Operating Systems - 2021

Project 2 – Carwash

Implemented in [Java] using [IntelliJ IDEA]

Alireza Khorshidi (9735413)

Link to the project on GitHub
Click

Table of Contents

Introduction

This documentation contains information on an implementation of the second project of the course *Operating Systems* in Spring 2021.

The goal is to practice resource allocation to threads that run simultaneously using semaphores. Java is used to implement the following solution.

More details on the project are specified in the assignment file & the following Project overview section.

Project overview

The goal is to simulate a number of cars entering a carwash that which has a limited capacity for cars. The carwash has 3 booths; A & B are for washing and C is for drying. Each booth has a maximum capacity of 1 car at a time. Every car must first be washed in either A or B (the one that's free) for 100 milliseconds and then proceed to booth C in order to be dried in 300 milliseconds, and finally proceed to exit, making room for the remaining cars to enter the carwash.

The booths and the carwash have dedicated semaphores which will "wait" upon a car's entrance to them and "signal" upon a car's exit from them. While implementing the solution, other rules of concurrency like Mutual Exclusion have been taken into consideration, no thread can modify another thread's values.

The carwash can be initialized in 2 modes, one with booth C automated and the other with it requiring a worker. The worker will rest in short intervals and announces that he's taking a break. Note that no car is washed during the worker's break.

Classes

Main

Start the application from here.

- public void Main(String args[])
  Gets input from the user, initializes the cars and the carwash and runs the cars.

Carwash

Responsible for holding references to semaphores and the worker for all cars to interact with.

- public final Semaphore capacity
  The number of cars that can be present in the carwash simultaneously.

- public final Semaphore boothA
  The number of cars that can be present in the booth A simultaneously.

- public final Semaphore boothB
  The number of cars that can be present in the booth B simultaneously.

- public final Worker worker
  Reference to booth C's worker (is set to null if booth C is automated).

- private long washingSpan
  The amount of time it takes for Booths A and B to finish washing a car (milliseconds).

- private long dryingSpan
  The amount of time it takes for Booths C to finish drying a car (milliseconds).

- public Game(int carwashCapacity, int boothACapacity
  , int boothBCapacity, int boothCCapacity, Boolean isAutomated)
  Constructor. Initializes the semaphores with corresponding integer values. Initializes the woker only if the carwash is not automated, otherwise leaves it as null. Also sets washingSpan to 100 milliseconds and dryingSpan to 300 milliseconds by default.

- public long getWashingSpan()
  Returns the washingSpan.

- public long getDryingSpan()
  Return the dryingSpan.

Car

A model of every car. It also is a thread as well.

- private String carName
  The name of the car. Helps identify each car.

- private Carwash carwash
  A reference to the carwash which the car wants to enter.

- private Message messageTerminal
  A dedicated object used for announcing the car's current state to terminal.

- private CarState carState
  The car's current state.

- public Car (int id, String name, Game parentGame)
  Constructor. Assigns the arguments' values to their corresponding class fields. Also sets threadName to carName using the parent's constructor (Thread.super). Also initializes messageTerminal to a new object & sets the current state to CarState.BORN.

- public void run()
  An implementation of Thread.run() for the *Car* class. Its functionality is pretty straightforward; Request and wait for proper resources using the corresponding semaphore by the carwash reference, update the state machine, and send messages to the terminal accordingly. If the carwash booth C is not automated, then at the time of entrance to that booth, waits for the worker to finish resting.

Worker

A model for a worker. It also is a thread as well.

- private Carwash carwash
  A reference to the carwash at which the worker operates booth C.

- public Worker (Carwash carwash)
  Constructor. Assigns the arguments' values to their corresponding class fields.

