CSE 344
System Programming
3rd Assignment
Report

Author

Mert Emir ŞEKER

200104004085

Date

16.05.2024

Table Of Contents

1. **Makefile** **3**

2. **Makefile Commands** **3**

3. **How to run the code?** **3**

4. **Problem-solving approach and program functionality** **4**

5. **Code Explanation** **5**

5.1 Global variables and synchronization primitives 5

5.2 Initialize_semaphores 5

5.3 carOwner 6

5.4 carAttendant 7

5.5 carExit 8

5.6 destroy_semaphores 9

5.7 check_all_spots_full 9

5.8 main 10

5.9 signal_handler 11

6. **Example Outputs** **12**

6.1 Automobile or pickup arrives 12

6.2 Parking temporary spots 13

6.3 Moving temporary to permanent by valet 14

6.4 Pickup or automobile exits the temporary park spot 15

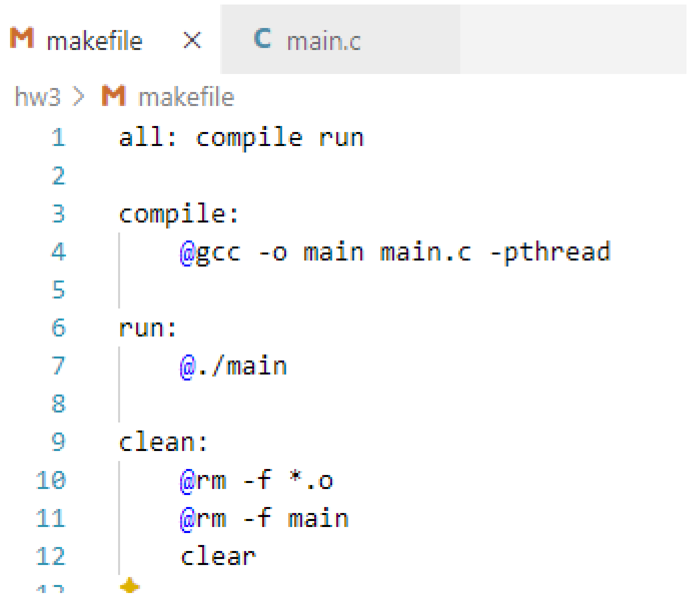
6.5 Full output 16

6.5.1 Part 1 16

6.5.2 Part 2 17

6.5.3 Part 3 18

1. Makefile:



```
all: compile run

compile:
	@gcc -o main main.c -pthread

run:
	@./main

clean:
	@rm -f *.o
	@rm -f main
	clear
```

2. Makefile Commands:

All: Compiles and runs the code.

***Compile**:* Compiles the code.

Run: Runs the code.

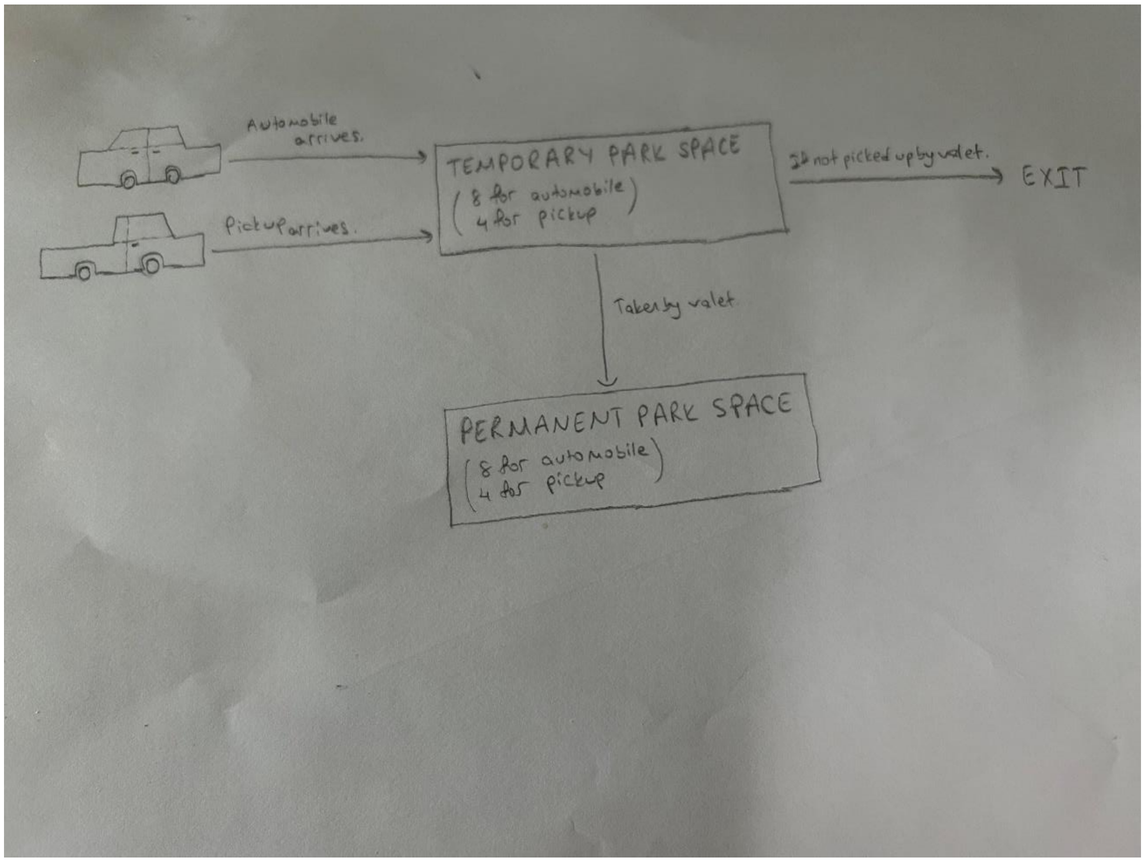
Clean: Clears terminal.

3. How to run the code?

To run the program, you can write **make all** in terminal or you can do it like this first **make compile** and than **make run**.

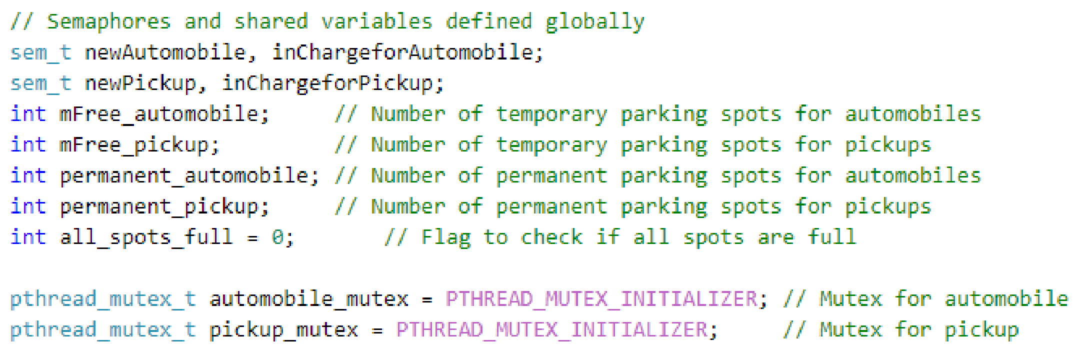
4. Problem-solving approach and program functionality:

When I first read the assignment PDF, I thought of simulating a parking lot with 24 parking spaces. Out of these 24 parking spaces, 12 are temporary parking spots (8 for automobiles and 4 for pickups) and 12 are permanent parking spots (8 for automobiles and 4 for pickups). In our program, valets move automobiles or pickups from the temporary parking area to the permanent parking area after a certain period of time. If automobiles or pickups in the temporary parking area are not taken by the valet, they will exit the temporary parking area after a random period between 1 and 5 seconds. Automobiles or pickups that are moved to the permanent parking area do not leave the parking lot. The arrival of automobiles or pickups in the temporary parking area is determined by generating a random number; if the number is 0, a pickup arrives, and if the number is 1, an automobile arrives. If the permanent parking areas are full, automobiles or pickups park in the temporary parking area. If all 24 parking spaces in the program are full, the program terminates. Until these 24 spaces are full, automobiles or pickups continue to arrive. I am implementing this program termination because if it doesn't terminate, it will run indefinitely and keep printing the same messages as automobiles or pickups continue to arrive constantly.



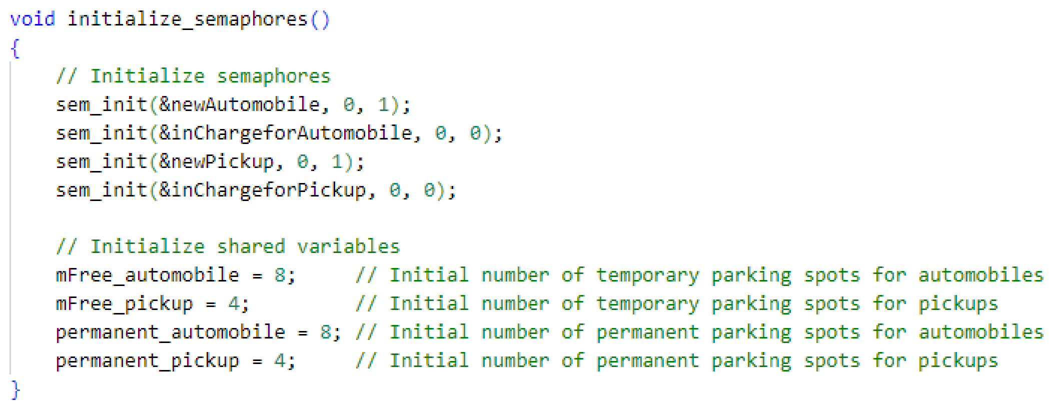
5. Code Explanation:

5.1. Global variables and synchronization primitives:
The global variables and synchronization primitives needed to control the parking lot simulation are defined in this section. Access to temporary parking spaces is managed by the semaphores (sem_t newAutomobile, inChargeforAutomobile, newPickup, inChargeforPickup), which also notify valets to shift cars to permanent locations. The number of temporary and permanent parking spaces for cars and pickups is kept track of by the counters (int mFree_automobile, mFree_pickup, permanent_automobile, permanent_pickup). Mutexes (pthread_mutex_t automobile_mutex, pickup_mutex) guard access to shared variables, guaranteeing synchronized access and averting race situations. The flag int all_spots_full shows if all spots are full.



