GEBZE TECHNICAL UNIVERSITY
COMPUTER ENGINEERING FACULTY

CSE 344 SYSTEM PROGRAMMING

HOMEWORK 3 REPORT

ENES AYSU
1901042671

Program Description

This project presents a multithreaded car parking management system designed to handle automobiles and pickups using synchronization mechanisms to ensure orderly parking and retrieval. The system is modeled with a limited number of parking spots for automobiles and pickups, managed by separate attendant threads. Key components of the implementation include:

- Parking Spots Management: The system maintains counters for available parking spots, with 8 spots for automobiles and 4 spots for pickups.
- Temporary Lot Coordination: A temporary lot is used to control vehicle entry into the parking system. Only one vehicle can occupy this lot at any given time, ensuring synchronized entry.
- Thread Synchronization: Semaphores are employed to signal the arrival of new vehicles and to coordinate the parking and retrieval process by attendants. Mutexes protect shared counters and critical sections, preventing race conditions.
- Vehicle Owner Threads: Each vehicle owner is represented by a thread that attempts to park in the temporary lot. If the lot is free, the thread signals the appropriate attendant and waits for parking confirmation. If the lot is occupied, the thread exits, simulating the owner's departure due to no available spots.
- Attendant Threads: Separate threads for automobile and pickup attendants wait for signals from vehicle owners. Upon receiving a signal, attendants check for available spots, park the vehicle, and update the spot counters. Attendants also create additional threads to handle the duration of the vehicle's stay in the parking lot.

The implementation ensures efficient and synchronized management of parking operations, highlighting the use of concurrency primitives in a real-world simulation of a car parking system.

Function and Processes

Global Variables and Initializations

Shared Counters and Semaphores:

- int mFree_automobile (Keeps track of the number of available automobile spots)
- int mFree_pickup (Keeps track of the number of available pickup spots)
- int temporary_lot (Indicates whether the temporary lot is occupied (1 if occupied, 0 if not))

Semaphores:

- sem_t newAutomobile (Signals the arrival of a new automobile)
- sem_t newPickup (Signals the arrival of a new pickup)
- sem_t inChargeforAutomobile (Signals that an automobile has been parked)
- sem_t inChargeforPickup (Signals that a pickup has been parked)

Mutexes:

- pthread_mutex_t automobile_mutex (Protects access to mFree_automobile)
- pthread_mutex_t pickup_mutex (Protects access to mFree_pickup)
- pthread_mutex_t temporary_lot_mutex (Protects access to temporary_lot)

carOwner():

Function that simulates the behavior of a car owner.

- It locks the temporary_lot_mutex to check if the temporary lot is free.
- If the lot is free, it posts the appropriate semaphore (newAutomobile or newPickup) to signal the arrival of a vehicle and waits for the attendant to park the car.
- If the lot is not free, the owner leaves due to no available spots.

carStayHandler():

Function that handles the duration of a vehicle's stay in the parking lot.

- It sleeps for 10 seconds to simulate the parking duration.
- After the sleep, it increments the appropriate spot counter (mFree_automobile or mFree_pickup) and prints a message indicating the vehicle has left.

carAttendant():

Function that simulates the behavior of a car attendant.

- It checks if it needs to exit by locking exit_mutex and checking the exitAttendants flag.
- If not exiting, it waits on the appropriate semaphore (newAutomobile or newPickup) to handle new arrivals.
- It locks the automobile_mutex or pickup_mutex to check and update the number of available spots.
- If a spot is available, it parks the vehicle, decreases the spot count, and posts the appropriate semaphore (inChargeforAutomobile or inChargeforPickup).
- It then creates a new thread to handle the vehicle's stay duration.

main():

Initializes the semaphores and creates two car attendant threads.

- Simulates the arrival of car owners by creating 12 carOwner threads, with a 1-second delay between each.
- Waits for all car attendant threads to finish.

Execution Steps of the Program

1) Initialization:

- Initialize shared counters for parking spots: “mFree_automobile” (8 spots), “mFree_pickup” (4 spots), and “temporary_lot” (0 indicating the lot is free).
- Initialize semaphores:
    - “newAutomobile” and “newPickup” to signal new arrivals of automobiles and pickups, respectively.
    - “inChargeforAutomobile” and “inChargeforPickup” for attendants to signal the completion of parking.
- Initialize mutexes to protect shared resources (automobile_mutex, pickup_mutex, temporary_lot_mutex).

