**Question 2 (60 Marks)**

Write a complete Java program based on the UML class diagram given in Figure 1 and driver program named **FinalGolf2.java** (refer to Figure 2). Your program should be able to produce the output shown in Figure 3.

**<<abstract>> GolfShop**

- brand: String

- price: double

- quantity: int

+ GolfShop()

+ GolfShop(String,double, int)

+ setPrice(double): void

+ getBrand(): String

+ getPrice(): double

+ getQuantity(): int

+ abstract display(): void

+ abstract calcPrice(): double

1..10

buy()

**CustomerA**

- CustName: String

- itemList: GolfShop[]

- numOfItem: int

- totalPay: double

+ CustomerA(String)

+ getName(): String

+ getTotalPay(): double

+ getNumOfItem(): int

+ buy(GolfShop): void

+ toString(): String

+ print(): void



**KayuGolf**

**Ball**

- KayuType: int

- numOfCan: int

+ KayuGolf()

+ KayuGolf(String,double,int,int)

+ Ball()

+ Ball(String, double, int, int)

**FinalGolf2**

**<<interface>> DiscConsiderable**

+ static main (String []): void

RATE = 0.15: double

calcDisc(): double

**Figure 1:** UML class diagram



**public class FinalGolf2**

**{**

**public static void main(String[] args)**

**{**

**GolfShop p1 = new KayuGolf("GolfDriver Wood", 0, 3, 2);**

**GolfShop p2 = new Ball("Wilson Anne", 225.00, 3, 24); GolfShop p3 = new Ball("Cobra", 240.00, 5, 18);**

**GolfShop p4 = new KayuGolf("Fairway Wood", 0, 4, 5);**

**CustomerA c = new CustomerA("Mikael Nadal"); System.out.println(c.toString());**

**c.buy(p1);**

**c.buy(p2);**

**c.buy(p3); c.buy(p4); c.print();**

**}**

**}**

**Figure 2: FinalGolf2.java**

You must write five complete Java programs, **GolfShop.java**, **Ball.java**, **KayuGolf.java**, **CustomerA.java** and **DiscConsiderable.java** based on the instruction given below:

(a) Write an abstract superclass named **GolfShop**. The class provides the following methods:

(i) Write the codes for a constructor with no argument that will do nothing.

(ii) Write the codes for a constructor with arguments that will initialize all the member attributes to the values received as arguments.

(iii) Write suitable codes for the accessor and mutator methods, if applicable.

(b) Write a subclass **Ball** and implements interface with the following codes:

(i) Write the codes for a constructor with no argument that will do nothing.

(ii) Write the codes for a constructor with arguments that will initialize all the member attributes for the class, including the superclass’s attributes.

(iii) Write the codes for **calcDisc()**method that will calculate and return the new price after the discount deducted, that discount received which is 15 percent of the price of the balls.

(iv) Write the codes for the abstract method **display()**that will print the brand, price, number of can case per set, discount received, and quantity of set of balls can case purchased. The method will also print the total price of set of balls purchased after the

discount deducted by calling **calcPrice()**.

(v) Write the codes for **calcPrice()** that will call **calcDisc()**and return the total price of balls purchased after the discount deducted.

(c) Write a subclass **KayuGolf** with the following codes:

(i) Write the codes for a constructor with no argument that will do nothing.

(ii) Write the codes for a constructor with arguments that will initialize all the member attributes for the class, including the superclass’s attributes.

(iii) Write the codes for the abstract method **display()**that will print the brand, name of racket type, price, and quantity of rackets purchased. Table 1 shows the prices vary according to the type of racket. The method will also print the total price of rackets purchased by calling **calcPrice()**.

**Table 1:** KayuGolf type and price

| **Kayu Type** | **Name of Kayu Golf Type** | **Price (RM)** |
| --- | --- | --- |
| 1 | Driver Wood | 80.00 |
| 2 | Fairway Wood | 110.00 |
| 3 | Iron | 235.00 |
| 4 | Putter | 185.00 |

(iv) Write the codes for **calcPrice()** that will calculate and return the total price of rackets purchased.

(d) Write an interface named **DiscConsiderable** with the following codes: (i) Declaration of one final static variable

(ii) Declaration of one abstract method

(e) Write a class named **CustomerA**. The class provides the following methods:

(i) Write the codes for a constructor with arguments that accepts the customer’s name.

This value should be assigned to the customer’s name member variable.

(ii) Write the codes for **buy(GolfShop product)** method that will receive one argument of instance, update the array named **itemList** to include the new instance of **GolfShop** in the array, update number of tennis items’ in the array **itemList**, and print the information of the item bought (using **display()** method) and update the total payment for all items bought (including discount).

(iii) Write the codes for **toString()** method to return message “*Welcome the customer*”

and “*List of Golf Items Bought*”.

(iv) Write the codes for **print()** method that will display message “*TOTAL: RMxx.xx*”.



The assessment criteria are shown in **Table 2**.

**Table 2:** Assessment criteria

| **Item** | **Criteria** | **Marks** |
| --- | --- | --- |
| **A** | The program is able to run and produce the correct outputs | 5 |
| **GolfShop.java** | | |
| **B** | Defining abstract class | 2.5 |
| Creating a constructors | 3 |
| Writing a suitable accessor and/ or mutator methods | 4 |
| Defining an abstract methods | 2 |
| **Ball.java** | | |
| **C** | Defining subclass of **Ball** | 2 |
| Creating a constructors | 3 |
| Implement and defining method **calcDisc()** | 2 |
| Implement and defining method **calcPrice()** | 2 |
| Implement and defining method **display()** | 4 |
| **KayuGolf.java** | |  |
| **D** | Defining subclass of **KayuGolf** | 1.5 |
| Creating a constructors | 3 |
| Implement and defining method **display()** | 8.5 |
| Implement and defining method **calcPrice()** | 2 |
| **DiscConsiderable.java** | |  |
| **E** | Defining an interface class | 1 |
| Declaration of one final, static field | 0.5 |
| Declaration of one abstract method | 0.5 |
| **CustomerA.java** | |  |
| **F** | Defining a class | 2.5 |
| Creating a constructor | 2.5 |
| Defining method **buy()** | 4.5 |
| Defining method **toString()** | 2 |
| Defining method **print()** | 2 |
| **Total** | | **60** |