Assessment details COIT 20245

Assessment item 2—JAVA Program using array of objects

Due date:	**Week 11 T318** – Midnight, **Friday** (1/2/19)	**ASSESSMENT**
	Refer below for complete assessment item 2 requirements (**Assignment Two**)	
Weighting:	30%	
Length:	N/A	**2**

Objectives

This assessment item relates to the unit learning outcomes as stated in the Unit 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: NemoReefToursMenu.java and is available on the unit 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:

- **Booking.java**
- **NemoReefTourMenu.java**

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

A report including an UML class diagram of your Booking class, how long it took to create the whole program, any problems encountered and screen shots of the output produced including annotations. (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.

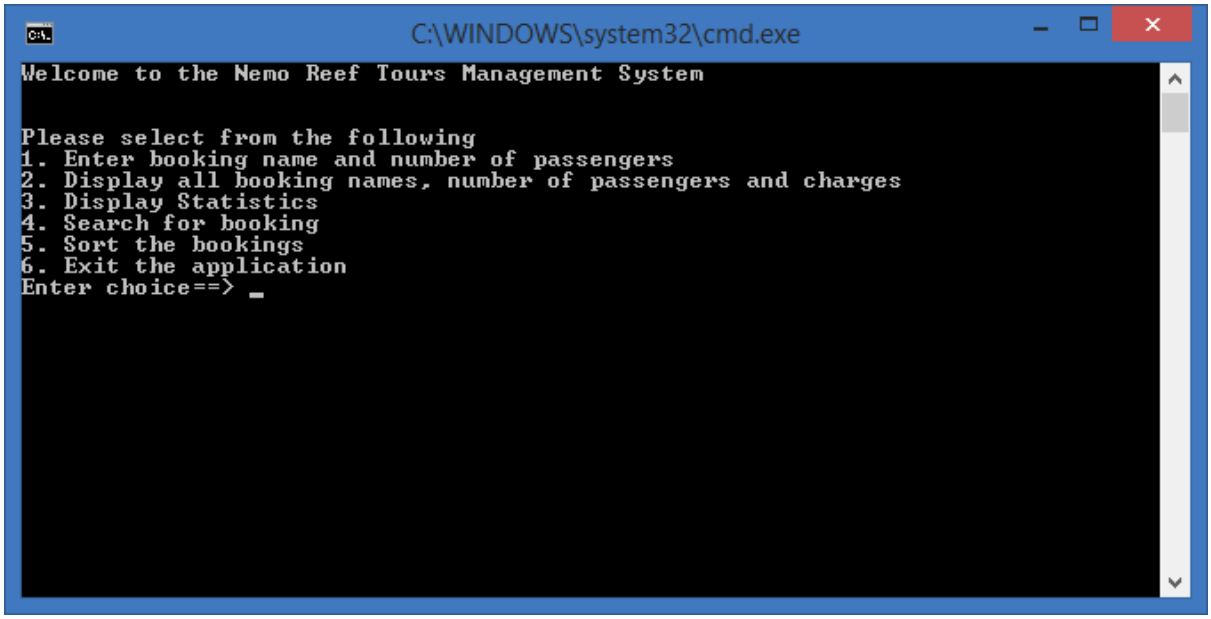
- **ReportAss2.docx**

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 bookings for Nemo Reef Tours. We are going to extend this application so the booking names and number of passengers per booking can be stored in an array of objects, **do not use ArrayList**.

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



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.

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

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

- bookingName as a String
- passengers as an integer

You should also have constants for the charge and discount values.

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 charge for the booking. This calculation will be the same as in assignment one. Use constants for all numeric literals.