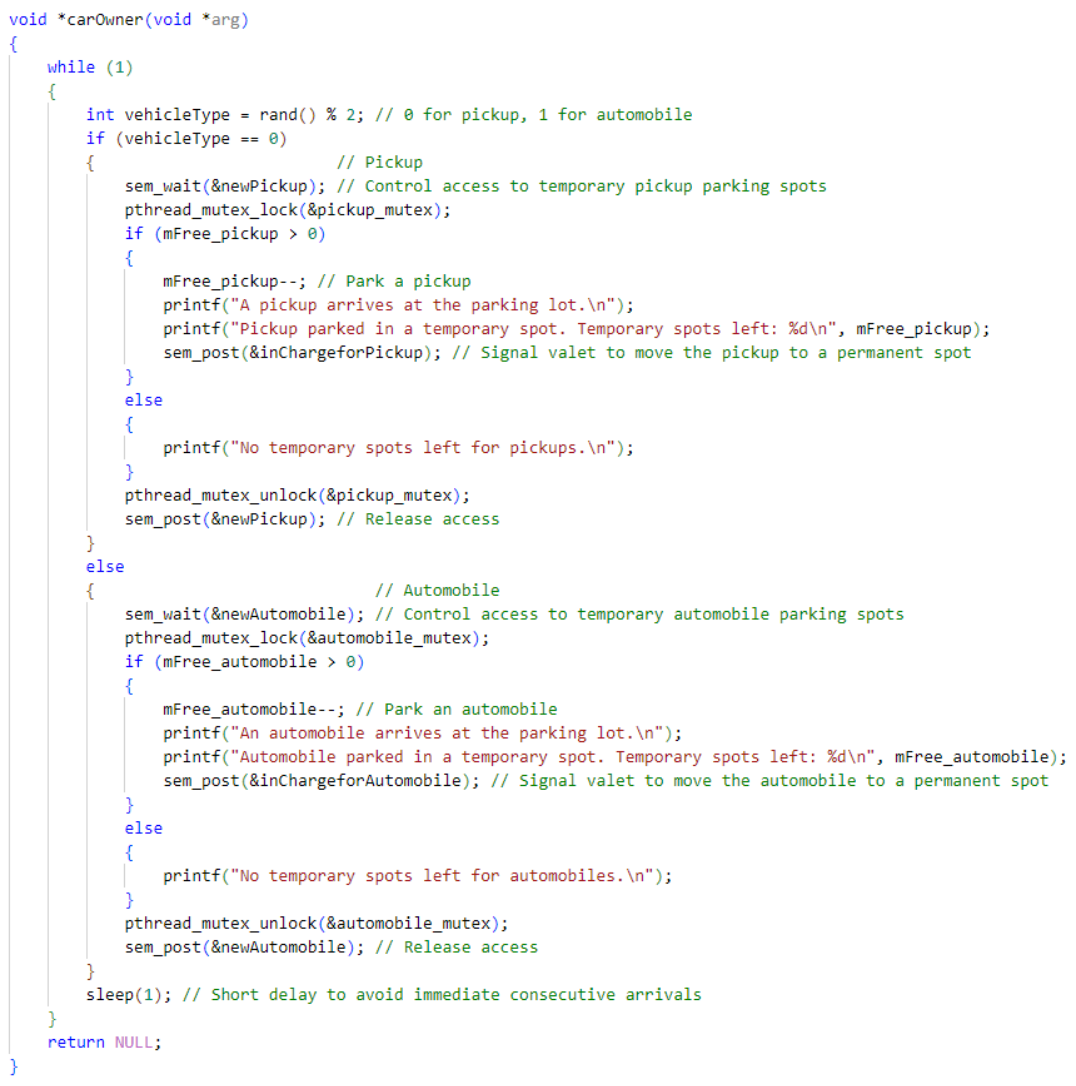
```c
// Semaphores and shared variables defined globally
sem_t newAutomobile, inChargeforAutomobile;
sem_t newPickup, inChargeforPickup;
int mFree_automobile;     // Number of temporary parking spots for automobiles
int mFree_pickup;         // Number of temporary parking spots for pickups
int permanent_automobile; // Number of permanent parking spots for automobiles
int permanent_pickup;     // Number of permanent parking spots for pickups
int all_spots_full = 0;       // Flag to check if all spots are full

pthread_mutex_t automobile_mutex = PTHREAD_MUTEX_INITIALIZER; // Mutex for automobile
pthread_mutex_t pickup_mutex = PTHREAD_MUTEX_INITIALIZER;     // Mutex for pickup
```

5.2. initialize_semaphores:
The shared variables used in the parking lot simulation are initialized and the semaphores are set up by the initialize_semaphores function. The following is how the semaphores are initialized: Cars and trucks are given access to temporary parking spaces by sem_init(&newAutomobile, 0, 1) and sem_init(&newPickup, 0, 1), respectively, while the valets are signaled by sem_init(&inChargeforAutomobile, 0, 0) and sem_init(&inChargeforPickup, 0, 0) to relocate the vehicles to permanent locations. The initial values of the common variables mFree_automobile, mFree_pickup, permanent_automobile, and permanent_pickup indicate the number of temporary and permanent parking spaces that are available, which is 8 for cars and 4 for pickups in both categories. This method makes sure that before any parking lot actions start, all semaphores and counters are properly initialized.

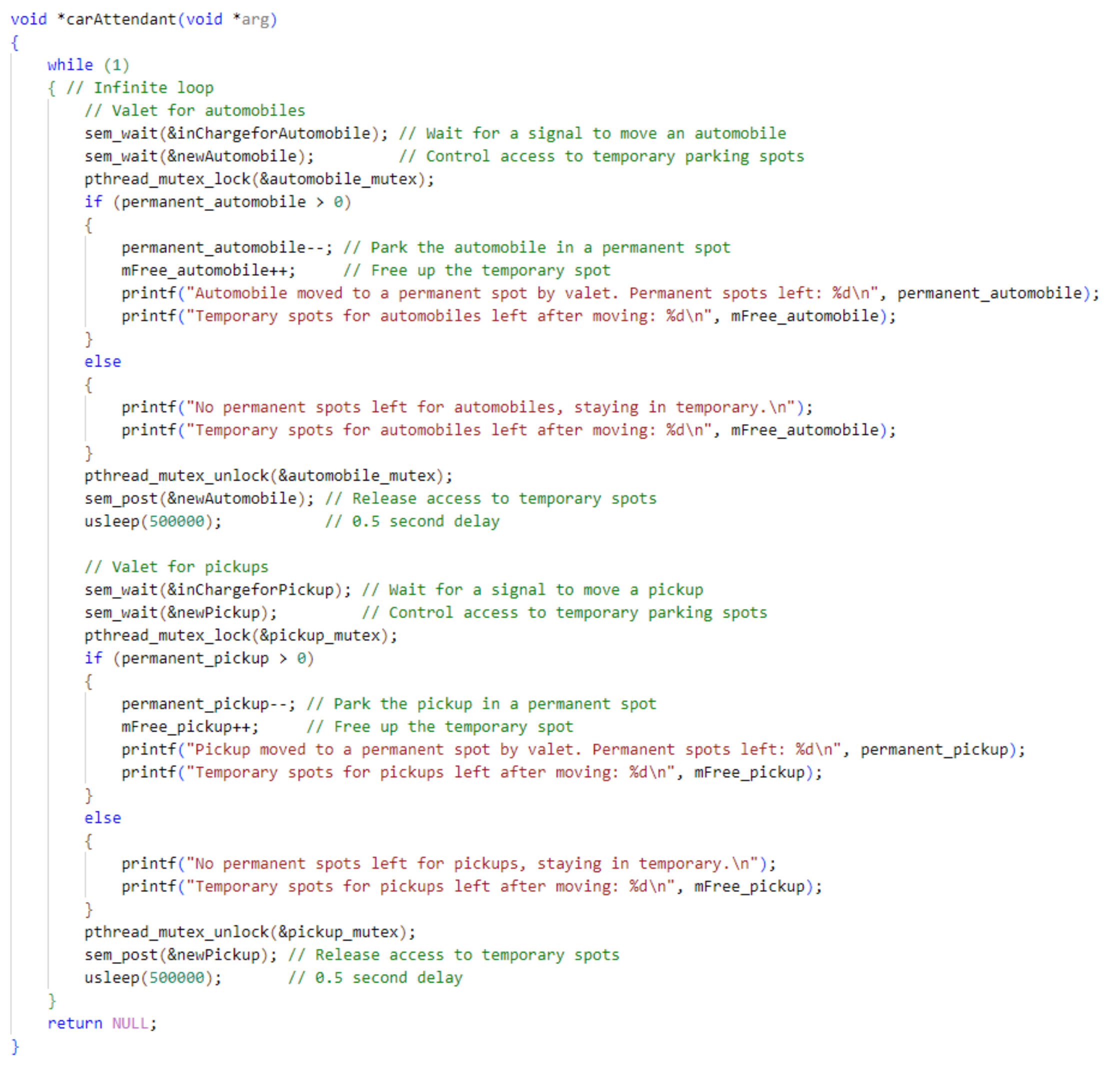


```c
void initialize_semaphores()
{
    // Initialize semaphores
    sem_init(&newAutomobile, 0, 1);
    sem_init(&inChargeforAutomobile, 0, 0);
    sem_init(&newPickup, 0, 1);
    sem_init(&inChargeforPickup, 0, 0);

    // Initialize shared variables
    mFree_automobile = 8;     // Initial number of temporary parking spots for automobiles
    mFree_pickup = 4;         // Initial number of temporary parking spots for pickups
    permanent_automobile = 8; // Initial number of permanent parking spots for automobiles
    permanent_pickup = 4;     // Initial number of permanent parking spots for pickups
}
```

5.3 carOwner: The carOwner function runs indefinitely to continually create automobiles, simulating the arrival of vehicle owners at the parking lot. It is decided at random whether each vehicle is a pickup or a car (int vehicleType = rand() % 2). When a pickup occurs, the thread locks the pickup mutex, acquires the semaphore that governs access to temporary pickup locations (sem_wait(&newPickup)), and determines if a temporary space is available. In such case, it prints a notice, parks the pickup by decreasing mFree_pickup, and instructs a valet to transfer the pickup to a permanent location (sem_post(&inChargeforPickup)). Next, the semaphore and mutex are freed. Comparably, for an automobile, the thread locks the automobile_mutex, obtains the semaphore for temporary automobile spots (sem_wait(&newAutomobile)), and looks for spaces that are available.When a place becomes available, it publishes a notice, signals a valet (sem_post(&inChargeforAutomobile)), and parks the car by decreasing mFree_automobile. Next, the semaphore and mutex are freed. The thread simulates the time it takes for trucks to arrive by sleeping briefly after each operation.

