University of Westminster
School of Electronics and Computer Science

4COSC005/W Software Development 2 – Coursework	
Module leader	Artie Basukoski
Weighting:	50% of the module
Qualifying mark	30%
Description	Coursework
Learning Outcomes Covered in this Assignment:	LO1, LO3, LO4, LO5.
Handed Out:	21st February, 2022
Due Date	Code due on Blackboard coursework upload Monday 11th April 2022, 1pm.
Expected deliverables	a) Zip the project directory of each implementation and upload as w1234557_arrays_only.zip, and w1234567_classes.zip. b) Submit test results (pdf or doc) c) Online demo
Method of Submission:	Blackboard
Type of Feedback and Due Date:	Written feedback and marks 15 working days (3 weeks) after the submission deadline. All marks will remain provisional until formally agreed by an Assessment Board.

Assessment regulations

Refer to section 4 of the "How you study" guide for undergraduate students for a clarification of how you are assessed, penalties and late submissions, what constitutes plagiarism etc.

Penalty for Late Submission

If you submit your coursework late but within 24 hours or one working day of the specified deadline, 10 marks will be deducted from the final mark, as a penalty for late submission, except for work which obtains a mark in the range 40 – 49%, in which case the mark will be capped at the pass mark (40%). If you submit your coursework more than 24 hours or more than one working day after the specified deadline you will be given a mark of zero for the work in question unless a claim of Mitigating Circumstances has been submitted and accepted as valid.

It is recognised that on occasion, illness or a personal crisis can mean that you fail to submit a piece of work on time. In such cases you must inform the Campus Office in writing on a mitigating circumstances form, giving the reason for your late or non-submission. You must provide relevant documentary evidence with the form. This information will be reported to the relevant Assessment Board that will decide whether the mark of zero shall stand. For more detailed information regarding University Assessment Regulations, please refer to the following website:**http://www.westminster.ac.uk/study/current-students/resources/academic-regulations**

Coursework Description

Cruise Ship Boarding

You are to write a boarding system for a Cruise ship in java. The system will print a menu with which the user will interact to assign passengers to each cabin of a cruise ship.

Task 1. Arrays solution. Design a program for a Cruise Ship with **twelve** cabins using code like the code provided near the end of your notes. Start by checking that the code works.

Once the basic code runs, put the code for 'Views All cabins' and 'Adds customer to cabin', into separate procedures, and test it works. You can build up your test cases as you develop your program (see testing below).

Then add a menu system which will allow the user to choose what they want to select. Enter an 'A' to add a customer to a cabin, and a 'V' to view all cabins. Implement each as a method. When an 'A' is pressed, it should do the Add method; a 'V' should do the View method.

One by one, add extra methods to do each of the following. The user should be able to choose from the menu what the program does.

E: Display Empty cabins
D: Delete customer from cabin
F: Find cabin from customer name
S: Store program data into file
L: Load program data from file
O: View passengers Ordered alphabetically by name. (**Implement your own sort algorithm**)

Task 2. Classes solution. Create a second version of the Cruise Ship program using an *array of Cabin objects*. Create a class called Cabin and another class called Passenger. Each cabin should hold up to 3 passengers with the following additional information.

i. First Name.
ii. Surname.
iii. Expenses.

Add an additional option to the menu. 'T' that will give the user the option to print the expenses per passenger as well as the total expenses of all passengers. Otherwise, the program should function as in Task 1.

Task 3. Add Queue. Add a waiting list to your Cruise Ship class version.
Modify your 'A: Add' and 'D: Delete' as follows:

- When you press 'A' to add a new customer, the customer should be added to the Waiting List queue if the Cruise Ship is full.
- When you press 'D' to delete a customer from a cabin, the next customer in the Waiting List queue should be automatically placed in the cabin.

Extra marks will be awarded if you implement the waiting list queue as a circular queue.

Task 4. Testing. Create a table of test cases showing how you tested your program (see below for example). Write a brief (no more than one page) discussion of how you chose your test cases to ensure that your tests cover all aspects of your program. Additionally, write a few paragraphs (up to 1 page) explaining which of the solutions (Array or Class) you think is better and why. Think about

which is easier to read/understand, which is easier to modify if necessary, and any other comments you can think of.

Test Case	Expected Result	Actual Result	Pass/Fail
(Cabins Initialised correctly) After program starts, Press 'V'	Displays 'e' for all cabins	Displays 'e' for all cabins	Pass
(Add customer "Bob" to cabin 5) Select A, enter "Bob"	Press 'v' Displays "Bob" for cabin 5	Displays "Bob" for cabin 4	X

Note: Solutions should be java console applications (not windows).

Marking scheme

The coursework will be marked based on the following marking criteria:

Criteria	Max for Subcomponent	Max Subtotal
Task 1 Three marks for each option (A,V,E,D,F,S,L,O) Menu works correctly	24 6	**(30)**
Task 2 Cabin class correctly implemented. Passenger class correctly implemented. Expenses correctly reported.	14 10 6	**(30)**
Task 3 Waiting list queue implementation "A: Add"works correctly "D: Delete"works correctly Circular queue implementation	10 3 3 4	**(20)**
Task 4 Test case coverage and reasons Writeup on which version is better and why.	6 4	**(10)**
Coding Style (Comments, indentation, style) Complete the self-evaluation form indicating what you have accomplished to ensure appropriate feedback.	7 3	**(10)**
Totals		**(100)**

Demo: At the discretion of your tutor, you may be called on to give a demo of your work to demonstrate understanding of your solutions. If you cannot explain your code and are unable to point to a reference within your code of where this code was found (i.e., in a textbook or on the internet) then significant marks will be lost for that marking component.