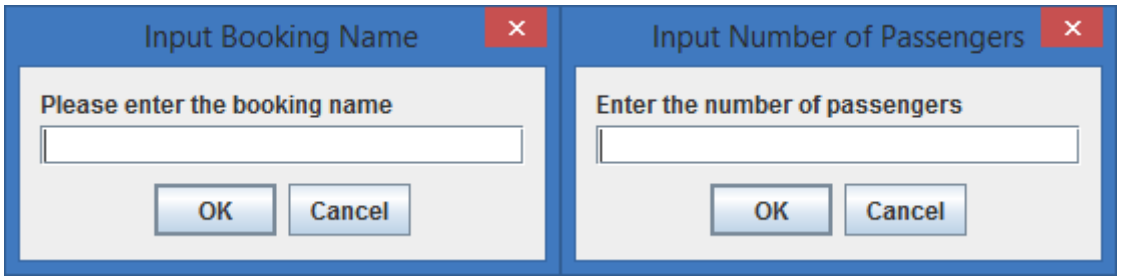
  Charge per person $85.50.
  One to two passengers: no discount.
  From three to five passengers: 10% discount.
  From six to ten passengers: 15% discount.
  More than ten passengers: 20% discount.

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

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

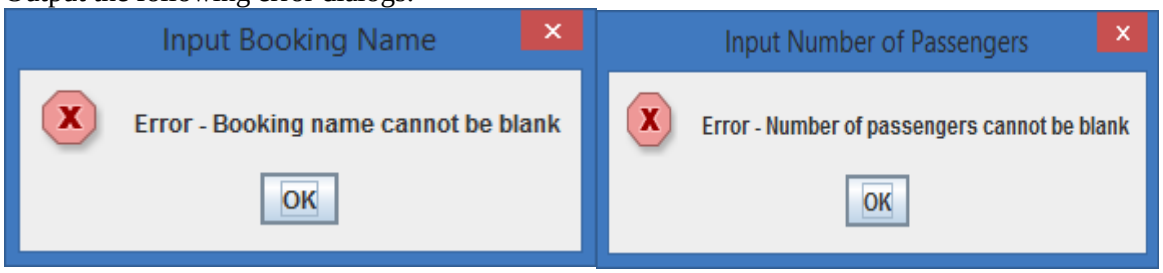
Data structures
For this assignment we are going to store the booking names and number of passengers in an array of Booking objects.
Declare an array of Booking objects as an instance variable of NemoReefToursMenu class the array should hold **ten** bookings.
You will need another instance variable (integer) to keep track of the number of the bookings being entered and use this for the index into the array of Booking objects.

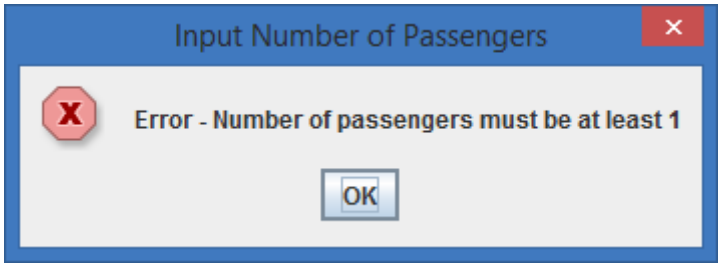
Menu options
1. **Enter booking name and number of passengeers**: `enterBooking()`
For assignment two we are going to use the GUI dialog showInputDialog() for our input. You will need to create the following two dialogs to receive the input from the user. You will not implement the functionality of the cancel key (need to use exceptions for this). Use `Integer.parseInt()` to convert the number as a string into an integer.



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 booking name cannot be blank i.e. not null and the number of passengers cannot be blank and the number of passengers needs to be greater than zero, the same as assignment one.

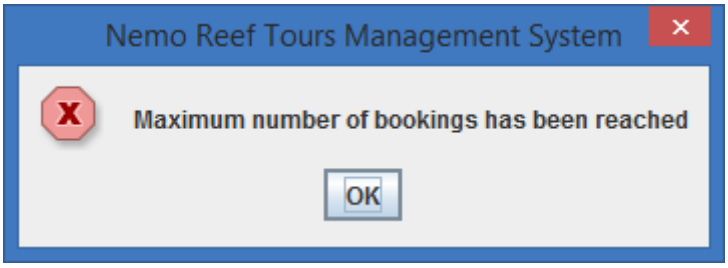
Output the following error dialogs:



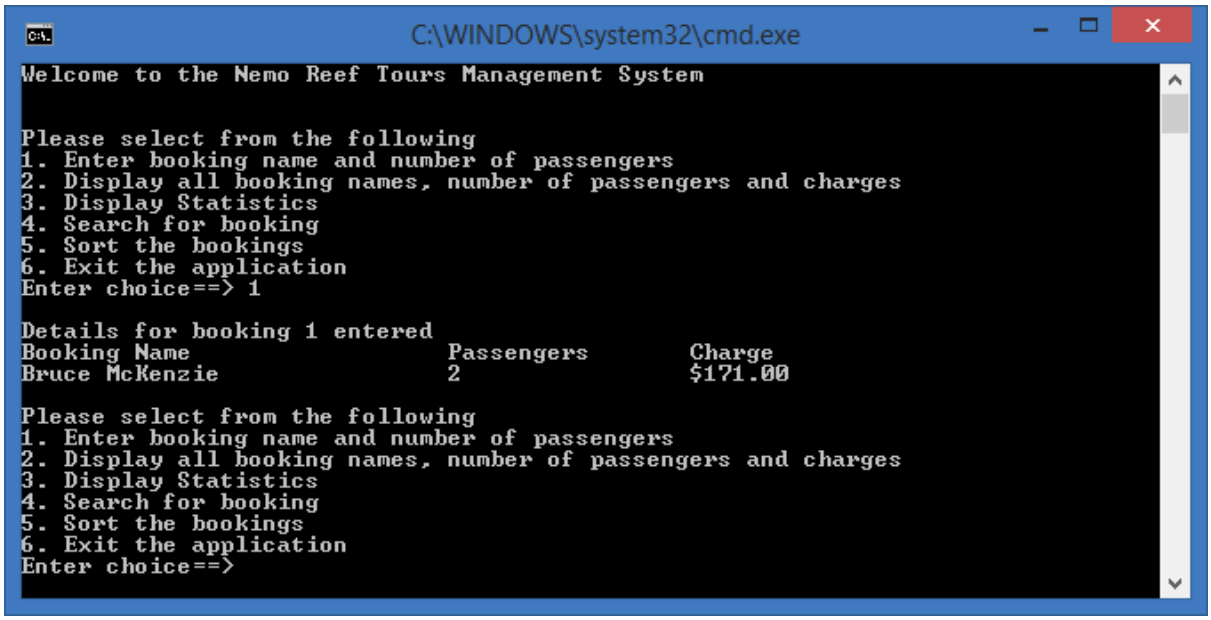


When the booking name and number of passengers have been entered successfully into two local variables you will need to add these values into the bookings object array, you will also need to increment a counter to keep track of the number of bookings you have entered and the position in the array of the next booking to be entered.

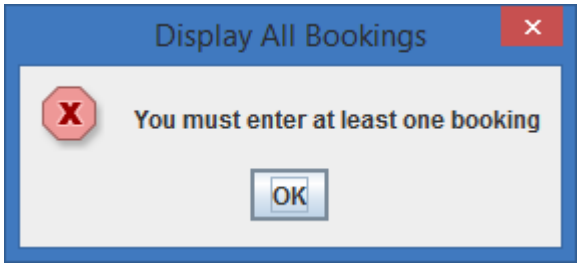
When the maximum number of bookings is reached do not attempt to add any more bookings and give the following error message:

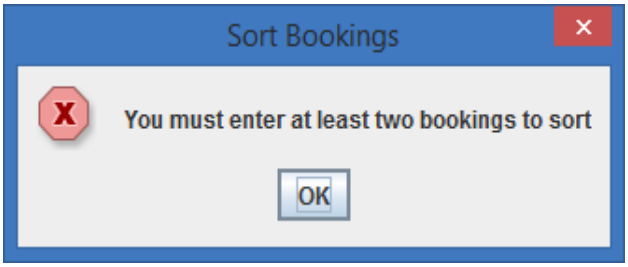


When the booking details have been successfully entered, display the details of the booking and the charge on the console screen.



