**Problem 2. Parking**

Exam problems for the [“JavaScript Advanced” course @ SoftUni](https://softuni.bg/courses/javascript-advanced). Submit your solutions in the SoftUni Judge system at <https://judge.softuni.bg/Contests/Compete/Index/2590#2>

Write a **class Parking**, which implements the following functionality:

**Functionality**

**constructor ( capacity )**

Should have these **2** properties:

* **capacity** – **number**;
* **vehicles** – **array**;

**Hint:** You can add more properties to help you finish the task.

**addCar( carModel, carNumber )**

The **carModel** and **carNumber** are of type **string**.

* If there's **not enough parking spots** for the car the park, **throw an Error**:

**"Not enough parking space."**

* Otherwise this function should **add** the car, with the properties: **carModel**, **carNumber**, **payed**: **default false**, to the vehicles arrayand **return:**

**"The {carModel}, with a registration number {carNumber}, parked."**

**removeCar( carNumber )**

* If the car is not found, throw an Error:

**"The car, you're looking for, is not found."**

* If the car hasn't payed, throw an Error:

**"{carNumber} needs to pay before leaving the parking lot."**

* Otherwise, this function should **remove** the car from the vehicles arrayand **return:**

**"{carNumber} left the parking lot."**

**pay( carNumber )**

* If the car is not found, throw an Error:

**"{carNumber} is not in the parking lot."**

* If the car has already payed, throw an Error:

**"{carNumber}'s driver has already payed his ticket."**

* Otherwise, this function set payed to true on the found car and **return:**

**"{carNumber}'s driver successfully payed for his stay."**

**getStatistics(carNumber)**

This **method** can be called **with one parameter** or **without** any.

If **NO** parameter is provided, the method should **return** the full information of the parking lot.

* At the first line:

**"The Parking Lot has { emptySlots } empty spots left."**

* On the lines, display information about each vehicle**, sorted alphabetically ascending** by their **carModel:**

**"{carModel} == {carNumber} - {Has payed / Not payed}"**

If the method is called with **parameter** for **carNumber**:

* **return only** the **information about the car with the given carNumber:**

**"{carModel} == {carNumber} - {Has payed / Not payed}"**

**Examples**

|  |
| --- |
| **Sample code usage** |
| **const parking = new Parking(12);**  **console.log(parking.addCar("Volvo t600", "TX3691CA"));**  **console.log(parking.getStatistics());**  **console.log(parking.pay("TX3691CA"));**  **console.log(parking.removeCar("TX3691CA"));** |
| **Corresponding output** |
| **The Volvo t600, with a registration number TX3691CA, parked.**  **The Parking Lot has 11 empty spots left.**  **Volvo t600 == TX3691CA - Not payed**  **TX3691CA's driver successfully payed for his stay.**  **TX3691CA left the parking lot.** |