### BIT100 / DIP222 Assignment 2

|  |  |
| --- | --- |
| **Due date:** |  |
| **Value:** | 10% |
| **Mode:** | Individual Assignment |

**Rationale**

This assignment is designed to reinforce the subject material covered in the learning modules and encourage students to work consistently throughout the semester. This assignment has been designed to give students the opportunity to demonstrate their skill in:

* solving a fairly complex problem involving the design of more than one user defined class;
* using list to maintain a collection of objects and allow management of them.
* managing an application which involves a collection of objects
* writing and using methods which enable objects to show desired behaviors
* using and complying with a supplied specification for classes to be written.
* using good programming style.

This assignment assesses the following learning outcomes:

* CLO2: Discuss elements of good programming style
* CLO4: Apply the concept of object orientation as an approach to data abstraction
* CLO5: Write programs to solve basic computing problems

|  |
| --- |
| **SUBMISSION REQUIREMENT**  Your assignment has to be submitted to Turnitin, with the following all contained in a single Word document (.doc or .docx) file:   1. All your Python source files, printed in Word document format 2. Printed output (showing your interactivity with your program) is to be included at the end of your Python source files, in Word document created in (1)   **Turnitin Report (http://www.turnitin.com)**  Register yourself in BIT100/DIP222 Sem 1, 2019 using the following details:  **class ID :**  **Enrollment password:** |

**Overview**

For this assignment, you are going to develop an application that keeps track of a fund manager’s daily operations. To keep the process as simple as possible, we will assume that there is only a single portfolio/ investment made and that is in the stock market.

The problem focuses on a container class of user-defined objects and the main features being assessed include your ability to handle several classes working together, the dynamic adding of new objects to a list and searching the list for particular objects.

The specifications given below are to be followed, even though they might differ from any known investment system.

**Task**

You are to write **two** source files as described below:

* The first file named **stockApp.py**, should contain the following two classes:
* The first is a class called Stock which defines a simple object type representing a stock being traded in a stock market.
* The second class called StockPortfolio defines objects which are containers of Stock objects.
* The second file named **stockDriver.py** defines a Python application, with main method, which creates one StockPortfolio object and allows the various methods of Stock to be called. This class will be an interactive application using the keyboard and the screen to interact with a human operator. It will not do calculations itself but will immediately pass user inputs as arguments to methods of StockPortfolio class.

**NOTE**: The final application will only execute correctly when both files have been defined completely and correctly but don't wait until you have completely written all three before you start compiling and testing your code. It is recommended that you save both source code files in the same directory on your file system and compile and test each class as you develop it using small separate programs to create and test objects of each class.

**The files**

The files you will require are:

**Class Stock** [saved in **stockApp.py**]

Each stock purchased will have 4 instance variables:

* stock name (a string)
* amount purchased in lot (an integer, where 1 lot refers to 1000 units)
* price purchased (per unit, a float, in RM)
* date purchased (a string, where the format is “MM/DD/YYYY”).

The methods of class Stock should include:

* An initialiser (constructor) method (\_\_init\_\_) which accepts FOUR attributes for a stock - name, amount purchased, price purchased, and date purchased. The method should initialise its object's attributes with these parametric values.
* A reader (getter) method for each of the four attributes. That is, a simple 'getter' method for the name, amount purchased, price purchased, and date purchased.
* A writer (setter) method for each of the attributes.
* A method named value that will return the value of this stock. A stock’s value is calculated by multiplying the purchased price with the amount purchased (in units). For example, a stock purchased with a price of RM3.50 for 2 lots (2000 units) will have a value of RM7000.00
* A method \_\_eq\_\_ that takes another Stock as parameter, and returns true if both stocks are equal (same), false otherwise. Two stocks are considered equal if they have the same name.
* A method \_\_lt\_\_ that takes another Stock as parameter, and returns true if this stock is having a value less than the parametric stock.
* A method \_\_le\_\_ that takes another Stock as parameter, and returns true if this stock is having a value less than or equal to the parametric stock.
* A string method (\_\_str\_\_) which return a single string containing the details of a stock. Such a string can be formed by concatenating the values of the four attributes name, amount purchased, price purchased, and date purchased in the format:

<name> (<amountInLots> lots @<price>/unit) bought <date>

One sample output is as shown below:

**HELP** (**2** lots @RM**2.35**/unit) bought in **10/10/2010**

It is recommended that once you have written the Stock class, you create a tiny program to test it. The testing program should be used to create one or two Stock objects and call some of the Stock methods. Compile and run the test program to check your work.

