| MODULE NAME: | MODULE CODE: |
|---|---|
| PROGRAMMING 1B | PROG6112 |

| | |
|---|---|
| **ASSESSMENT TYPE:** | **TEST (PAPER ONLY)** |
| **TOTAL MARK ALLOCATION:** | **60 MARKS** |
| **TOTAL HOURS:** | **1.5 HOURS (+10 minutes reading time)** |

**INSTRUCTIONS:**

1. *Please adhere to all instructions in the assessment booklet.*
2. *Independent work is required.*
3. *Five minutes per hour of the assessment to a maximum of 15 minutes is dedicated to reading time before the start of the assessment. You may make notes on your question paper, but not in your answer sheet. Calculators may not be used during reading time.*
4. *You may not leave the assessment venue during reading time, or during the first hour or during the last 15 minutes of the assessment.*
5. *Ensure that your name is on all pieces of paper or books that you will be submitting. Submit all the pages of this assessment's question paper as well as your answer script.*
6. *Answer all the questions on the answer sheets or in answer booklets provided. The phrase 'END OF PAPER' will appear after the final set question of this assessment.*
7. *Remember to work at a steady pace so that you are able to complete the assessment within the allocated time. Use the mark allocation as a guideline as to how much time to spend on each section.*

***Additional instructions:***

1. *This is an OPEN BOOK assessment.*
2. *Calculators are allowed.*
3. *For open book assessments the students may have open access to all resources inclusive of notes, books (hardcopy and e-books) and the internet. These resources may be accessed as hard copies or as electronic files on electronic devices. All electronic devices batteries must be fully charged before the assessment as no charging of devices will be permitted during the sitting of the assessment. The IIE and associated brands accept no liability for the loss or damage incurred to electronic devices used during open book assessments.*
4. *Answer All Questions.*
5. *Instructions for assessments including practical computer work:*
   - *Use of good programming practice and comments in code is compulsory.*
   - *Save your application in the location indicated by the administrator (e.g. the Z:\ drive or your local drive).*
   - *Create a folder as follows: use the module code and your own student number and create a folder with a folder name as per the format shown here:*
   - ***StudentNumber_ModuleCode_Test**. Save all files (including any source code files, template files, design files, image files, text files, database files, etc.) within this folder.*

- *E.g. if your student number is 12345, and you are writing an examination for the module PROG121, create a folder named* ***12345_Prog121_Test*** *and use this throughout the session to save all of your files.*

6. ***Important:*** *Upon completion of your assessment, you must save and close all your open files and double click the ExamLog application on your desktop. You must follow the instructions carefully to ensure that the information about the files that you have submitted for this assessment has been logged on the network. Specify the location of your source code on your question paper.*

## Learning Unit: Advanced Array Concepts

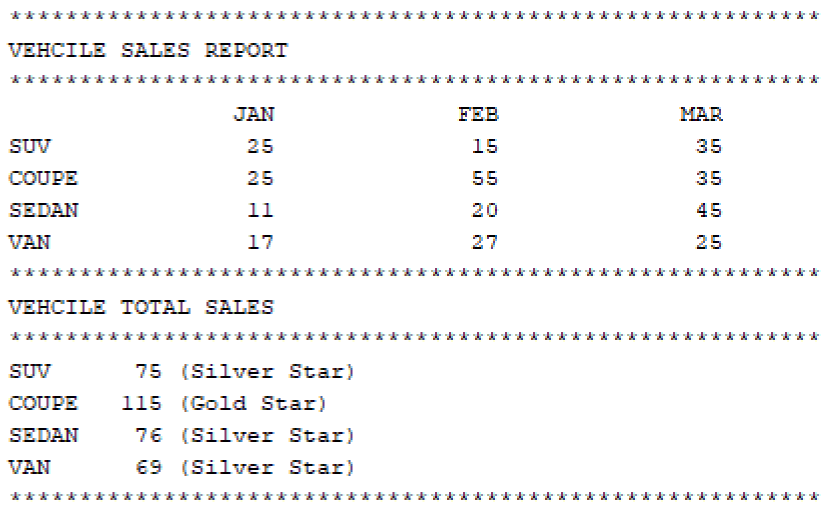
**Question 1** **(Marks: 30)**

Write a Java program to display three monthly sales of different vehicle types. The rows and columns represent the monthly sales of each vehicle type.

| | JAN | FEB | MAR |
|---|---|---|---|
| SUV | 25 | 15 | 35 |
| COUPE | 25 | 55 | 35 |
| SEDAN | 11 | 20 | 45 |
| VAN | 17 | 27 | 25 |

Using a Two-Dimensional array, produce the vehicle type sales report and the total sales made for each vehicle type. If the total sales made per month are greater than or equal to 100, **gold status** is awarded. If the monthly sales are less than 100, **silver status** is awarded.