2) Create Attendant Threads:

- Create two threads for attendants: one for automobiles and one for pickups. Each thread continuously waits for a signal of new vehicle arrivals and parks the vehicles accordingly.

3) Simulate Car Owners Arriving:

- Create 12 threads representing vehicle owners. Each thread:
    - Randomly decides the type of vehicle (automobile or pickup).
    - Tries to park in the temporary lot. If the lot is free, signals the appropriate attendant and waits for confirmation. If the lot is occupied, exits and simulates the owner leaving due to no available spots.

4) Attendant Thread Operations:

- Each attendant thread waits for a signal (newAutomobile or newPickup).
- Checks the availability of parking spots and the temporary lot status.
- If a spot is available, parks the vehicle (decrements the respective counter) and signals the vehicle owner.
- Creates a new thread to handle the vehicle's stay duration, which:
    - Sleeps for a fixed duration (10 seconds).
    - Upon waking up, increments the respective counter, indicating that the spot is now free.

5) Wait for Attendant Threads to Finish:

- Main thread waits for all attendant threads to complete using pthread_detach with while loop.

6) Cleanup:

- Destroy all semaphores.
- Destroy all mutexes and the condition variable.

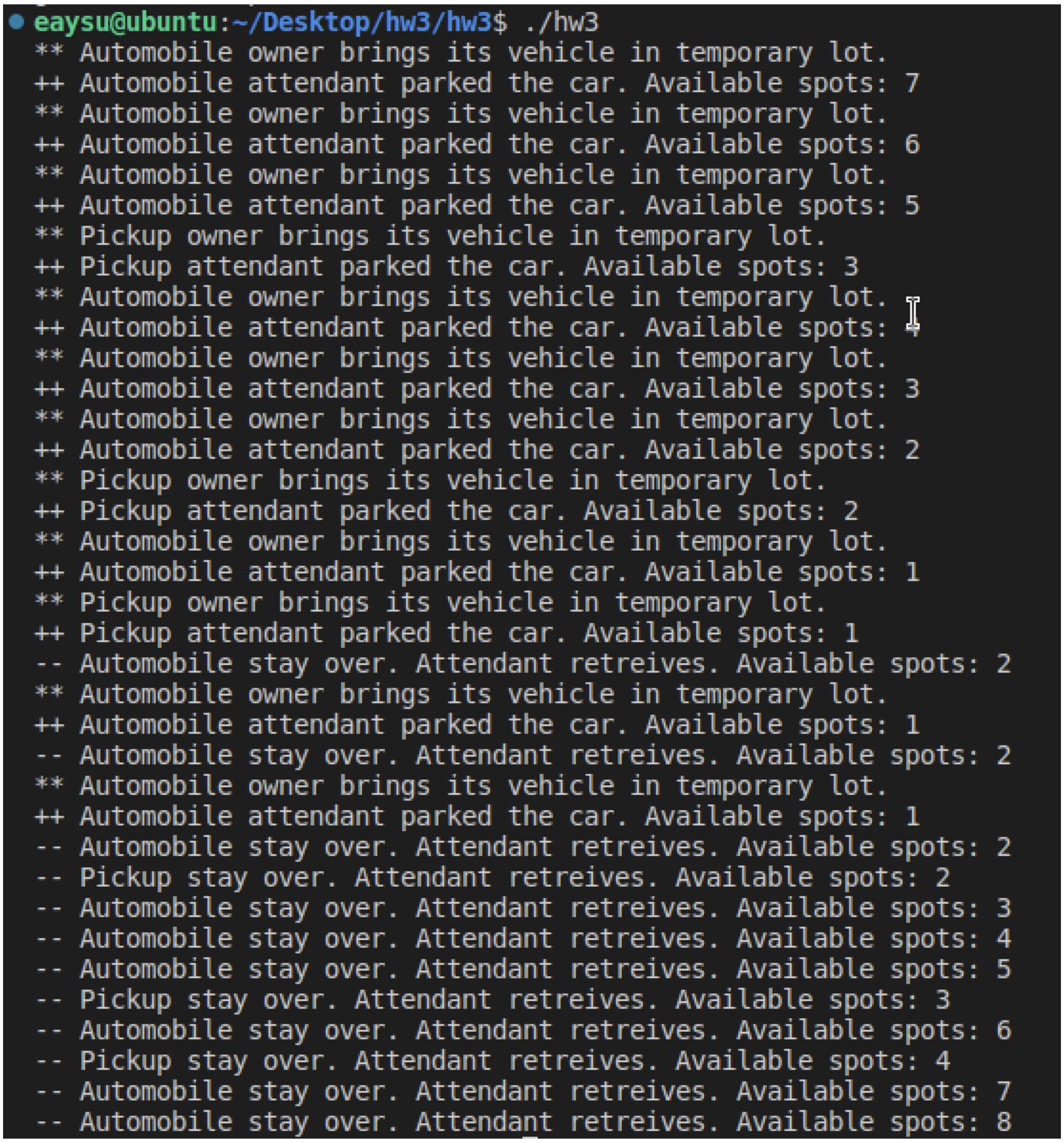
Tests

For Compilation

$ make

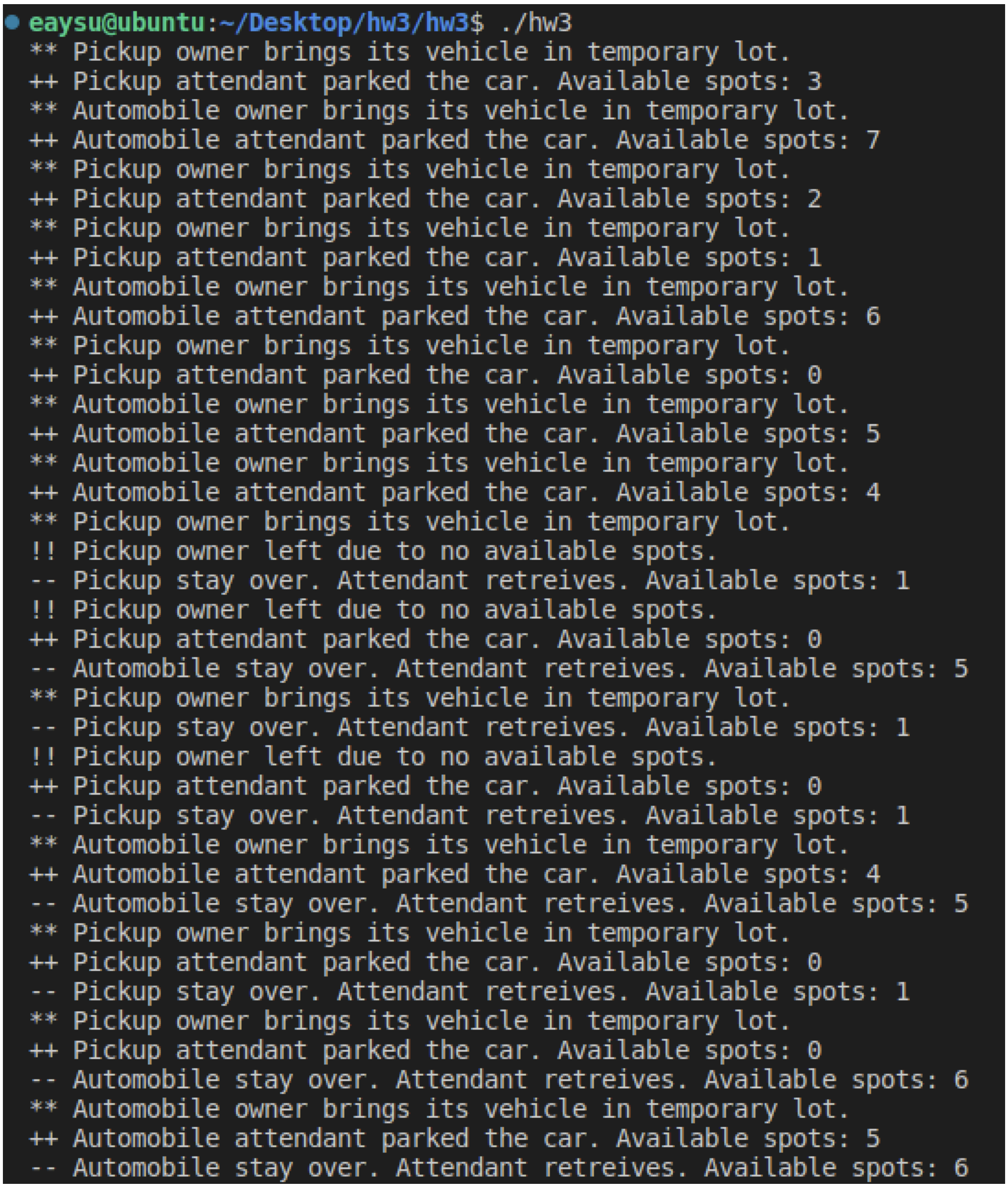
$ ./hw3

Execution 1 (Car Owners (12) comes with 1 second interval)

