's hair.

[Unisex barber: 1]      went to sleep.
```



```
[Customer: 5]    male customer entered the shop.
[Customer: 5]   waits in waiting room.

[Male barber: 2]        started cutting 5. male customer's hair.

[Unisex barber: 2]      went to sleep.

[Customer: 6]    male customer entered the shop.
[Customer: 6]   waits in waiting room.

[Male barber: 3]        started cutting 6. male customer's hair.

[Unisex barber: 3]      went to sleep.

[Customer: 7]    male customer entered the shop.
[Customer: 7]   waits in waiting room.

[Unisex barber: 1]      started cutting 7. male customer's hair.

[Customer: 8]    male customer entered the shop.
[Customer: 8]   waits in waiting room.

[Unisex barber: 2]      started cutting 8. male customer's hair.

[Customer: 9]    male customer entered the shop.
[Customer: 9]   waits in waiting room.

[Unisex barber: 3]      started cutting 9. male customer's hair.

[Customer: 10]   male customer entered the shop.
[Customer: 10]  waits in waiting room.

[Customer: 11]   male customer entered the shop.
[Customer: 11]  could not find a seat in waiting room, leaving the barber shop.

[Customer: 12]   male customer entered the shop.
[Customer: 12]  could not find a seat in waiting room, leaving the barber shop.

[Customer: 13]   male customer entered the shop.
[Customer: 13]  could not find a seat in waiting room, leaving the barber shop.

[Customer: 14]   male customer entered the shop.
[Customer: 14]  could not find a seat in waiting room, leaving the barber shop.

[Customer: 15]   male customer entered the shop.
[Customer: 15]  could not find a seat in waiting room, leaving the barber shop.

[Customer: 16]   male customer entered the shop.
[Customer: 16]  could not find a seat in waiting room, leaving the barber shop.

[Customer: 17]   male customer entered the shop.
[Customer: 17]  could not find a seat in waiting room, leaving the barber shop.

[Customer: 18]   male customer entered the shop.
[Customer: 18]  could not find a seat in waiting room, leaving the barber shop.

[Customer: 19]   male customer entered the shop.
[Customer: 19]  could not find a seat in waiting room, leaving the barber shop.

[Customer: 20]   male customer entered the shop.
[Customer: 20]  could not find a seat in waiting room, leaving the barber shop.

[Female barber: 1]      finished 1. female customer's hair.

[Female barber: 1]      went to sleep.
```


```
[Female barber: 2]      finished 2. female customer's hair.

[Female barber: 2]      went to sleep.

[Female barber: 3]      finished 3. female customer's hair.

[Female barber: 3]      went to sleep.

[Male barber: 1]        finished 4. male customer's hair.

[Male barber: 1]        started cutting 10. male customer's hair.

[Male barber: 2]        finished 5. male customer's hair.

[Male barber: 2]        went to sleep.

[Male barber: 3]        finished 6. male customer's hair.

[Male barber: 3]        went to sleep.

[Unisex barber: 1]      finished 7. male customer's hair.

[Unisex barber: 1]      went to sleep.

[Unisex barber: 2]      finished 8. male customer's hair.

[Unisex barber: 2]      went to sleep.

[Unisex barber: 3]      finished 9. male customer's hair.

[Unisex barber: 3]      went to sleep.

[Male barber: 1]        finished 10. male customer's hair.

[Male barber: 1]        went to sleep.

All customers were served. Barber shop is closed. Barbers left the shop.
osboxes@osboxes:~/Documents/Lab5$
```