1st SIT COURSEWORK 1 QUESTION PAPER:	**Year Long 2022/2023**

Module Code:	**CS5003NI**
Module Title:	**Data Structures and Specialist Programming**
Module Leader:	**Mr. Prithivi Maharjan**

Coursework Type:	**Individual**
Coursework Weight:	This coursework accounts for **30%** of your total module grades.
Submission Date:	**Week 12**
When Coursework is given out:	**Week 7**
Submission Instructions:	Submit the following to Islington College RTE department before the due date: ● **Files should be in .pdf** ● **.java, and .class file should be zipped**
Warning:	London Metropolitan University and Islington College takes Plagiarism seriously. Offenders will be dealt with sternly.

© London Metropolitan University

Plagiarism Notice

You are reminded that there exist regulations concerning plagiarism.

Extracts from University Regulations on Cheating, Plagiarism and Collusion

Section 2.3: “The following broad types of offence can be identified and are provided as indicative examples …...

(i) Cheating: including copying coursework.
(ii) Falsifying data in experimental results.
(iii) Personation, where a substitute takes an examination or test on behalf of the candidate. Both candidate and substitute may be guilty of an offence under these Regulations.
(iv) Bribery or attempted bribery of a person thought to have some influence on the candidate’s assessment.
(v) Collusion to present joint work as the work solely of one individual.
(vi) Plagiarism, where the work or ideas of another are presented as the candidate’s own.
(vii) Other conduct calculated to secure an advantage on assessment.
(viii) Assisting in any of the above.

Some notes on what this means for students:

(i) Copying another student's work is an offence, whether from a copy on paper or from a computer file, and in whatever form the intellectual property being copied takes, including text, mathematical notation and computer programs.
(ii) Taking extracts from published sources without attribution is an offence. To quote ideas, sometimes using extracts, is generally to be encouraged. Quoting ideas is achieved by stating an author's argument and attributing it, perhaps by quoting, immediately in the text, his or her name and year of publication, e.g., " e = mc2 (Einstein 1905)". A reference section at the end of your work should then list all such references in alphabetical order of authors' surnames. (There are variations on this referencing system which your tutors may prefer you to use.) If you wish to quote a paragraph or so from published work then indent the quotation on both left and right margins, using an italic font where practicable, and introduce the quotation with an attribution.

Further information in relation to the existing London Metropolitan University regulations concerning plagiarism can be obtained from http://www.londonmet.ac.uk/academic-regulations

Deliverables

You are required to submit two components before the submission deadline.

1. The software with the entire coursework project containing Java classes source codes. For this you can zip your NetBeans Java application projects.
2. Report in PDF format.

Case Study

Using appropriate classes from the Java Collections framework, you must create a Java software system. Information about **e-wallet accounts** is managed by the system. It should have the capacity to maintain the general card information for an indefinite number of accounts. It should have the information of the last five transactions for each account, as mentioned below. The system does not require storing the data to disk in order to make things simpler, but you can add this functionality if you choose. The following information is required to be included in a store's data:

Account general details:

1) Account ID
2) Account holder name
3) Account Type (e-sewa, Khalti and so on)
4) Account holder address
5) Account holder number
6) Account opening date
7) Account current balance
8) Account status (active, inactive)
9) Total Transactions

Transaction details:

1) Transaction from (e-sewa or khalti)
2) Transaction to (Account name)
3) Transaction amount
4) Transaction date
5) Transaction type (deposit or withdrawal)

Development

Part 1: System Features

1) The **user interface** of a system should be appropriately shown.

2) Can **create a new account** on the system.

3) Can **delete the closed account** from the system.

4) **Display a list of all the existing accounts** with a general description of each account on the computer screen. For example, Account holders' id, name, type, status, and so on as required.

5) If the detail of the account holder is mistaken then the **update** feature should be available to change the account holder detail. The unique identifier detail like account holder ID, phone number and others should not be changed.

6) You are required to use **proper data structure** to store the information of account holders and transaction information. For example, if you've used LinkedList you are not permitted to use the LinkedList provided by built-in library. You must create your own LinkedList functionality to add, remove and so on.

7) A **search** functionality should be there to search the e-wallet account by holder name. **Binary search algorithm** must be used in this feature. You can also add other searching criteria like by account type and account status.

8) The system should also have the functionality of **sorting**. You can **use any sorting algorithm that is present (taught) in our module**. You can sort either by ascending or descending or both. You are required to sort the data according to the account balance.

Part 2: System enhancement

To complete this part of the coursework, you must provide your own implementation of a data structure and searching/sorting algorithm.

As required by Part 1 of this project, you should have used some classes and its methods from the Java Collections framework in your implementation of the system. You should perform the following in this section:

Following requirements are must complete to complete part 2:

1. Validation and exception handling

For example, if a new account is registering then you should check if the account is already existing or not. In this case you can use a unique identifier like phone number to validate.

2. Programming styles

In this section correct naming & its convention must be used in overall Java code. Comments must be used appropriately.

3. Proper use of classes and methods

Proper utilisation of OOP concepts is required. Code redundancy should be minimised via use of methods and classes.

4. Use of proper messages

For example, when adding a new account, if some information is invalid (for example, only a number is entered in the full name field) then a proper message should be shown.

Report

A reflective report (1000 words), which concisely documents:

1. Well-structured report which includes font size, font family, alignment, introduction, conclusion and other report essentials.
2. Proper description of the tools and techniques used like NetBeans, Java and others. About the tool and how you have used them and what its benefits must be explained with proper images.
3. Detailed explanation of the **algorithm** you used including **how and why** you implemented them. Step wise explanation with diagram, flowchart diagram and pseudocode are required.
4. Detailed explanation of the **data structure** you used including **how and why** you implemented them. Step wise explanation with diagram, flowchart diagram and pseudocode are required.
5. Class diagram and detailed description of the classes' purpose, properties and methods is required.
6. Test cases with proper evidence (screenshot of the system) is required. At least 5 test cases are required.
7. A reflection of your experience of the development task, what issues you experienced, your solution to overcome it and any lessons learned.

Marking Scheme for CS5003 Individual Coursework 1

This coursework counts for 30% of the module mark. Please see the table below for the marking criteria and its weighting.

Marking Criteria		Weightage
Development Requirements		**70**
Part 1	**Feature**	**50**
1	Application User Interface	10
2	Create a new account	5
3	Delete closed account	5
4	Display the list of existing accounts	5
5	Update existing account information	5
6	Appropriate use of Data Structure	5
7	Use of binary search algorithm	10
8	Use of any sorting algorithm	5
Part 2	**Quality and style**	**20**
9	Validation and exception handling	5
10	Programming styles (comments and naming)	5
11	Proper use of classes and methods	5
12	Proper messages shown to user	5
Report Requirements		**30**
1	Well-structured report	5
2	Test Cases	5
3	Algorithm Explanation	5
4	Class diagram	5
5	Method description	5
6	Development process with tools used	5
	Total Mark	100