การทดลองที่ 4

Chapter 9 Objects and Classes – Chapter 10 Object-Oriented Thinking

Programming Exercise 9.7 p.361 (The Account class)

Design a class named Account that contains:

- A private int data field named id for the account (default 0).
- A private double data field named balance for the account (default 0).
- A private double data field named annualInterestRate that stores the currentinterest rate (default 0). Assume all accounts have the same interest rate.
- A private Date data field named dateCreated that stores the date when the account was created.
- A no-arg constructor that creates a default account.
- A constructor that creates an account with the specified id and initial balance.
- The accessor and mutator methods for id, balance, and annualInterestRate.
- The accessor method for dateCreated.
- A method named getMonthlyInterestRate() that returns the monthly interest rate.
- A method named getMonthlyInterest() that returns the monthly interest.
- A method named withdraw that withdraws a specified amount from the account.
- A method named deposit that deposits a specified amount to the account.

Draw the UML diagram for the class and then implement the class. (Hint: The method getMonthlyInterest() is to return monthly interest, not the interest rate. Monthly interest is balance * monthlyInterestRate. monthlyInterestRate is annualInterestRate / 12.

Note that annualInterestRate is a percentage, e.g., like 4.5%. You need to divide it by 100.)

Write a test program that creates an Account object with an account ID of 1122, a balance of $20,000, and an annual interest rate of 4.5%. Use the withdraw method to withdraw $2,500, use the deposit method to deposit $3,000, and print the balance, the monthly interest, and the date when this account was created.

Programming Exercise 9.9 p.362 (Geometry: n-sided regular polygon)

In an n-sided regular polygon, all sides have the same length and all angles have the same degree (i.e., the polygon is both equilateral and equiangular). Design a class named RegularPolygon that contains:

All private data fields
All constructors
The accessor and mutator methods for all data fields.
The method getPerimeter() that returns the perimeter of the polygon.
The method getArea() that returns the area of the polygon. The formula for computing the area of a regular polygon is

$$Area = \frac{n \times s^2}{4 \times \tan(\frac{\pi}{n})}$$

Draw the UML diagram for the class and then implement the class. Write a test program that creates three RegularPolygon objects, created using the no-arg constructor, using RegularPolygon(6, 4), and using RegularPolygon(10, 4, 5.6, 7.8). For each object, display its perimeter and area.

Programming Exercise 9.11 p.363 (Algebra: 2 * 2 linear equations)

Design a class named LinearEquation for a 2 * 2 system of linear equations:

$$\begin{aligned} ax + by &= e \\ cx + dy &= f \end{aligned} \qquad x = \frac{ed - bf}{ad - bc} \qquad y = \frac{af - ec}{ad - bc}$$

The class contains:

- All private data fields.
- All constructors.
- Six getter methods for all private data fields.
- The accessor and mutator methods for all data fields.
- A method named isSolvable() that returns true if ad - bc is not 0.
- Methods getX() and getY() that return the solution for the equation.

Draw the UML diagram for the class and then implement the class. Write a test program that prompts the user to enter a, b, c, d, e, and f and displays the result. If ad - bc is 0, report that “The equation has no solution.” See sample run

$$\begin{aligned} 3.4x + 50.2y &= 44.5 \\ 2.1x + 0.55y &= 5.9 \end{aligned}$$

Programming Exercise 10.7 p.401 (Game: ATM machine)

Use the Account class created in Programming Exercise 9.7 to simulate an ATM machine. Create ten accounts in an array with id 0, 1, ..., 9, and initial balance $100. The system prompts the user to enter an id. If the id is entered incorrectly, ask the user to enter a correct id. Once an id is accepted, the main menu is displayed as shown in the sample run. You can enter a choice 1 for viewing the current balance, 2 for withdrawing money, 3 for depositing money, and 4 for exiting the main menu. Once you exit, the system will prompt for an id again. Thus, once the system starts, it will not stop.

```
Enter an id: 4 <Enter>

Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 1 <Enter>
The balance is 100.0

Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 2 <Enter>
Enter an amount to withdraw: 3 <Enter>

Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 1 <Enter>
The balance is 97.0
```

```
Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 3 <Enter>
Enter an amount to deposit: 10 <Enter>

Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 1 <Enter>
The balance is 107.0

Main menu
1: check balance
2: withdraw
3: deposit
4: exit
Enter a choice: 4 <Enter>
Enter an id: <Enter>
```