การทดลองที่ 7-2

Chapter 11 Inheritance and Polymorphism

Program # 1 (Exercise 11.1 p.445 The Triangle class)

Design a class named Triangle that extends GeometricObject. The class contains: Three double data fields named side1, side2, and side3 with default values 1.0 to denote three sides of the triangle.

- A no-arg constructor that creates a default triangle.
- A constructor that creates a triangle with the specified side1, side2, and side3.
- The accessor methods for all three data fields.
- A method named getArea() that returns the area of this triangle.
- A method named getPerimeter() that returns the perimeter of this triangle.
- A method named toString() that returns a string description for the triangle.

For the formula to compute the area of a triangle, see Programming Exercise 2.19. The toString() method is implemented as follows: return "Triangle: side1 = " + side1 + " side2 = " + side2 + " side3 = " + side3;

Draw the UML diagrams for the classes Circle, Rectangle, Triangle and GeometricObject and implement the classes.

1) Write a test program that prompts the user to enter three sides of the triangle, a color, and a Boolean value to indicate whether the triangle is filled. The program should create a Triangle object with these sides and set the color and filled properties using the input.
2) The program should display the area, perimeter, color, and true or false to indicate whether it is filled or not. Also by using a method named displayObject(Object object), the program should displays the area, perimeter, and diameter if the object is a circle, and the length of each side, area, and perimeter if the object is a rectangle or a triangle.

Program # 2 (Exercise 11.8 p.446 New Account class)

An Account class was specified in Programming Exercise 9.7. Design a new Account class as follows:

- Add a new data field name of the String type to store the name of the customer.
- Add a new constructor that constructs an account with the specified name, id, and balance.
- Add a new data field named transactions whose type is ArrayList that stores the transaction for the accounts. Each transaction is an instance of the Transaction class. The Transaction class is defined as shown in Figure 11.6.

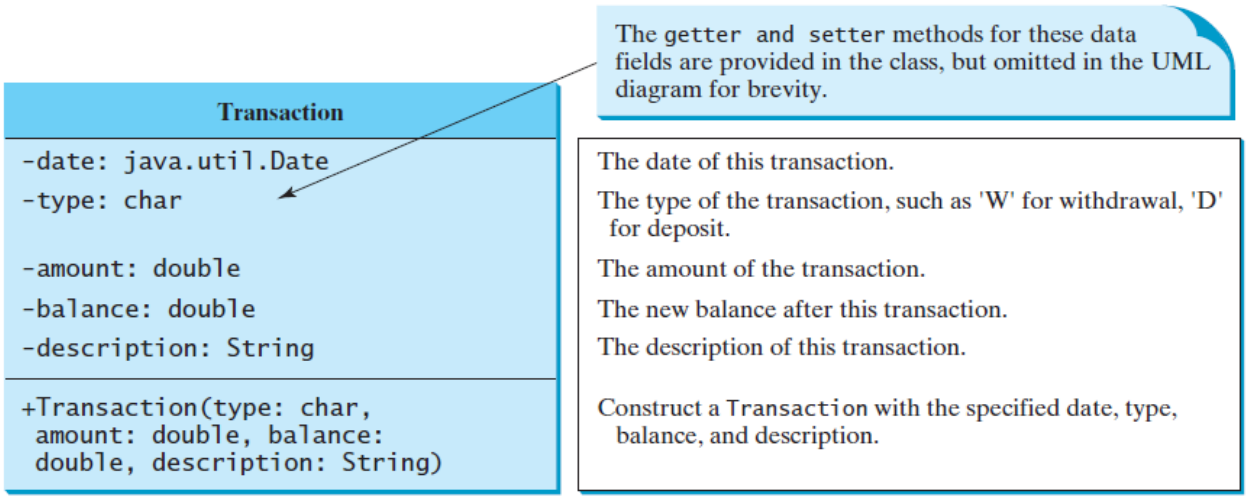
The getter and setter methods for these data fields are provided in the class, but omitted in the UML diagram for brevity.
Transaction
-date: java.util.Date
-type: char
-amount: double
-balance: double
-description: String
+Transaction(type: char, amount: double, balance: double, description: String)
The date of this transaction.
The type of the transaction, such as 'W' for withdrawal, 'D' for deposit.
The amount of the transaction.
The new balance after this transaction.
The description of this transaction.
Construct a Transaction with the specified date, type, balance, and description.

FIGURE 11.6 The Transaction class describes a transaction for a bank account.

- Modify the withdraw and deposit methods to add a transaction to the transactions array list.
- All other properties and methods are the same as in Programming Exercise 9.7.

Write a test program that creates an Account with annual interest rate 1.5%, balance 1000, id 1122, and name George. Deposit $30, $40, and $50 to the account and withdraw $5, $4, and $2 from the account. Print an account summary that shows account holder name, interest rate, balance, and all transactions.

ตัวอย่างผลลัพธ์การทำงานของโปรแกรม

```
Name: George
Account ID: 1122
Annual interest rate: 1.65
Balance: 1109.00
Date                           Type        Amount      Balance
Sun Mar 07 20:47:59 ICT 2021   D           30.00       1030.00
Sun Mar 07 20:47:59 ICT 2021   D           40.00       1070.00
Sun Mar 07 20:47:59 ICT 2021   D           50.00       1120.00
Sun Mar 07 20:47:59 ICT 2021   W           5.00        1115.00
Sun Mar 07 20:47:59 ICT 2021   W           4.00        1111.00
Sun Mar 07 20:47:59 ICT 2021   W           2.00        1109.00
```