**Example of output:**



```
**********************************************************
VEHCILE SALES REPORT
**********************************************************
                JAN            FEB            MAR
SUV              25             15             35
COUPE            25             55             35
SEDAN            11             20             45
VAN              17             27             25
**********************************************************
VEHCILE TOTAL SALES
**********************************************************
SUV       75  (Silver Star)
COUPE    115  (Gold Star)
SEDAN     76  (Silver Star)
VAN       69  (Silver Star)
**********************************************************
```

| **Marking Guideline** | **Mark** | **Examiner** | **Moderator** |
|---|---|---|---|
| **Declaration and Population of Two-Dimensional Array**<br>Excellent: Correct declaration – 8 marks<br>Good: Correct but just a minor error – 5 to 7 marks<br>Developing: Attempted but not correct – 2 to 4 marks<br>Poor: Not attempted – 0 marks | **8** | | |
| **Printing of the rows and columns**<br>Excellent: Correct print of rows and columns – 7 marks<br>Good: Correct but just a minor error – 5 to 6 marks<br>Developing: Attempted but not correct – 2 to 4 marks<br>Poor: Not attempted – 0 marks | **7** | | |
| **Accumulating totals**<br>Excellent: Correct total calculations – 7 marks<br>Good: Correct but just a minor error – 4 to 6 marks<br>Developing: Attempted but not correct – 2 to 3 marks<br>Poor: Not attempted – 0 marks | **7** | | |
| **Printing the total sales**<br>Excellent: Correct printing of total sales – 3 marks<br>Good: Correct but just a minor error – 2 marks<br>Developing: Attempted but not correct – 1 mark<br>Poor: Not attempted – 0 marks | **3** | | |
| **Report produced as per sample**<br>Excellent: Report produced correctly – 3 marks<br>Good: Attempted but not correct – 2 marks<br>Developing: Attempted but not correct – 1 mark<br>Poor: Not attempted – 0 marks | **3** | | |
| **Comments** | **2** | | |
| **TOTAL** | **30** | | |

**Learning Unit: Inheritance**

**Question 2** **(Marks: 30)**

Design a console application that will print a movie ticket for a customer. Make use of an abstract class named Tickets that contains variables to store the customer name, movie title, customer age and price of the movie. Create a constructor that accepts the customer name, movie title, customer age and price of the movie as parameters and create get methods for the variables. The Tickets class must implement an iTickets interface that contains the following:

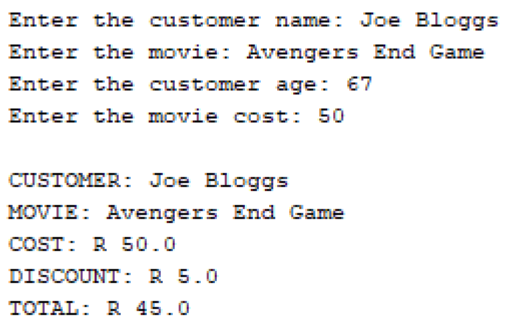
```
public interface iTickets
{
   public void print_tickets();
}
```

Create a subclass called TicketSales that extends the Tickets class. The TicketSales class must contain a constructor to accept the customer name, movie title, customer age and price of the movie as parameters. Write code for the print_tickets method which displays the customer name, movie title, movie price, discount and final cost due.

A discount of 10% is applied if the customer's age is greater than or equal to 65. If the customer is under 65 years old, no discount is given.

Finally, write a Movie_Tickets class to instantiate the TicketSales class. Sample output is shown below and you may use the same values to test your application.

**Example of input and output:**



```
Enter the customer name: Joe Bloggs
Enter the movie: Avengers End Game
Enter the customer age: 67
Enter the movie cost: 50

CUSTOMER: Joe Bloggs
MOVIE: Avengers End Game
COST: R 50.0
DISCOUNT: R 5.0
TOTAL: R 45.0
```

| Marking Guideline | Mark | Examiner | Moderator |
|---|---|---|---|
| **iTickets interface class created**<br>Excellent: Correct declaration – 6 marks<br>Good: Correct but just a minor error – 4 to 5 marks<br>Developing: Attempted but not correct – 2 to 3 marks<br>Poor: Not attempted – 0 marks | **6** | | |
| **Abstract class created with Constructor, Variables, Methods and Input and Output. Implements the Interface.**<br>Excellent: Correct – 10 marks<br>Good: Correct but just a minor error – 5 to 9 marks<br>Developing: Attempted but not correct – 2 to 4 marks<br>Poor: Not attempted – 0 marks | **10** | | |
| **Ticket Sales class created that extends the Ticket class**<br>Excellent: Correct – 7 marks<br>Good: Correct but just a minor error – 4 to 6 marks<br>Developing: Attempted but not correct – 2 to 3 marks<br>Poor: Not attempted – 0 marks | **7** | | |
| **Use the Movie Tickets class to capture input and instantiate the TicketSales class**<br>Excellent: Correct – 3 marks<br>Good: Correct but just a minor error – 2 marks<br>Developing: Attempted but not correct – 1 mark<br>Poor: Not attempted – 0 marks | **3** | | |
| **Output produced as per sample**<br>Excellent: Correct – 3 marks<br>Good: Attempted but not correct – 2 marks<br>Developing: Attempted but not correct – 1 mark<br>Poor: Not attempted – 0 marks | **3** | | |
| **Comments** | **1** | | |
| **TOTAL** | **30** | | |

**END OF PAPER**