ÇAĞRI YILDIZ 1901042630

CSE 344

HOMEWORK 3 REPORT

Note: Due to some ambiguities in the class group regarding the assignment requirements, I have implemented the project as per my understanding. I have defined parking areas with capacities of 4 for pickups and 8 for automobiles, for both temporary and permanent parking areas.

Project Structure

This report consists of two parts:

- Project Requirements: Detailed overview of the assignment's expectations and objectives.
- Terminal Outputs: Explanation and demonstration of how the program operates, including descriptions of the outputs observed during its execution.

Project Requirements

Objective

The objective of this assignment is to simulate a parking lot system managed by two parking attendants with specific capacities for pickups and automobiles. The synchronization between car owners and attendants should be handled using semaphores.

1. Compilation:

The provided code should compile successfully without any errors. To compile the code, you can use the following commands:

- Compile the code: *make*
- Run the code: *make run*
- Check for memory leaks using Valgrind: *make valgrind*

2. Separate Spots for Cars and Pickups:

The program should maintain separate parking spots for automobiles and pickups. This is demonstrated in the following code snippet, where the counters for temporary and permanent parking spots are defined:

```
9   #define MAX_AUTOMOBILE 8
10  #define MAX_PICKUP 4
11
12  // Shared parking space counters
13  int mFreeTemp_automobile = MAX_AUTOMOBILE; // Number of available temporary parking spots for automobiles
14  int mFreeTemp_pickup = MAX_PICKUP; // Number of available temporary parking spots for pickups
15  int mFreePerm_automobile = MAX_AUTOMOBILE; // Number of available permanent parking spots for automobiles
16  int mFreePerm_pickup = MAX_PICKUP; // Number of available permanent parking spots for pickups
17
```

3. Random Vehicle Type Generation:

The randomVehicleType function generates a random vehicle type (0 for pickup, 1 for automobile):

```
93  // Function to generate a random vehicle type (0: Pickup, 1: Automobile)
94  int randomVehicleType() {
95      return rand() % 2; // 0: Pickup, 1: Automobile
96  }
```

4. Single Vehicle Entry:

Vehicles are created with a delay to simulate staggered arrivals, ensuring that only one vehicle enters the system at a time:

```
40      // Create carOwner threads (simulating vehicle arrivals)
41      pthread_t owners[20];
42      for (int i = 0; i < 20; i++) {
43          pthread_create(&owners[i], NULL, carOwner, (void*) (long) randomVehicleType());
44          usleep(100000); // Simulate staggered arrivals (introduce a delay between arrivals)
45      }
```

5. Synchronization using Semaphores:

The code uses semaphores to synchronize the car owners and attendants. Semaphores are initialized correctly, and mutexes are used to protect shared resources:

```
65  void initialize() {
66      // Initialize semaphores
67      sem_init(&newPickup, 0, 0);
68      sem_init(&newAutomobile, 0, 0);
69      sem_init(&inChargeforPickup, 0, 1);
70      sem_init(&inChargeforAutomobile, 0, 1);
71      // Initialize mutex locks
72      pthread_mutex_init(&lockTempAutomobile, NULL);
73      pthread_mutex_init(&lockTempPickup, NULL);
74      pthread_mutex_init(&lockPermAutomobile, NULL);
75      pthread_mutex_init(&lockPermPickup, NULL);
76      // Seed for random number generation
77      srand(time(NULL));
78  }
79
80  void destroy() {
81      // Destroy semaphores
82      sem_destroy(&newPickup);
83      sem_destroy(&newAutomobile);
84      sem_destroy(&inChargeforPickup);
85      sem_destroy(&inChargeforAutomobile);
86      // Destroy mutex locks
87      pthread_mutex_destroy(&lockTempAutomobile);
88      pthread_mutex_destroy(&lockTempPickup);
89      pthread_mutex_destroy(&lockPermAutomobile);
90      pthread_mutex_destroy(&lockPermPickup);
91  }
```