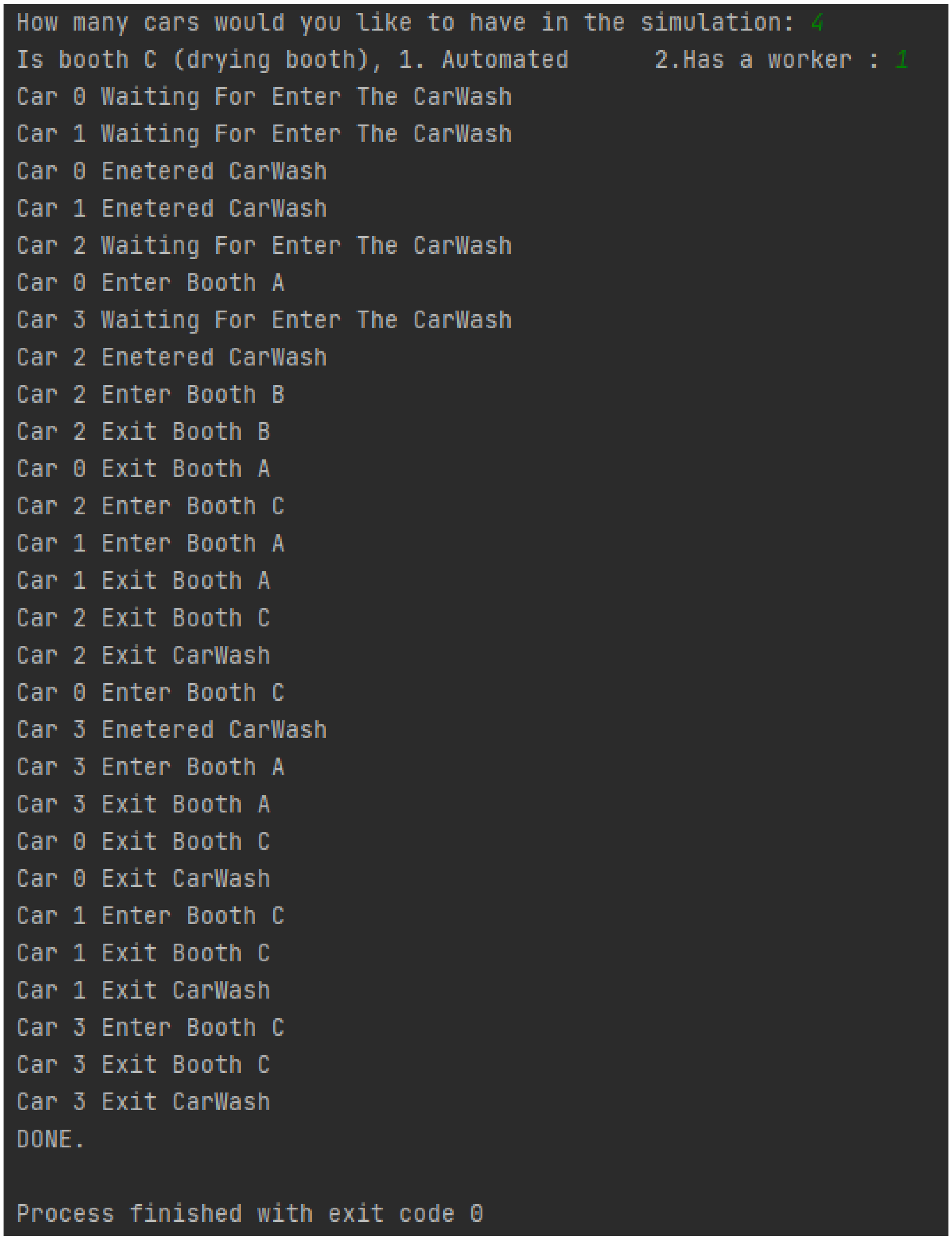
- public void run()
  An implementation of Thread.run() for the *Car* class. Puts the thread to sleep for 150 milliseconds (The duration of a break for the worker).

Message

Used for printing a car's status to the terminal.

