MA1008 Introduction to Computational Thinking
Mini Project: A Badminton Ladder
Semester 2, AY 2020/2021, Week 10 – Week 13

1. Introduction

The objective of the mini project is for you to produce a program of a moderate size that will require you to utilised what you have learned in this course, and more, to do something interesting. Through this, you should learn to design, manage and execute a sizable program.

2. The Project

This is a data management project with graphical display. You are to write a program for managing a badminton ladder, which is a form of badminton competition where the players are placed in an ordered list and challenge players higher in the order. Here are the rules in operating the ladder:

i. A player on the ladder may challenge another player up to three places above. A challenge to players beyond three places above is forbidden.
ii. If the challenger wins, then s/he moves into the position of the challenged. The challenged and the players in between each moves down the ladder by one place. Other players are not affected.
iii. If the challenger loses, the ladder remains unchanged.
iv. A new player joins the ladder at the bottom.
v. When a player leaves, everyone below moves up by one place to occupy the vacated spot.

Initially, the ladder is populated randomly. After numerous challenges, the ladder would assume an order of strength. The picture on the right shows a physical rack for manual recording of a badminton ladder. This project aims to provide an online replacement, which also offers a lot more facilities.

The Program

Your program manages the ladder and displays useful information. This means your program must

i. Provide means for a player to take one of these actions:
    a. Issue a challenge (stating opponent and play-by date)
    b. Record the result of a challenge
    c. Join the ladder
    d. Withdraw from the ladder
    e. Make a query
ii. Provide means of recording and displaying the result of a challenge.
iii. Adjust the ladder according to the result.
iv. Register a new player and place him/her on the ladder.
v. Accept the withdrawal of a player from the ladder and update the ladder accordingly.

Your program must produce a graphical display with these elements (imagine this is a web page where players go for their information and conduct ladder business):

i. The ladder, with the first player on top and the last player at the bottom.
ii. Facilities for a player to take an action as stated above.
iii. A display showing the yet-to-play challenges.
iv. A means for making queries and displaying the answers. Your program should include these queries:

a. The order of the ladder on a specific date
b. The data of a specific challenge based on the names of the players
c. The data of a specific challenge based on the date
d. The list of matches a player has played, with all the associated data
e. The most active player, i.e. the player with the most matches
f. The least active player, i.e. the player with the least matches
g. The list of matches played in a specific date range
h. You may add more queries as you see fit

Furthermore,

v. The data of the ladder are stored in the two data files specified below, and they must be updated as events take place.
vi. When the program runs, the display should show the current ladder order as well as the results of the latest n challenges (you may choose n, n >= 2).

Within reason, you may choose your own design of this display and what to include in it.

The Data Files

The data of the ladder are in two files which are to be updated every time an action is taken. One file, ladder.txt, contains the current order of the ladder, which is an ordered list of the names of the players. The second file, data.txt, contains data on every challenge and the coming and going of players. The data are in chronological order, with the latest data inserted at the end of the file. Specifically, each line of this second file contains one of these three items:

i. The data on a challenge. The syntax of the line is:

```
Name of challenger p1 / Name of challenged p2/ date / scores
```

The "/" separates the different fields. The names are strings. p1 and p2 after the names are integers showing the current positions of the players on the ladder. The date format is DD-MM-YYYY, and the scores format is xx-yy, where xx and yy represent the numerical scores of the challenger and the challenged respectively. A match may have two or three of such scores.
Example: `CW Lee 5/M Frost 3/20-07-2000/20-22 18-21`

If a score is missing from such a line, it means that the challenge has been issued but the match has not been played, and the date in the line is the play-by date. If this date is in the past, it means that the challenged has not taken up the challenge and therefore forfeits the match; the challenger is automatically the winner.

(Note: Badminton matches are played to the best of three games. The player reaching 21 points first in a game wins the game. If a game reaches 20-20, then the player leading by two points subsequently wins the game.)

ii. The name of a new player who joins and the date of joining, with a + sign before the name.
Example: `+NEW Player/18-05-2020`

iii. The name of a player who leaves and his/her position on the ladder, and the date of leaving, with a – sign before the name.
Example: `-EX Player 12/18-05-2020`

The existing data in ladder.txt contains the current order of the list of players in there. The data in data.txt contains the chronological record of the challenges, new players joining and existing players leaving to date. As your program runs, the data in these files will change.

You may choose to use your own data format, in which case you must specify this format clearly.

3. Resources

The main external resource that you will need for this project is for the graphical display, for which you need the graphics library provided with Python, called **Turtle**. Turtle capabilities and functions are described fully in Section 24.1 of the Python documentation, which you can access via the IDLE interactive shell by clicking Help > Python Docs and then search for "Turtle". Turtle provides the functions required for drawing, such as the creation of shapes and posting of text, and controlling their attributes such as location, colour, and more. You should study the different Turtle functions that are required for your graphics. There are many Turtle functions, you only need to learn the ones you require. It is therefore necessary that you work out what you need beforehand.

4. What you need to submit

1. You need to submit **a working Python program** that manages the dynamics of a badminton ladder and displays the status appropriately, as stated above.
2. The two data files generated at the end of a sequence of activities, which should contain more data than the file first handed out.
3. You also need to submit **a report in a Word file**, providing
   i. a guide on how to run your program
   ii. the data of a sequence of runs of the program, including the challenge at each run and the insertion and removal of players, as they come and go
   iii. pictures of your graphics window, with display of the different information
   iv. highlights of the **key strengths and limitations** of the program.

The **deadline** for project submission is **the end of the day, Friday 16 April**.

5. Grading Rubrics

Here are what the graders will be looking for when grading:

1. The quality of the graphics display.
2. The ease in interacting with the program, which includes issuing and recording the results of a challenge, registering or removing a player, and making queries on the data.
3. The quality of your program, which includes:
   - How easy it is to read and understand your program. This means your program should be adequately commented with good choice of variable and function names.
   - Modularisation of the program including appropriate use of functions.
   - The design of the functions which include the appropriateness of parameters in the functions.

6. Epilogue

You should start working on the program immediately. Do not wait till the last week. What you produce depends on the time you spend on the project and your ability in creating good algorithms and writing good code.

This is an individual project. You may discuss and consult with your classmates, but everyone must write their own program. Any program deemed to have been copies of each other will be penalised heavily, regardless of who is doing the copying.