Detailed Workflow

Car Owner Function:

The car owner function locks the respective temporary parking mutex, checks for availability, and signals the valet if a spot is available:

```c
void* carOwner(void* arg) {
    int vehicleType = (int)(long) arg; // Vehicle type (0: Pickup, 1: Automobile)
    int isAutomobile = vehicleType; // Is it an automobile? (0: Pickup, 1: Automobile)

    if (isAutomobile) {
        pthread_mutex_lock(&lockTempAutomobile); // Lock temporary automobile parking
        if (mFreeTemp_automobile > 0) {
            // If there is space in the temporary lot, reduce available spots and signal the valet
            mFreeTemp_automobile--;
            printf("Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: %d\n", mFreeTemp_automobile);
            pthread_mutex_unlock(&lockTempAutomobile);
            sem_post(&newAutomobile);
            sem_wait(&inChargeforAutomobile); // Wait for the valet to park the automobile
        } else {
            pthread_mutex_unlock(&lockTempAutomobile);
            printf("Car Owner (Automobile) leaves due to no temporary spots.\n");
        }
    } else {
        pthread_mutex_lock(&lockTempPickup); // Lock temporary pickup parking
        if (mFreeTemp_pickup > 0) {
            // If there is space in the temporary lot, reduce available spots and signal the valet
            mFreeTemp_pickup--;
            printf("Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: %d\n", mFreeTemp_pickup);
            pthread_mutex_unlock(&lockTempPickup);
            sem_post(&newPickup);
            sem_wait(&inChargeforPickup); // Wait for the valet to park the pickup
        } else {
            pthread_mutex_unlock(&lockTempPickup);
            printf("Car Owner (Pickup) leaves due to no temporary spots.\n");
        }
    }

    return NULL;
}
```

Car Attendant Function:

The car attendant function waits for a signal of a new vehicle arrival, locks the respective permanent parking mutex, and updates the spot counters:

```c
void* carAttendant(void* arg) {
    char* type = (char*) arg; // Valet type (Automobile or Pickup)
    int isAutomobile = (strcmp(type, "Automobile") == 0); // Does the valet handle automobiles?

    while (running) {
        if (isAutomobile) {
            sem_wait(&newAutomobile); // Wait for a new automobile to arrive
            if (!running) break;
            pthread_mutex_lock(&lockPermAutomobile); // Lock permanent automobile parking
            if (mFreePerm_automobile > 0) {
                // If there is space in the permanent lot, reduce available spots and increase temporary parking
                mFreePerm_automobile--;
                pthread_mutex_lock(&lockTempAutomobile);
                mFreeTemp_automobile++;
                printf("Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: %d, Temporary Automobile Spots: %d\n",
                    mFreePerm_automobile, mFreeTemp_automobile);
                pthread_mutex_unlock(&lockTempAutomobile);
            } else {
                printf("No permanent spots available for Automobile.\n");
            }
            pthread_mutex_unlock(&lockPermAutomobile);
            sem_post(&inChargeforAutomobile); // Signal that the automobile has been parked
        } else {
            sem_wait(&newPickup); // Wait for a new pickup to arrive
            if (!running) break;
            pthread_mutex_lock(&lockPermPickup); // Lock permanent pickup parking
            if (mFreePerm_pickup > 0) {
                // If there is space in the permanent lot, reduce available spots and increase temporary parking
                mFreePerm_pickup--;
                pthread_mutex_lock(&lockTempPickup);
                mFreeTemp_pickup++;
                printf("Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: %d, Temporary Pickup Spots: %d\n",
                    mFreePerm_pickup, mFreeTemp_pickup);
                pthread_mutex_unlock(&lockTempPickup);
            } else {
                printf("No permanent spots available for Pickup.\n");
            }
            pthread_mutex_unlock(&lockPermPickup);
            sem_post(&inChargeforPickup); // Signal that the pickup has been parked
        }
    }

    return NULL;
}
```

Test Scenario (Expected Results)

Scenario Overview:

1. Separate Spots for Cars and Pickups:

The code defines separate counters for temporary and permanent parking spots for automobiles and pickups.

```
9   #define MAX_AUTOMOBILE 8
10  #define MAX_PICKUP 4
11
12  // Shared parking space counters
13  int mFreeTemp_automobile = MAX_AUTOMOBILE; // Number of available temporary parking spots for automobiles
14  int mFreeTemp_pickup = MAX_PICKUP; // Number of available temporary parking spots for pickups
15  int mFreePerm_automobile = MAX_AUTOMOBILE; // Number of available permanent parking spots for automobiles
16  int mFreePerm_pickup = MAX_PICKUP; // Number of available permanent parking spots for pickups
17
```

2. Random Vehicle Type Generation at the Entrance:

The randomVehicleType function generates a random vehicle type (0 for pickup, 1 for automobile).

```
93  // Function to generate a random vehicle type (0: Pickup, 1: Automobile)
94  int randomVehicleType() {
95      return rand() % 2; // 0: Pickup, 1: Automobile
96  }
```

3. Single Vehicle Entry:

Vehicles are created with a delay to simulate staggered arrivals, ensuring that only one vehicle enters the system at a time:

```
40      // Create carOwner threads (simulating vehicle arrivals)
41      pthread_t owners[20];
42      for (int i = 0; i < 20; i++) {
43          pthread_create(&owners[i], NULL, carOwner, (void*) (long) randomVehicleType());
44          usleep(100000); // Simulate staggered arrivals (introduce a delay between arrivals)
45      }
46
```

Workflow:

1. carOwner Represents Vehicle Owners:

The carOwner function simulates vehicle owners arriving at the parking lot.

```
void* carOwner(void* arg) {
    int vehicleType = (int)(long) arg; // Vehicle type (0: Pickup, 1: Automobile)
    int isAutomobile = vehicleType; // Is it an automobile? (0: Pickup, 1: Automobile)

    if (isAutomobile) {
        pthread_mutex_lock(&lockTempAutomobile); // Lock temporary automobile parking
        if (mFreeTemp_automobile > 0) {
            // If there is space in the temporary lot, reduce available spots and signal the valet
            mFreeTemp_automobile--;
            printf("Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: %d\n", mFreeTemp_automobile);
            pthread_mutex_unlock(&lockTempAutomobile);
            sem_post(&newAutomobile);
            sem_wait(&inChargeforAutomobile); // Wait for the valet to park the automobile
        } else {
            pthread_mutex_unlock(&lockTempAutomobile);
            printf("Car Owner (Automobile) leaves due to no temporary spots.\n");
        }
    } else {
        pthread_mutex_lock(&lockTempPickup); // Lock temporary pickup parking
        if (mFreeTemp_pickup > 0) {
            // If there is space in the temporary lot, reduce available spots and signal the valet
            mFreeTemp_pickup--;
            printf("Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: %d\n", mFreeTemp_pickup);
            pthread_mutex_unlock(&lockTempPickup);
            sem_post(&newPickup);
            sem_wait(&inChargeforPickup); // Wait for the valet to park the pickup
        } else {
            pthread_mutex_unlock(&lockTempPickup);
            printf("Car Owner (Pickup) leaves due to no temporary spots.\n");
        }
    }

    return NULL;
}
```

2. carAttendant Represents Valets:

