

Assessment details COIT 11222

Assessment item 2—JAVA Program using array of objects

Due date:	Week 11 T116 – Midnight, Friday (27/5/16)	ASSESSMENT
	Refer below for complete assessment item 2 requirements (Assignment Two)	
Weighting:	20%	
Length:	N/A	2

Objectives

This assessment item relates to the course learning outcomes as stated in the Course Profile.

Details

For this assignment, you are required to develop a **Menu Driven Console Java Program** to demonstrate you can use Java constructs including input/output via GUI dialogs, Java primitive and built-in types, Java defined objects, arrays, selection and looping statements and various other Java commands. Your program must produce the correct results.

The code for the menu and option selection is supplied and is available on the course website, you must write the underlying code to implement the program. The menu selections are linked to appropriate methods in the given code. Please spend a bit of time looking at the given code to familiarise yourself with it and where you have to complete the code. You will need to write comments in the supplied code as well as your own additions.

What to submit for this assignment

The Java source code:

- **Car.java**
- **CarPark.java**

If you submit the source code with an incorrect name you will lose marks.

A report including an UML class diagram of your Car class (see text p 493), how long it took to create the whole program, any problems encountered and screen shots of the output produced. (Use Alt-PrtScrn to capture just the application window and you can paste it into your Word document) You should test every possibility in the program.

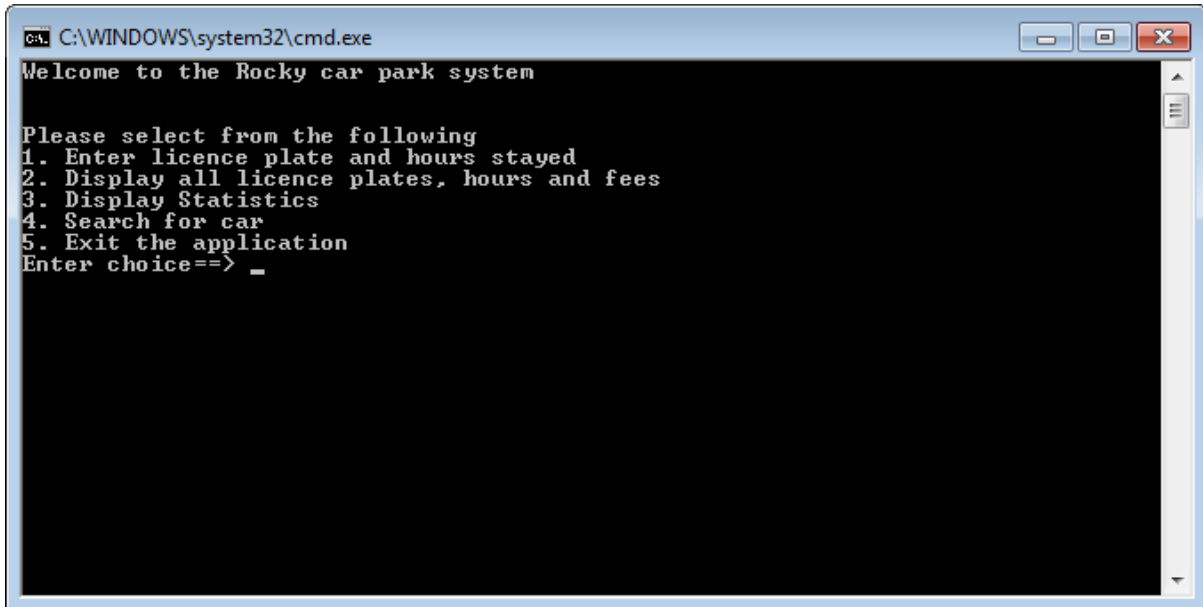
- **ReportAss2.docx**

You will submit your files by the due date using the “**Assignment 2**” link on the Moodle course website under **Assessment ... Assignment 2 Submission**.

Assignment Specification

You have completed the console program for processing cars parked at the Rocky car park. We are going to extend this application so the car's licence plates and hours spent parked can be stored in an array of objects.