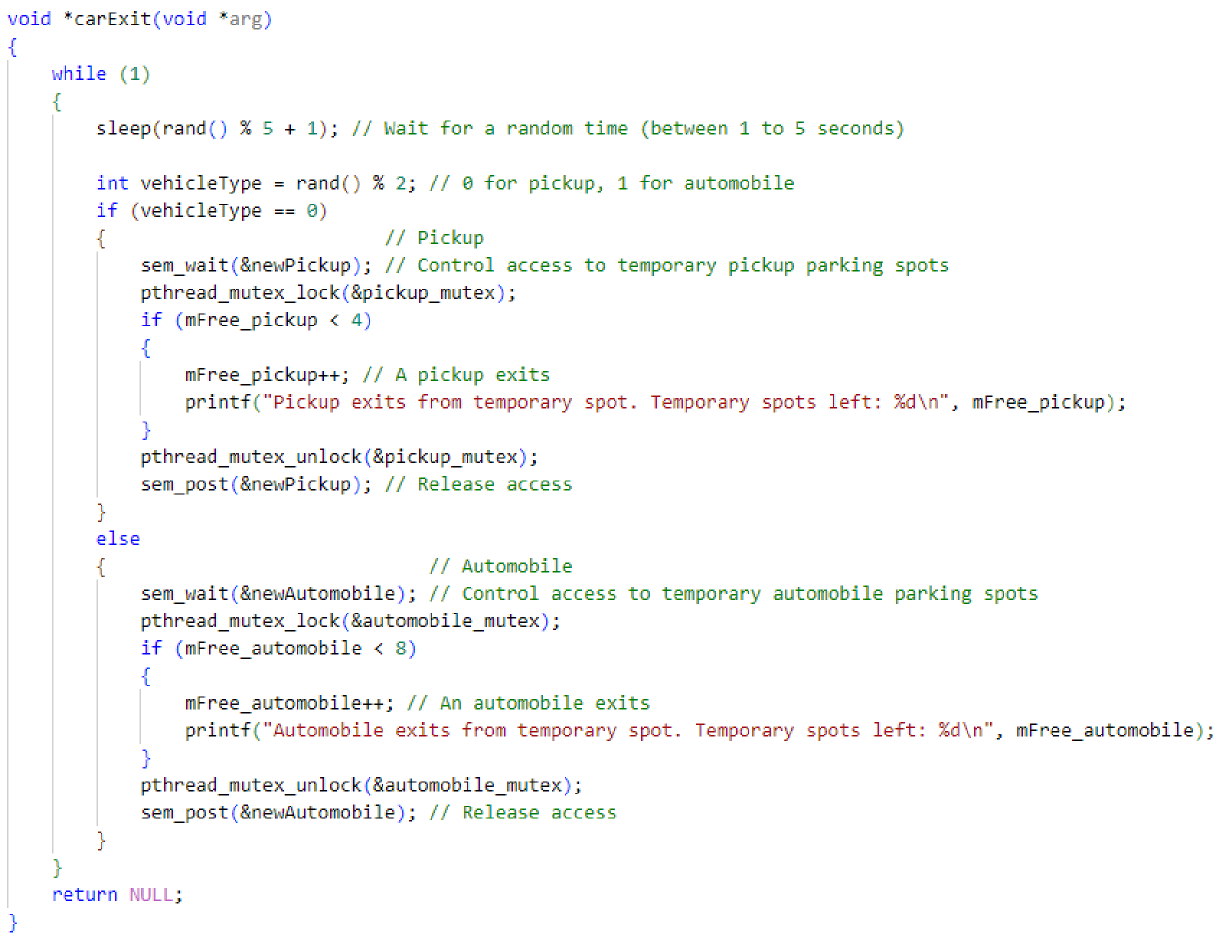

```c
void *carOwner(void *arg)
{
    while (1)
    {
        int vehicleType = rand() % 2; // 0 for pickup, 1 for automobile
        if (vehicleType == 0)
        {                           // Pickup
            sem_wait(&newPickup); // Control access to temporary pickup parking spots
            pthread_mutex_lock(&pickup_mutex);
            if (mFree_pickup > 0)
            {
                mFree_pickup--; // Park a pickup
                printf("A pickup arrives at the parking lot.\n");
                printf("Pickup parked in a temporary spot. Temporary spots left: %d\n", mFree_pickup);
                sem_post(&inChargeforPickup); // Signal valet to move the pickup to a permanent spot
            }
            else
            {
                printf("No temporary spots left for pickups.\n");
            }
            pthread_mutex_unlock(&pickup_mutex);
            sem_post(&newPickup); // Release access
        }
        else
        {                               // Automobile
            sem_wait(&newAutomobile); // Control access to temporary automobile parking spots
            pthread_mutex_lock(&automobile_mutex);
            if (mFree_automobile > 0)
            {
                mFree_automobile--; // Park an automobile
                printf("An automobile arrives at the parking lot.\n");
                printf("Automobile parked in a temporary spot. Temporary spots left: %d\n", mFree_automobile);
                sem_post(&inChargeforAutomobile); // Signal valet to move the automobile to a permanent spot
            }
            else
            {
                printf("No temporary spots left for automobiles.\n");
            }
            pthread_mutex_unlock(&automobile_mutex);
            sem_post(&newAutomobile); // Release access
        }
        sleep(1); // Short delay to avoid immediate consecutive arrivals
    }
    return NULL;
}
```

5.4 carAttendant: Valet attendants are represented by the carAttendant function, which moves cars in an endless cycle from temporary to permanent parking spaces. The valet first obtains access to the temporary parking spaces for cars by waiting for a signal to move a vehicle (sem_wait(&inChargeforAutomobile)) and then moves the vehicle (sem_wait(&newAutomobile)). After locking the automobile_mutex and determining whether any permanent slots are available, the attendant transfers an automobile by increasing mFree_automobile and decreasing permanent_automobile. Printed messages now reflect these modifications. A notification stating that the car stays in the temporary location is printed if there are no permanent slots available. Following the release of the mutex and semaphore, the valet waits for a brief while (usleep(500000)).Similarly, the procedure for pickups is repeated by the valet: sem_wait(&inChargeforPickup) for signal, sem_wait(&newPickup) for access to temporary places, lock the pickup_mutex, move pickups if permanent spots become available, update the counts, and print the status messages. The mutex and semaphore are freed following each action, and the valet waits a short while before proceeding.



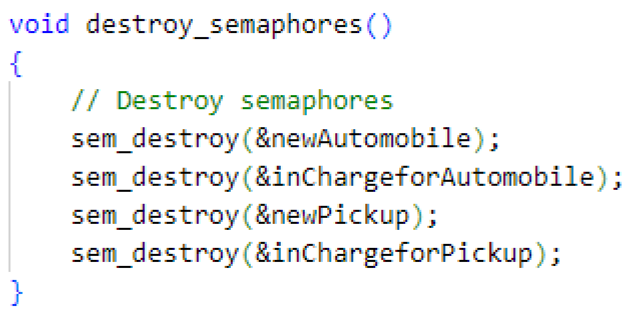
```c
void *carAttendant(void *arg)
{
    while (1)
    { // Infinite loop
        // Valet for automobiles
        sem_wait(&inChargeforAutomobile); // Wait for a signal to move an automobile
        sem_wait(&newAutomobile);         // Control access to temporary parking spots
        pthread_mutex_lock(&automobile_mutex);
        if (permanent_automobile > 0)
        {
            permanent_automobile--; // Park the automobile in a permanent spot
            mFree_automobile++;     // Free up the temporary spot
            printf("Automobile moved to a permanent spot by valet. Permanent spots left: %d\n", permanent_automobile);
            printf("Temporary spots for automobiles left after moving: %d\n", mFree_automobile);
        }
        else
        {
            printf("No permanent spots left for automobiles, staying in temporary.\n");
            printf("Temporary spots for automobiles left after moving: %d\n", mFree_automobile);
        }
        pthread_mutex_unlock(&automobile_mutex);
        sem_post(&newAutomobile); // Release access to temporary spots
        usleep(500000);           // 0.5 second delay

        // Valet for pickups
        sem_wait(&inChargeforPickup); // Wait for a signal to move a pickup
        sem_wait(&newPickup);         // Control access to temporary parking spots
        pthread_mutex_lock(&pickup_mutex);
        if (permanent_pickup > 0)
        {
            permanent_pickup--; // Park the pickup in a permanent spot
            mFree_pickup++;     // Free up the temporary spot
            printf("Pickup moved to a permanent spot by valet. Permanent spots left: %d\n", permanent_pickup);
            printf("Temporary spots for pickups left after moving: %d\n", mFree_pickup);
        }
        else
        {
            printf("No permanent spots left for pickups, staying in temporary.\n");
            printf("Temporary spots for pickups left after moving: %d\n", mFree_pickup);
        }
        pthread_mutex_unlock(&pickup_mutex);
        sem_post(&newPickup); // Release access to temporary spots
        usleep(500000);       // 0.5 second delay
    }
    return NULL;
}
```

5.5 carExit: The carExit function runs in an infinite loop to create vehicle exits on a regular basis, simulating cars pulling out of the parking lot. A random delay of one to five seconds (sleep(rand() % 5 + 1)) is applied to each exit. (int vehicleType = rand() % 2) determines the vehicle type at random. When a pickup occurs, the thread locks the pickup mutex, obtains the semaphore governing access to temporary pickup locations (sem_wait(&newPickup)), and determines if there are less than four temporary spaces. If so, it publishes a message with the new number of available places and increases mFree_pickup to make a temporary spot available. Next, the semaphore and mutex are freed.Comparably, for an automobile, the thread locks the automobile_mutex, obtains the semaphore for temporary automobile spots (sem_wait(&newAutomobile)), and determines if there are less than eight temporary spaces. When a slot opens up, it publishes a message indicating the change and increases mFree_automobile. Next, the semaphore and mutex are freed. This feature makes sure that as cars leave the parking lot, the number of temporary places that are available is updated accurately.