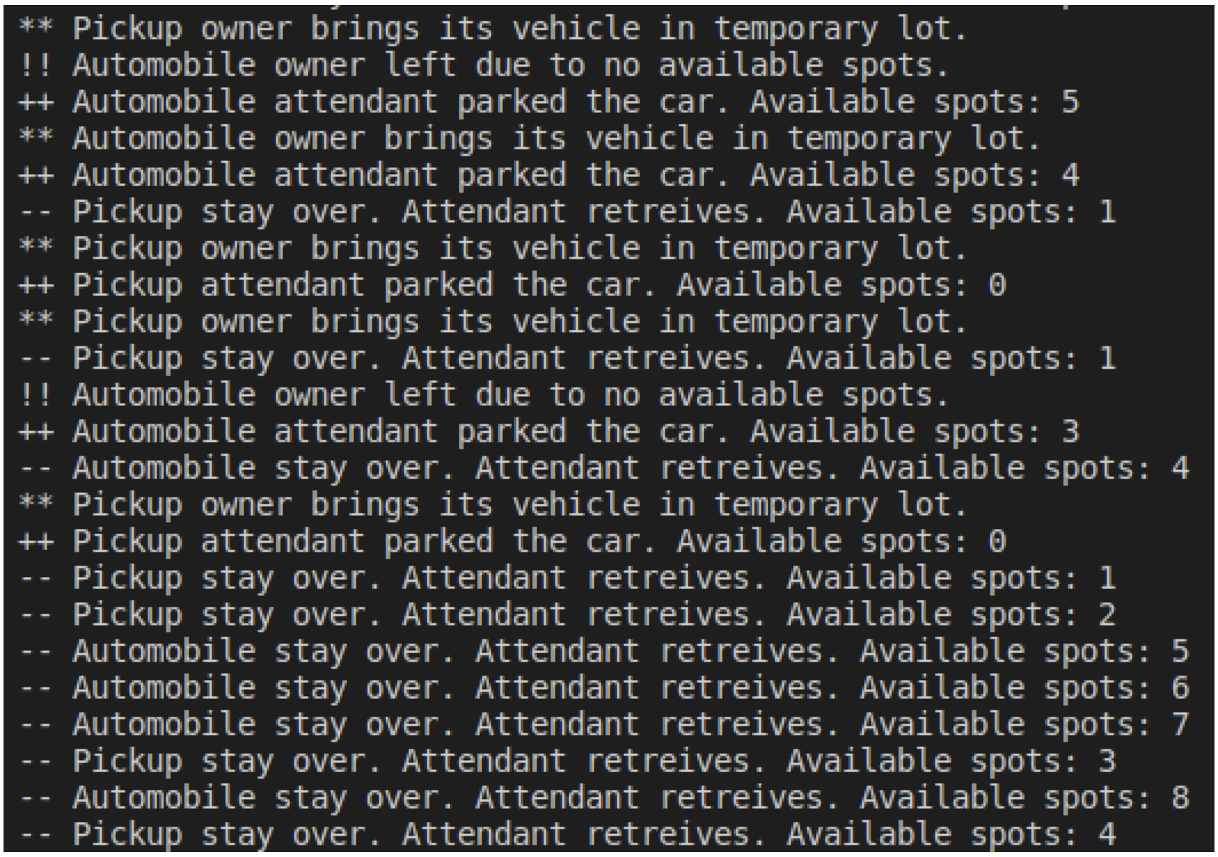


```
eaysu@ubuntu:~/Desktop/hw3/hw3$ ./hw3
** Automobile owner brings its vehicle in temporary lot.
++ Automobile attendant parked the car. Available spots: 7
** Automobile owner brings its vehicle in temporary lot.
++ Automobile attendant parked the car. Available spots: 6
** Automobile owner brings its vehicle in temporary lot.
++ Automobile attendant parked the car. Available spots: 5
** Pickup owner brings its vehicle in temporary lot.
++ Pickup attendant parked the car. Available spots: 3
** Automobile owner brings its vehicle in temporary lot.
++ Automobile attendant parked the car. Available spots: 4
** Automobile owner brings its vehicle in temporary lot.
++ Automobile attendant parked the car. Available spots: 3
** Automobile owner brings its vehicle in temporary lot.
++ Automobile attendant parked the car. Available spots: 2
** Pickup owner brings its vehicle in temporary lot.
++ Pickup attendant parked the car. Available spots: 2
** Automobile owner brings its vehicle in temporary lot.
++ Automobile attendant parked the car. Available spots: 1
** Pickup owner brings its vehicle in temporary lot.
++ Pickup attendant parked the car. Available spots: 1
-- Automobile stay over. Attendant retreives. Available spots: 2
** Automobile owner brings its vehicle in temporary lot.
++ Automobile attendant parked the car. Available spots: 1
-- Automobile stay over. Attendant retreives. Available spots: 2
** Automobile owner brings its vehicle in temporary lot.
++ Automobile attendant parked the car. Available spots: 1
-- Automobile stay over. Attendant retreives. Available spots: 2
-- Pickup stay over. Attendant retreives. Available spots: 2
-- Automobile stay over. Attendant retreives. Available spots: 3
-- Automobile stay over. Attendant retreives. Available spots: 4
-- Automobile stay over. Attendant retreives. Available spots: 5
-- Pickup stay over. Attendant retreives. Available spots: 3
-- Automobile stay over. Attendant retreives. Available spots: 6
-- Pickup stay over. Attendant retreives. Available spots: 4
-- Automobile stay over. Attendant retreives. Available spots: 7
-- Automobile stay over. Attendant retreives. Available spots: 8
```