Sample Tests

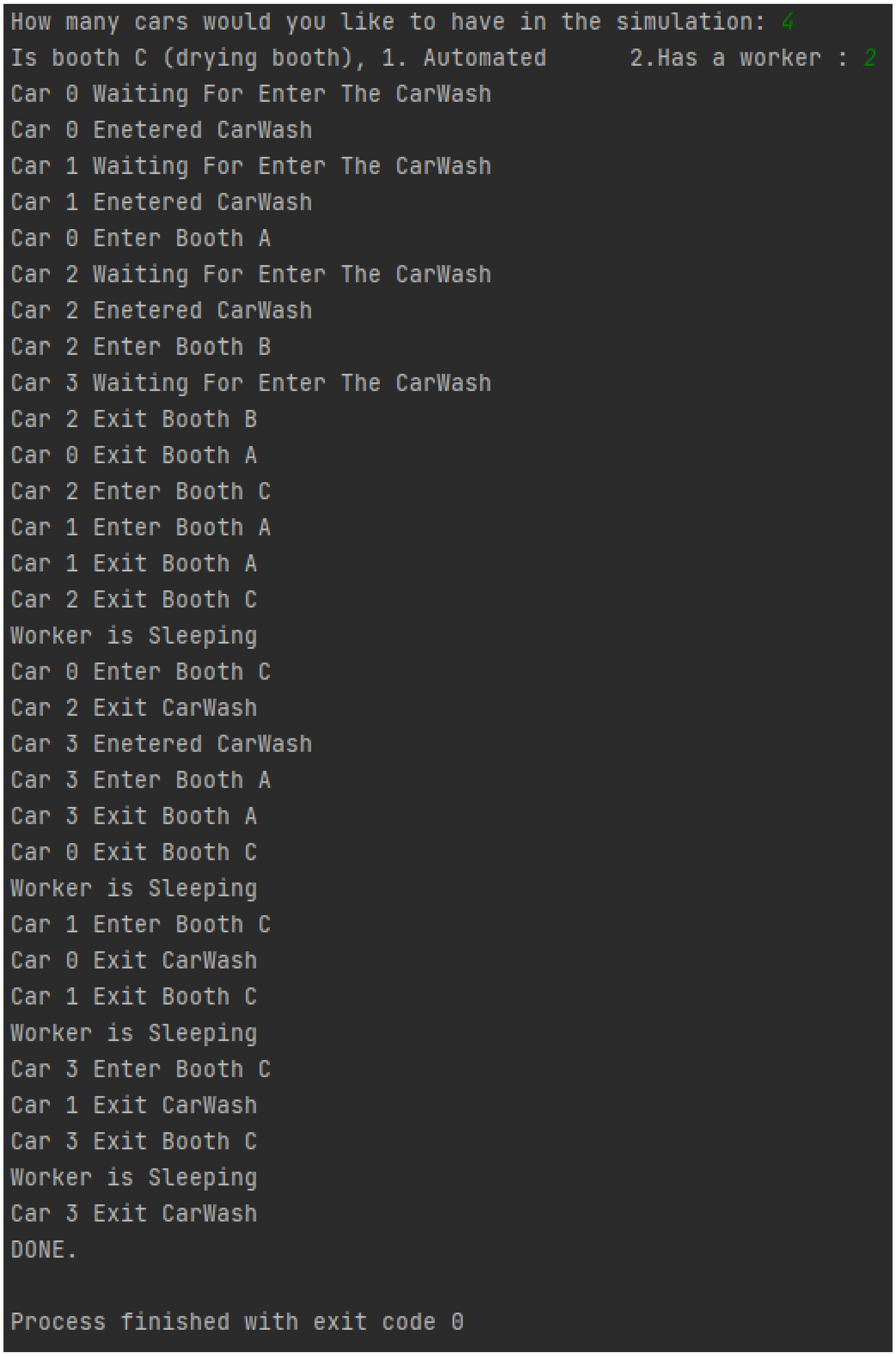
With 4 cars & Automated booth C:



```
How many cars would you like to have in the simulation: 4
Is booth C (drying booth), 1. Automated      2.Has a worker : 1
Car 0 Waiting For Enter The CarWash
Car 1 Waiting For Enter The CarWash
Car 0 Enetered CarWash
Car 1 Enetered CarWash
Car 2 Waiting For Enter The CarWash
Car 0 Enter Booth A
Car 3 Waiting For Enter The CarWash
Car 2 Enetered CarWash
Car 2 Enter Booth B
Car 2 Exit Booth B
Car 0 Exit Booth A
Car 2 Enter Booth C
Car 1 Enter Booth A
Car 1 Exit Booth A
Car 2 Exit Booth C
Car 2 Exit CarWash
Car 0 Enter Booth C
Car 3 Enetered CarWash
Car 3 Enter Booth A
Car 3 Exit Booth A
Car 0 Exit Booth C
Car 0 Exit CarWash
Car 1 Enter Booth C
Car 1 Exit Booth C
Car 1 Exit CarWash
Car 3 Enter Booth C
Car 3 Exit Booth C
Car 3 Exit CarWash
DONE.

Process finished with exit code 0
```

With 4 cars & a manual booth C (requires a worker):



```
How many cars would you like to have in the simulation: 4
Is booth C (drying booth), 1. Automated      2.Has a worker : 2
Car 0 Waiting For Enter The CarWash
Car 0 Enetered CarWash
Car 1 Waiting For Enter The CarWash
Car 1 Enetered CarWash
Car 0 Enter Booth A
Car 2 Waiting For Enter The CarWash
Car 2 Enetered CarWash
Car 2 Enter Booth B
Car 3 Waiting For Enter The CarWash
Car 2 Exit Booth B
Car 0 Exit Booth A
Car 2 Enter Booth C
Car 1 Enter Booth A
Car 1 Exit Booth A
Car 2 Exit Booth C
Worker is Sleeping
Car 0 Enter Booth C
Car 2 Exit CarWash
Car 3 Enetered CarWash
Car 3 Enter Booth A
Car 3 Exit Booth A
Car 0 Exit Booth C
Worker is Sleeping
Car 1 Enter Booth C
Car 0 Exit CarWash
Car 1 Exit Booth C
Worker is Sleeping
Car 3 Enter Booth C
Car 1 Exit CarWash
Car 3 Exit Booth C
Worker is Sleeping
Car 3 Exit CarWash
DONE.

Process finished with exit code 0
```