The program will run via a menu of options, the file **CarPark.java** has been supplied (via the Moodle web site) which supplies the basic functionality of the menu system.



```
C:\WINDOWS\system32\cmd.exe
Welcome to the Rocky car park system

Please select from the following
1. Enter licence plate and hours stayed
2. Display all licence plates, hours and fees
3. Display Statistics
4. Search for car
5. Exit the application
Enter choice==> _
```

Look at the code supplied and trace the execution and you will see the menu is linked to blank methods (stubs) which you will implement the various choices in the menu.

Car class

First step is to create a class called Car (Car.java).

The Car class will be very simple it will contain two private instance variables:

- licencePlate as a String
- hours as an integer

You can also have constants for the fee values i.e \$7.50 and \$4.50.

The following public methods will have to be implemented:

- A default constructor
- A parameterised constructor
- Two set methods (mutators)
- Two get methods (accessors)
- A method to calculate and return the fee. This calculation will be the same as in assignment one i.e. First hour is \$7.50 and the remaining hours will be \$4.50 per hour.

Note: following basic database principles calculated values are not usually stored so in this case we will not store the fee as a instance variable, but use the `calculateFee()` method when we want to access the fee.

CarPark class

Once the Car class is implemented and fully tested we can now start to implement the functionality of the menu system.

Data structures

For this assignment we are going to store the licence plates and hours in an array of Car objects.

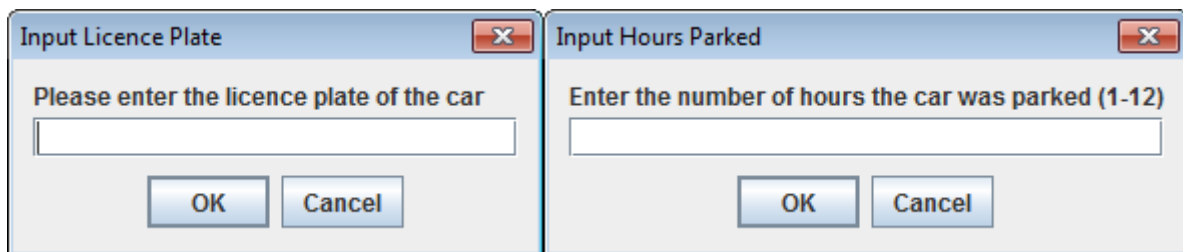
Declare an array of Car objects as an instance variable of CarPark class the array should hold twenty cars.

You will need another instance variable (integer) to keep track of the number of the cars being entered and use this for the index into the array of Car objects.

Menu options

1. Enter licence plate and hours parked: `enterCar()`

For assignment two we are going to use the GUI dialog `showInputDialog()` for our input. You will need to create the following two dialogs to receive the input from the user. You will not implement the functionality of the cancel key (need to use exceptions for this).



Data validation (you can implement this after you have got the basic functionality implemented)

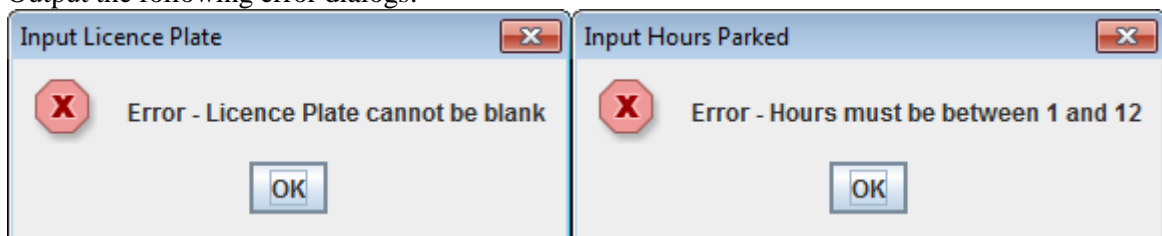
You will need to validate the user input using a validation loop.

The licence plate cannot be blank i.e. not null and the hours parked needs to be between one and twelve (1-12) inclusive.

Use the following pseudocode as a guide how to do this.