The carAttendant function simulates valets managing the parking lot.

```
void* carAttendant(void* arg) {
    char* type = (char*) arg; // Valet type (Automobile or Pickup)
    int isAutomobile = (strcmp(type, "Automobile") == 0); // Does the valet handle automobiles?

    while (running) {
        if (isAutomobile) {
            sem_wait(&newAutomobile); // Wait for a new automobile to arrive
            if (!running) break;
            pthread_mutex_lock(&lockPermAutomobile); // Lock permanent automobile parking
            if (mFreePerm_automobile > 0) {
                // If there is space in the permanent lot, reduce available spots and increase temporary parking
                mFreePerm_automobile--;
                pthread_mutex_lock(&lockTempAutomobile);
                mFreeTemp_automobile++;
                printf("Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: %d, Temporary Automobile Spots: %d\n",
                       mFreePerm_automobile, mFreeTemp_automobile);
                pthread_mutex_unlock(&lockTempAutomobile);
            } else {
                printf("No permanent spots available for Automobile.\n");
            }
            pthread_mutex_unlock(&lockPermAutomobile);
            sem_post(&inChargeforAutomobile); // Signal that the automobile has been parked
        } else {
            sem_wait(&newPickup); // Wait for a new pickup to arrive
            if (!running) break;
            pthread_mutex_lock(&lockPermPickup); // Lock permanent pickup parking
            if (mFreePerm_pickup > 0) {
                // If there is space in the permanent lot, reduce available spots and increase temporary parking
                mFreePerm_pickup--;
                pthread_mutex_lock(&lockTempPickup);
                mFreeTemp_pickup++;
                printf("Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: %d, Temporary Pickup Spots: %d\n",
                       mFreePerm_pickup, mFreeTemp_pickup);
                pthread_mutex_unlock(&lockTempPickup);
            } else {
                printf("No permanent spots available for Pickup.\n");
            }
            pthread_mutex_unlock(&lockPermPickup);
            sem_post(&inChargeforPickup); // Signal that the pickup has been parked
        }
    }

    return NULL;
}
```

Details:

1. Owners Confirm Parking Availability:

The car owners check for availability in the temporary parking lot before signaling the valet or leaving if no spots are available.

```
101
102     if (isAutomobile) {
103         pthread_mutex_lock(&lockTempAutomobile); // Lock temporary automobile parking
104         if (mFreeTemp_automobile > 0) {
105             // If there is space in the temporary lot, reduce available spots and signal the valet
106             mFreeTemp_automobile--;
107             printf("Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: %d\n", mFreeTemp_automobile);
108             pthread_mutex_unlock(&lockTempAutomobile);
109             sem_post(&newAutomobile);
110             sem_wait(&inChargeforAutomobile); // Wait for the valet to park the automobile
111         } else {
112             pthread_mutex_unlock(&lockTempAutomobile);
113             printf("Car Owner (Automobile) leaves due to no temporary spots.\n");
114         }
115     } else {
116         pthread_mutex_lock(&lockTempPickup); // Lock temporary pickup parking
117         if (mFreeTemp_pickup > 0) {
118             // If there is space in the temporary lot, reduce available spots and signal the valet
119             mFreeTemp_pickup--;
120             printf("Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: %d\n", mFreeTemp_pickup);
121             pthread_mutex_unlock(&lockTempPickup);
122             sem_post(&newPickup);
123             sem_wait(&inChargeforPickup); // Wait for the valet to park the pickup
124         } else {
125             pthread_mutex_unlock(&lockTempPickup);
126             printf("Car Owner (Pickup) leaves due to no temporary spots.\n");
127         }
128     }
```

2. Occupancy of Temporary and Permanent Spots is Handled Correctly:

The valets manage the occupancy of temporary and permanent spots, updating the counters accordingly.

```
138         if (isAutomobile) {
139             sem_wait(&newAutomobile); // Wait for a new automobile to arrive
140             if (!running) break;
141             pthread_mutex_lock(&lockPermAutomobile); // Lock permanent automobile parking
142             if (mFreePerm_automobile > 0) {
143                 // If there is space in the permanent lot, reduce available spots and increase temporary parking
144                 mFreePerm_automobile--;
145                 pthread_mutex_lock(&lockTempAutomobile);
146                 mFreeTemp_automobile++;
147                 printf("Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: %d, Temporary Automobile Spots: %d\n",
148                     mFreePerm_automobile, mFreeTemp_automobile);
149                 pthread_mutex_unlock(&lockTempAutomobile);
150             } else {
151                 printf("No permanent spots available for Automobile.\n");
152             }
153             pthread_mutex_unlock(&lockPermAutomobile);
154             sem_post(&inChargeforAutomobile); // Signal that the automobile has been parked
155         } else {
156             sem_wait(&newPickup); // Wait for a new pickup to arrive
157             if (!running) break;
158             pthread_mutex_lock(&lockPermPickup); // Lock permanent pickup parking
159             if (mFreePerm_pickup > 0) {
160                 // If there is space in the permanent lot, reduce available spots and increase temporary parking
161                 mFreePerm_pickup--;
162                 pthread_mutex_lock(&lockTempPickup);
163                 mFreeTemp_pickup++;
164                 printf("Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: %d, Temporary Pickup Spots: %d\n",
165                     mFreePerm_pickup, mFreeTemp_pickup);
166                 pthread_mutex_unlock(&lockTempPickup);
167             } else {
168                 printf("No permanent spots available for Pickup.\n");
169             }
```

