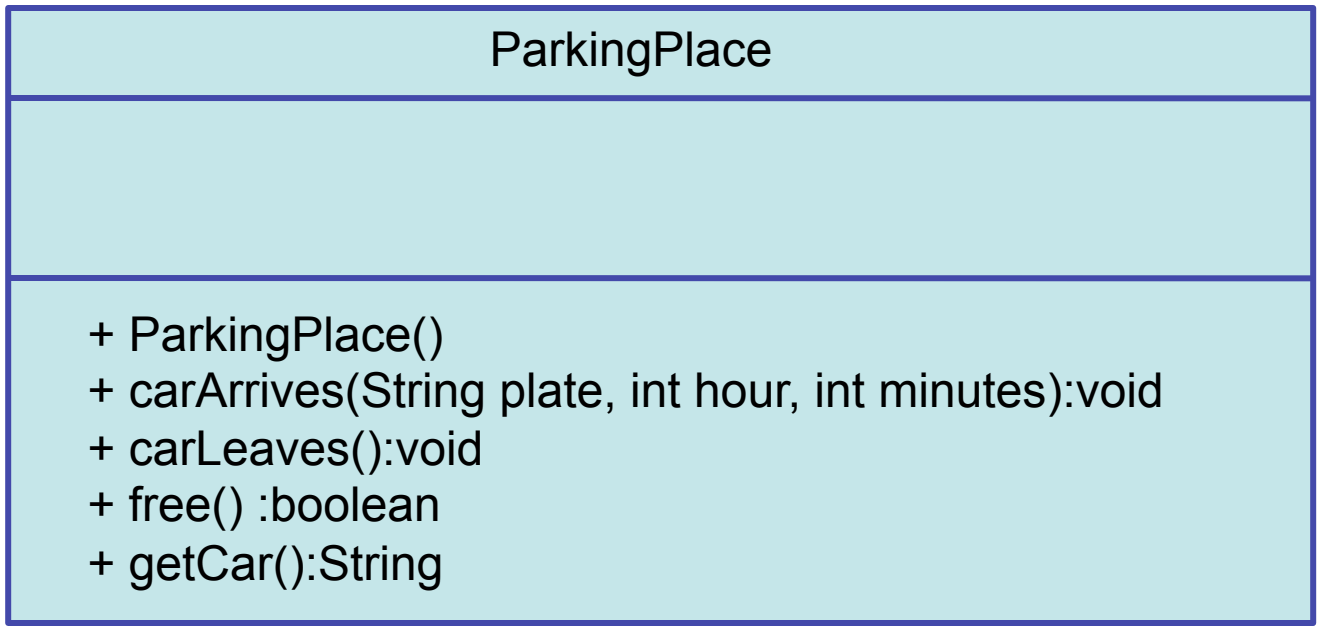
Lab 11 – Arrays of Objects

This exercise is a continuation of the last exercise presented in the previous lab. Assume the ParkingPlace class in this lab is given. You can find it in ParkingPlace.java file.



Methods of ParkingPlace class

•ParkingPlace() : constructor without parameters that constructs a parking place that is initially free;
•String toString() : that returns "-------" if the parking place is free, and the licence plate of the car, if the parking place is occupied;
•void carArrives(String plate, int hour, int minutes) : modifies the state of the parking place by setting it to occupied, sets the plate of the car that occupies the place to plate, and sets the time since when it is occupied to hour and minutes; if the parking place is already occupied, the method does nothing;
•void carLeaves() : modifies the state of the parking place by setting it to free;
•boolean free() : returns true if the parking place is free, false otherwise;
•String getCar() : returns the plate of the car that occupies the parking place, if the place is occupied, null otherwise;
•int getHour() : returns the hour since when the parking place is occupied, if the place is occupied, -1 otherwise;
•int getMinutes() : returns the minutes since when the parking place is occupied, if the place is occupied, -1 otherwise;

Exercise 1

- Write a class UseParkingPlace that contains various public static methods that are clients of ParkingPlace. In the description of all methods below, we assume that a parking lot is always represented as an array of parking places. The class should contain the following methods:
  - static int firstFreePlace(ParkingPlace[] p) : that, given a parking lot p, returns the index of the first free parking place in p;
  - static int countFreePlaces(ParkingPlace[] p) : that, given a parking lot p, returns the number of free parking places in p;
  - static int[] freePlaces(ParkingPlace[] p) : that, given a parking lot p, returns an array of integers containing the indices of all free parking places in p;
  - static void carEnters(ParkingPlace[] p, String a, int hour, int minutes) : that, given a parking lot p, modifies the array p by inserting the car a (represented by its plate) in the first free parking place available in p, assigning hour and minutes as arrival time; if there is no free parking place inp, the method does nothing;

Exercise 1 (Con' d)

- static void carExits(ParkingPlace[] p, String a) : that, given a parking lot p, modifies the array p by freeing the parking place where the car ais parked; if the car a is not present in the parking lot, the method does nothing;

- static int longestParkedCar(ParkingPlace[] p) : that, given a parking lot p, returns the index of the parking place in which the longest parked car is present; if there is more than one such car, the method should return one of the indexes (chosen arbitrarily); if there is no car parked in the parking lot, the method should return -1;
- static String[] allParkedCars(ParkingPlace[] p) : that, given a parking lot p, returns an array of strings that represent the plates of all cars present in p;

Exercise 2

Realize a class ParkingLot to represent parking lots, and whose objects support the same functionalities through instance methods as those implemented in UseParkingPlace through static methods. The class should have a constructor that takes as parameter a positive integer n and constructs a parking lot with n parking places that initially are all empty.

Hint: Note that you do not have to "recode" all the methods. To create instance methods in ParkingLot class you can reuse the developed static methods. Eg:

```
public class ParkingLot{
private ParkingPlace[] places;
…….
public int firstFreePlace() {
    return UseParkingPlace.firstFreePlace(this.places);
}
```