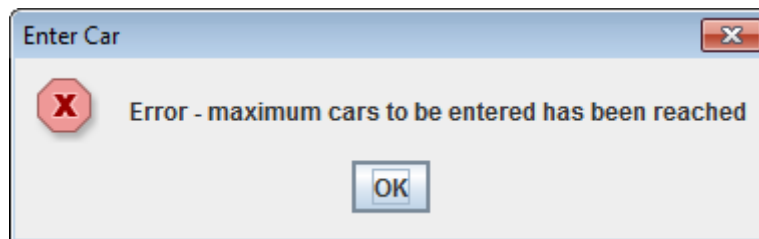
```
READ value
WHILE value is incorrect
    OUTPUT Error message
    READ value
END WHILE
```

Output the following error dialogs:



When the licence plate and hours parked has been entered successfully into two local variables you will need to add these values into the Car object array, you will also need to increment a counter to keep track of the number of cars you have entered and the position in the array of the next car to be entered.

When the maximum number of cars is reached do not attempt to add any more cars and give the following error message:



When the car details have been successfully entered display the details of the car and the fee to be charged on the console screen.

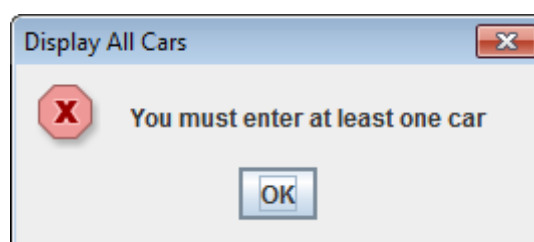
```
C:\WINDOWS\system32\cmd.exe
Welcome to the Rocky car park system

Please select from the following
1. Enter licence plate and hours stayed
2. Display all licence plates, hours and fees
3. Display Statistics
4. Search for car
5. Exit the application
Enter choice==> 1

Details for car 1 entered
Licence Plate      Hours      Fee
AAA111             3          $16.50

Please select from the following
1. Enter licence plate and hours stayed
2. Display all licence plates, hours and fees
3. Display Statistics
4. Search for car
5. Exit the application
Enter choice==>
```

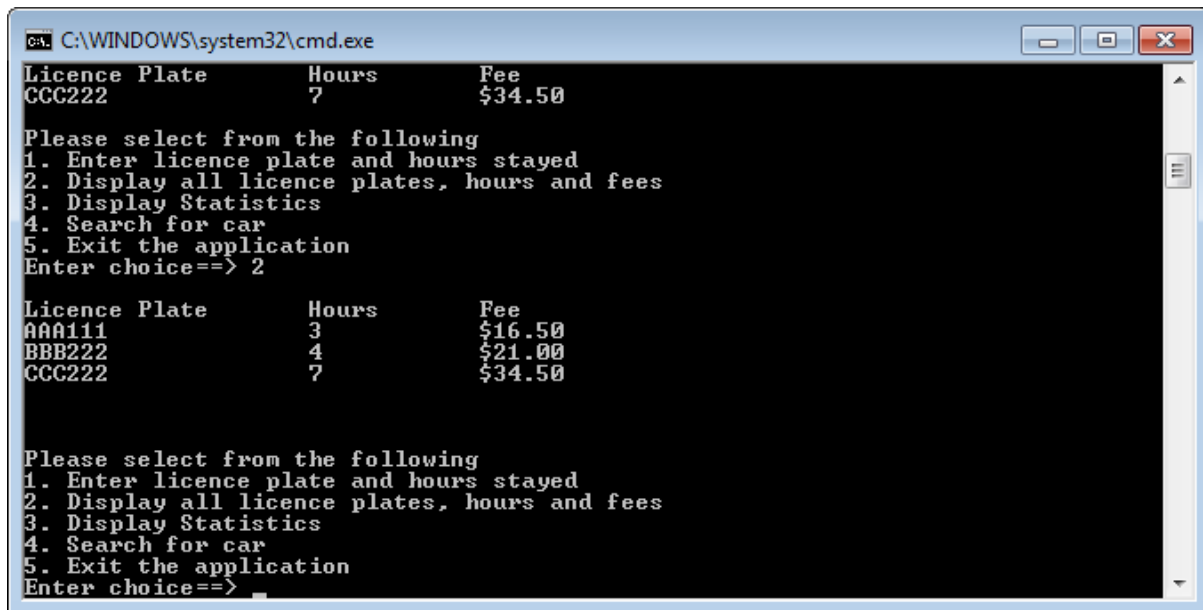
Note: For the next three options you should ensure at least one car has been entered and give an appropriate error message if there are no cars entered, for example:



2. Display all licence plates, hours and fees: `displayAllCars()`

When this option is selected display all of the cars which have been entered so far.