3. Semaphores are Initialized and Used Appropriately:

The initialization and destruction of semaphores are handled correctly to ensure proper synchronization.

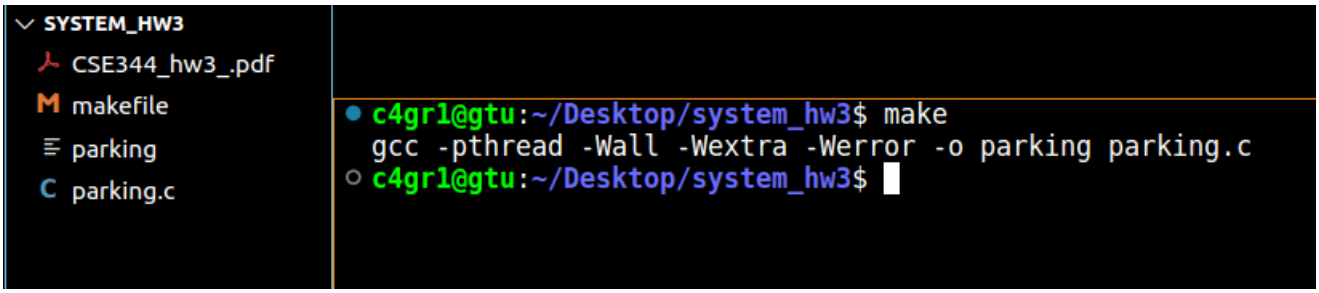
```c
void initialize() {
    // Initialize semaphores
    sem_init(&newPickup, 0, 0);
    sem_init(&newAutomobile, 0, 0);
    sem_init(&inChargeforPickup, 0, 1);
    sem_init(&inChargeforAutomobile, 0, 1);
    // Initialize mutex locks
    pthread_mutex_init(&lockTempAutomobile, NULL);
    pthread_mutex_init(&lockTempPickup, NULL);
    pthread_mutex_init(&lockPermAutomobile, NULL);
    pthread_mutex_init(&lockPermPickup, NULL);
    // Seed for random number generation
    srand(time(NULL));
}

void destroy() {
    // Destroy semaphores
    sem_destroy(&newPickup);
    sem_destroy(&newAutomobile);
    sem_destroy(&inChargeforPickup);
    sem_destroy(&inChargeforAutomobile);
    // Destroy mutex locks
    pthread_mutex_destroy(&lockTempAutomobile);
    pthread_mutex_destroy(&lockTempPickup);
    pthread_mutex_destroy(&lockPermAutomobile);
    pthread_mutex_destroy(&lockPermPickup);
}
```

