NTOU Java Programming Homework 1

Spring 2024

SOSEL

Service-Oriented Software Engineering Lab

Homework 1-1

- Please develop a program that runs 1000 games of craps (Fig. 6.8 Craps.java) and answers the following questions:
  - a) How many games are won on the first roll, second roll, ..., twentieth roll and after the twentieth roll?
  - b) How many games are lost on the first roll, second roll, ..., twentieth roll and after the twentieth roll?
  - c) What are the chances of winning at craps? (贏的機率有多高)
  - d) What is the average length of a game of craps? (平均一次遊戲擲幾次骰子)

Sample Output

```
224 games won and 99 games lost on roll #1
74 games won and 119 games lost on roll #2
50 games won and 96 games lost on roll #3
33 games won and 54 games lost on roll #4
23 games won and 47 games lost on roll #5
22 games won and 37 games lost on roll #6
18 games won and 13 games lost on roll #7
8 games won and 18 games lost on roll #8
7 games won and 14 games lost on roll #9
5 games won and 6 games lost on roll #10
5 games won and 6 games lost on roll #11
4 games won and 3 games lost on roll #12
1 games won and 3 games lost on roll #13
1 games won and 0 games lost on roll #14
0 games won and 4 games lost on roll #15
1 games won and 0 games lost on roll #16
0 games won and 0 games lost on roll #17
0 games won and 1 games lost on roll #18
0 games won and 0 games lost on roll #19
0 games won and 0 games lost on roll #20
3 games won and 1 games lost on rolls after the 20th roll

The chances of winning are 479 / 1000 = 47.90%
The average game length is 3.37 rolls.
```

Hint

- Fig. 6.8 can be renamed and slightly modified to record necessary information.
- An array needs to be created to record the number of occurrences of different dice rolls.
- For example, diceRollingWonTimes[5] = 23 means there are 23 times for "5 rolls and win", and diceRollingLost Times[5] = 47 means there are 47 times for "5 rolls and lose".
- During the simulation, the number of rolls can be continuously accumulated to calculate the average.

Homework 1-2₁

- Create a class called *Complex* for performing arithmetic with complex numbers. Complex numbers have the form

  *realPart + imaginaryPart * i*

  - Use *double* variables *real* and *imaginary* to represent the private data of the class.
  - Provide a constructor that enables an object of this class to be initialized when it is declared. (以設定實部與虛部)
  - Provide a no-argument constructor with random positive double values (from 0 to 1).
    - Please use *java.security.SecureRandom* to generate random numbers.

Homework 1-2₂

- Provide public methods that perform the following operations:
  - Return the conjugates (共軛) of a Complex number
  - Return the absolute number (絕對值) of a Complex number
  - Add two Complex numbers
  - Subtract two Complex numbers
  - Multiply two Complex numbers
  - Divide two Complex numbers
  - Print Complex numbers in the form a + bi , where a is the real part and b is the imaginary part. (顯示到小數點後兩位)
- References:
  - https://en.wikipedia.org/wiki/Complex_number
  - https://zh.wikibooks.org/zh-tw/複數

Sample Output

```
a = 1.10 + 2.20i                 c = 0.29 + 0.46i
Conjugates of a = 1.10 - 2.20i   Conjugates of c = 0.29 - 0.46i
Absolute value of a = 2.46       Absolute value of c = 0.54

b = 3.30 - 4.40i                 d = 0.79 + 0.68i
Conjugates of b = 3.30 + 4.40i   Conjugates of d = 0.79 - 0.68i
Absolute value of b = 5.50       Absolute value of d = 1.04

a + b = 4.40 - 2.20i             c + d = 1.08 + 1.14i
a - b = -2.20 + 6.60i            c - d = -0.49 - 0.23i
a × b = 13.31 + 2.42i            c × d = -0.08 + 0.56i
a ÷ b = -0.20 + 0.40i            c ÷ d = 0.50 + 0.14i
```

- *No user input is required. Just create a new Complex object in the Test class, call its related methods, and print the output.*
- *a and b are directly assgned, and c and d are random numbers.*

Hint

- For the *conjugates, add, subtract, multiply, and divide* methods of class *Complex*, return a new Complex object with the results of the calculations.
  - You need *java.lang.Math*
- For the *absoluteValue* method of class *Complex*, return a double value.
- To print an object, please add a public method: *toString()* that returns a String.
  - You can refer to https://www.javatpoint.com/understanding-toString()-method
- Please ensure that executing ComplexTest.java can produce the expected output.

Homework 1-3$_1$

- Please develop a Java project to simulate a course management system.
  - Each course includes a course name, a list of students, and a maximum number of students.
  - A Student class containing a student name is provided.

Homework 1-3$_2$

- Please define a Course class containing a course name (final), a list of students (ArrayList), and a maximum number of students (final).
  - Please provide appropriate constructors and methods.
  - Please include methods to add, remove students, and set the course name.
  - Please ensure appropriate exception handling for invalid inputs such as null course names or negative maximum students.
  - Please design an appropriate *toString()* method.

Homework 1-3₃

- Please define a CourseManager class to manage all courses.
  - This class should include a static variable to track all courses.
  - This class should include methods to add courses, query courses, and print out all courses' detailed information.

Homework 1-3$_4$

- □ Please complete the “//TODO” parts in the given code to produce the expected output in homework-1-3-results.txt.

Hint

- Please use the following mechanisms:
  - Object composition (three layers)
  - ArrayList
  - Exception handling.

Homework Requirements

- The naming should conform to the CamelCase style.
- “Package“ is required: ntou.cs.java2024.
- There must be comments in the class (at least your student number and name, and brief descriptions for each class).
- Each assignment must have more than two classes, one of which is a test class (only including main)
  - Class Names:
    - 1-1: CrapsAnalysis and CrapsAnalysisTest
    - 1-2: Complex and ComplexTest
    - 1-3: Student, Course, CourseManager, and CourseManagerTest
- Please submit files including .java files and .class files (upload them to TronClass).
- Code that fails to compile or execute is not accepted.