Review the text p.933 for the use of the `printf` method and using a format string. You could also use `String.format()` to format your output.



```
C:\WINDOWS\system32\cmd.exe
Licence Plate      Hours      Fee
CCC222             7          $34.50

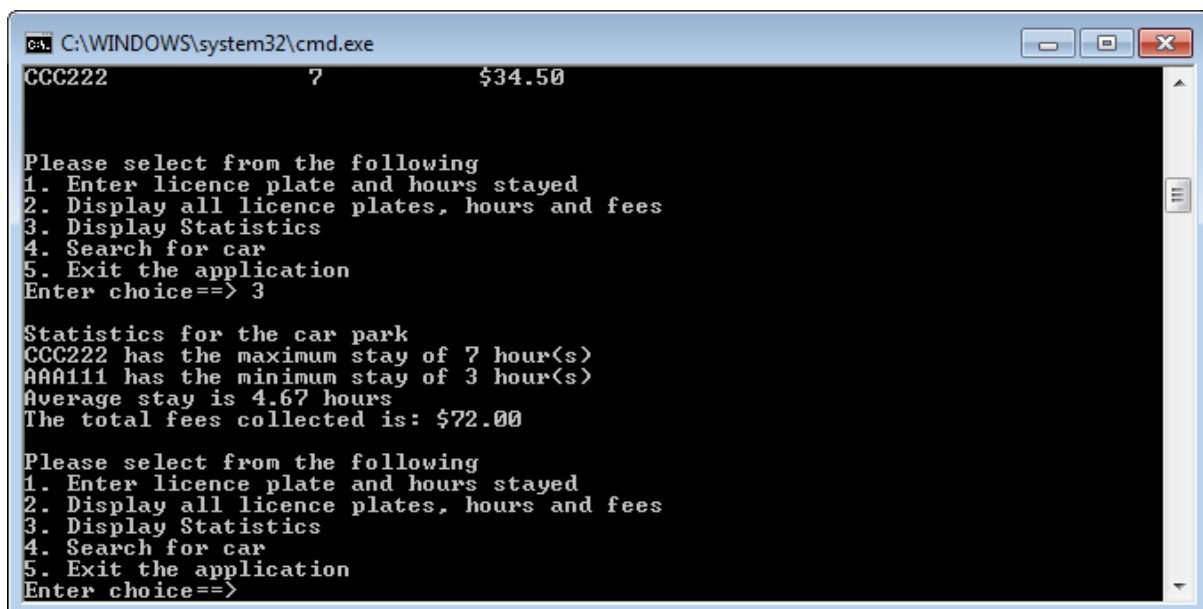
Please select from the following
1. Enter licence plate and hours stayed
2. Display all licence plates, hours and fees
3. Display Statistics
4. Search for car
5. Exit the application
Enter choice==> 2

Licence Plate      Hours      Fee
AAA111             3          $16.50
BBB222             4          $21.00
CCC222             7          $34.50

Please select from the following
1. Enter licence plate and hours stayed
2. Display all licence plates, hours and fees
3. Display Statistics
4. Search for car
5. Exit the application
Enter choice==>
```