Terminal Outputs

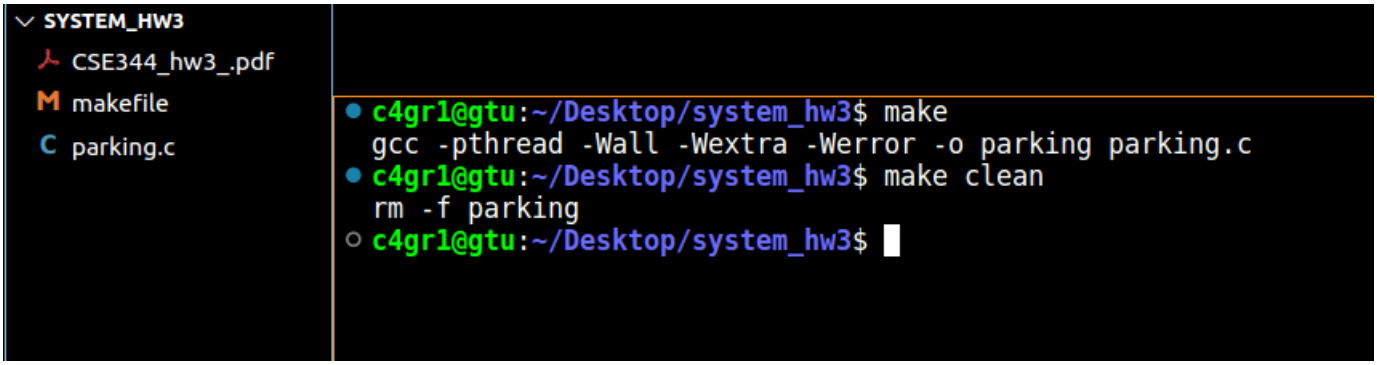
1. make or make all

This command compiles your source file (parking_lot.c) into the executable named parking. It uses the GCC compiler with pthread support enabled.



2. make clean

This command removes the compiled executable and cleans up any intermediate files. It is helpful for cleaning up your project directory and ensuring that subsequent builds start from a clean state.



3. make run

This command runs the compiled executable parking.

```
c4gr1@gtu:~/Desktop/system_hw3$ make run
gcc -pthread -Wall -Wextra -Werror -o parking parking.c
./parking
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: 3, Temporary Pickup Spots: 4
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 7, Temporary Automobile Spots: 8
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: 2, Temporary Pickup Spots: 4
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: 1, Temporary Pickup Spots: 4
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: 0, Temporary Pickup Spots: 4
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 6, Temporary Automobile Spots: 8
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
No permanent spots available for Pickup.
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 2
No permanent spots available for Pickup.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 5, Temporary Automobile Spots: 8
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 1
No permanent spots available for Pickup.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 4, Temporary Automobile Spots: 8
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 3, Temporary Automobile Spots: 8
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 2, Temporary Automobile Spots: 8
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 1, Temporary Automobile Spots: 8
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 0
No permanent spots available for Pickup.
Car Owner (Pickup) leaves due to no temporary spots.
Car Owner (Pickup) leaves due to no temporary spots.
Car Owner (Pickup) leaves due to no temporary spots.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 0, Temporary Automobile Spots: 8
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
No permanent spots available for Automobile.
c4gr1@gtu:~/Desktop/system_hw3$
```

```
c4gr1@gtu:~/Desktop/system_hw3$ make run
./parking
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: 3, Temporary Pickup Spots: 4
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: 2, Temporary Pickup Spots: 4
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 7, Temporary Automobile Spots: 8
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 6, Temporary Automobile Spots: 8
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: 1, Temporary Pickup Spots: 4
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 5, Temporary Automobile Spots: 8
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: 0, Temporary Pickup Spots: 4
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
No permanent spots available for Pickup.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 4, Temporary Automobile Spots: 8
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 2
No permanent spots available for Pickup.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 3, Temporary Automobile Spots: 8
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 1
No permanent spots available for Pickup.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 2, Temporary Automobile Spots: 8
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 0
No permanent spots available for Pickup.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 1, Temporary Automobile Spots: 8
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 0, Temporary Automobile Spots: 8
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
No permanent spots available for Automobile.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 6
No permanent spots available for Automobile.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 5
No permanent spots available for Automobile.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 4
No permanent spots available for Automobile.
c4gr1@gtu:~/Desktop/system_hw3$
```