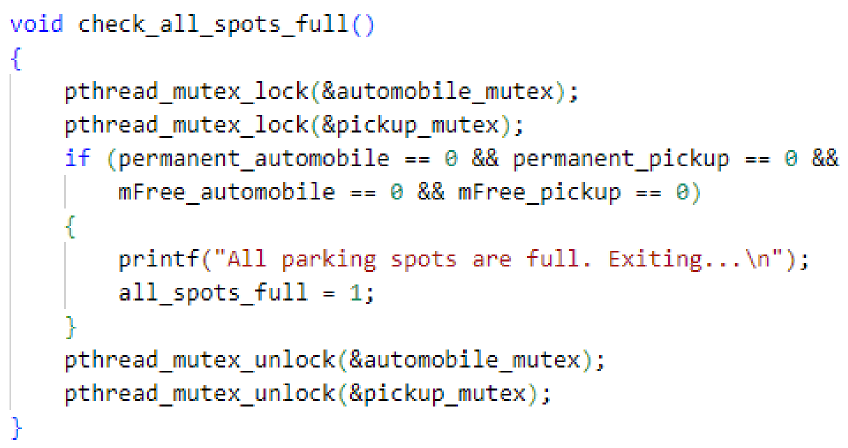
```c
void *carExit(void *arg)
{
    while (1)
    {
        sleep(rand() % 5 + 1); // Wait for a random time (between 1 to 5 seconds)

        int vehicleType = rand() % 2; // 0 for pickup, 1 for automobile
        if (vehicleType == 0)
        {                         // Pickup
            sem_wait(&newPickup); // Control access to temporary pickup parking spots
            pthread_mutex_lock(&pickup_mutex);
            if (mFree_pickup < 4)
            {
                mFree_pickup++; // A pickup exits
                printf("Pickup exits from temporary spot. Temporary spots left: %d\n", mFree_pickup);
            }
            pthread_mutex_unlock(&pickup_mutex);
            sem_post(&newPickup); // Release access
        }
        else
        {                             // Automobile
            sem_wait(&newAutomobile); // Control access to temporary automobile parking spots
            pthread_mutex_lock(&automobile_mutex);
            if (mFree_automobile < 8)
            {
                mFree_automobile++; // An automobile exits
                printf("Automobile exits from temporary spot. Temporary spots left: %d\n", mFree_automobile);
            }
            pthread_mutex_unlock(&automobile_mutex);
            sem_post(&newAutomobile); // Release access
        }
    }
    return NULL;
}
```

5.6 destroy_semaphores: In order to liberate any resources that the semaphores in the parking lot simulation may be retaining, they must be appropriately destroyed via the destroy_semaphores method. To prevent resource leaks and ensure a clean program exit, this is crucial. On all four semaphores—newAutomobile, inChargeforAutomobile, newPickup, and inChargeforPickup—the code executes sem_destroy. The function guarantees that all resources associated with semaphores are eliminated, averting possible memory leaks and guaranteeing a smooth program termination by eliminating these semaphores.



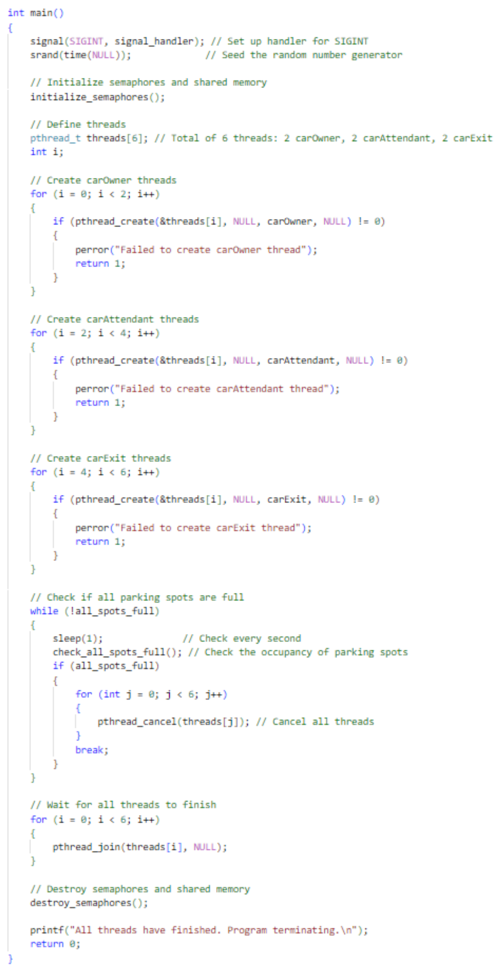
```
void destroy_semaphores()
{
    // Destroy semaphores
    sem_destroy(&newAutomobile);
    sem_destroy(&inChargeforAutomobile);
    sem_destroy(&newPickup);
    sem_destroy(&inChargeforPickup);
}
```

5.7 check_all_spots_full: The check_all_spots_full function checks to see if every parking space in the lot—permanent and temporary—is taken. Initially, it locks the pickup_mutex and car_mutex to provide synchronized access to common variables. The method then determines if the counts of temporary locations for both (mFree_automobile and mFree_pickup) and permanent spots for cars (permanent_automobile and permanent_pickup) are zero. The function outputs a message and changes the all_spots_full flag to 1 if all of these counts are zero, which indicates that all parking spots are full. The mutexes are finally unlocked. This feature is essential for keeping an eye on the condition of the parking lot and setting off alarms when it fills up entirely.



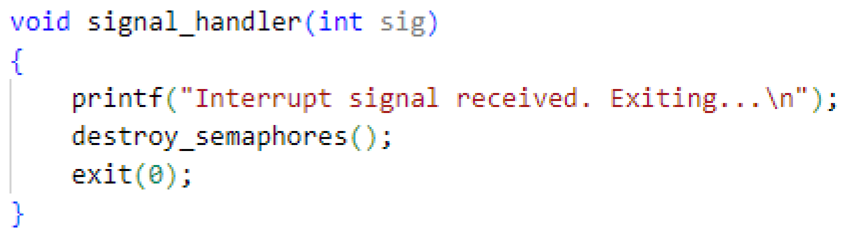
```
void check_all_spots_full()
{
    pthread_mutex_lock(&automobile_mutex);
    pthread_mutex_lock(&pickup_mutex);
    if (permanent_automobile == 0 && permanent_pickup == 0 &&
        mFree_automobile == 0 && mFree_pickup == 0)
    {
        printf("All parking spots are full. Exiting...\n");
        all_spots_full = 1;
    }
    pthread_mutex_unlock(&automobile_mutex);
    pthread_mutex_unlock(&pickup_mutex);
}
```

5.8 main: The main function coordinates the start and stop of threads and initializes and controls the whole parking lot simulation. In order to guarantee a smooth exit, it first configures a signal handler for SIGINT, which is normally activated by pressing Ctrl+C. It then seeds the random number generator with the current time in order to generate random vehicle kinds and exit timings. The code then calls initialize_semaphores to initialize the shared variables and semaphores. The array of six threads is then defined, two for automobile owners (carOwner), two for valets (carAttendant), and two for vehicle exits (carExit). These threads are created by the function in three iterations, while it checks for faults throughout the thread creation process and prints an error message if any of the threads fail to start.Every second, the while loop calls check_all_spots_full to see if all parking spaces are taken. It cancels all threads and exits the loop if it finds that every slot is taken. The code waits for all threads to complete using pthread_join after breaking out of the loop. In order to free resources, it runs destroy_semaphores at the end and outputs a message stating that the application is ending. The primary function guarantees the seamless running of the program by appropriately coordinating thread operations and managing resource cleaning at program end.



```c
int main()
{
    signal(SIGINT, signal_handler); // Set up handler for SIGINT
    srand(time(NULL));              // Seed the random number generator

    // Initialize semaphores and shared memory
    initialize_semaphores();

    // Define threads
    pthread_t threads[6]; // Total of 6 threads: 2 carOwner, 2 carAttendant, 2 carExit
    int i;

    // Create carOwner threads
    for (i = 0; i < 2; i++)
    {
        if (pthread_create(&threads[i], NULL, carOwner, NULL) != 0)
        {
            perror("Failed to create carOwner thread");
            return 1;
        }
    }

    // Create carAttendant threads
    for (i = 2; i < 4; i++)
    {
        if (pthread_create(&threads[i], NULL, carAttendant, NULL) != 0)
        {
            perror("Failed to create carAttendant thread");
            return 1;
        }
    }

    // Create carExit threads
    for (i = 4; i < 6; i++)
    {
        if (pthread_create(&threads[i], NULL, carExit, NULL) != 0)
        {
            perror("Failed to create carExit thread");
            return 1;
        }
    }

    // Check if all parking spots are full
    while (!all_spots_full)
    {
        sleep(1);              // Check every second
        check_all_spots_full(); // Check the occupancy of parking spots
        if (all_spots_full)
        {
            for (int j = 0; j < 6; j++)
            {
                pthread_cancel(threads[j]); // Cancel all threads
            }
            break;
        }
    }

    // Wait for all threads to finish
    for (i = 0; i < 6; i++)
    {
        pthread_join(threads[i], NULL);
    }

    // Destroy semaphores and shared memory
    destroy_semaphores();

    printf("All threads have finished. Program terminating.\n");
    return 0;
}
```

