海大資工 Java程式設計課程

SOSEL

Service-Oriented Software Engineering Lab

Fall

2017

Homework 2-1

- Write an application that runs 1000 games of craps (Fig. 6.8) 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)

Example

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 可略做修改，以紀錄必要的資訊。
- □ 需宣告陣列，以紀錄不同骰子投擲次數的出現次數（如diceRollingWonTimes[5] = 23代表擲5次、結果為勝有23次、 diceRollingLostTimes[5] = 47代表擲擲5次、結果為負有47次)。
- □ 模擬過程中可持續累加每次的骰子投擲次數，即可算平均值。

Homework 2-2$_1$

- http://www.youtube.com/watch?v=dicgjskLVJc

- 請設計一個縮減版的UNO遊戲(共48張牌)的基本功能，分為藍色、綠色、紅色、黃色四種顏色：
  - 字牌
    - 每種顏色各10張牌，牌面為0~9的數字
  - 指令牌
    - 4張 Draw 2：要求下家抽取兩張牌 (四種顏色各1)
    - 4張 Skip：此輪不需抽牌 (四種顏色各1)

Homework 2-2$_2$

- 請設計此UNO牌之資料結構類別: UNOCard，並設計一個類別DeckOfUNOCards 代表整副牌，以及測試用的類別UNOCardsTest
  - UNOCard.java:
    - UNOCard需包含三個field: type、coloredSuit、number
      - type型態為enum型態UNOType，其允許的constant為NUMBER、SKIP、DRAW2
      - coloredSuit型態為enum型態ColorType，其允許的constant為BLUE、GREEN、RED、YELLOW
      - number型態為int，且範圍需限定為-1~9之間 (-1代表為功能牌)
        - 請提供一個public static final的int變數INVALID，設定其值為-1，在程式其他部分請勿出現”-1”之設定或判斷。
    - 請設計合適的constructor，可以設定上述之field。
    - 請提供toString功能，可以直接列印UNO物件之詳細資訊。

Homework 2-2$_3$

- DeckOfUNOCards.java
  - constructor中需初始化一副具備48張牌 (依題目規定)的UNO牌
  - 提供洗牌(shuffle)功能
  - 提供發牌(dealCard)功能
    - 開始的發牌與後來的摸牌均可透過此功能

Homework 2-2$_4$

- UNOCardsTest.java
  - 顯示未洗牌(出廠狀態)的全部UNO牌
  - 顯示洗牌後的全部UNO牌
  - 設定玩家為四人，各發4張牌後，個別顯示四個玩家目前手中的牌，以及目前在排堆中的牌(依照由上往下的順序，應有32張)

Example

□ 未洗牌:

BLUE 0
BLUE 1
BLUE 2
…
BLUE SKIP
BLUE DRAW2
GREEN 0
GREEN 1
GREEN 2
…

□ 洗牌後:

YELLOW 9
BLUE 4
BLUE 2
RED 0
RED SKIP
GREEN 5
RED DRAW2
…

□ 發牌後:

First Player:
YELLOW 9
RED SKIP

Second Player:
BLUE 4
GREEN 5
…

Remaining Cards:
YELLOW 4
BLUE 3
…

Hint

- (1) 建議將UNOType與ColorType都設定為public enum，並存成UNOType.java以及ColorType.java，以便利其他類別存取
- (2) 可透過values()方法取出enum型態的所有constants
- (3) 直接把constant當成string印出，就會顯示constant名稱
  - 可用String.format()將轉換後的字串放在變數中 (用法非常類似printf)
  - 參考範例：

```
String output = "";
for ( UNOType type : UNOType.values() )
{
    output += String.format("%s", type) + "\n";
}
System.out.println(output);
```

Homework 2-3

- 你要設計一個可產生迴旋矩陣的程式，分為幾種需求：
  - 輸入邊長與以及順時針方向與外迴旋設定，可輸出順時針方向外迴旋矩陣
    - 邊長設定低於100
  - 輸入邊長與以及逆時針方向與外迴旋設定，可輸出逆時針方向外迴旋矩陣
    - 邊長設定低於100
  - 輸入邊長與以及順時針方向與內迴旋設定，可輸出順時針方向內迴旋矩陣
    - 邊長設定低於100且需為奇數
  - 輸入邊長與以及逆時針方向與內迴旋設定，可輸出逆時針方向內迴旋矩陣
    - 邊長設定低於100且需為奇數
- 輸出矩陣之處理必須要用Enhanced For結構

Example: 邊長為6的順時針外迴旋矩陣

```
請輸入順時針(1)或逆時針(2)？1
請輸入內迴旋(1)或外迴旋(2)？2
請輸入邊長(<100)？6

1       2       3       4       5       6
20      21      22      23      24      7
19      32      33      34      25      8
18      31      36      35      26      9
17      30      29      28      27      10
16      15      14      13      12      11
```

Example: 邊長為6的逆時針外迴旋矩陣

```
請輸入順時針(1)或逆時針(2)？2
請輸入內迴旋(1)或外迴旋(2)？2
請輸入邊長(<100)？6

1      20     19     18     17     16
2      21     32     31     30     15
3      22     33     36     29     14
4      23     34     35     28     13
5      24     25     26     27     12
6      7      8      9      10     11
```

Example: 邊長為7的順時針內迴旋矩陣

```
請輸入順時針(1)或逆時針(2)？1
請輸入內迴旋(1)或外迴旋(2)？1
請輸入邊長(奇數且<100)？6
邊長應為奇數，請重新輸入！
請輸入邊長(奇數且<100)？7

43      44      45      46      47      48      49
42      21      22      23      24      25      26
41      20      7       8       9       10      27
40      19      6       1       2       11      28
39      18      5       4       3       12      29
38      17      16      15      14      13      30
37      36      35      34      33      32      31
```

Example: 邊長為7的逆時針內迴旋矩陣

```
請輸入順時針(1)或逆時針(2)？2
請輸入內迴旋(1)或外迴旋(2)？1
請輸入邊長(奇數且<100)？7

49      48      47      46      45      44      43
26      25      24      23      22      21      42
27      10      9       8       7       20      41
28      11      2       1       6       19      40
29      12      3       4       5       18      39
30      13      14      15      16      17      38
31      32      33      34      35      36      37
```

作業要求

- 命名都要符合Camel Case
- 類別都要設定package，名稱為ntou.cs.java2017.你的英文名字.hw2
- 類別內要有註解，至少要簡述此類別與每個方法
- 每題都至少要有兩個類別(可以更多)，一個為主要類別，一個為測試類別(只包含main)
  - 2-1: CrapsSimulation.java, CrapsSimulationTest.java
  - 2-2: UNOCard.java, DeckOfUNOCards, UNOCardsTest.java
  - 2-3: SpriralMatrix.java, SpriralMatrixTest.java
- 請繳交電子檔 ，電子檔包含.java檔與依套件階層目錄擺放之.class檔(上傳至TronClass)
- 屍體(無法compile或執行)不計分