**Class StockPortfolio** [written and saved in the same file as **StockPortfolio** in **stockApp.py**]

This class is to be defined in the same file as class Stock. It declares a class of object with maintains a list of Stock objects. It will contain methods which enable the list to show the appropriate behaviours as required by the menu.

The StockPortfolio class should have a client’s name (of type string), and a collection of Stock objects bought by the client, no additional attribute is required:

The StockPortfolio class must also contain some methods which allow the collection of stocks to be managed. The methods of class StockPortfolio should include:

1. An initialiser (constructor) (\_\_init\_\_) with one argument of type string, which is used to initialise the client’s name. The constructor should also initialise an empty list for storing stocks purchased.
2. Getter (reader) methods for accessing the attributes, name and stock list, but only setter (writer) method for name.
3. A method named **addStock** which accepts as an argument an object of class Stock. This method will store a reference to this Stock object into the list.
4. A method named **noOfStocks** which returns the number of Stock objects currently stored in the list.
5. A method named **allStocks** that does not take any parameters, but returns a string containing all the details of all Stock objects, one per line.
6. A method named **totalValue** that does not take any parameters, but returns the total value of all the Stock objects in the portfolio.
7. A method named **mostExpensiveStock** that finds and returns the details of the stock with the highest value. This method does not take any parameters.

**NOTE**: You may assume that there will always be only one stock with the highest value. There will not be a tie between two stocks on values.

1. A method named **stockSummary** that takes a parametric string, representing a stock name. This method will find and return the total value, number of lots, and average value of the purchased stocks with the parametric name.
2. A method named **sellStock** that takes a **positive** integer representing the location of the stock to be sold (i.e deleted from the list). If the parametric integer is invalid or out of bounds, then the method should return **None**.
3. A method **sortedStocks** that takes a parameter of type string. The string represents the criteria to be used in sorting the stock list. The method will return a new list containing all the stocks sorted accordingly to the criteria mentioned in the parametric string in ascending order. The criteria could be **lots**, **price**, or **value**.
4. A method **saveToFile** that accepts a string representing the filename to save, and all Stock objects will be saved to a text file, one per line, with each attribute separated by a comma.
5. A method **loadFromFile** that accepts a string representing the filename where the data is to load from.

When you have written the StockPortfolio class - test it by creating a StockPortfolio object and invoking the methods from a small test program.

**stockDriver.py**

The aim of this file is to provide a user-interface for a modest application which uses a StockPortfolio container class. The user-interface is written as a 'console' application using the normal screen and keyboard to interact with a user via a simple text-based menu.

The user-interface should create a single StockPortfolio object and provide a menu of choices to the user.

**The Menu**

Stock portfolio for <user’s name>

---------------------------------

1 Buy stock

2 Display all stocks’ details

3 Display summary information about stock list

4 Display stocks with user-specified stock name

5 Sell a stock based on a given index

6 Display all stocks, sorted according to Stock values

7 Load stock information from file

8 Save stock information to file

0 Quit

Your choice?

**Sample Run**

**Please note - I’ve truncated display of menu to save space! The menu should display on the screen in full each time it is displayed. User’s input is in blue, and bold.**

Enter client’s name: **Jackson Howard**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

2 Display all stocks’ details

3 Display summary information about stock list

4 Display stocks with user-specified stock name

5 Sell a stock based on a given index

6 Display all stocks, sorted according to stock values

7 Load stock information from file

8 Save stock information to file

0 Quit

Your choice? **2**

**No stocks are in the list yet**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

:

8 Save stock information to file

0 Quit

Your choice? **8**

**No stocks are in the list yet**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

:

8 Save stock information to file

0 Quit

Your choice? **1**

Buying a new stock

Stock Name: **Weyland Consortium**

Amount of Lots: **2**

Price: **4.5**

Date: **3/2/2019**

... **bought successfully.**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

:

8 Save stock information to file

0 Quit

Your choice? **1**

Buying a new stock

Stock Name: **Haas-Bioroid**

Amount of Lots: **5**

Price: **2.75**

Date: **17/1/2019**

... **bought successfully.**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

