IN-COURSE ASSESSMENT (ICA) SPECIFICATION

Module Title Object Oriented Programming (Semester 2)	**Module Leader**	Jackie Barker
	Module Code	CIS4037-N
	Submission Final Date	Wednesday, 29th March 2023
Module Title ICA Element1: Console Applications	**Submission Method** Online (Blackboard) ☑ Middlesbrough Tower ☐	

Online Submission Notes

- Please carefully follow the instructions given in this Assignment Specification.
- Extensions or MITS should be requested using the Extenuating Circumstances (EC) form available at https://www.tees.ac.uk/sections/stud/handbook/extenuating_circumstances.cfm
- Instructions for submitting completed EC forms:
  - Short extensions requests are emailed to either Module Leader or Course Leader
  - Long Extension requests are submitted to Course Leader or scedt-assessments@tees.ac.uk
  - MITS requests are submitted to SLSMitigatingCircumstances@tees.ac.uk
- If Extenuating Circumstances (extension or MITS) is granted, a fully completed and signed Extenuating Circumstances form must be emailed to scedt-assessments@tees.ac.uk or submitted to the School Reception.

Module Assessment Notes

The module assessment is composed of two elements:

- Element 1 is a Portfolio of Team programming tasks and Individual written task; contributing 30% to the module mark
- Element 2 is a Portfolio of Individual programming tasks and written task; contributing 70% to the module mark

To pass the module students need to attain a minimum overall mark of 40%. If a student has attained 40% overall, but failed one of the elements, they will still pass if the student submitted a genuine attempt for the failed element.

FULL DETAILS OF THE ASSIGNMENT ARE ATTACHED

INCLUDING MARKING & GRADING CRITERIA

Contents

1. Introduction

The assessment for the module is comprised of two elements.

1.1 ICA Element 1 Overview

The first element is worth 30% of the module's mark. Working as part of a team, students are given three programming tasks that will require each student to do some research, experimentation, and development.

Students will be provided applications that have some functional similarities to the application prototypes your teams will produce; this will help focus the research into features of Java and associated APIs. This research and practice from this element will be of help with element 2.

Students will individually write and submit reflective documents discussing the research and development process and outcomes.

1.2 ICA Element 2 Overview

The second element is worth 70% of the module's mark. Each student will individually develop a small Java application to simulate buying and selling of sports items.

The emphasis is on demonstrating:

- The use of the Java programming language.
- The use of Java APIs.
- Good use of object-oriented programming techniques.
- Basic GUI creation using the Swing API.
- Selecting and implementing an appropriate design pattern to improve the overall architecture of the solution

Students will also write a short technical and self-reflective report.

2. Element 1 [30%] – Console Application

2.1 Task Summary

Students will work in a team of two or three to design a develop a console application for Asher Sports Consortium to sell and restock items. The application will be developed in three stages, corresponding to (programming) tasks 1, 2 and 3. Students using Apache NetBeans, or raw coded using a text editor,

Table 1: ICA Element 1 tasks

ICA Task	Description	Submission	ICA Weighting
1	Skeleton Application	Week 05	5 %
2	Mock Application	Week 07	5 %
3	Prototype Application	Week 09	5%
4	Reflection	Week 10	15%

The programming tasks (1, 2 and 3) will be assessed in each team's practical session in the weeks stated in Table 1 above. Students will also need to submit their java project as a zip file to Blackboard by the end of the same week. Task 4 is an individual reflection to be submitted to Blackboard in Week 10.

Each team member will take on the role of Team Leader for one of the programming tasks, with following responsibilities

- Make design decisions for the task and assign coding tasks based on design to all member (including themselves)
- Keep a record of regular progress reports from the other team members.
- Provide support to the other team members.
- Record actions taken if insufficient progress was being made to achieve the required outcome.
- Develop test plan for task (which will be used in demonstration)

Although the Team leader has final decision on what is to be done, it is required that all team members will contribute ideas for design and implementation for the task.

