COS70006: Object Oriented Programming
Project 1 - Semester 2, 2023

20% of your final mark

Due by at the end of Week 6 (11:59pm Sunday 10 Sept 2023)

NOTE: Extensions must be applied for before the due date by emailing the unit convener. A request for extension after the due time or without a valid medical certificate will **NOT** be responded to.

Project 1 – Parking Spot System for Selling Used Cars

Requirements

A company for selling used cars has a car park site. A small system is required that will help manage used cars at a parking site for the company. You are to develop a system that should has following four classes:

- Application class
- CarPark class
- ParkingSlot class
- Car class

Application is the Console (Text Based) Interface class including the main() method and handling all inputs and outputs.

CarPark is responsible for maintaining a list of available parking slots. You should be able to add a slot, delete a slot, provide a list of all slots included in the car park, park a car, find / remove a car by registration number, and find cars by the make.

A parking slot must have an identifier, which starts with a capital letter, followed by a three-digit number e.g. “D001”, “E127”. A parking slot should also know if a car is parked in the slot. You must be able to add a car to the slot and remove a car from the slot.

A car will be identified by its registration number. A registration number always starts with a capital letter, followed by a four-digit number e.g. “T1234”. A car should also have its make, model and year.

For this assessment, you do NOT need to maintain a list of parked cars in any of your classes.

Basic required functions (80 marks):
The office manager requires a simple Console (Text Based) Interface that will have the following menu items.

1. Add a parking slot, all information provided by users
2. Delete a parking slot by slot ID (only if not occupied)
3. List all slots in a well-defined format with information including slot ID, whether occupied, and if occupied, show the car’s registration & make.
4. Park a car into a slot (provide slot ID and car information)

5. Find a car by registration number and show the slot and the registration number if the car is in.
6. Remove a car by registration number.
7. Find cars by car's make and show the slot and car's registration number, make, model & year if the cars are in
8. Exit. Need to show "Program end!" before exit.

Required conditions to be checked for user inputs:

1. User inputs for menu options, car information, and parking slot information should not crash the program
2. Parking slot number must be an uppercase letter followed by 3 digits.
3. Car registration number must be an uppercase letter followed by 4 digits
4. Each slot should have a unique slot number
5. A parking slot cannot be deleted if there is a car being parked there
6. A car can only be parked in an unoccupied slot
7. A car can only be parked in one slot

Popper messages for user inputs and outputs:

e.g. show proper format for slot ID / car registration number for user inputs;
show proper out messages to indicate something wrong / success.

Advanced features (20 marks):

Note: Make sure you get all the basics required above working first before you attempt the advanced features below:

- To **park a car**, the current time is recorded in the Car object and output to the screen (e.g. 2023-08-20 16:48:05).
- In the **list of all slots** and **find a car by make**, the parking time length for the occupied cars should also be shown up (e.g. 0 hours 25 minutes and 20 seconds)

Code:

- The solution must be a BlueJ Project.

Some Expectations.

1. All classes and methods include Javadoc
2. Code is well structured and object oriented.
3. User interface class (Application) is broken down into single purposed methods
4. User interface class (Application) is separated from business logic classes
5. The user input is safe and will not crash the program
6. The user should be well informed about what he/she is expected to enter and the feedback of their action.
7. Pre-condition checking is included in the class methods.

Plagiarism

THIS IS AN INDIVDUAL PROJECT.

- The submitted work must be your own work.
- You must keep your own work from other students.
- You may NOT view the code of other students.
- You may discuss the work with teaching staff.
- You may discuss the big picture with peers but the final design should be yours.
- You must name and code attributes and operations on your own.
- There will be absolutely no tolerance of plagiarism.
- Any person that presents any work that is not their own or is not properly referenced will be awarded 0 marks for the project.

Submission

Zip you BlueJ project into a file and submit it to ESP by the due time

Some sample runs:

```
Please select an option (1-7): 1
Please enter the slot ID (e.g. A001):
B002
The slot is added successfully

1: add a car slot
2: Delete a car slot
3: List all car slots
4: Park a car
5: Find a car by reg
6: Remove a car by reg
7: Find cars by make
8: Exit
Please select an option (1-7): 3
SlotID A001 is not occupied
SlotID A002 is not occupied
SlotID B002 is not occupied
```

```
Please select an option (1-7): 4
Please enter the slot ID (e.g. A001) that you want to park at: B002
Please enter the car registreation number (such as D1234): D1234
Please enter the car's make (e.g. Toyota):
Toyota
Please enter the car's model (e.g. Corolla):
Corolla
Please enter the car's year (e.g. 2009):
2009
The car has been parked successfully.
```

```
Please select an option (1-7): 3
SlotID A001 is occupied by a car with reg: W1234, make: Toyota
SlotID A002 is not occupied
SlotID B002 is occupied by a car with reg: D1234, make: Toyota
```

```
Please select an option (1-7): 5
Please enter the car registreation number (such as D1234): D1234
The car with reg=D1234 is parked on the slot=B002
```

```
Please select an option (1-7): 7
Please neter the car's make (e.g. Toyota):
Toyota
Slot ID A001: reg=W1234, Toyota, Camry, 2019
Slot ID B002: reg=D1234, Toyota, Corolla, 2009
```

```
Please select an option (1-7): 7
Please neter the car's make (e.g. Toyota):
Mazda
The model (Mazda) is not found!
```

Marking Scheme

Total marks: 100

Note: Your program should not crash - (marks deduction will be up to 100 marks)

Items	Max Marks
Code readability: • Javadoc for all class headers and methods headers • Proper comments for variables and blocks • Proper indentations and use of blank lines • Proper Java naming conventions and meaningful names	3
Appropriate implementation of functionality using well-structured OO classes • User Interface class separated from business logic classes. • Broken down into single purposed methods. • Each class is defined as required. • Logic of code can be followed.	3
Each function in the menu **works** as required and required **conditions checked and proper user messages.** • Add a parking slot *(10 marks)* • Delete a parking slot (only if not occupied) *(10 marks)* • List all slots in a well-defined format with information including slot ID, whether occupied, and if occupied, show the car's registration & make. *(10 marks)* • Park a car to a specified slot *(10 marks)* • Find a car by registration number and show the slot and the registration number if the car is in. *(10 marks)* • Remove a car *(10 marks)* • Find cars by car's make and show the slot and car's registration number, make, model & year if the cars are in *(12 marks)* • Exit. Show "Program end!" before exit. *(2 marks)*	74
Advanced features *(8+12 marks)*	20
Total	100