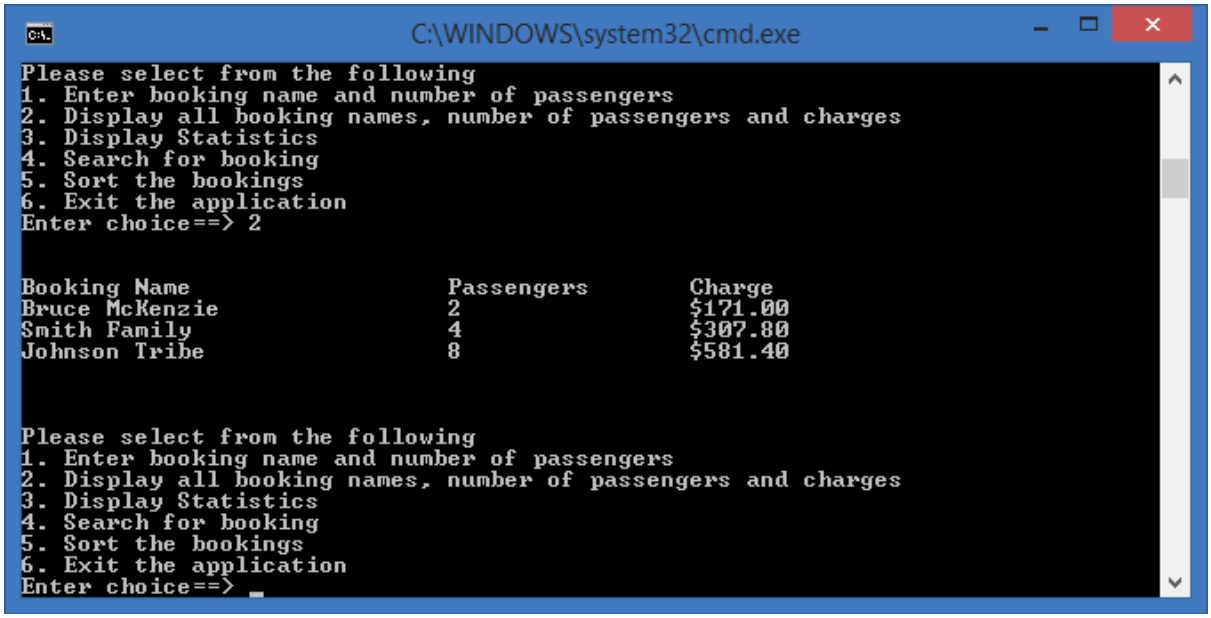
Note: For the next three options, display all, statistics and search you should ensure at least one booking has been entered and give an appropriate error message if it there are no bookings entered and for the sorting option you must ensure at least two bookings have been entered, for example:



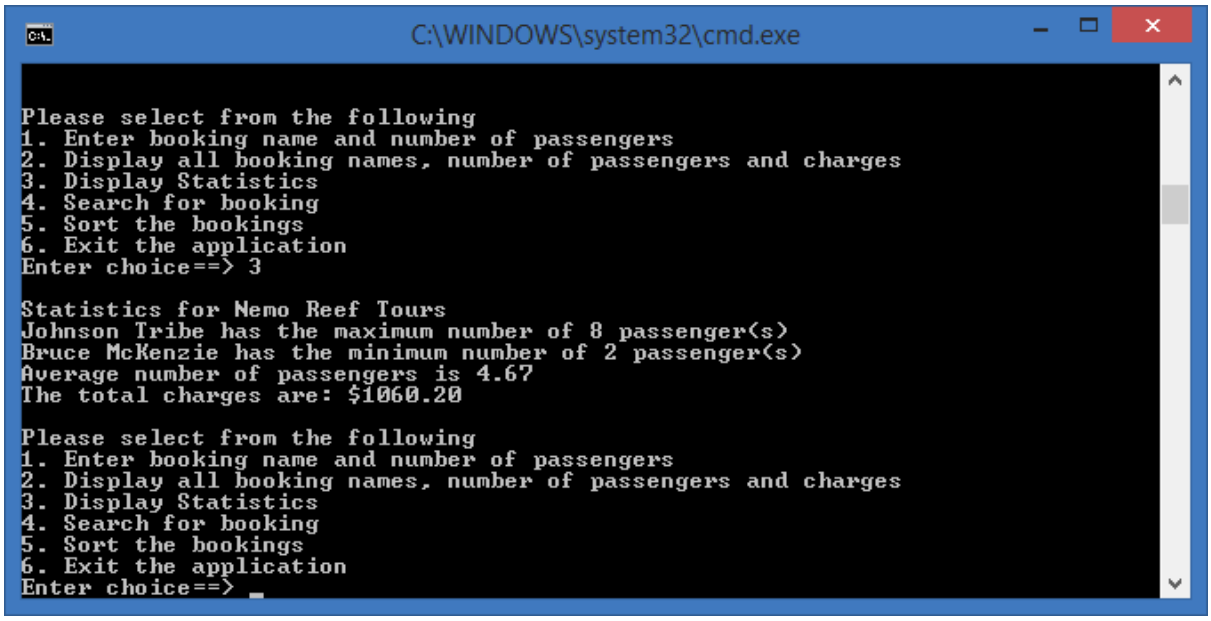


2. **Display all booking names, number of passengers and charges**: `displayAllBookings()`

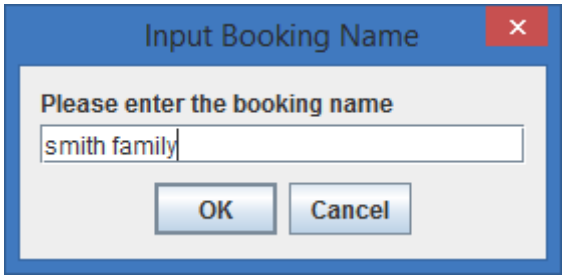
When this option is selected display all of the bookings which have been entered so far.



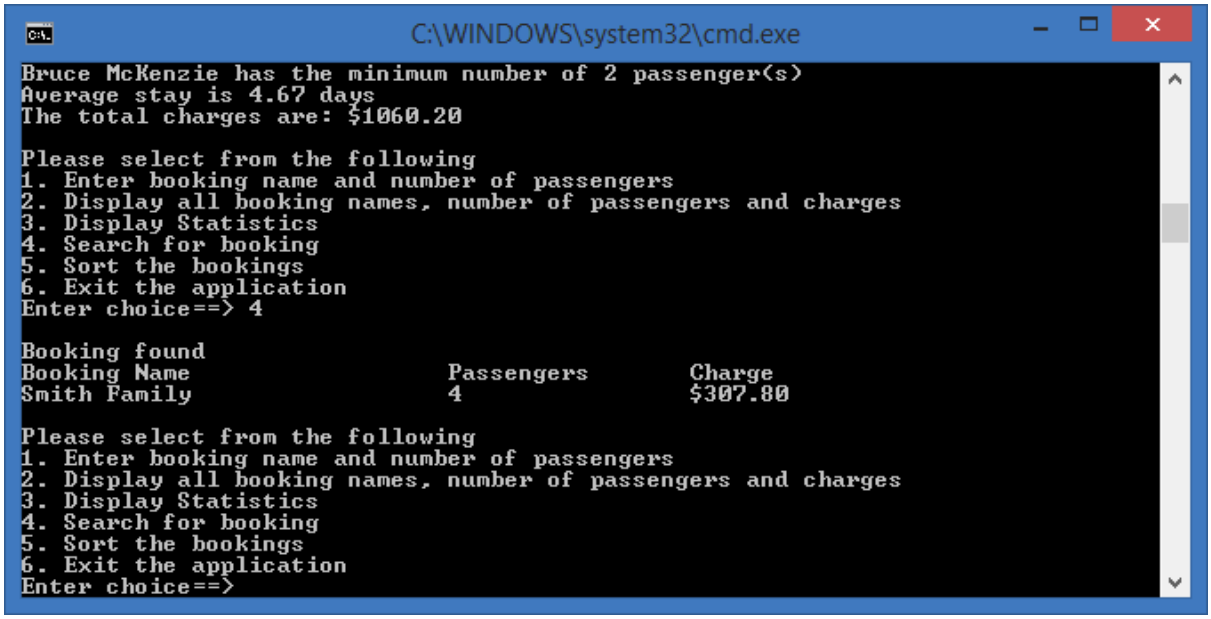
3. **Display statistics**: `displayStatistics()`
When this option is selected you will display the statistics as per assignment one. You can loop through your array of objects to calculate this information.