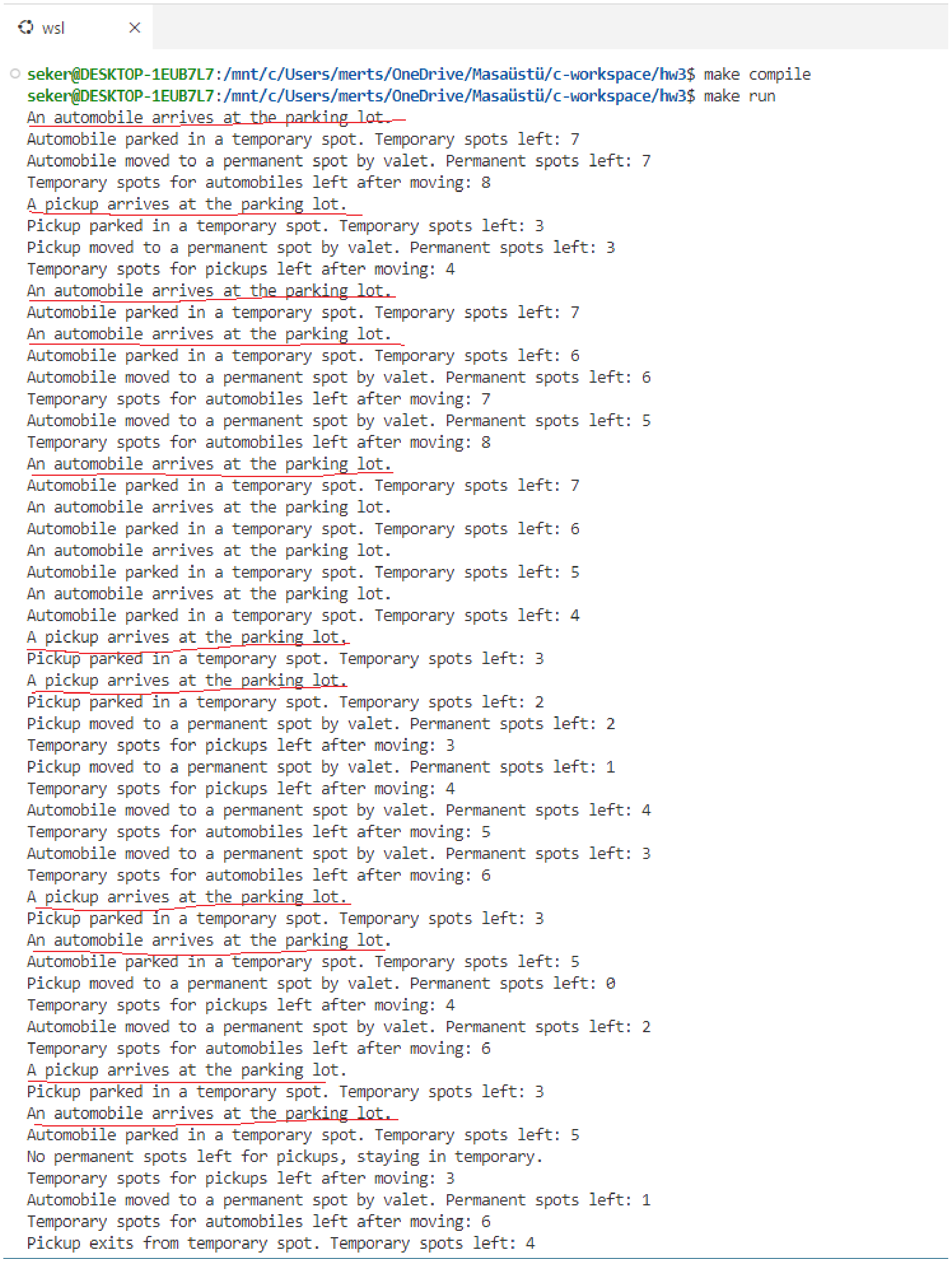
5.9 signal_handler: To guarantee that the application ends smoothly, the signal_handler function handles interrupt signals (like SIGINT, which is normally triggered by pressing Ctrl+C). The function outputs a message ("Interrupt signal received") when it receives an interrupt signal. "Leaving...") to let the user know. The resources that the semaphores were holding are then released and cleaned up by calling the destroy_semaphores method. Ultimately, the function ends the program by using exit(0). This feature makes sure that the application can cleanly handle sudden termination requests, such as those that are made by hitting Ctrl+C, preventing resource leaks and guaranteeing a correct shutdown.



```
void signal_handler(int sig)
{
    printf("Interrupt signal received. Exiting...\n");
    destroy_semaphores();
    exit(0);
}
```

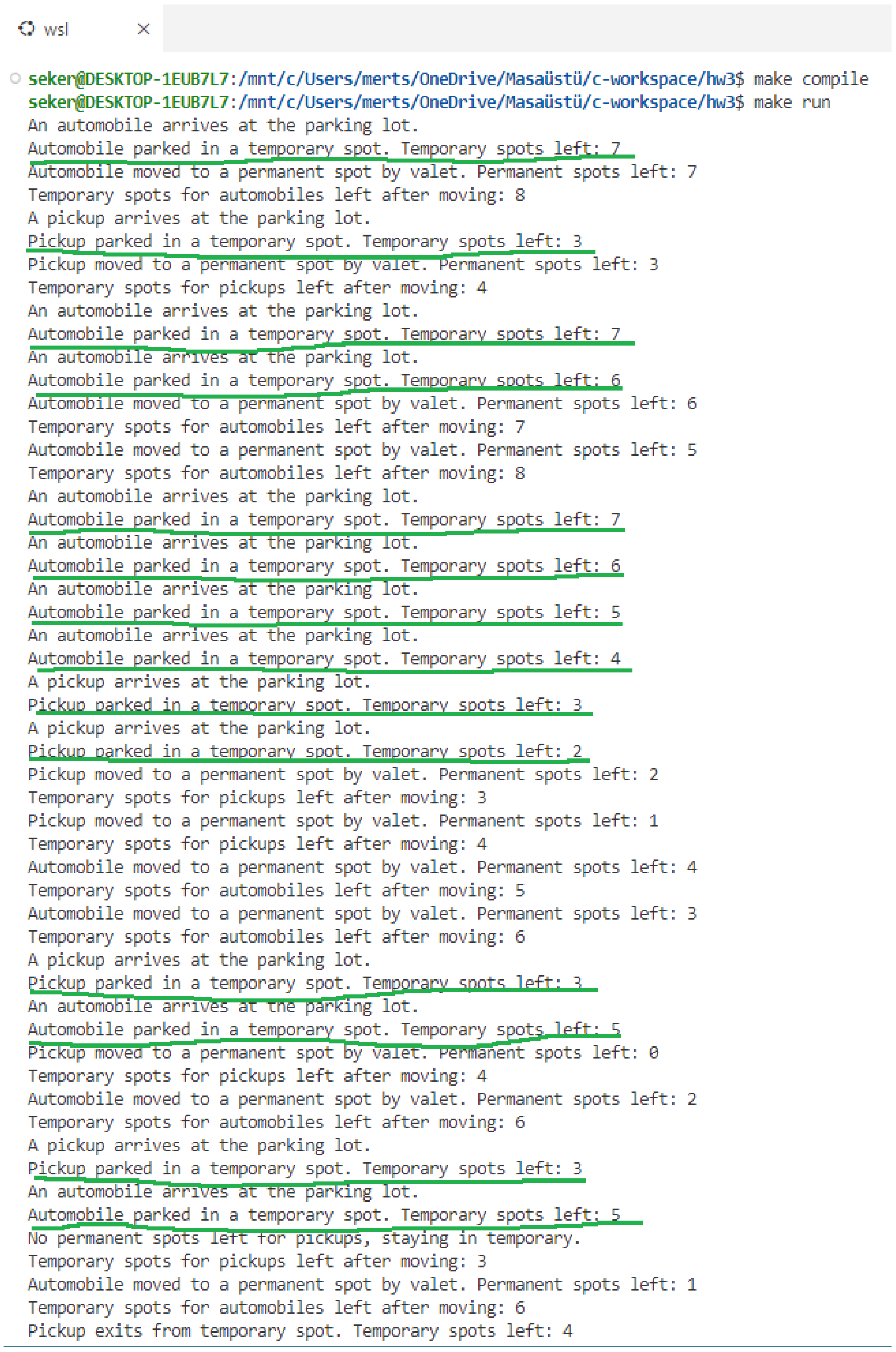
6. Example Outputs:

6.1. Automobile or pickup arrives: Automobiles or pickups keep arriving until all the temporary and permanent spots are filled.



```
seker@DESKTOP-1EUB7L7:/mnt/c/Users/merts/OneDrive/Masaüstü/c-workspace/hw3$ make compile
seker@DESKTOP-1EUB7L7:/mnt/c/Users/merts/OneDrive/Masaüstü/c-workspace/hw3$ make run
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 7
Automobile moved to a permanent spot by valet. Permanent spots left: 7
Temporary spots for automobiles left after moving: 8
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
Pickup moved to a permanent spot by valet. Permanent spots left: 3
Temporary spots for pickups left after moving: 4
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 7
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 6
Automobile moved to a permanent spot by valet. Permanent spots left: 6
Temporary spots for automobiles left after moving: 7
Automobile moved to a permanent spot by valet. Permanent spots left: 5
Temporary spots for automobiles left after moving: 8
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 7
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 6
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 5
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 4
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 2
Pickup moved to a permanent spot by valet. Permanent spots left: 2
Temporary spots for pickups left after moving: 3
Pickup moved to a permanent spot by valet. Permanent spots left: 1
Temporary spots for pickups left after moving: 4
Automobile moved to a permanent spot by valet. Permanent spots left: 4
Temporary spots for automobiles left after moving: 5
Automobile moved to a permanent spot by valet. Permanent spots left: 3
Temporary spots for automobiles left after moving: 6
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 5
Pickup moved to a permanent spot by valet. Permanent spots left: 0
Temporary spots for pickups left after moving: 4
Automobile moved to a permanent spot by valet. Permanent spots left: 2
Temporary spots for automobiles left after moving: 6
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 5
No permanent spots left for pickups, staying in temporary.
Temporary spots for pickups left after moving: 3
Automobile moved to a permanent spot by valet. Permanent spots left: 1
Temporary spots for automobiles left after moving: 6
Pickup exits from temporary spot. Temporary spots left: 4
```

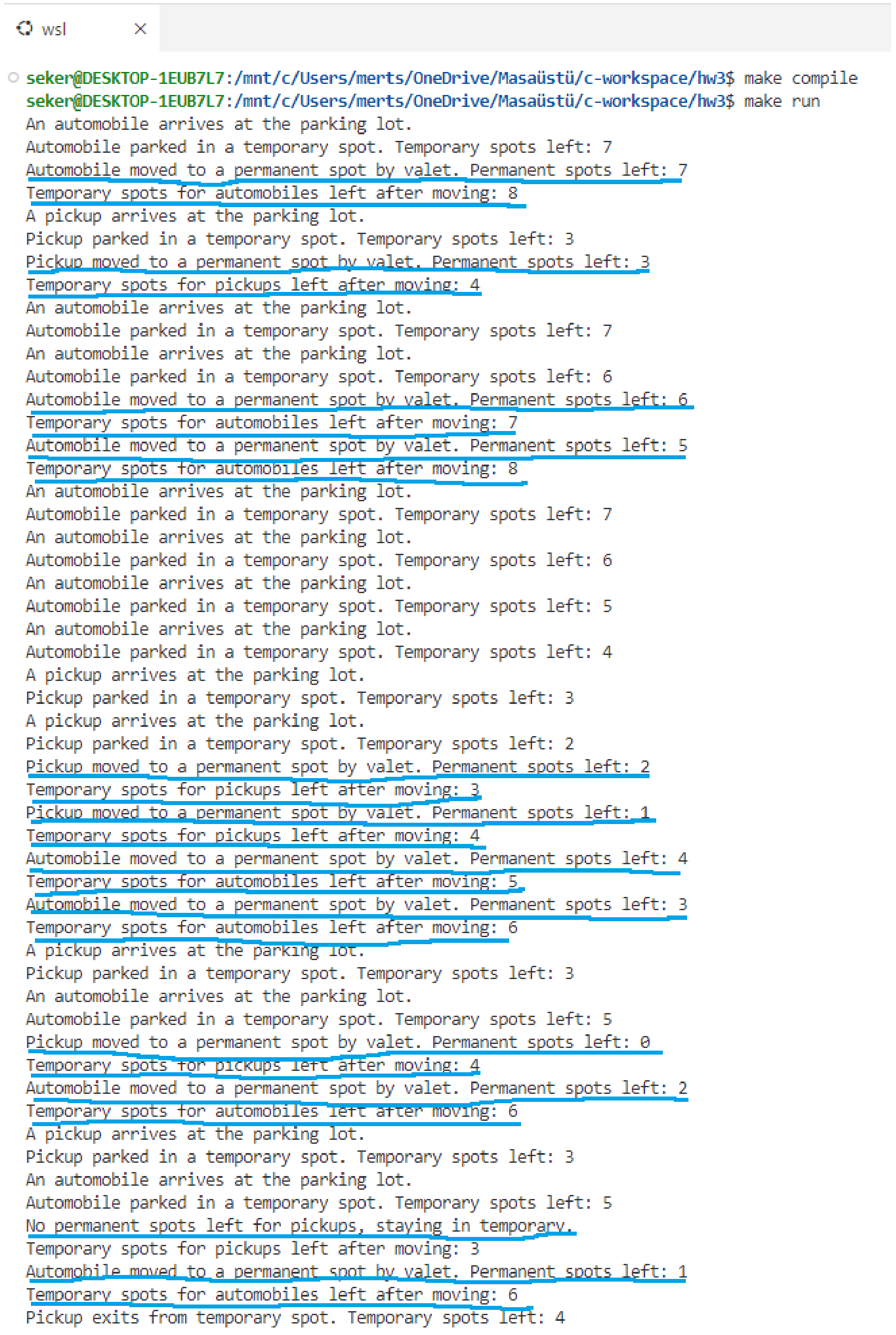
6.2. Parking temporary spots:



wsl

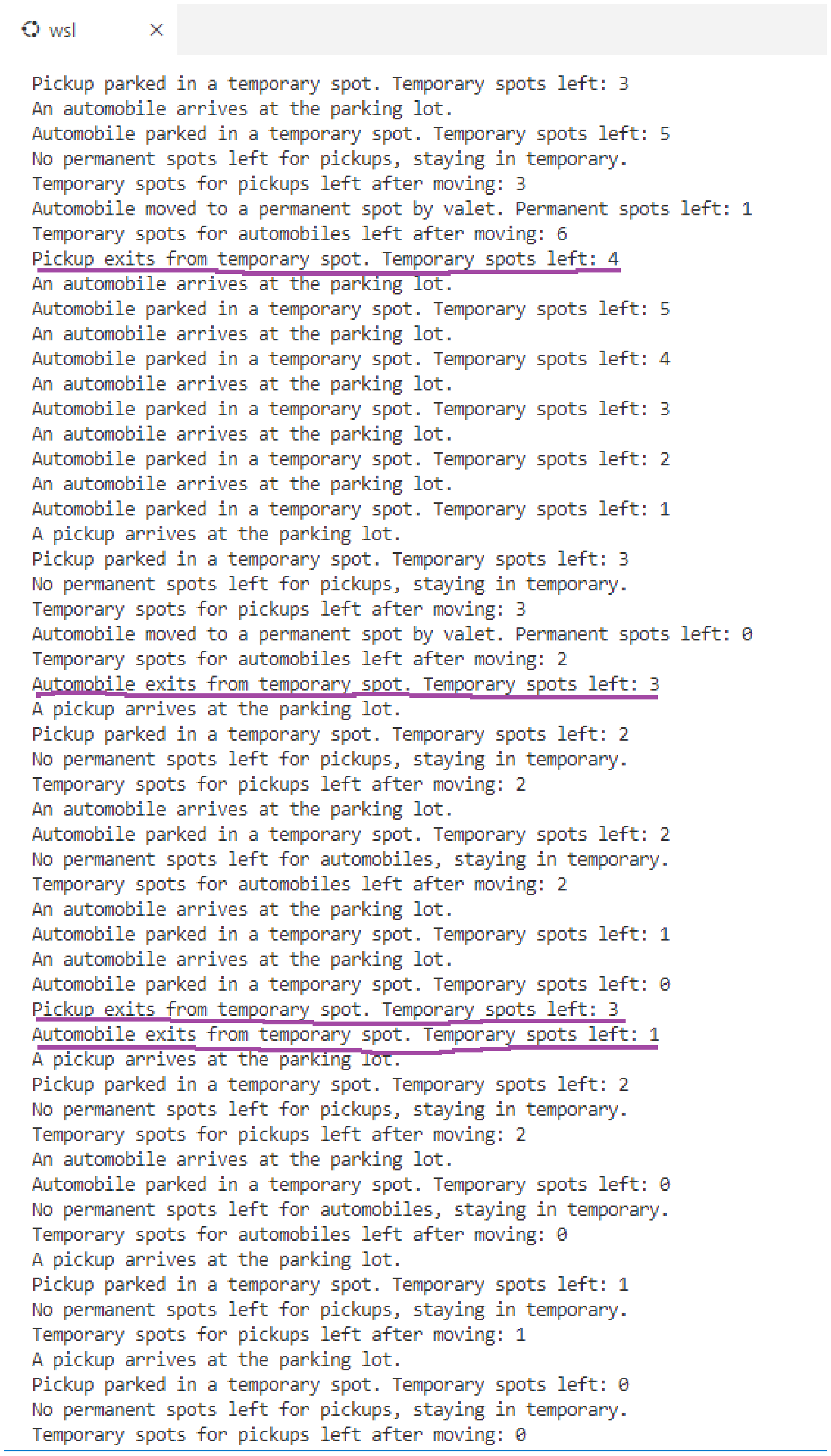
```
seker@DESKTOP-1EUB7L7:/mnt/c/Users/merts/OneDrive/Masaüstü/c-workspace/hw3$ make compile
seker@DESKTOP-1EUB7L7:/mnt/c/Users/merts/OneDrive/Masaüstü/c-workspace/hw3$ make run
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 7
Automobile moved to a permanent spot by valet. Permanent spots left: 7
Temporary spots for automobiles left after moving: 8
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
Pickup moved to a permanent spot by valet. Permanent spots left: 3
Temporary spots for pickups left after moving: 4
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 7
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 6
Automobile moved to a permanent spot by valet. Permanent spots left: 6
Temporary spots for automobiles left after moving: 7
Automobile moved to a permanent spot by valet. Permanent spots left: 5
Temporary spots for automobiles left after moving: 8
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 7
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 6
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 5
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 4
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 2
Pickup moved to a permanent spot by valet. Permanent spots left: 2
Temporary spots for pickups left after moving: 3
Pickup moved to a permanent spot by valet. Permanent spots left: 1
Temporary spots for pickups left after moving: 4
Automobile moved to a permanent spot by valet. Permanent spots left: 4
Temporary spots for automobiles left after moving: 5
Automobile moved to a permanent spot by valet. Permanent spots left: 3
Temporary spots for automobiles left after moving: 6
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 5
Pickup moved to a permanent spot by valet. Permanent spots left: 0
Temporary spots for pickups left after moving: 4
Automobile moved to a permanent spot by valet. Permanent spots left: 2
Temporary spots for automobiles left after moving: 6
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 5
No permanent spots left for pickups, staying in temporary.
Temporary spots for pickups left after moving: 3
Automobile moved to a permanent spot by valet. Permanent spots left: 1
Temporary spots for automobiles left after moving: 6
Pickup exits from temporary spot. Temporary spots left: 4
```

