

# **EXERCISE: USER DEFINED CLASS**

**STIA1123 –PROGRAMMING 2**

**TASK 1**

Type, compile and analyze the following class Student:

public class Student{

//list of data members

private String matricNo;

private double test1, test2, averageMark;

//constructor

public Student(String matric, double ts1, double ts2 ){

matricNo = matric;

test1 = ts1;

test2 = ts2;

}

//method that returns some of students’ info

public String getStudentInfo()

{

return "\t"+matricNo+"\t\t"+averageMark;

}

//method that calculates the average of 2 tests

public void calculateAverage()

{

averageMark = (test1 + test2)/2;

}

}//end of class Student

1. Complete the following class (class TestStudent) that invokes the methods of the Student class and applies that concept of an array of objects. The program should produce the following expected outputs:

Matric No : s1111  
 Test 1 : 67  
 Test 2 : 45   
 Matric No : s2222  
 Test 1 : 67  
 Test 2 : 98  
 Matric No : s3333  
 Test 1 : 67  
 Test 2 : 90  
 

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 STUDENT INFORMATION  
 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 Matric No AverageMark  
 s1111 56.0  
 s2222 82.5  
 s3333 78.5

import java.util.Scanner;

public class TestStudent {

public static void main (String [] args) {

Scanner input = new Scanner(System.in);

String matricNo;

double test1,test2;

Student [] studDegree = new Student[3];

//complete your code here

}//end of main method

}//end of class TestStudent

1. Draw the UML class diagram for the class Student.
2. Draw a status memory of the array studDegree after Student’s objects are created.

**TASK 2**

A telecommunication provider, Maxcom, charges the following rates for telephone calls:

|  |  |
| --- | --- |
| **Rate Category** | **Rate per Minute (RM)** |
| **1 (Daytime )** | 0.07 |
| **2 (Evening)** | 0.12 |
| **3 (Off-Peak)** | 0.05 |

You are assigned by Maxcom to write a program that can calculate the charge for a call made by the user. The user needs to enter the duration in minutes of the call and the rate category. The program will then display output containing the rate category, the duration and the calculated charge. Then, the program will ask whether the user want to continue with the next calculation or quit the program. If the user chooses to continue, the program will repeat again with a new calculation.

An example of the program running is as follows:



**Note:** The underlined values are entered by the user.

In the following page, you are given **TWO (2)** incomplete classes :

i) MaxcomApp – the class representing the program.

ii) CallChargeCalculator – the class definition for CallChargeCalculator object to be used in MaxcomApp.

You have to complete both classes based on the given information.





**TASK 3**

Implement a class Average which consists of the main method and average method. Inside the main method, call average method. Print the value return by the average method.

Next, overload the average method such that if three integers are provided as parameters, the method returns the average of all three. Modify main method by adding a statement to invoke the overload method.

Then, create another overload average method to accept four integer parameters and return their average. Modify main method by adding a statement to invoke the latest overload method.

**TASK 4**

Given below is a class definition forBook. Type and compile the class, and answer the following questions:



Write a program in a class TestBook that can do the following:

1. Read an integer, n (representing the number of books), from the keyboard and then create an array name myLibrary of type Book with size n.
2. Using loop, create n objects of type Book and put them in the array myLibrary after reading the title and price for each book from the keyboard.

3. Print out the title of the book with the highest price.

4. Print out all titles of books that contain the term “Java” in their title.