2.2 Programming Tasks

Task 1: Skelton Application

The skeleton application will display relevant details of Asher Sports Consortium followed by a command line menu with the following options:

- View Items
- Buy Item
- Add Stock
- Quit

A suitable number should be allocated to each option and displayed alongside the option in the menu, as depicted by figure 1 below.

```
ASHER SPORTS CONSORTIUM

MAIN MENU: Please select a menu option to continue
        [1] View Items
        [2] Buy Item
        [3] Add Stock
        [0] Quit
```

Figure 1: Task1 command line menu

When the user enters an option number the application should:

- Display the name of the option
- Redisplay the Menu

```
3

ADD STOCK

MAIN MENU: Please select a menu option to continue
        [1] View Items
        [2] Buy Item
        [3] Add Stock
        [0] Quit
```

Figure 2: Add Stock operation

If the user enters an non integer value or an option that is not specified than a suitable validation menu is displayed followed by the menu

```
ASHER SPORTS CONSORTIUM

MAIN MENU: Please select a menu option to continue
        [1] View Items
        [2] Buy Item
        [3] Add Stock
        [0] Quit

quit
!!!!! Please select a valid menu option !!!!!

MAIN MENU: Please select a menu option to continue
        [1] View Items
        [2] Buy Item
        [3] Add Stock
        [0] Quit
```

Figure 3: Dealing with Invalid option

If the user selects the **Quit** Option, in which case a suitable message is displayed before the application terminates.

```
ASHER SPORTS CONSORTIUM

MAIN MENU: Please select a menu option to continue
        [1] View Items
        [2] Buy Item
        [3] Add Stock
        [0] Quit

0


***** Thank you for your visit *****


------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  3.425 s
Finished at: 2023-01-22T21:48:30Z
------------------------------------------------------------------------
```

Figure 4: Quit selected by user

Note:

- Project Name: `Task01`
- Group ID: `oop`
- Package name: `oop.ica.e1`
- Main Class Name: `SportsShopSystem.java`
- Authors: Name and User Id of all team members should be specified as comments at top of any java file coded
- Students should use a modular approach within their design.
- Students should ensure suitable validation and error handling is in included in the code
- Task to be assessed in Week 5
- Submit to Blackboard by 4.00pm on Friday, 24 February 2023
- Worth: Contributes 5% towards Module mark

Task 2: Mock Application

The Mock Application will work with a Data Class and a Generic ArrayList to enable actions for the menu options from task 1. Students should create a copy of their Task01 Project and rename the copied project to `Task02`

Data Class

Student will add and code a data class based on the following design and notes:

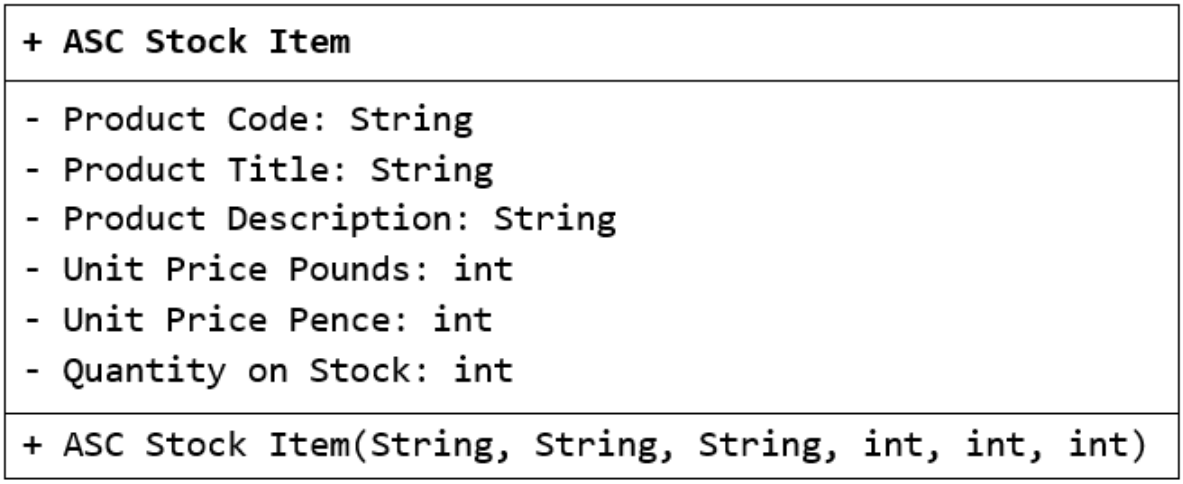

Figure 5: Generalised Class Diagram

Note:

- Above identifiers are in sentence form
  - Students should use appropriate Java naming convention for class name, field, and methods names
  - Class and constructor name should be Java version of that specified above
  - Field names can be appropriate shorten Java versions of that specified above
  - but can modify the field names
- In addition to constructor, students will need to provide appropriate:
  - Accessor Methods, e.g. `getPounds()`
  - Service Methods, e.g. `getPrice()`

There are various restrictions on the value that can be stored by each field, see Table 2 below.