6.3. Moving temporary to permanent by valet:


wsl

```
seker@DESKTOP-1EUB7L7:/mnt/c/Users/merts/OneDrive/Masaüstü/c-workspace/hw3$ make compile
seker@DESKTOP-1EUB7L7:/mnt/c/Users/merts/OneDrive/Masaüstü/c-workspace/hw3$ make run
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 7
Automobile moved to a permanent spot by valet. Permanent spots left: 7
Temporary spots for automobiles left after moving: 8
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
Pickup moved to a permanent spot by valet. Permanent spots left: 3
Temporary spots for pickups left after moving: 4
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 7
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 6
Automobile moved to a permanent spot by valet. Permanent spots left: 6
Temporary spots for automobiles left after moving: 7
Automobile moved to a permanent spot by valet. Permanent spots left: 5
Temporary spots for automobiles left after moving: 8
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 7
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 6
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 5
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 4
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 2
Pickup moved to a permanent spot by valet. Permanent spots left: 2
Temporary spots for pickups left after moving: 3
Pickup moved to a permanent spot by valet. Permanent spots left: 1
Temporary spots for pickups left after moving: 4
Automobile moved to a permanent spot by valet. Permanent spots left: 4
Temporary spots for automobiles left after moving: 5
Automobile moved to a permanent spot by valet. Permanent spots left: 3
Temporary spots for automobiles left after moving: 6
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 5
Pickup moved to a permanent spot by valet. Permanent spots left: 0
Temporary spots for pickups left after moving: 4
Automobile moved to a permanent spot by valet. Permanent spots left: 2
Temporary spots for automobiles left after moving: 6
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 5
No permanent spots left for pickups, staying in temporary.
Temporary spots for pickups left after moving: 3
Automobile moved to a permanent spot by valet. Permanent spots left: 1
Temporary spots for automobiles left after moving: 6
Pickup exits from temporary spot. Temporary spots left: 4
```

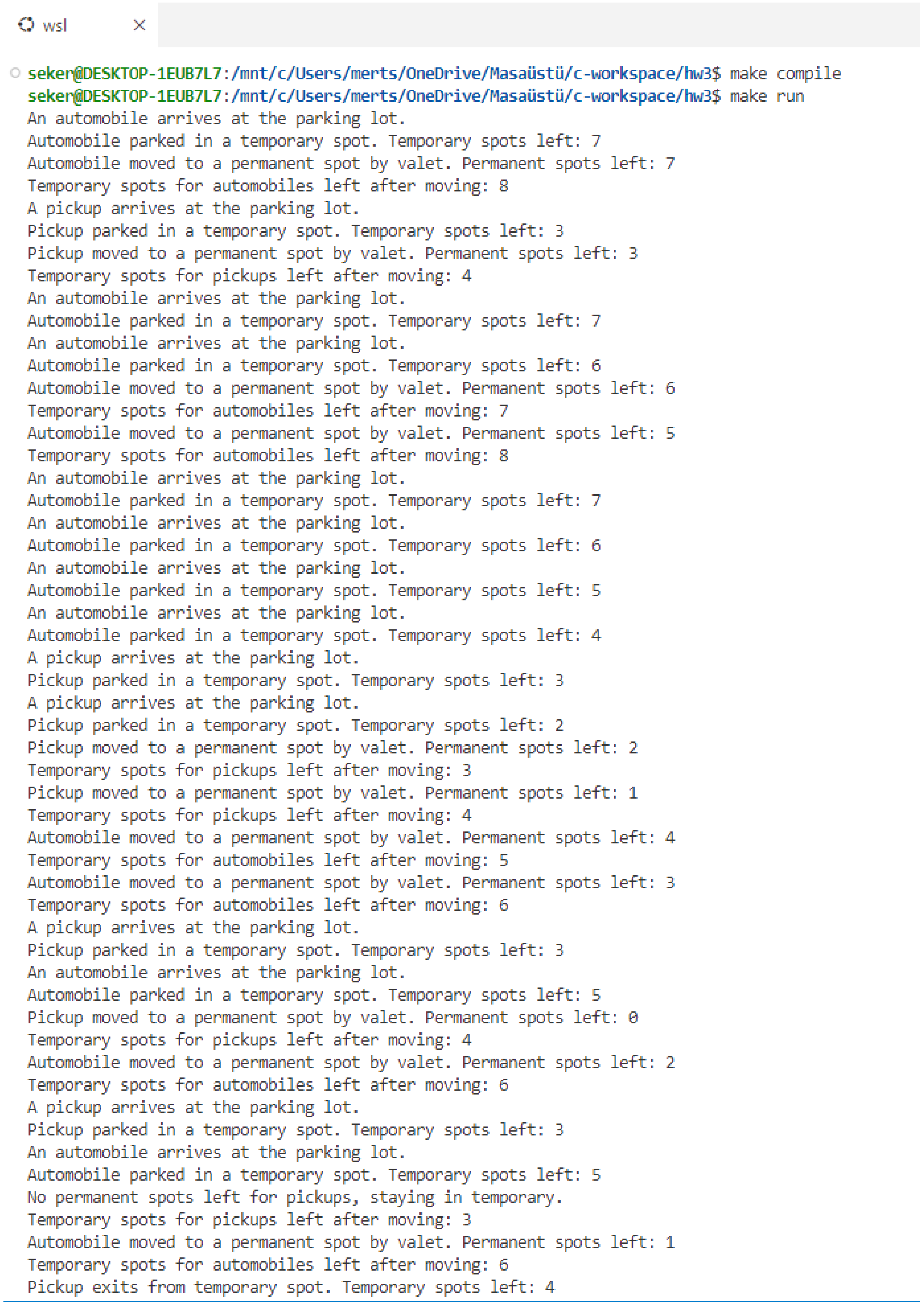
6.4. Pickup or automobile exits the temporary park spot:


wsl

```
Pickup parked in a temporary spot. Temporary spots left: 3
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 5
No permanent spots left for pickups, staying in temporary.
Temporary spots for pickups left after moving: 3
Automobile moved to a permanent spot by valet. Permanent spots left: 1
Temporary spots for automobiles left after moving: 6
Pickup exits from temporary spot. Temporary spots left: 4
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 5
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 4
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 3
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 2
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 1
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
No permanent spots left for pickups, staying in temporary.
Temporary spots for pickups left after moving: 3
Automobile moved to a permanent spot by valet. Permanent spots left: 0
Temporary spots for automobiles left after moving: 2
Automobile exits from temporary spot. Temporary spots left: 3
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 2
No permanent spots left for pickups, staying in temporary.
Temporary spots for pickups left after moving: 2
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 2
No permanent spots left for automobiles, staying in temporary.
Temporary spots for automobiles left after moving: 2
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 1
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 0
Pickup exits from temporary spot. Temporary spots left: 3
Automobile exits from temporary spot. Temporary spots left: 1
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 2
No permanent spots left for pickups, staying in temporary.
Temporary spots for pickups left after moving: 2
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 0
No permanent spots left for automobiles, staying in temporary.
Temporary spots for automobiles left after moving: 0
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 1
No permanent spots left for pickups, staying in temporary.
Temporary spots for pickups left after moving: 1
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 0
No permanent spots left for pickups, staying in temporary.
Temporary spots for pickups left after moving: 0
```

