Homework 5

Moodle Submission Deadline: 2024/6/3 (Monday) 23:59

Problem 1: Minesweeper & Blackjack [hw5_1m.py, hw5_1b.py]

Your tasks are solving HW4 again, i.e., Minesweeper and Blackjack. This time, you are asked to use functions and dictionary. In each problem, you need to design **at least three functions.** Besides, you **must use dictionary to store the board in Minesweeper and to store the current draw cards of players in Blackjack.** The sample input and output are the same as HW4.

Problem 2: Two-Player Dice Rice [hw5_2.py]

Create a board game where two players move around a board by rolling a dice. The board consists of squares that are either safe or have penalties for landing on them. The player who first reaches the end of the board wins. If both players reach the end in the same turn, both wins. You need to write at least four functions in your code.

Requirements:

(1) The board should be represented by a list or a dictionary, with each element or key representing a square on the board. The value should indicate whether the square is *safe* or has a *penalty*. Safe and penalty squares should be determined randomly with a probability of 0.3 for penalties. The length of the board is 30.

(2) There are two players, starting at the left-most square and aiming to reach the right-most.

(3) Each turn, a player rolls a dice to move a number of squares on the board. The dice should be simulated using the *randint* function from the *random* module.

(4) If a player lands on a safe square, their turn ends. If a player lands on a penalty square, they lose their next turn. If both players land on penalty squares, both lose the next turn.

(5) After each turn, print the current board and both players' numbers with markings:
- If Player A's landing is safe, mark the square as 'A'; otherwise, mark as 'a'.
- If Player B's landing is safe, mark the square as 'B'; otherwise, mark as 'b'.
- If both players land at the same safe square, mark it as 'X'.
- If both players land at the same penalty square, mark it as 'x'.
- If squares are never landed on, print '_' for the empty square.

(6) If Player A reaches the end of the board, Player B can still have their last turn, provided they were not on a penalty square in their previous turn. It is because Player A moves first.

(7) The game ends when one or both players reach the end of the board. If both players reach the end in the same turn, they both win. Winner(s) need to stay at the right-most square.

(8) When the game ends, print the hidden penalty squares with the following markings:
- Mark penalty squares as 'P', and mark safe squares as '_'.
- Do not show the players' final squares when printing the penalty squares.

Sample input & output

```
c:\workspace>python hw5_2.py
__x_____________________________ (A: 2, B: 2)
__x_____________________________ (A: 0, B: 0)
___b_A__________________________ (A: 3, B: 1)
___b_____A______________________ (A: 6, B: 0)
_____b_________A________________ (A: 6, B: 3)
_____b___________A______________ (A: 2, B: 0)
___________b_____a______________ (A: 1, B: 6)
___________b_____a______________ (A: 0, B: 0)
_________________B___a__________ (A: 2, B: 6)
___________________Ba___________ (A: 0, B: 3)
_____________________B_A________ (A: 5, B: 4)
_____________________B__A_______ (A: 3, B: 1)

Player A wins!

PPPPP_PPP___P__P____P_PP____P_

c:\workspace>
```

```
c:\workspace>python hw5_2.py
_B_a____________________________ (A: 3, B: 1)
__Ba____________________________ (A: 0, B: 1)
____B__A________________________ (A: 5, B: 3)
_________B_A____________________ (A: 5, B: 6)
_____________Ba_________________ (A: 3, B: 4)
______________a___b_____________ (A: 0, B: 6)
__________________x_____________ (A: 5, B: 0)
__________________aB____________ (A: 0, B: 1)
____________________A___b_______ (A: 2, B: 6)
________________________a_b_____ (A: 3, B: 0)
________________________a__b____ (A: 0, B: 6)

Player B wins!

__P_____P___P_P____P__PPPPPP

c:\workspace>
```

```
c:\workspace>python hw5_2.py
____b_A_________________________ (A: 6, B: 4)
____b__a________________________ (A: 1, B: 0)
_______a_b______________________ (A: 0, B: 5)
________Ab______________________ (A: 1, B: 0)
_________ab_____________________ (A: 1, B: 1)
_________ab_____________________ (A: 0, B: 0)
___________BA___________________ (A: 4, B: 2)
_____________A_B________________ (A: 2, B: 5)
___________________X____________ (A: 6, B: 4)
_____________________B_a________ (A: 4, B: 2)
_______________________x________ (A: 0, B: 2)
_______________________b___A____ (A: 4, B: 0)

Player A wins!

PP__PP_P_PPP__P_P__P_____P____

c:\workspace>
```

```
c:\workspace>python hw5_2.py
___A__B_________________________ (A: 3, B: 6)
_______B_A______________________ (A: 6, B: 1)
_________b___A__________________ (A: 5, B: 3)
_________b_____A________________ (A: 3, B: 0)
__________B_________A___________ (A: 3, B: 1)
_____________B_____A____________ (A: 1, B: 5)
_____________________B____A_____ (A: 6, B: 6)
_______________________X_________ (A: 1, B: 6)
_________________________x_______ (A: 6, B: 3)

Both players win!

PP___P____P__P____P______P___P

c:\workspace>
```

```
c:\workspace>python hw5_2.py
_____X__________________________ (A: 6, B: 6)
______x_________________________ (A: 1, B: 1)
______x_________________________ (A: 0, B: 0)
_________BA_____________________ (A: 4, B: 3)
__________Ba____________________ (A: 1, B: 1)
___________x____________________ (A: 0, B: 1)
___________b__A_________________ (A: 3, B: 0)
_____________b_____A____________ (A: 6, B: 2)
_____________b_____________a____ (A: 6, B: 0)
_____________________b_____a____ (A: 0, B: 3)
_____________________b__________a (A: 2, B: 0)

Player A wins!

_PP_PP_PPP__PPP_PP__P_PP_PPP_P

c:\workspace>
```

```
c:\workspace>python hw5_2.py
__X_____________________________ (A: 2, B: 2)
_____aB_________________________ (A: 4, B: 5)
_____a___b______________________ (A: 0, B: 4)
__________x_____________________ (A: 5, B: 0)
__________a__B__________________ (A: 0, B: 4)
____________a___B_______________ (A: 2, B: 3)
____________a_____B_____________ (A: 0, B: 2)
_____________a_______B__________ (A: 1, B: 2)
_____________a_____________B_____ (A: 0, B: 6)
_____________________a__________b (A: 3, B: 6)

Player B wins!

P__PPPP____PPPP__P__________P

c:\workspace>python hw5_2.py
```