Table 2: Field Restriction(s)

Field	Restriction
Product Code	String value in the format `ABC1234567` `ABC` is a three-letter code indicating the department • `CYC` (cycling) • `FIT` (fitness) • `FTB` (football) • `GLF` (golf) • `RUN` (running) • `SWM` (swimming) The numeric part is an unique seven-digit number
Product Title	String value which has maximum length of 120 characters
Product Description	String value which has maximum length of 500 characters
Unit Price Pounds	Integer value which specifies the pounds component of the item price
Unit Price Pence	Integer value which specifies the pence component of the item price
Quantity in Stock	Integer value which specifies the quantity of the item in stock

The above restrictions will be implemented by students in ICA Element 2. For ICA Element 1, data is provided which complies with the restrictions.

Main Class

In the main class students will need to declare an Generic ArrayList based on the Data class. Both the ArrayList will need to be visible to all methods in the main class.

Two additional methods are required:

- `loadData()`
- `displayItems()`

The `loadData()` method will need code which manually populates the ArrayList with three data class objects. The field values for the objects are displayed in Table 3.

Table 3: Item Values

Product Code	Product Title	Product Description	Unit Price Pounds	Unit Price Pence	Quantity in Stock
RUN1234567	RunTech Shorts	High quality running shorts	10	0	10
CYC1111111	Cycle4ever Cycling Jacket	Ultra lightweight jacket for the urban cyclist	50	0	20
SWM2222222	7oceans Goggles	Super HiTech goggles for the extreme swimmer	24	99	8

The above values occupy the first three lines of the `AsherSportsConsortium.csv` file. The load data method should be called before the menu is displayed

The `displayItems()` method will display in a tabular layout the following detail for each Stock Item in the ArrayList

- Item ID – a simple integer count starting at 1
- Item Title
- Item Price in Pounds and Pence format (to two decimal places)
- Quantity in Stock

The `displayItems()` method will be used for the Buy Item and Add Stock main menu operations.

Main Menu Operations

When the user selects the **View Items** option, then after the option title full details should be displayed for each Item in a tabular layout After which the menu is redisplayed, as depicted by Figure 6

```
ASHER SPORTS CONSORTIUM

MAIN MENU: Please select a menu option to continue
        [1] View Items
        [2] Buy Item
        [3] Add Stock
        [0] Quit

1

VIEW ITEMS
Stock Code      Title                           Description                                 Price           Quantity
RUN1234567      RunTech Shorts                  High quality running shorts                 10.00           10
CYC1111111      Cycle4ever Cycling Jacket       Ultra lightweight jacket for the urban cyclist      50.00           20
SWM2222222      7oceans Goggles                 Super HiTech goggles for the extreme swimmer        24.00           8


MAIN MENU: Please select a menu option to continue
        [1] View Items
        [2] Buy Item
        [3] Add Stock
        [0] Quit
```

Figure 6. Display from modified View Items Menu Option

When the user selects the **Buy Item** option, after the option title the dislayItems() method is called. The operation should prompt the user for which item they wish to buy. If a valid item number is provided, then one item is sold and the stock for that item suitably amended. A suitable message with relevant details should be displayed, after which the menu is redisplayed, as depicted by Figure 7.

```
ASHER SPORTS CONSORTIUM

MAIN MENU: Please select a menu option to continue
        [1] View Items
        [2] Buy Item
        [3] Add Stock
        [0] Quit

2

BUY ITEM
Stock Items
ID    Item                                  Price      Quantity
1     RunTech Shorts                        10.00      10
2     Cycle4ever Cycling Jacket             50.00      20
3     7oceans Goggles                       24.00      8

Select item if to buy, or 0 to return to main menu
1
Sale of RunTech Shorts for £10.00 confirmed. Quantity remaining: 9

MAIN MENU: Please select a menu option to continue
        [1] View Items
        [2] Buy Item
        [3] Add Stock
        [0] Quit
```