4. make valgrind

This command runs your program under Valgrind to check for memory leaks. This is crucial for ensuring your program manages memory correctly and doesn't have leaks, which can cause problems in production environments.

```
c4gr1@gtu:~/Desktop/system_hw3$ make valgrind
valgrind --leak-check=full --show-leak-kinds=all --track-origins=yes ./parking
==38735== Memcheck, a memory error detector
==38735== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==38735== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==38735== Command: ./parking
==38735==
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 7, Temporary Automobile Spots: 8
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: 3, Temporary Pickup Spots: 4
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: 2, Temporary Pickup Spots: 4
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: 1, Temporary Pickup Spots: 4
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: 0, Temporary Pickup Spots: 4
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
No permanent spots available for Pickup.
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 2
No permanent spots available for Pickup.
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 1
No permanent spots available for Pickup.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 6, Temporary Automobile Spots: 8
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 5, Temporary Automobile Spots: 8
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 0
No permanent spots available for Pickup.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 4, Temporary Automobile Spots: 8
Car Owner (Pickup) leaves due to no temporary spots.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 3, Temporary Automobile Spots: 8
Car Owner (Pickup) leaves due to no temporary spots.
Car Owner (Pickup) leaves due to no temporary spots.
Car Owner (Pickup) leaves due to no temporary spots.
Car Owner (Pickup) leaves due to no temporary spots.
Car Owner (Pickup) leaves due to no temporary spots.
Car Owner (Pickup) leaves due to no temporary spots.
==38735==
==38735== HEAP SUMMARY:
==38735==     in use at exit: 0 bytes in 0 blocks
==38735==   total heap usage: 23 allocs, 23 frees, 7,008 bytes allocated
==38735==
==38735== All heap blocks were freed -- no leaks are possible
==38735==
==38735== For lists of detected and suppressed errors, rerun with: -s
==38735== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
c4gr1@gtu:~/Desktop/system_hw3$
```

```
c4gr1@gtu:~/Desktop/system_hw3$ make valgrind
valgrind --leak-check=full --show-leak-kinds=all --track-origins=yes ./parking
==38933== Memcheck, a memory error detector
==38933== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==38933== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==38933== Command: ./parking
==38933==
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: 3, Temporary Pickup Spots: 4
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: 2, Temporary Pickup Spots: 4
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: 1, Temporary Pickup Spots: 4
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
Valet parks a Pickup in the permanent lot. Available Permanent Pickup Spots: 0, Temporary Pickup Spots: 4
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 7, Temporary Automobile Spots: 8
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 6, Temporary Automobile Spots: 8
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 3
No permanent spots available for Pickup.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 5, Temporary Automobile Spots: 8
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 2
No permanent spots available for Pickup.
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 1
No permanent spots available for Pickup.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 4, Temporary Automobile Spots: 8
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 3, Temporary Automobile Spots: 8
Car Owner (Pickup) arrived at temporary lot. Available Temporary Pickup Spots: 0
No permanent spots available for Pickup.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 2, Temporary Automobile Spots: 8
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 1, Temporary Automobile Spots: 8
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
Valet parks an Automobile in the permanent lot. Available Permanent Automobile Spots: 0, Temporary Automobile Spots: 8
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 7
No permanent spots available for Automobile.
Car Owner (Pickup) leaves due to no temporary spots.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 6
No permanent spots available for Automobile.
Car Owner (Automobile) arrived at temporary lot. Available Temporary Automobile Spots: 5
No permanent spots available for Automobile.
==38933==
==38933== HEAP SUMMARY:
==38933==     in use at exit: 0 bytes in 0 blocks
==38933==   total heap usage: 23 allocs, 23 frees, 7,008 bytes allocated
==38933==
==38933== All heap blocks were freed -- no leaks are possible
==38933==
==38933== For lists of detected and suppressed errors, rerun with: -s
==38933== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
c4gr1@gtu:~/Desktop/system_hw3$
```

THANKS FOR READING

ÇAĞRI YILDIZ

1901042630