Problem 3: Movie Data Analysis [hw5_3.py]

Write a program for answering the following questions based on IMDB Movie data (**IMDB-Movie-Data.csv**). You are required to write a function for each of the following question. You may want to use Python's `String`, `Loops`, `File`, `List`, `Functions`, and `Dictionary` to solve the questions. Note that there is not specific output format in this problem. You are allowed to simply print the answers using any format. Nevertheless, you must indicate top 1, top 2, …, etc. when printing the results. Besides, to ensure you have correct answers, you can check the answers with other teams. Be careful not to lend or borrow code to/from other teams to avoid plagiarism.

	Questions
(1)	Top-3 movies with the highest ratings in **2016**?
(2)	The director who **involves in the most movies**?
(3)	The actor generating the **highest total revenue**?
(4)	The average rating of **Emma Watson**'s movies?
(5)	Top-4 actors **playing the most movies**?
(6)	Top-7 frequent **collaboration pairs of director & actor**?
(7)	Top-3 directors who **collaborate with the most actors**?
(8)	Top-6 actors playing in the most **genres of movies**?
(9)	Top-3 actors whose movies lead to the **largest** *maximum gap of years*? *Example of "maximum gap of years":* Tom Cruise has movies: "Edge of Tomorrow" in 2014, "Mission: Impossible - Rogue Nation" in 2015, "Oblivion" in 2013, "Jack Reacher" in 2012, "Mission: Impossible III" in 2006, "Jack Reacher: Never Go Back" in 2016, "Rock of Ages" in 2012, "Mission: Impossible - Ghost Protocol" in 2011. **The maximum gap of years is 2016-2006 = 10**

Problem 4: Arithmetic Expression Evaluator [hw5_4.py]

Write a Python program that evaluates arithmetic expressions containing integers, addition, subtraction, multiplication, and division operators, as well as parentheses. The program should handle different types of errors and print appropriate error messages. The program should continue running until the user inputs 'q' to quit.

Requirements:

(1) The program should allow the user to input arithmetic expressions containing integers, addition (+), subtraction (-), multiplication (*), division (/), and parentheses.

(2) The program should evaluate the expression and print the result.

(3) The program should handle and print error messages for the following errors or exceptions:

- Division by zero error: Division by zero (ZeroDivisionError) in the expression.
- Operand error: Missing an operand before or after an operator.
- Unbalanced parentheses error: Number of opening and closing parentheses is not equal.
- Unsupported character error: The expression contains characters other than integers, addition, subtraction, multiplication, division, and parentheses.

(4) The program should continue running and allow the user to input expressions until the user inputs 'q' to quit.

Sample input & output

```
c:\workspace>python hw5_4.py
Enter an expression to evaluate or 'q' to quit: 10+10
Result: 20
Enter an expression to evaluate or 'q' to quit: 5*(2+8)/4
Result: 12.5
Enter an expression to evaluate or 'q' to quit: (10-3)*(6+2)/4
Result: 14.0
Enter an expression to evaluate or 'q' to quit: ((5+3)/4)*5
Result: 10.0
Enter an expression to evaluate or 'q' to quit: 100
Result: 100
Enter an expression to evaluate or 'q' to quit: 100/(20-20)
Error: Division by zero
Enter an expression to evaluate or 'q' to quit: 10+
Error: Operand error
Enter an expression to evaluate or 'q' to quit: /20
Error: Operand error
Enter an expression to evaluate or 'q' to quit: ((10+5
Error: Unbalanced parentheses
Enter an expression to evaluate or 'q' to quit: ((((10)*3)-1)+2)
Result: 31
Enter an expression to evaluate or 'q' to quit: ((((10)*3)-1)+2))
Error: Unbalanced parentheses
Enter an expression to evaluate or 'q' to quit: 10^2
Error: Unsupported character ^
Enter an expression to evaluate or 'q' to quit: (10+5-2)%3
Error: Unsupported character %
Enter an expression to evaluate or 'q' to quit: 3/(((10*10)/2)-50)
Error: Division by zero
Enter an expression to evaluate or 'q' to quit: q

c:\workspace>
```

Need to write proper comments for each problem!

Note

This is a homework for each **individual**. 必須於程式檔內註解註明系及姓名學號。

How to Submit Your Homework?

Submission in NCKU Moodle

Before submitting your homework, please zip the files in a zip file, and name the file as "學號_hw4.zip". For example, if your 學號 is H12345678, then your file name is:

"H12345678_hw**5**.zip" or "H12345678_hw**5**.rar"

When you zip your files, please follow the instructions provided by TA's slides to submit your file using NCKU Moodle platform http://moodle.ncku.edu.tw .

Have Questions about This Homework?

Please feel free to visit TAs, and ask/discuss any questions in their office hours. We will be more than happy to help you.