Lab Exercise Four – CIS*2430 F24

For this lab exercise, we will continue adding to our DiscussionBoard program with hashmaps and exceptions.

- As our program grows in popularity, we may find that linearly searching for posts by a specific username to be too slow. As such we will implement a HashMap for storing the index of each user's posts
    - The keys for the HashMap should be the username of the User
    - The values should be an ArrayList of Integers containing the indices of the posts the User has made
- When displaying all posts we should still loop through all values in all ArrayLists
- When displaying all posts by user we should only have to loop through the indices in the corresponding HashMap entry
- We can no longer guarantee that the user will input the correct type of data (e.g., entering a float when prompted for an integer). As a result, we will need to employ defensive mechanisms to ensure our program does not crash.
    - One good pattern is to create a function that will continue to prompt the user until the correct data type is input. We can use try-catch blocks to facilitate the flow of our function and only return when no exception occurs. You can then reuse the function whenever you need to collect that type of input!
- We should also make sure that User and Post objects are created with meaningful data for our program (e.g., Certain attributes of a post such as "title" shouldn't be blank, and certain attributes of a user such as "username" can't be blank either)
    - You **MUST** throw exceptions from any User or Post constructor/mutator if the operation would cause an invalid object. The exceptions must be checked in the calling methods. We should **NOT** be checking if the values the user entered are legal outside of the User or Post class
    - Also remember that the username can be left blank in favour of the program generating a default username, so this is not an illegal input
    - If the constructor or mutator generates an error, we should either prompt our user for a new set of input or return to the main menu loop

* This lab will only be tested with TextPosts. If you have not implemented PollPosts in lab 3 you do not need to implement them in lab 4 to achieve full marks!

* A word is defined as a set of alpha-numeric characters surrounded by spaces

* All posts will be a single line of words, all fullnames will be a single line of words, and all usernames will be single words

* Keywords are matched by ignoring cases but word-by-word (e.g., "program" and "programming" are not a match), implying that string tokenization is needed

Sample Menu

(1) Create new user
(2) Create new post
(3) View all posts
(4) View all posts with a given username
(5) End Program

Sample Text Post Printout

Post #<post_id>

Created By: <user_full_name> (@<user_name>)

Title: <post_title>

<post_content>

Sample Poll Post Printout (if you are building your implementation from Lab 3 Exercise):

Post #<post_id>

Created By: <user_full_name> (@<user_name>)

Title: <post_title>

<option_1_text>: <option_1_count>

<option_2_text>: <option_2_count>

<option_3_text>: <option_3_count>

Marking Rubric

10 marks total

- **3 marks** for HashMap implementation
- **2 marks** for User exceptions
- **2 marks** for Post exceptions
- **3 marks** for defensive programming

Submission Requirements

Please submit all related java files + runnable jar as a zip file on CourseLink. Please place your main method in DiscussionBoard.java with your name, student id, and compile and run commands as comments at the top. If you are building Lab 4 implementation on top of Lab 3 exercise, please also submit a save file named “cis2430.dboard” in a “boards” folder inside the zip file.

Due Dates

Lab Section	Due Date
0101 & 0201	November 8th, 2024 @ 11:59pm
0102 & 0202	November 7th, 2024 @ 11:59pm
0103 & 0203	November 13th, 2024 @ 11:59pm
0104 & 0204	November 9th, 2024 @ 11:59pm
0105 & 0205	November 12th, 2024 @ 11:59pm
0106 & 0206	November 13th, 2024 @ 11:59pm
0107 & 0207	November 12th, 2024 @ 11:59pm