3. Display statistics: `displayStatistics()`

When this option is selected you will display the statistics as per assignment one. You can loop through your array of objects to calculate this information.



```
C:\WINDOWS\system32\cmd.exe
CCC222             7          $34.50

Please select from the following
1. Enter licence plate and hours stayed
2. Display all licence plates, hours and fees
3. Display Statistics
4. Search for car
5. Exit the application
Enter choice==> 3

Statistics for the car park
CCC222 has the maximum stay of 7 hour(s)
AAA111 has the minimum stay of 3 hour(s)
Average stay is 4.67 hours
The total fees collected is: $72.00

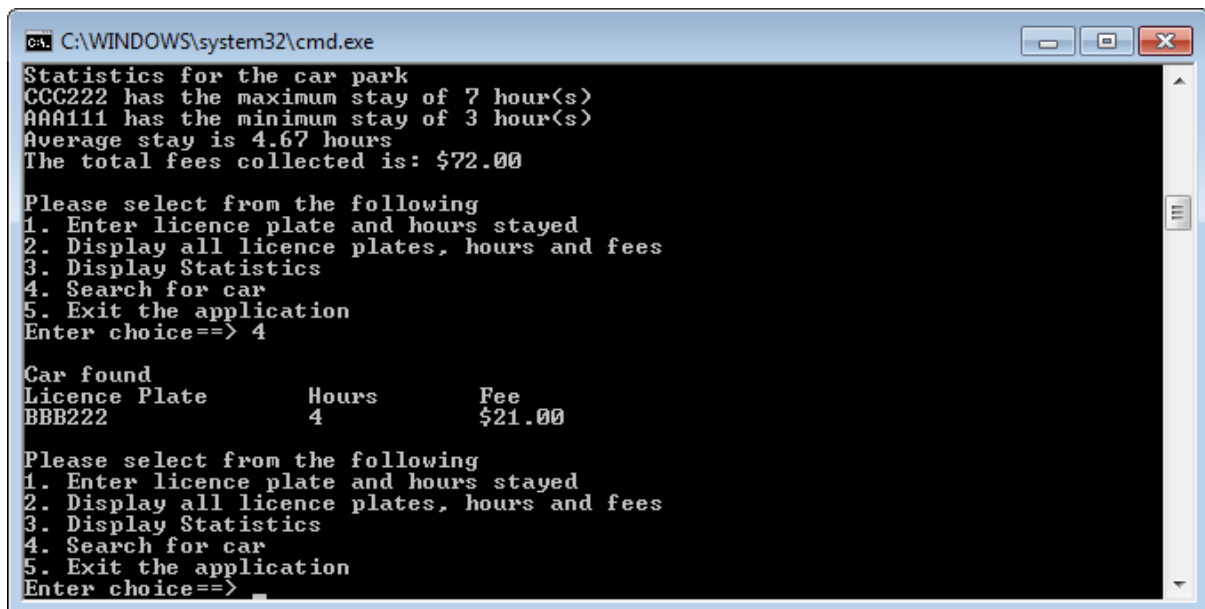
Please select from the following
1. Enter licence plate and hours stayed
2. Display all licence plates, hours and fees
3. Display Statistics
4. Search for car
5. Exit the application
Enter choice==>
```

4. Search for car: `searchCar()`

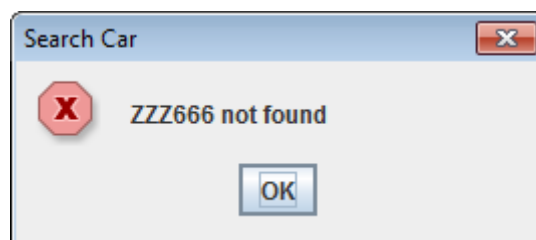
You can just use a simple linear search which will be case insensitive. Use the `showInputDialog()` method to input the name (you can share this functionality from `enter car`).



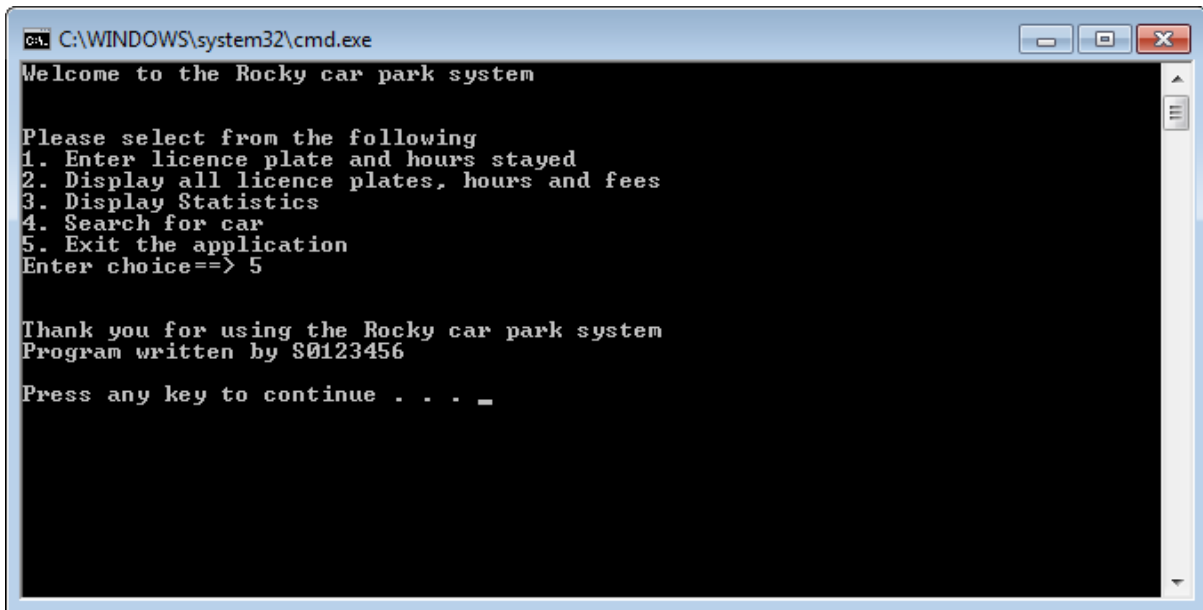
If the search is successful display the details about the car.