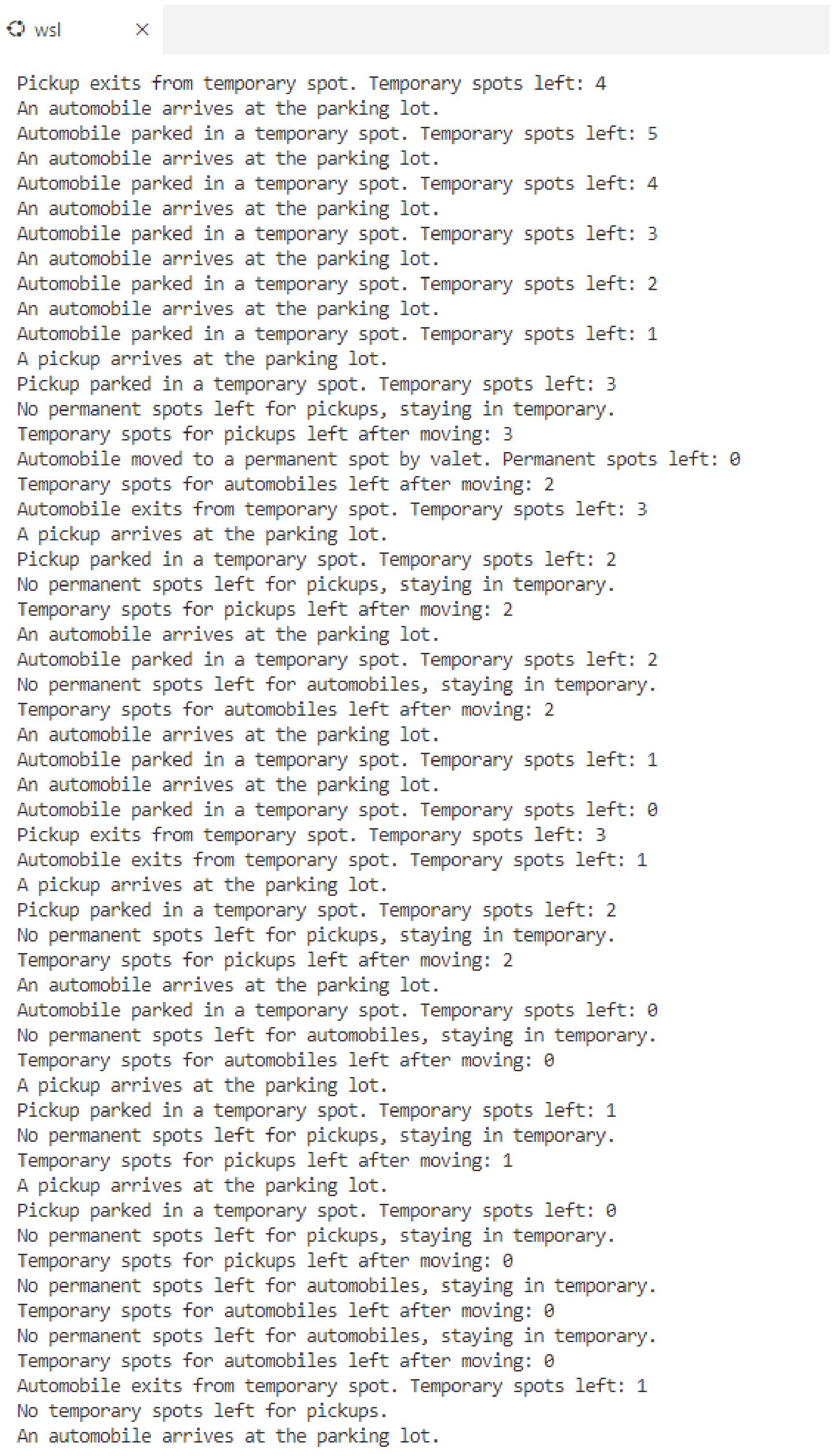
6.5. Full output: The full output is in 3 parts. it's long so I couldn't fit it in one screenshot. it's all a continuation of each other.

6.5.1. Part 1:



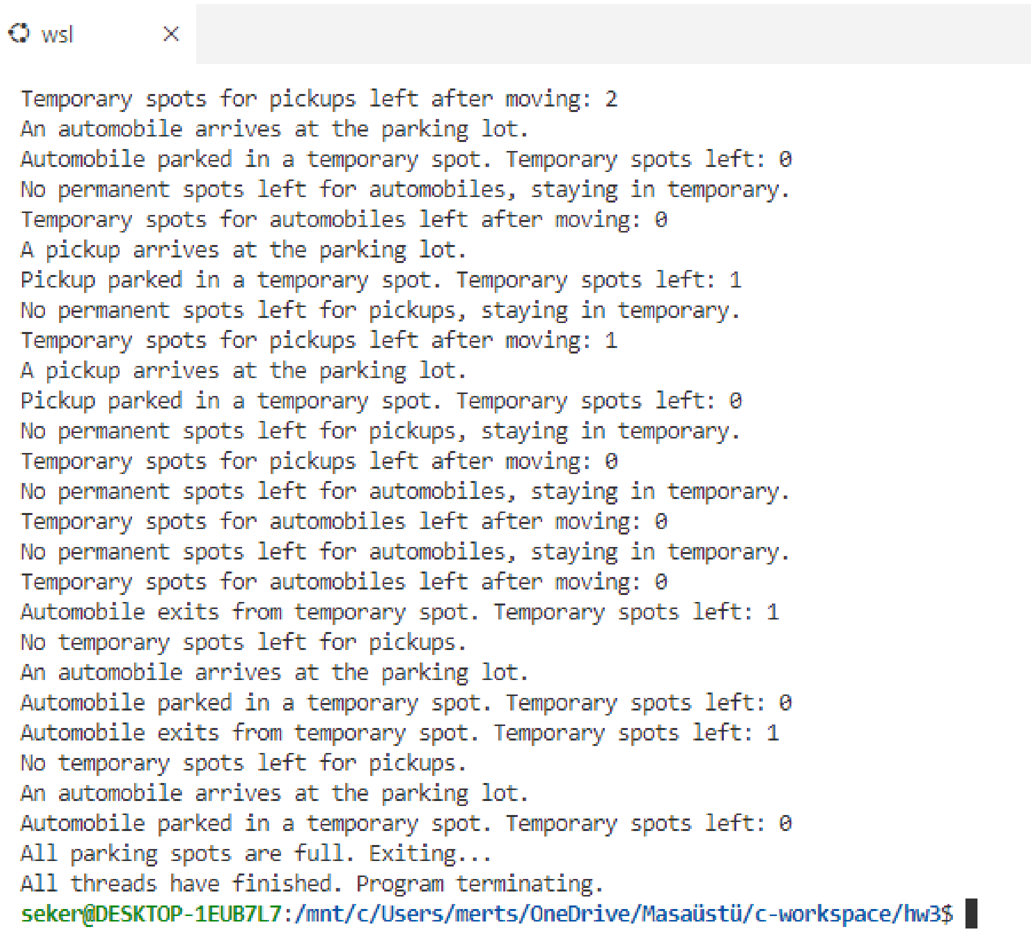
```
seker@DESKTOP-1EUB7L7:/mnt/c/Users/merts/OneDrive/Masaüstü/c-workspace/hw3$ make compile
seker@DESKTOP-1EUB7L7:/mnt/c/Users/merts/OneDrive/Masaüstü/c-workspace/hw3$ make run
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 7
Automobile moved to a permanent spot by valet. Permanent spots left: 7
Temporary spots for automobiles left after moving: 8
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
Pickup moved to a permanent spot by valet. Permanent spots left: 3
Temporary spots for pickups left after moving: 4
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 7
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 6
Automobile moved to a permanent spot by valet. Permanent spots left: 6
Temporary spots for automobiles left after moving: 7
Automobile moved to a permanent spot by valet. Permanent spots left: 5
Temporary spots for automobiles left after moving: 8
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 7
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 6
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 5
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 4
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 2
Pickup moved to a permanent spot by valet. Permanent spots left: 2
Temporary spots for pickups left after moving: 3
Pickup moved to a permanent spot by valet. Permanent spots left: 1
Temporary spots for pickups left after moving: 4
Automobile moved to a permanent spot by valet. Permanent spots left: 4
Temporary spots for automobiles left after moving: 5
Automobile moved to a permanent spot by valet. Permanent spots left: 3
Temporary spots for automobiles left after moving: 6
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 5
Pickup moved to a permanent spot by valet. Permanent spots left: 0
Temporary spots for pickups left after moving: 4
Automobile moved to a permanent spot by valet. Permanent spots left: 2
Temporary spots for automobiles left after moving: 6
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 5
No permanent spots left for pickups, staying in temporary.
Temporary spots for pickups left after moving: 3
Automobile moved to a permanent spot by valet. Permanent spots left: 1
Temporary spots for automobiles left after moving: 6
Pickup exits from temporary spot. Temporary spots left: 4
```

6.5.2. Part 2:



```
Pickup exits from temporary spot. Temporary spots left: 4
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 5
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 4
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 3
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 2
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 1
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 3
No permanent spots left for pickups, staying in temporary.
Temporary spots for pickups left after moving: 3
Automobile moved to a permanent spot by valet. Permanent spots left: 0
Temporary spots for automobiles left after moving: 2
Automobile exits from temporary spot. Temporary spots left: 3
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 2
No permanent spots left for pickups, staying in temporary.
Temporary spots for pickups left after moving: 2
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 2
No permanent spots left for automobiles, staying in temporary.
Temporary spots for automobiles left after moving: 2
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 1
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 0
Pickup exits from temporary spot. Temporary spots left: 3
Automobile exits from temporary spot. Temporary spots left: 1
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 2
No permanent spots left for pickups, staying in temporary.
Temporary spots for pickups left after moving: 2
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 0
No permanent spots left for automobiles, staying in temporary.
Temporary spots for automobiles left after moving: 0
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 1
No permanent spots left for pickups, staying in temporary.
Temporary spots for pickups left after moving: 1
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 0
No permanent spots left for pickups, staying in temporary.
Temporary spots for pickups left after moving: 0
No permanent spots left for automobiles, staying in temporary.
Temporary spots for automobiles left after moving: 0
No permanent spots left for automobiles, staying in temporary.
Temporary spots for automobiles left after moving: 0
Automobile exits from temporary spot. Temporary spots left: 1
No temporary spots left for pickups.
An automobile arrives at the parking lot.
```

6.5.3. Part 3:


wsl

```
Temporary spots for pickups left after moving: 2
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 0
No permanent spots left for automobiles, staying in temporary.
Temporary spots for automobiles left after moving: 0
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 1
No permanent spots left for pickups, staying in temporary.
Temporary spots for pickups left after moving: 1
A pickup arrives at the parking lot.
Pickup parked in a temporary spot. Temporary spots left: 0
No permanent spots left for pickups, staying in temporary.
Temporary spots for pickups left after moving: 0
No permanent spots left for automobiles, staying in temporary.
Temporary spots for automobiles left after moving: 0
No permanent spots left for automobiles, staying in temporary.
Temporary spots for automobiles left after moving: 0
Automobile exits from temporary spot. Temporary spots left: 1
No temporary spots left for pickups.
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 0
Automobile exits from temporary spot. Temporary spots left: 1
No temporary spots left for pickups.
An automobile arrives at the parking lot.
Automobile parked in a temporary spot. Temporary spots left: 0
All parking spots are full. Exiting...
All threads have finished. Program terminating.
seker@DESKTOP-1EUB7L7:/mnt/c/Users/merts/OneDrive/Masaüstü/c-workspace/hw3$
```