COM498 Algorithms & Data Structures

Assignment 1 – Part 2

You have been provided with the skeleton of the file *CarPark.java*, which defines a `CarPark` data type as a `Bag` of `String` objects representing a collection of names representing the drivers currently parked. In this implementation, the car park only has 3 spaces available, and no driver can use more than one space (i.e., all names stored in the data structure are unique).

The following methods on `CarPark` objects are available

`public CarPark(int size)`

> *A constructor that creates a new Car Park of* `size` *spaces.*

`public String showAllCars()`

> *A method that returns a string representation of the current car park users*

`public boolean parkCar(String name)`

> *A method that adds a driver to the car park, representing a parking event. In this implementation, all driver names are assumed to be unique, and no name can be stored more than once in the* `CarPark` *structure. The method returns* `true` *if the operation is successful, and* `false` *otherwise.*

`public boolean removeCar(String name)`

> *A method to remove a driver from the car park, representing a vehicle leaving the parking space. The method returns* `true` *if the operation is successful, and* `false` *otherwise.*

`public boolean isParked(String name)`

> *A method that returns* `true` *if the parameter is contained in the structure, and* `false` *otherwise.*

`public boolean replace(String leaving, String parking)`

> *A method that replaces the car matching the* `leaving` *parameter with one matching the* `parking` *parameter, representing a new driver using a previously occupied car park space. The method returns* `true` *if the operation is successful, and* `false` *otherwise.*

`public int getTotalParked()`

> *A method that returns an integer representing the total number of car park users – i.e., all vehicles that have been parked.*

In addition, you have the file *CarParkTest.java*, which creates a new Car Park consisting of 3 spaces and simulates a number of car park events.

Your task is to provide the remaining code for the class `CarPark`, to have the methods exhibit the behaviour as described above.

Tasks

1. Create a new Java Project in IntelliJ called **Assignment1** and copy the files *CarPark.java*, *CarParkTest.java*, *BagInterface.java* and *ArrayBag.java* into its **src** folder.

2. Provide the code for **`parkCar()`** to ensure that each name stored in the structure is unique. If a duplicate name is presented as a parameter, the method should return **`false`** and the element should not be added.

3. Provide the code for **`removeCar()`** to remove the element matching the string parameter. If the parameter is not contained in the structure, the method should return **`false`**.

4. Provide the code for **`isParked()`** to return **`true`** if the parameter is stored in the structure and **`false`** otherwise.

5. Provide the code for **`replace()`** to remove the element matching the **`leaving`** parameter and add an element corresponding to the **`parking`** parameter – while observing the rule on unique names. If, for any reason, the replace operation cannot be completed, the method should return **`false`** and the data structure should be unchanged, otherwise **`true`** is returned.

6. Provide additional code wherever it is needed so that the method **`getTotalParked()`** returns the total number of parking events – i.e. the total should be incremented each time a driver name is added to the structure.

   Note: If you are unable to provide code for any of these, marks may be awarded for relevant pseudocode provided as comments.

7. Run the **`main()`** method in the file *CarParkTest.java*, which conducts tests on your new methods. If any of these tests fail, you should spend any remaining time debugging your code and re-running the tests. (Note: The output you should see after a successful test is shown on the following page.)

8. Take a screenshot of your final attempt at running **CarParkTest** and upload this, along with your source code from *CarPark.java* to the link provided on Blackboard.

You should modify ONLY the code in CarPark.java. No other Java files should be modified.

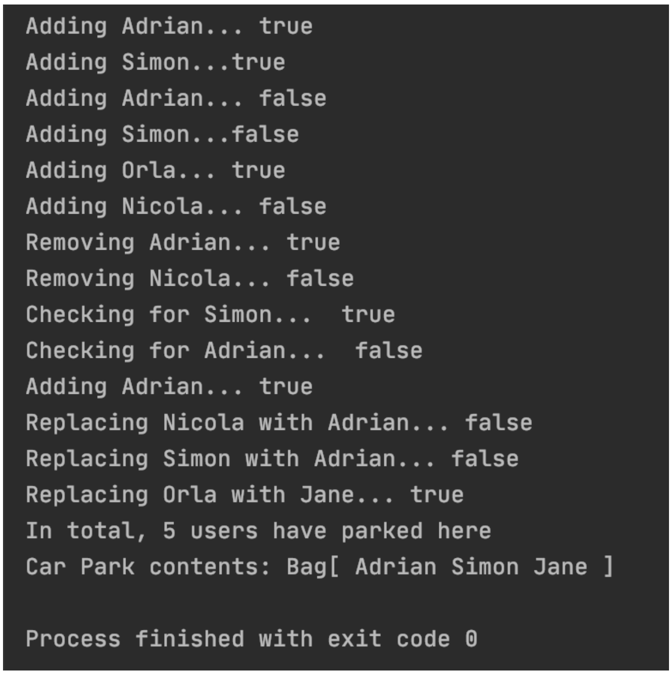
Marking Scheme

60% of the marks for Assignment 1 are available from this exercise. Marks will be allocated as follows:

Element	Marks
Implementation of **`parkCar()`** method	12
Implementation of **`removeCar()`** method	12
Implementation of **`isParked()`** method	12
Implementation of **`replace()`** method	12
Implementation of **`getTotalParked()`** method	12

Target output

If your **`CarPark`** class has been properly implemented, running the **`main()`** method in the **`CarParkTest`** class should generate the following output.



```
Adding Adrian... true
Adding Simon...true
Adding Adrian... false
Adding Simon...false
Adding Orla... true
Adding Nicola... false
Removing Adrian... true
Removing Nicola... false
Checking for Simon...  true
Checking for Adrian...  false
Adding Adrian... true
Replacing Nicola with Adrian... false
Replacing Simon with Adrian... false
Replacing Orla with Jane... true
In total, 5 users have parked here
Car Park contents: Bag[ Adrian Simon Jane ]

Process finished with exit code 0
```

Submission

Please submit a screenshot of the output from your final attempt at running **`CarparkTest`** and your file *CarPark.java* to the **Assessment 1 Part 2** link in the **Assessment** section on Blackboard.