Execution 2 (Car Owners (24) comes with 1 second interval)

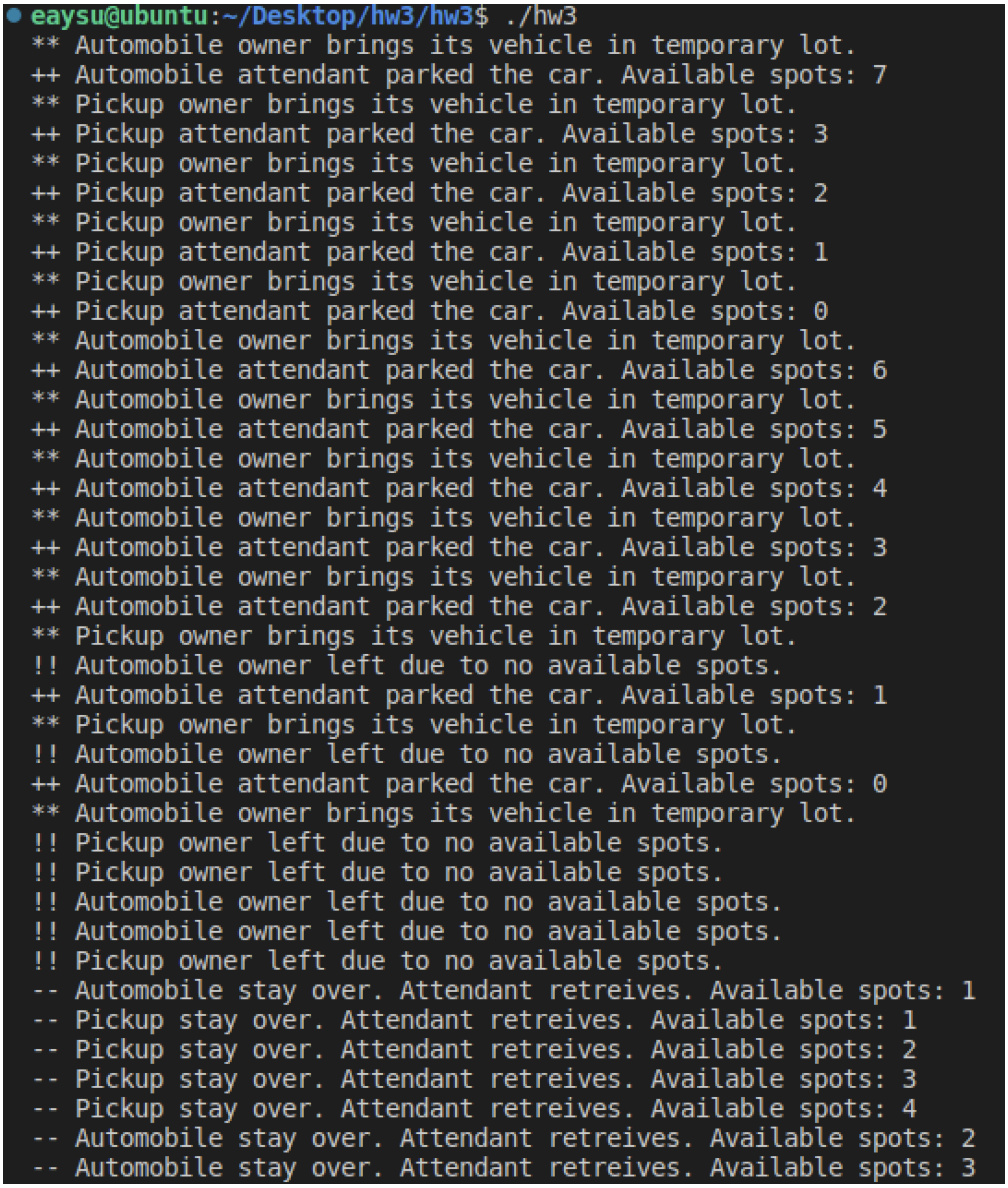


```
eaysu@ubuntu:~/Desktop/hw3/hw3$ ./hw3
** Pickup owner brings its vehicle in temporary lot.
++ Pickup attendant parked the car. Available spots: 3
** Automobile owner brings its vehicle in temporary lot.
++ Automobile attendant parked the car. Available spots: 7
** Pickup owner brings its vehicle in temporary lot.
++ Pickup attendant parked the car. Available spots: 2
** Pickup owner brings its vehicle in temporary lot.
++ Pickup attendant parked the car. Available spots: 1
** Automobile owner brings its vehicle in temporary lot.
++ Automobile attendant parked the car. Available spots: 6
** Pickup owner brings its vehicle in temporary lot.
++ Pickup attendant parked the car. Available spots: 0
** Automobile owner brings its vehicle in temporary lot.
++ Automobile attendant parked the car. Available spots: 5
** Automobile owner brings its vehicle in temporary lot.
++ Automobile attendant parked the car. Available spots: 4
** Pickup owner brings its vehicle in temporary lot.
!! Pickup owner left due to no available spots.
-- Pickup stay over. Attendant retreives. Available spots: 1
!! Pickup owner left due to no available spots.
++ Pickup attendant parked the car. Available spots: 0
-- Automobile stay over. Attendant retreives. Available spots: 5
** Pickup owner brings its vehicle in temporary lot.
-- Pickup stay over. Attendant retreives. Available spots: 1
!! Pickup owner left due to no available spots.
++ Pickup attendant parked the car. Available spots: 0
-- Pickup stay over. Attendant retreives. Available spots: 1
** Automobile owner brings its vehicle in temporary lot.
++ Automobile attendant parked the car. Available spots: 4
-- Automobile stay over. Attendant retreives. Available spots: 5
** Pickup owner brings its vehicle in temporary lot.
++ Pickup attendant parked the car. Available spots: 0
-- Pickup stay over. Attendant retreives. Available spots: 1
** Pickup owner brings its vehicle in temporary lot.
++ Pickup attendant parked the car. Available spots: 0
-- Automobile stay over. Attendant retreives. Available spots: 6
** Automobile owner brings its vehicle in temporary lot.
++ Automobile attendant parked the car. Available spots: 5
-- Automobile stay over. Attendant retreives. Available spots: 6
```

