TUNKU ABDUL RAHMAN UNIVERSITY OF MANAGEMENT AND TECHNOLOGY

FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY

ACADEMIC YEAR 2024/2025

MAY/JUNE EXAMINATION

AACS2204 OBJECT-ORIENTED PROGRAMMING TECHNIQUES

SATURDAY, 24 MAY 2025 TIME: 2.00 PM – 4.00 PM (2 HOURS)

DIPLOMA IN COMPUTER SCIENCE

DIPLOMA IN INFORMATION TECHNOLOGY

DIPLOMA IN SOFTWARE ENGINEERING

Instructions to Candidates:

Answer **ALL** questions. All questions carry equal marks.

This question paper consists of 4 questions on 6 printed pages.

Question 1

a) List and explain the **THREE (3)** basic principles of Object-Oriented Design. (9 marks)

b) Explain **THREE (3)** differences between the Procedural Paradigm and the Object-Oriented Paradigm in software design. (6 marks)

c) Explain the relationship between a class and an object in Object-Oriented Programming. (4 marks)

d) (i) Describe the role of visibility modifiers in achieving encapsulation in Object-Oriented Programming. (2 marks)

(ii) List any **TWO (2)** visibility modifiers and explain their visibility levels. (4 marks)

[Total: 25 marks]

Question 2

Consider the following UML class diagram in *Figure 1*, which represents a class Employee and its details.

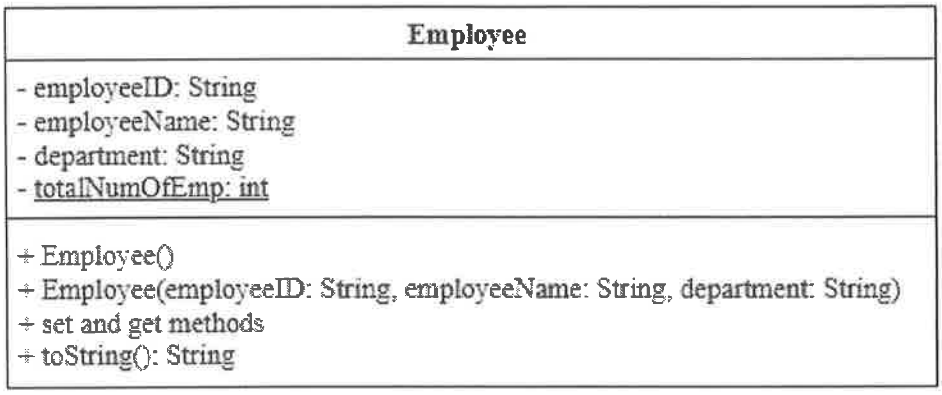


Figure 1: Class diagram

a) Based on *Figure 1*, construct a class named Employee with the following requirements:

- The constructors should increment the static variable totalNumOfEmp by 1 each time a new Employee object is created.
- Include set and get methods for all data members.
- Implement a toString() method that returns all information about an employee.

(16 marks)

Question 2 (Continued)

b) Write a driver program to test the class that you have created in Question 2 a). Your program should perform the following tasks:

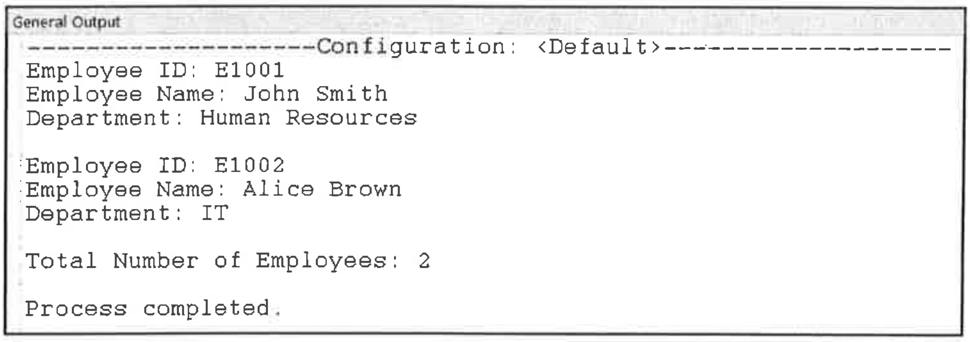
- Use the **no-argument constructor** to create an Employee object and set the values using set methods. Assign the following values:

Employee ID	Employee Name	Department
E1001	John Smith	Human Resources

- Use the **parameterised constructor** to create another Employee object with the following values:

Employee ID	Employee Name	Department
E1002	Alice Brown	IT

- Display the information of both Employee Objects, including the total number of employees. The sample output is shown in *Figure 2*.



```
--------------------Configuration: <Default>--------------------
Employee ID: E1001
Employee Name: John Smith
Department: Human Resources

Employee ID: E1002
Employee Name: Alice Brown
Department: IT

Total Number of Employees: 2

Process completed.
```

Figure 2: Sample Output

(9 marks)

[Total: 25 marks]

Question 3

The GourmetBites Restaurant requires an information system to manage customer billing. The billing is categorised into Dine-in Billing and Take Away Billing.

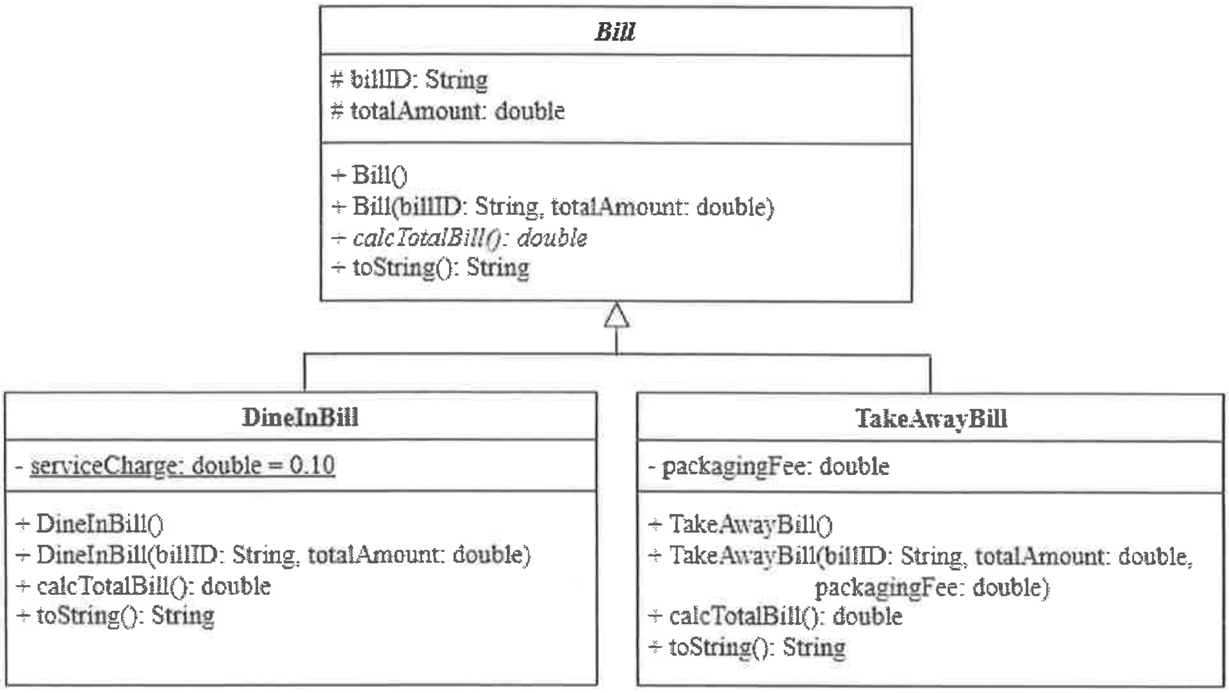

Bill
# billID: String
# totalAmount: double
+ Bill()
+ Bill(billID: String, totalAmount: double)
+ calcTotalBill(): double
+ toString(): String
DineInBill
- serviceCharge: double = 0.10
+ DineInBill()
+ DineInBill(billID: String, totalAmount: double)
+ calcTotalBill(): double
+ toString(): String
TakeAwayBill
- packagingFee: double
+ TakeAwayBill()
+ TakeAwayBill(billID: String, totalAmount: double, packagingFee: double)
+ calcTotalBill(): double
+ toString(): String

Figure 3: Class diagram

Each class must include the following:

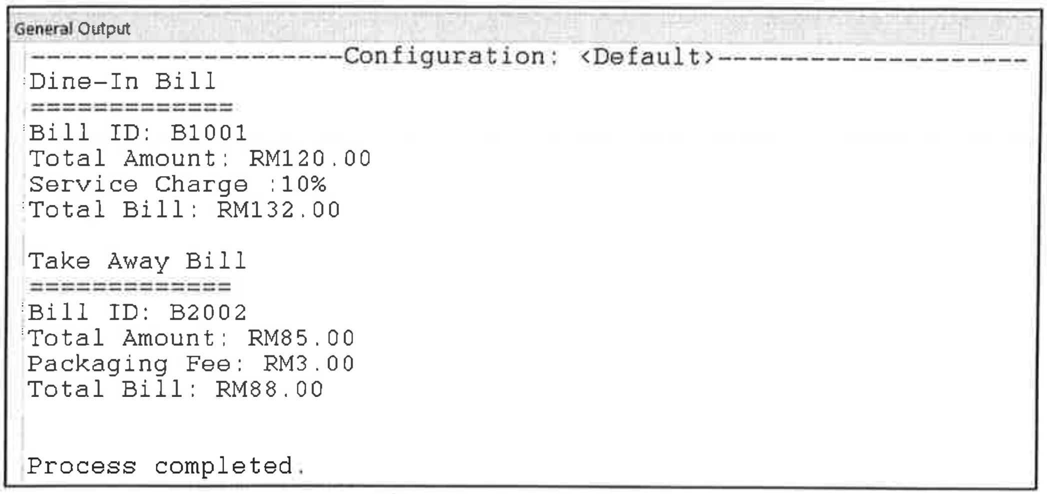
- A no-argument constructor.
- A parameterised constructor with the maximum number of parameters.
- A calcTotalBill() method. The formulas to calculate the total bill for both DineInBill class and TakeAwayBill class are shown below:
  - For DineInBill class:
    total bill = totalAmount + (totalAmount × serviceCharge)
  - For TakeAwayBill class:
    total bill = totalAmount + packagingFee
- A toString() method that returns all data member values of the bill.

a) Construct an **abstract superclass** named Bill that stores the common data members and methods.
Note: Exclude all set and get methods. (8 marks)

b) Construct a **subclass** named TakeAwayBill.
Note: Exclude all set and get methods. (9 marks)

Question 3 (Continued)

c) Assume that a **subclass** named DineInBill has been implemented based on the UML diagram in *Figure 3*. Construct a driver program to test the DineInBill and TakeAwayBill classes. The program should declare a polymorphic array to store a DineInBill object and a TakeAwayBill object. Using a for loop, determine the type of each object and print an appropriate label, such as "Dine-In Bill" or "Take Away Bill," followed by the bill details, including the total bill, as shown in *Figure 4*.



```
--------------------Configuration: <Default>--------------------
Dine-In Bill
=============
Bill ID: B1001
Total Amount: RM120.00
Service Charge :10%
Total Bill: RM132.00

Take Away Bill
=============
Bill ID: B2002
Total Amount: RM85.00
Packaging Fee: RM3.00
Total Bill: RM88.00


Process completed.
```

Figure 4: Sample Output

(8 marks)

[Total: 25 marks]

Question 4

a) (i) Create an **interface** named Discountable, which declares a **constant variable** named DISCOUNT_RATE with the value 0.05 and a method named calcPriceAfterDiscount(), which returns a double value. (3 marks)

(ii) Construct a class named Item that implements the Discountable interface. The following should be included in the class:

- An instance variable named price.
- The calcPriceAfterDiscount() method should calculate and return the price after applying the discount using the formula:

price after discount = price × (1 - DISCOUNT_RATE)

(5 marks)

b) Identify the appropriate class relationship for each of the following entity pairs and illustrate the relationships with a UML class diagram:

(i) Sale and Product: a Sale has multiple Products, but each Product can be listed in multiple Sales. (3 marks)

(ii) Vehicle and Car: a Car is a type of Vehicle. (3 marks)

Question 4 b) (Continued)

(iii) House and Room: a House has multiple Rooms, and each Room exists as a part of the House. (3 marks)

(iv) Doctor and Patient: a Doctor treats multiple Patients, and each Patient can be treated by multiple Doctors. (3 marks)

c) Analyse the following Java program and determine the output.

```
public class StringTest {
   public static void main(String[] args) {
      String str = "Welcome to Java Programming";

      System.out.println(str.length());
      System.out.println(str.indexOf("Java"));
      System.out.println(str.charAt(11));
      System.out.println(str.substring(11));
      System.out.println(str.lastIndexOf('m'));
   }
}
```

(5 marks)

[Total: 25 marks]