2 Display all stocks’ details

:

8 Save stock information to file

0 Quit

Your choice? **2**

Details of all stocks:

**Weyland Consortium (2 lots @RM4.50/unit) bought in 3/2/2019**

**Haas-Bioroid (5 lots @RM2.75/unit) bought in 17/1/2019**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

2 Display all stocks’ details

3 Display summary information about stock list

:

8 Save stock information to file

0 Quit

Your choice? **3**

Summary Information

Number of stocks: **2**

Total value: **RM22750.00**

Most expensive stock: **Haas-Bioroid (5 lots @RM2.75/unit) bought in 17/1/2019 with a total value of RM13750.00**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

:

4 Display stocks with user-specified stock name

:

8 Save stock information to file

0 Quit

Your choice? **4**

Stock Name: **Weyland Consortium**

The stocks are:

**Weyland Consortium (2 lots @RM4.50/unit) bought in 3/2/2019 with a total value of RM9000**

**Average value of Weyland Consortium stocks: RM9000**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

:

4 Display stocks with user-specified stock name

:

8 Save stock information to file

0 Quit

Your choice? **4**

Stock Name: **Jinteki Biotech**

**No stock with that name could be found**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

:

5 Sell a stock based on a given index

:

8 Save stock information to file

0 Quit

Your choice? **5**

Stock to sell (1..2)? **0**

**Out of range...**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

:

8 Save stock information to file

0 Quit

Your choice? **1**

Buying a new stock

Stock Name: **Globalsec**

Amount of Lots: **6**

Price: **3.24**

Date: **8/1/2019**

... **bought successfully.**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

:

8 Save stock information to file

0 Quit

Your choice? **1**

Buying a new stock

Stock Name: **Jinteki Biotech**

Amount of Lots: **3**

Price: **6.22**

Date: **10/3/2019**

... **bought successfully.**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

:

8 Save stock information to file

0 Quit

Your choice? **1**

Buying a new stock

Stock Name: **Weyland Consortium**

Amount of Lots: **7**

Price: **1.01**

Date: **10/3/2019**

... **bought successfully.**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

2 Display all stocks’ details

:

8 Save stock information to file

0 Quit

Your choice? **2**

Details of all stocks:

**Weyland Consortium (2 lots @RM4.50/unit) bought in 3/2/2019**

**Haas-Bioroid (5 lots @RM2.75/unit) bought in 17/1/2019**

**Globalsec (6 lots @RM3.24/unit) bought in 8/1/2019**

**Jinteki Biotech (3 lots @RM6.22/unit) bought in 10/3/2019**

**Weyland Consortium (7 lots @RM1.01/unit) bought in 10/3/2019**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

2 Display all stocks’ details

3 Display summary information about stock list

:

8 Save stock information to file

0 Quit

Your choice? **3**

Summary Information

Number of stocks: **5**

Total value: **RM67920.00**

Most expensive stock: **Globalsec (6 lots @RM3.24/unit) bought in 8/1/2019 with a total value of RM19440.00**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

:

4 Display stocks with user-specified stock name

:

8 Save stock information to file

0 Quit

Your choice? **4**

Stock Name: **Weyland Consortium**

The stocks are:

**Weyland Consortium (2 lots @RM4.50/unit) bought in 3/2/2019 with a total value of RM9000**

**Weyland Consortium (7 lots @RM1.01/unit) bought in 10/3/2019 with a total value of RM7070**

**Average value of Weyland Consortium stocks: RM8035**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

:

6 Display all stocks, sorted according to stock values

7 Load stock information from file

8 Save stock information to file

0 Quit

Your choice? **6**

Sorting stocks

- criteria (lots, price, value)? **lots**

**Weyland Consortium (2 lots @RM4.50/unit) bought in 3/2/2019**

**Jinteki Biotech (3 lots @RM6.22/unit) bought in 10/3/2019**

**Haas-Bioroid (5 lots @RM2.75/unit) bought in 17/1/2019**

**Globalsec (6 lots @RM3.24/unit) bought in 8/1/2019**

**Weyland Consortium (7 lots @RM1.01/unit) bought in 10/3/2019**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

:

6 Display all stocks, sorted according to stock values

7 Load stock information from file

8 Save stock information to file

0 Quit

Your choice? **6**

Sorting stocks

- criteria (lots, price, value)? **price**

**Weyland Consortium (7 lots @RM1.01/unit) bought in 10/3/2019**

**Haas-Bioroid (5 lots @RM2.75/unit) bought in 17/1/2019**

**Globalsec (6 lots @RM3.24/unit) bought in 8/1/2019**

**Weyland Consortium (2 lots @RM4.50/unit) bought in 3/2/2019**

**Jinteki Biotech (3 lots @RM6.22/unit) bought in 10/3/2019**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

:

6 Display all stocks, sorted according to stock values

7 Load stock information from file

8 Save stock information to file

0 Quit

Your choice? **6**

Sorting stocks

- criteria (lots, price, value)? **value**

**Weyland Consortium (7 lots @RM1.01/unit) bought in 10/3/2019 with a total value of RM7070.00**

**Weyland Consortium (2 lots @RM4.50/unit) bought in 3/2/2019 with a total value of RM9000.00**

**Haas-Bioroid (5 lots @RM2.75/unit) bought in 17/1/2019 with a total value of RM13750.00**

**Jinteki Biotech (3 lots @RM6.22/unit) bought in 10/3/2019 with a total value of RM18660.00**

**Globalsec (6 lots @RM3.24/unit) bought in 8/1/2019 with a total value of RM19440.00**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

2 Display all stocks’ details

:

8 Save stock information to file

0 Quit

Your choice? **2**

Details of all stocks:

**Weyland Consortium (2 lots @RM4.50/unit) bought in 3/2/2019**

**Haas-Bioroid (5 lots @RM2.75/unit) bought in 17/1/2019**

**Globalsec (6 lots @RM3.24/unit) bought in 8/1/2019**

**Jinteki Biotech (3 lots @RM6.22/unit) bought in 10/3/2019**

**Weyland Consortium (7 lots @RM1.01/unit) bought in 10/3/2019**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

:

8 Save stock information to file

0 Quit

Your choice? **8**

File name to save to: **stockfile.txt**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

:

5 Sell a stock based on a given index

:

8 Save stock information to file

0 Quit

Your choice? **5**

Stock to sell (1..5)? **6**

**Out of range...**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

:

5 Sell a stock based on a given index

:

8 Save stock information to file

0 Quit

Your choice? **5**

Stock to sell (1..2)? **3**

**Globalsec (6 lots @RM3.24/unit) bought in 8/1/2019 has been sold.**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

2 Display all stocks’ details

:

8 Save stock information to file

0 Quit

Your choice? **2**

Details of all stocks:

**Weyland Consortium (2 lots @RM4.50/unit) bought in 3/2/2019**

**Haas-Bioroid (5 lots @RM2.75/unit) bought in 17/1/2019**

**Jinteki Biotech (3 lots @RM6.22/unit) bought in 10/3/2019**

**Weyland Consortium (7 lots @RM1.01/unit) bought in 10/3/2019**

Stock portfolio for Jackson Howard

----------------------------------

1 Buy stock

:

8 Save stock information to file

0 Quit

Your choice? **0**

**Adios...**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Enter client’s name: **Lily Lockwell**

Stock portfolio for Lily Lockwell

---------------------------------

1 Buy stock

2 Display all stocks’ details

3 Display summary information about stock list

4 Display stocks with user-specified stock name

5 Sell a stock based on a given index

6 Display all stocks, sorted according to stock values

7 Load stock information from file

8 Save stock information to file

0 Quit

Your choice? **2**

**No stocks are in the list yet**

Stock portfolio for Lily Lockwell

---------------------------------

1 Buy stock

:

7 Load stock information from file

8 Save stock information to file

0 Quit

Your choice? **7**

File name to load from: **stockfile.txt**

Stock portfolio for Lily Lockwell

---------------------------------

1 Buy stock

2 Display all stocks’ details

:

8 Save stock information to file

0 Quit

Your choice? **2**

Details of all stocks:

**Weyland Consortium (2 lots @RM4.50/unit) bought in 3/2/2019**

**Haas-Bioroid (5 lots @RM2.75/unit) bought in 17/1/2019**

**Globalsec (6 lots @RM3.24/unit) bought in 8/1/2019**

**Jinteki Biotech (3 lots @RM6.22/unit) bought in 10/3/2019**

**Weyland Consortium (7 lots @RM1.01/unit) bought in 10/3/2019**

Stock portfolio for Lily Lockwell

---------------------------------

1 Buy stock

:

5 Sell a stock based on a given index

:

8 Save stock information to file

0 Quit

Your choice? **5**

Stock to sell (1..2)? **2**

**Haas-Bioroid (5 lots @RM2.75/unit) bought in 17/1/2019 has been sold.**

Stock portfolio for Lily Lockwell

---------------------------------

1 Buy stock

2 Display all stocks’ details

:

8 Save stock information to file

0 Quit

Your choice? **2**

Details of all stocks:

**Weyland Consortium (2 lots @RM4.50/unit) bought in 3/2/2019**

**Globalsec (6 lots @RM3.24/unit) bought in 8/1/2019**

**Jinteki Biotech (3 lots @RM6.22/unit) bought in 10/3/2019**

**Weyland Consortium (7 lots @RM1.01/unit) bought in 10/3/2019**

Stock portfolio for Lily Lockwell

----------------------------------

1 Buy stock

:

8 Save stock information to file

0 Quit

Your choice? **0**

**Adios...**

**Notes**

1. Selecting menu item 6 **does not** affect the original data set, it merely shows how the data would look if sorted. That is selecting item 2 immediately after 6 should show the data set again unchanged.
2. Apart from the necessary entry of data via menu item 1, menu items should be selectable in any order.

**IMPORTANT**! The two sample runs shown above are to be used as a guide to check whether your program is correct. It should be correct when your program produces the same output as shown. When you submit your own execution, you should use your OWN DATA, and not the one shown above. Marks will be deducted if you are using the sample run data.

**Documentation**

* You should include comments in your code stating what each method does and explaining any complex sections of code.
* You should also include your student ID as comments within the code.
* You should of course use meaningful variable names so that your code is to some extent self-documenting.

**Submission Requirements**

You should submit the following:

* A cover-sheet stating your name and student number.
* A marking scheme stating you name and student number.
* Printouts of your code using Courier-New 10-point size font. Make sure that your long Python statements, if any, do not have the second line printed starting from the left-hand margin. You must break the long statement in appropriate lengths.
* You are **NOT** allowed to print in **landscape** orientation.
* Printouts demonstrating real interaction between yourself and your application.
* A Turnitin Similarity Report.
* A compressed file containing the source **.py** files uploaded to e-learning.

**Note about marking**

In order to obtain the marks for a given level, you must demonstrate that your program produces correct results. If you do, you will receive all of the marks indicated in the marking scheme below. If you do not, you will receive no marks.

**If your program does not meet the requirements by the due date you should obtain help from the lecturer and notify the lecturer that you will submit the assignment late (marks will be deducted).**

**Note about testing and plagiarism**

It is very important that you complete this assignment alone. You may of course obtain general assistance from the lecturing staff in the subject and your peers, but the coding must be carried out yourself. It is normally quite easy to detect when two or more students work together on their coding.

It is also very important that the demonstration of the results of your program using the given test data is produced using the identical version of the program to the printout of your source code.

**Students who hand in substantially similar assignments or whose programs do not match their demonstration of testing will fail the assignment.**

Any student suspected of copying, or of not producing the work himself or herself, can be called for **oral examination**, where the student will be expected to **demonstrate sufficient knowledge of the application** to show that it is his or her own original work.