Figure 7. Example display from successful Buy Item Menu Operation

When the user selects the **Add Stock** option, after the option title the dislayItems() method is called. The operation should prompt the user for which item they wish to add to. If a valid item number is provided, then one item is is added to the stock. A suitable message with relevant details should be displayed, after which the menu is redisplayed, as depicted by Figure 8.

```
ASHER SPORTS CONSORTIUM

MAIN MENU: Please select a menu option to continue
        [1] View Items
        [2] Buy Item
        [3] Add Stock
        [0] Quit

3

ADD STOCK
Stock Items
ID    Item                                  Price      Quantity
1     RunTech Shorts                        10.00      10
2     Cycle4ever Cycling Jacket             50.00      20
3     7oceans Goggles                       24.00      8

Select item if to add, or 0 to return to main menu
2
New quantity for Cycle4ever Cycling Jacket is: 21

MAIN MENU: Please select a menu option to continue
        [1] View Items
        [2] Buy Item
        [3] Add Stock
        [0] Quit
```

Figure 8. Example display from successful Add Stock Menu Operation

Note:

- Project Name: Task2
- Authors: Name and User Id of all team members should be specified as comments at top of any java file coded
- Students should ensure suitable validation and error handling is in included in the code
- Students will need to decide on what code from previous task to keep, modify, or remove
- Task to be assessed in Week 7
- Submit to Blackboard by 4.00pm on Friday, 10 March 2023
- Worth: Contributes 5% towards Module mark

Task 3: Protype Application

The Prototype Application will work with an input text file and an output text file to make the mock application dynamic in terms of data handling. Students should create a copy of their Task02 Project and rename the copied project to `Task03`

Students will need to modify the code two main class methods

- `loadData()`
- `main()`

Students will need to code one new main class method

- `saveData()`

The `loadData()` method will populate the generic ArrayList reading and processing the input file. This method should run before the application displays anything to the console. If the `loadData()` is successful in populating the ArrayList then the application should proceed with displaying and processing the menu. Otherwise, if there is an file error or no items are loaded, then an suitable error message should be displayed, and the application allowed to terminate.

```
ASHER SPORTS CONSORTIUM


!!!!! Error: 'AsherSportsConsortium.csv' does not exist !!!!!


------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.957 s
Finished at: 2023-01-22T22:41:50Z
------------------------------------------------------------------------
```

Figure 9. Example display due to file reading error

The `saveData()` method will save the contents of the generic ArrayList to the output file. This method should run when the user selects the quit option from the main menu. Suitable messages should be displayed if method is successful or unsuccessful

```
ASHER SPORTS CONSORTIUM

MAIN MENU: Please select a menu option to continue
        [1] View Items
        [2] Buy Item
        [3] Add Stock
        [0] Quit

0

Data written to file at: asc_output.txt


***** Thank you for your visit *****


------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  11.619 s
Finished at: 2023-01-22T23:23:09Z
------------------------------------------------------------------------
```

Figure 10. Example display when output file successfully written

No changes are required to the View Items, Buy Item and Add Stock operations

Note:

- Project Name: Task03
- Authors: Name and User Id of all team members should be specified as comments at top of any java file coded
- Input file and Output file should be placed in the Project folder and are delimited by a comma
    - Input file: `AsherSportsConsortium.csv`
    - Output file: `asc_output.txt`
- Students should ensure suitable validation and error handling is in included in the code
- Students will need to decide on what code from previous task to keep. modify or remove
- Task to be assessed in Week 9
- Submit to Blackboard by 4.00pm on Friday, 24 March 2023
- Worth: Contributes 5% towards module mark

Task 4: Reflection

Students will need to submit a reflection (either in essay or report format) on their experience as a Team Leader. The reflection should discuss the following

1. Alternative ideas considered for any of the programming tasks
2. Decisions made and why
3. Supporting team as team leader
4. What was test plan and how did it compare to demonstration

The above can be used as section headings for the reflection. Within each section clear examples should be provided for the reflective discussion and supported with evidence, e.g., screenshots, test plans, copy of messages, etc. In addition, students should conclude each section by indicating what they earnt and how that learning could be ut to good use in the future

The body text of the reflection should be 1000 words. No need to deploy a reflection model or include an appendices.