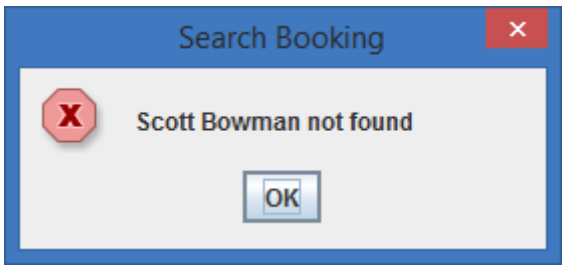
4. **Search for a booking**: `searchBookings()`
You can just use a simple linear search which will be **case insensitive**. Use the showInputDialog() method to input the booking name (you can share this functionality from enter booking).



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

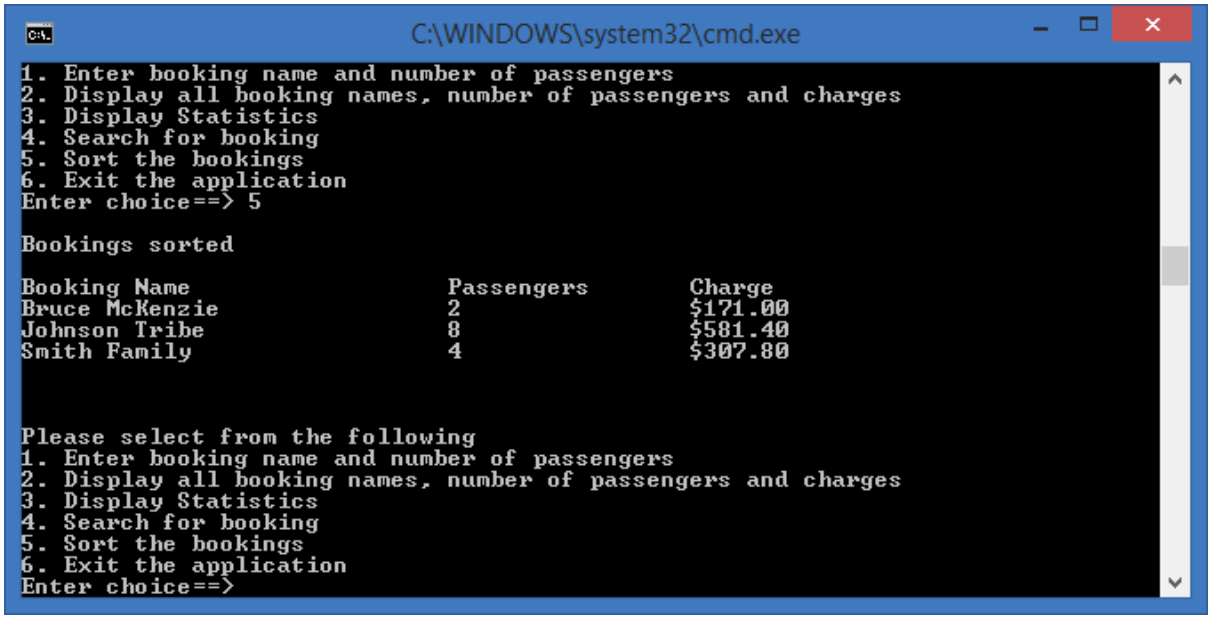


If the search is unsuccessful display an appropriate message.