**BIT100/DIP222 Assignment 2 Marking Scheme**

**Student Name: Student ID:**

|  |  |  |
| --- | --- | --- |
| **Marking Criteria** | **Allocated Marks** | **Awarded Marks** |
| **Stock Class (25%)** | | |
| Constructor with FOUR parameters: name, amount of lots, price, and date bought. | **2** |  |
| Getters and setters for the FOUR instance variables | **8** |  |
| Method named value for computing total value of a stock | **4** |  |
| method eq for testing equality of two stocks based on lots, price, and value | **3** |  |
| methods \_\_lt\_\_ and \_\_le\_\_ for comparing two stocks, based on price | **4** |  |
| string method \_\_str\_\_ that returns the detail of a property with required format | **4** |  |
| **StockPortfolio Class (33%)** | | |
| Constructor, with one parameter, and initialises an empty list | **2** |  |
| Getters for both name and list, but setter only for name | **3** |  |
| addStock for adding a new stock | **2** |  |
| noOfStocks returns number of stocks in the list | **2** |  |
| allStocks returns details of all stocks as a single string | **3** |  |
| totalValue returns total Value of all stocks in the list | **2** |  |
| mostExpensiveStockreturns the details of the stock with the highest value | **3** |  |
| stockSummary returns required properties as a string with required format; returns empty string if no property with the supplied stock name | **3** |  |
| sellStock returns the stock sold, based on location in the list | **3** |  |
| sortedStocks returns details of all stock sorted according to specific criteria, as a list | **4** |  |
| saveToFile - saving the properties to a text file, with appropriate format | **3** |  |
| loadFromFile - load properties from file | **3** |  |
| **Driver (27%)** | | |
| Program with main + various functions | **8** |  |
| Menu Display | **2** |  |
| Correct invocatons of methods of Stock and stockPortfolio (Deal with empty list properly at the beginning) | **12** |  |
| Allows repetition of Menu with appropriate loop structure | **2** |  |
| Selection of Functionality with appropriate conditional structure | **3** |  |
| **Quality of Code and Documentation (15%)** | | |
| Source code documentation includes student number, date and description of program | **2** |  |
| Function documentation includes purpose | **1** |  |
| Code are properly indented, and printed in Courier New 10pt | **2** |  |
| Meaningful identifier names | **3** |  |
| Cover-sheet with student name and number | **1** |  |
| Source code submitted together with sample output, with own data (not using data from sample runs) | **6** |  |
| Softcopy of assignment NOT uploaded to Turnitin. Report NOT attached with hardcopy. Compressed copy of your softcopy NOT uploaded to e-learning. | **-3** |  |
| **TOTAL** | **100** |  |

 **Assignment Cover Sheet**

Assignment No.: \_\_\_\_\_

|  |  |  |
| --- | --- | --- |
| **Student Information (For group assignment, please state names of all members)** | | **Grade/Marks** |
| **Name** | **ID** |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Module/Subject Information** | | **Office Acknowledgement** |
| **Module/Subject Code** |  |  |
| **Module/Subject Name** |  |  |
| **Lecturer/Tutor/Facilitator** | Dr. Tan Choon Ling |  |
| **Due Date** |  |  |
| **Assignment Title/Topic** | Assignment 2 |  |
| **Intake (where applicable)** |  |  |
| **Word Count** | n/a | **Date/Time** |

**Declaration**

* I/We have read and understood the Programme Handbook that explains on **plagiarism**, and I/we testify that, unless otherwise acknowledged, the work submitted herein is entirely my/our own.
* I/We declare that no part of this assignment has been written for me/us by any other person(s) except where such collaboration has been authorized by the lecturer concerned.
* I/We authorize the University to test any work submitted by me/us, using text comparison software, for instances of plagiarism. I/We understand this will involve the University or its contractors copying my/our work and storing it on a database to be used in future to test work submitted by others.

Note: 1) The attachment of this statement on any electronically submitted assignments will be deemed to have the same authority as a signed statement.

2) The Group Leader signs the declaration on behalf of all members.

|  |  |
| --- | --- |
| Signature: | Date: |
| E-mail: |  |

*ACA-F-020(010611:01)*

Page **1** of **2**

|  |
| --- |
| **Feedback/Comments\*** |
| **Main Strengths** |
|  |
|  |
|  |
|  |
|  |
|  |
| **Main Weaknesses** |
|  |
|  |
|  |
|  |
|  |
|  |
| **Suggestions for improvement** |
|  |
|  |
|  |
|  |
|  |
|  |

|  |  |
| --- | --- |
|  | **Student acknowledge feedback/comments** |
|  |
| Grader’s signature | Student’s signature: |
| Date: | Date: |

Note:

1. A soft and hard copy of the assignment shall be submitted.
2. The signed copy of the assignment cover sheet shall be retained by the marker.
3. If the Turnitin report is required, students have to submit it with the assignment. However, departments may allow students up to **THREE** (3) working days after submission of the assignment to submit the Turnitin report. The assignment shall only be marked upon the submission of the Turnitin report.

\*Use additional sheets if required.

*ACA-F-020(010611:01)*

Page **2** of **2**