Note:

- File formats: DOCX, DOC or PDF
- File Name:
  - Format: *surname*-*userID*-OOP-Task4
  - Example: Rashid-u0018369-OOP-Task4.pdf
- Submit to Blackboard by 4.00pm on Wednesday 29th March 2023 (week10)
- Worth: Contributes 15% towards module mark

Appendix A. Test Requirements

For each programming task students should test that the task works as would be expected. For tasks 1, 2 and 3 the team leader should construct test plans for each new and or revised feature of the application.

The test plan should use the following format:

Test Number	Description	Values / Input	Expected outcome	Actual outcome

Once the test plan has been completed, use it to check your design and application. If any of the tests fail, update your application and re-test (keep the original entry that failed in the plan along with the new entry for the re-test).

Students should have the test plan either printed or displayed when demonstrating a programming task.

Appendix B. Module Learning Outcomes

The following tables provide the learning outcomes for ‘Computer Technologies and Operating Systems’ module.

Personal and Transferable Skills	
1.	Produce appropriate software documentation to communicate the programming concepts and techniques underpinning their solutions.
2.	Demonstrate a sound understanding of the Java API and the ability to make good use of the information provided therein when building solutions
3.	Develop team leadership skills by taking the team lead role to identify and develop solutions to practical problems.

Research, Knowledge and Cognitive Skills	
4.	Analyse complex and/or incomplete problem specifications, justify the design and technical methodologies used, and recognise and argue for alternative approaches.
5.	Demonstrate a systematic and critical understanding of Object Oriented concepts, event handling and the development of Graphical User Interfaces.
6.	Select appropriate programming techniques and abstract Object-Oriented concepts, and critically evaluate their effectiveness in a given scenario.

Professional Skills	
7.	Design and implement efficient Java solutions to unfamiliar problem specifications and critically evaluate the processes and facilities used in an autonomous manner.

Appendix C. Marking Criteria

Tasks 1, 2 and 3 (15%)				
Programming Practice	**Distinction**	**Merit**	**Pass**	**Fail**
	The code is written to a very high and consistent standard, elegantly specified. The code is easy to read and understand by someone other than the original author. Comments are provided, clear descriptions of algorithmic implementations that are not immediately obvious by the code itself. The code is clearly demonstrating that it is very easy to extend and maintain.	The code is written to a good standard, although not always consistent. The code is easy to read and understand by someone other than the original author. Comments are provided, although it maybe inconsistent in clarity or usefulness. The code appears to be generally easy to extend or maintain with little effort.	It performs reasonably, although not consistently. The code is written to a reasonable standard, although the author may not have always considered that others would need to read and understand their code in the future. Some of the requirements have been met. Comments are present, although they may be inconsistent in their usefulness or appropriateness. It is not immediately clear, or it looks like quite a lot of effort would be required to extend and/or maintain the current code-base	Code is poorly written, lacking in in structure. There is little or no consideration that other programmers may need to be able to read and understand the code in the future. Few, if any, useful or appropriate comments are provided. The current code-base is messy and would require considerable effort to extend and/or maintain - probably a complete rewrite would be required. .

Task 4 (15%)				
Reflection	**Distinction**	**Merit**	**Pass**	**Fail**
	Full (but concise) explanation of the problem domain and the objectives of each task; clear insight into main challenges. A thorough reflection on the learning experience from researching and experimenting, and how these were used to guide you and your team to required exercise solutions. Excellent reflection of how the team interacted, worked together, fair and professional appraisal of other members of team performance.	Technical authorship presents a clear explanation of the problem; discussion/critique shows a good appreciation of main issues & objectives. A good reflection on the learning experience from researching and experimenting with the various examples and how these were used to guide you and your team to required exercise solutions. Good reflection of how the team interacted and worked together, fair and professional appraisal of other members of team performance.	Technical authorship is acceptable; discussion/critique demonstrates understanding of main issues & problems. Demonstrated an understanding of the problem tackled and their solution. Adequate reflection of how the team interacted and worked together. Appraisal of other members of team performance.	Technical authorship is below the standard expected of a Masters degree. There is little or no demonstration of understanding of the issues and problems. There is little or no evidence of teamwork.