If the search is unsuccessful display an appropriate message.



Remember the welcome and exit messages as per assignment one.



```
C:\WINDOWS\system32\cmd.exe
Welcome to the Rocky car park system

Please select from the following
1. Enter licence plate and hours stayed
2. Display all licence plates, hours and fees
3. Display Statistics
4. Search for car
5. Exit the application
Enter choice==> 5

Thank you for using the Rocky car park system
Program written by S0123456

Press any key to continue . . . _
```

Extra Hints

Your program should be well laid out, commented and uses appropriate and consistent names (camel notation) for all variables, methods and objects.

Make sure you have no repeated code (even writing headings in the output)

Constants must be used for all numbers in your code.

Look at the marking criteria to ensure you have completed all of the necessary items

Refer to a Java reference textbook and the course and lecture material (available on the course web site) for further information about the Java programming topics required to complete this assignment.

Check output, check code and add all of your comments, complete report and the UML class diagram.

Supplied Code

Download, compile and run the supplied code available from the course web site.

You will see the menu interface has been implemented and you have to implement the underlying code, use the supplied method stubs and add you own methods.

Again no code should be repeated in your program.

If you just submit the supplied code you will receive zero marks.

Good luck! Bruce McKenzie Course Coordinator T116 COIT 11222

Marking Scheme

		Marks allocated
	Total number of marks – 20	
1	Variables, constants and types	
	Variables have meaningful names and use camel notation	0.5
	Variables are the correct type and constants are used	0.5
	Array of objects is used	0.5
2	Code in general	
	Code is indented and aligned correctly	0.5
	Code is easy to read (use of vertical whitespace)	0.5
	Code has header comment which includes name, student ID, date, file name and purpose of the class	0.5
	Code is fully commented including all variables and methods	0.5
	No repeated code	0.5
3	Car object	
	Instance variables are correct and private	0.5
	Default and parameterised constructors are correct	0.5
	Method for calculating fees is correct	0.5
	Get and set methods are correct	1
4	Enter licence plate and hours parked	
	Licence plate is read correctly	0.5
	Hours parked is read correctly	0.5
	Data is added to the object array correctly	1
	Output resembles the specification	0.5
	Fees calculated correctly	0.5
	Error dialog when maximum cars is reached	0.5
	Error dialog when licence plate not entered	0.5
	Error dialog when hours parked is out of range	0.5
	Data validation loops are correct	0.5
5	Display all cars	
	All records displayed	1
	Output resembles the specification	0.5
6	Display statistics	
	Maximum and minimum are correct	0.5
	Average is correct	0.5
	Total fees is correct	0.5
	Output resembles the specification	0.5
7	Search	
	Search is correct and correct details returned	1
	Search is case insensitive	0.5
	No search result is handled correctly	0.5
8	General	
	Welcome and exit message (as per assignment one)	0.5
	No cars entered is handled correctly	0.5
	Correct files submitted including types and names	0.5
9	Report	
	UML class diagram of Car object is correct	0.5
	Screen shot(s) of testing	0.5
	Report presentation and comments including how long it took and any problems encountered	0.5
10	Penalties	
	Penalty for late submission is 5% per day (1 mark per day) enter negative number	