CSCI-201: Principles of Software Development

Spring 2022

Lab11: Hogwarts Dueling Club Tournament

Introduction:

Gilderoy Lockhart at Hogwarts School of Witchcraft and Wizardry is looking to organize a Dueling club tournament which can be completed in as less amount of time as possible. Since Lockhart specializes only in memory spells, he wants your help to run the tournament.

The dueling club currently has 8 wizards enrolled. Additional wizards can be enrolled later. A wizard can only cast one of the three spells – Expelliarmus, Stupefy or Protego. Following are the conditions for winning the match:

1. If Expelliarmus is cast against Stupefy, Expelliarmus wins
2. If Protego is cast against Expelliarmus, Protego wins
3. If Stupefy is cast against Protego, Stupefy wins.

In case of a tie, that is if both wizards cast the same spell, then the wizard with the higher rank wins. (Note, Rank 1 is higher than Rank 2).

Initially, 8 wizards will compete in 4 matches (2 wizards per match). Wizards are chosen to compete against each other sequentially i.e. Wizards at position 0 and position 1 will compete, wizards at position 2 and 3 will compete and so on. Winners from these matches will again compete.

```
Wizard 1 --------
                 | Winner 1 ----------------
Wizard 2 --------                           |
                                            | ---------Winner 1 -----------------
Wizard 3--------                            |                                    |
                 | Winner 2 ----------------                                     |
Wizard 4 --------                                                                |
                                                                                 |------------- Champion
Wizard 5 --------                                                                |
                 | Winner 3 -----------------                                    |
Wizard 6 --------                           |                                    |
                                            |---------Winner 2 ------------------
Wizard 7 --------                           |
                 | Winner 4------------------
Wizard 8 --------

LEVEL 1                  LEVEL 2               LEVEL 3
```

If you closely look at the figure above, matches at a particular level can take place at the same time in parallel. Once matches at a particular level are completed, you can start execution of matches on the next level. At the final level, only one match takes place and the champion is declared. In this lab you will learn:

- Parallel Processing
- ForkJoinPool
- RecursiveTask

Implementation:

Download the provided Java files (*Wizard.java, TournamentMatch.java, Lab11.java)* and drag them into your newly created Eclipse project.

Wizard.java is already implemented. You do not need to modify this file.

TournamentMatch.java extends RecursiveTask<T>. You are to implement the compute() method that performs the fork and join for processes. You are to also implement the playDuel() method in this class. Closely look at the comments given for better understanding.

Lab11.java is the main driver. In the main() method you create a ForkJoinPool() and invoke processes to play the matches. Shut down the pool once the champion is found.

A Magic Charm (Hint!): Winners of the duels are added to a list which is accessible in the main() method and a new pool is created from main to run the next set of duels.

Testing:

1. Comment line "Collections.shuffle()" in the main method and run the program. Below is one possible output (just a part) of your program. The final champion will be the same.

```
====Playing Level 1 duels====

Draco Malfoy: Stupefy vs Hermione Granger: Protego
Ronald Weasley: Stupefy vs Neville Longbottom: Protego
Winner is: Ronald Weasley

Winner is: Draco Malfoy

Luna Lovegood: Expelliarmus vs Cedric Diggory: Stupefy
Harry Potter: Expelliarmus vs Ginny Weasley: Stupefy
Winner is: Luna Lovegood

Winner is: Harry Potter

====Playing Level 2 duels====
Draco Malfoy: Stupefy vs Ronald Weasley: Stupefy
Winner is: Ronald Weasley

Harry Potter: Expelliarmus vs Luna Lovegood: Expelliarmus
Winner is: Harry Potter

====Playing Level 3 duels====

Ronald Weasley: Stupefy vs Harry Potter: Expelliarmus
Winner is: Harry Potter

THE CHAMPION is Harry Potter
```

2) Uncomment the “Collections.shuffle()” line in the main method. One possible output:

```
====Playing Level 1 duels====

Luna Lovegood: Expelliarmus vs Hermione Granger: Protego
Winner is: Hermione Granger

Ginny Weasley: Stupefy vs Ronald Weasley: Stupefy
Neville Longbottom: Protego vs Harry Potter: Expelliarmus
Cedric Diggory: Stupefy vs Draco Malfoy: Stupefy
Winner is: Neville Longbottom

Winner is: Ronald Weasley

Winner is: Draco Malfoy

====Playing Level 2 duels====

Hermione Granger: Protego vs Ronald Weasley: Stupefy
Draco Malfoy: Stupefy vs Neville Longbottom: Protego
Winner is: Ronald Weasley

Winner is: Draco Malfoy

====Playing Level 3 duels====

Ronald Weasley: Stupefy vs Draco Malfoy: Stupefy
Winner is: Ronald Weasley

THE CHAMPION is Ronald Weasley
```

Important Note: The output will vary on every run of the program. In the first test, the final champion will always be Harry Potter. In the second case, the final champion is not fixed. The print statements for matches played on a level might be garbled.

Grading Criteria:

Labs are graded based on your understanding of the course material. To receive full credit, you will need to:

1) complete the lab following the instructions above
2) show your understanding of the lab material by answering questions upon check-off

If there is a discrepancy between your understanding of the material and your implementation (i.e. if your code is someone else's work), you will receive a grade of 0 for the lab.

Implementation: **(8 Points)**

1) *Lab11.java*

a) 2 - the main() method is complete and working
b) 1 - the student is on the right track, but the implementation is incomplete
c) 0 - the student implements less than 50%

2) *Tournament.java*. In this class you grade two methods: compute() and playDuel().

a) 3 – for each method if it is complete and working
b) 1.5 - the student is on the right track, but the implementation is incomplete
c) 0 - the student implements less than 50%

Not a part of the grading criteria
If you had fun implementing the lab, why not try with 16, 32 or 256 or 4096 wizards?! Tweak your code and see the magic happen!
****

Check-off Questions: **(2 Points)**

Please randomly select one question.

Question 1
What will happen if use ForkJoinPool newPool = new ForkJoinPool(2) ?

Question 2
Is parallel execution always faster than sequential?

Question 3
What is the difference between RecursiveAction and RecursiveTask?

Question 4
How can you prevent getting garbled print statements between different processes?

Question 5
What is the main difference between the Executor framework and ForkJoinPool?