DEPARTMENT OF COMPUTER & INFORMATION SCIENCES

CS994 OBJECT ORIENTED PROGRAMMING 2024/25

INDIVIDUAL LAB TEST

Mock Test

Duration: 1 hour and 30 minutes (excluding time to submit your work via MyPlace)

Available marks: 100

Contribution to overall mark: This assignment contributes 50% towards your final module mark.

General instructions:

Please read the assignment brief carefully and **attempt all *"Assessed Tasks"***. Even if you do not complete everything, make sure that you submit all your code. This is an **open-book** programming lab test (i.e. you are allowed to use: the book, lecture videos, notes, source code from previous practicals/tutorials, mock lab test code etc.), but it is still **under exam conditions** (i.e. no communication among students is allowed). **This is an individual assignment. Plagiarism/collusion[1] checks will be performed on all submissions. Late submissions are NOT allowed.**

Aims:

The aim of this assignment is to implement (**in Java**) a number of classes under the paradigm of Object-Orientation.

Learning outcomes:

After completing this assignment, you will have demonstrated experience of:

- understanding and using objects in common object-oriented languages;
- understanding and developing programs using class based object-oriented programming.

[Assignment brief continues on next page]

[1] ***Penalties apply.***

IMPORTANT - Marking Criteria (breakdown of the 100 available marks):

Your submission will be marked for:

- **Completeness** (i.e. has all required functionality been implemented?), and **correctness** (i.e. does everything work as specified?): As specified by the marks below each ***"Assessed Task"*** – **Total of 85 Marks**
- **Commenting** (i.e. is everything (classes/methods) commented as it should?) – **Total 10 Marks**
- **Style** (i.e. code layout, naming conventions, meaningful messages) – **Total 5 Marks**

Submission:

Your lecturer will give you instructions on how to submit your code via MyPlace.

[Assignment brief continues on next page]

Assessed Tasks

Fun with text messages

1. Implement a class named **`TextMessage`** that holds three data fields: the text message (a `String`), the sender's name, and the message's size in KB (an integer). Write a constructor that sets all data fields to meaningful default values. Include methods to set and get the values for each data field.

(10 Marks)

2. Implement a second constructor in the **`TextMessage`** class that accepts three parameters and uses their values to initialize the respective data fields.

(5 Marks)

3. Implement a method in the **`TextMessage`** class that prints **ALL** the details of a **`TextMessage`** object, i.e. the values of all data fields along with some descriptive text.

(5 Marks)

4. Implement a class named **`TextMessageManager`** that holds an `ArrayList` of **`TextMessage`** objects as a data field. Include a method to get (return) the value of the data field. Implement a method that takes a **`TextMessage`** object as a parameter and works according to the following specification:
If the list **already** contains the parameter **`TextMessage`** object, the method should **reject** the parameter and **print** the message "**This TextMessage object is already in your collection!"** on the screen. **Otherwise**, the method should add the parameter **`TextMessage`** object to the end of the list and **print** the message "**TextMessage object added successfully to your collection!"** on the screen.

(15 Marks)

[Assignment brief continues on next page]

5. In the **`TextMessageManager`** class, implement a method that takes two parameters:

- `index`: an integer, which represents a position in the `ArrayList`, and

- a **`TextMessage`** object.

The method works according to the following specification:

If the list **already** contains the parameter **`TextMessage`** object, the method should **reject** the parameter and **print** the message "**This TextMessage object is already in your collection!"** on the screen. **Otherwise**, the method should add the parameter **`TextMessage`** object to position `index` of the list and **print** the message "**TextMessage object added successfully to your collection!"** on the screen.

What if the value of the `index` parameter is not a valid position in the `ArrayList`? Include the appropriate error checking and error messages.

(10 Marks)

6. Implement a method in the **`TextMessageManager`** class that takes no parameters and works according to the following specification:

The method **returns** `true` if the `ArrayList` is empty. Otherwise, it **returns** `false`.

(5 Marks)

7. Implement a method in the **`TextMessageManager`** class that prints **ALL** the details of **ALL** **`TextMessage`** objects in the list. Your implementation **must use a while loop**.

What if the `ArrayList` is empty? Include the appropriate checking and messages.

(10 Marks)

[Assignment brief continues on next page]

8. Implement a method in the **`TextMessageManager`** class that takes one parameter: a search string. The method works according to the following specification: it prints **ALL** the details of **ALL `TextMessage`** objects in the list with a sender's name **equal to** the search string **OR** with a size of **less than** 100 KB. Your implementation **must use a for-each loop**.

(25 Marks)

Good luck!!!

[End of assignment brief]