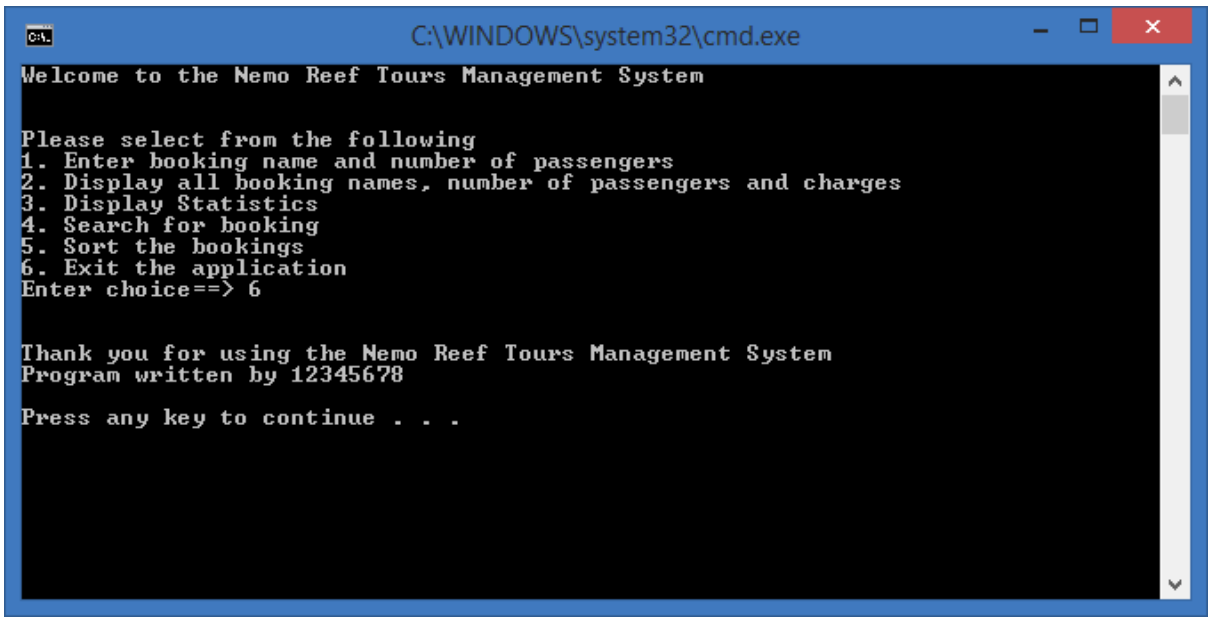


5. **Sort the bookings**: `sortBookings()`

This option you will sort the bookings alphabetically (ignore case) by the booking names, you can use any sorting algorithm which you like, **do not use any in-built sort methods**. Display the sorted list after the sort is complete.



Remember the welcome and exit messages as per assignment one.



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 (numeric literals) 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 unit 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 your 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 Unit Coordinator T318 COIT20245

b.mckenzie@cqu.edu.au

Marking scheme is on the following page.

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	1
2	**Code in general**	
	Code is indented and aligned correctly	1
	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	1
3	**Booking class**	
	Instance variables are correct and private	0.5
	Default and parameterised constructors are correct	0.5
	Method for calculating charges is correct	1
	Get and set methods are correct	1
4	**NemoReefToursMenu class -- enter booking**	
	Booking name is read correctly	0.5
	Number of passengers is read correctly	0.5
	Data is added to the object array correctly	2
	Output resembles the specification (two decimal points)	1
	Charges calculated correctly	0.5
	Error dialog when maximum bookings is reached	0.5
	Error dialog when booking name not entered	0.5
	Error dialog when number of passengers is blank or is out of range	0.75
	Data validation loops are correct	0.25
5	**Display all bookings**	
	All records displayed	2
	Output resembles the specification	1
6	**Display statistics**	
	Maximum and minimum are correct	0.5
	Average is correct	0.5
	Total charges is correct	0.5
	Output resembles the specification (two decimal points)	1
7	**Search**	
	Search is correct and correct details returned	2
	Search is case insensitive	0.5
	“<search key> not found” is correct	0.5
8	**Sort**	
	Sort is correct and inbuilt sorting methods are not used	2
	Sort is case insensitive	0.5
	Sorted list is displayed after sort	0.5
9	**General**	
	Welcome and exit message (as per assignment one)	0.5
	No bookings entered is handled correctly	0.5
	Correct files submitted including types and names	0.5
10	**Report**	
	UML class diagram of Booking class is correct	1
	Screen shot(s) of testing and annotations	0.5
	Report presentation and comments including how long it took and any problems encountered	0.5