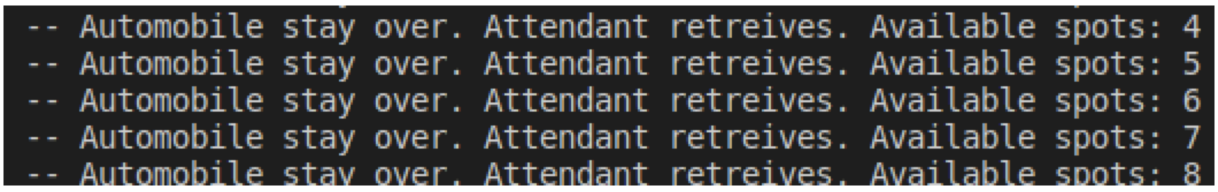


```
** Pickup owner brings its vehicle in temporary lot.
!! Automobile owner left due to no available spots.
++ Automobile attendant parked the car. Available spots: 5
** Automobile owner brings its vehicle in temporary lot.
++ Automobile attendant parked the car. Available spots: 4
-- Pickup stay over. Attendant retreives. Available spots: 1
** Pickup owner brings its vehicle in temporary lot.
++ Pickup attendant parked the car. Available spots: 0
** Pickup owner brings its vehicle in temporary lot.
-- Pickup stay over. Attendant retreives. Available spots: 1
!! Automobile owner left due to no available spots.
++ Automobile attendant parked the car. Available spots: 3
-- Automobile stay over. Attendant retreives. Available spots: 4
** Pickup owner brings its vehicle in temporary lot.
++ Pickup attendant parked the car. Available spots: 0
-- Pickup stay over. Attendant retreives. Available spots: 1
-- Pickup stay over. Attendant retreives. Available spots: 2
-- Automobile stay over. Attendant retreives. Available spots: 5
-- Automobile stay over. Attendant retreives. Available spots: 6
-- Automobile stay over. Attendant retreives. Available spots: 7
-- Pickup stay over. Attendant retreives. Available spots: 3
-- Automobile stay over. Attendant retreives. Available spots: 8
-- Pickup stay over. Attendant retreives. Available spots: 4
```

Execution 2 (Car Owners (20) comes with 1 second interval and park duration (20))



```
● eaysu@ubuntu:~/Desktop/hw3/hw3$ ./hw3
 ** Automobile owner brings its vehicle in temporary lot.
 ++ Automobile attendant parked the car. Available spots: 7
 ** Pickup owner brings its vehicle in temporary lot.
 ++ Pickup attendant parked the car. Available spots: 3
 ** Pickup owner brings its vehicle in temporary lot.
 ++ Pickup attendant parked the car. Available spots: 2
 ** Pickup owner brings its vehicle in temporary lot.
 ++ Pickup attendant parked the car. Available spots: 1
 ** Pickup owner brings its vehicle in temporary lot.
 ++ Pickup attendant parked the car. Available spots: 0
 ** Automobile owner brings its vehicle in temporary lot.
 ++ Automobile attendant parked the car. Available spots: 6
 ** Automobile owner brings its vehicle in temporary lot.
 ++ Automobile attendant parked the car. Available spots: 5
 ** Automobile owner brings its vehicle in temporary lot.
 ++ Automobile attendant parked the car. Available spots: 4
 ** Automobile owner brings its vehicle in temporary lot.
 ++ Automobile attendant parked the car. Available spots: 3
 ** Automobile owner brings its vehicle in temporary lot.
 ++ Automobile attendant parked the car. Available spots: 2
 ** Pickup owner brings its vehicle in temporary lot.
 !! Automobile owner left due to no available spots.
 ++ Automobile attendant parked the car. Available spots: 1
 ** Pickup owner brings its vehicle in temporary lot.
 !! Automobile owner left due to no available spots.
 ++ Automobile attendant parked the car. Available spots: 0
 ** Automobile owner brings its vehicle in temporary lot.
 !! Pickup owner left due to no available spots.
 !! Pickup owner left due to no available spots.
 !! Automobile owner left due to no available spots.
 !! Automobile owner left due to no available spots.
 !! Pickup owner left due to no available spots.
 -- Automobile stay over. Attendant retreives. Available spots: 1
 -- Pickup stay over. Attendant retreives. Available spots: 1
 -- Pickup stay over. Attendant retreives. Available spots: 2
 -- Pickup stay over. Attendant retreives. Available spots: 3
 -- Pickup stay over. Attendant retreives. Available spots: 4
 -- Automobile stay over. Attendant retreives. Available spots: 2
 -- Automobile stay over. Attendant retreives. Available spots: 3
```



```
-- Automobile stay over. Attendant retreives. Available spots: 4
-- Automobile stay over. Attendant retreives. Available spots: 5
-- Automobile stay over. Attendant retreives. Available spots: 6
-- Automobile stay over. Attendant retreives. Available spots: 7
-- Automobile stay over. Attendant retreives. Available spots: 8
```

Summary

The project successfully simulates a car parking system, providing insights into the challenges and solutions associated with concurrent programming. The use of threads and synchronization primitives demonstrates how to manage shared resources and coordinate operations in a multithreaded environment. This implementation can serve as a foundation for more complex parking management systems or other applications requiring synchronized resource management.