MIDTERM ASSIGNMENT

Subject : Classes, objects, encapsulation, inheritance, and polymorphism.
Instructor: Dr. Selim Yılmaz (selimyilmaz@mu.edu.tr) & Dr. Özgür Kılıç (ozgurkilic@mu.edu.tr)
Out Date: 05/03/2021 23:59:59
Due Date: 05/17/2021 23:59:59

DECLARATION OF HONOR CODE[1]

Student ID 180709005
Name Mehmet Kadri
Surname GOFRALILAR

In the course of Introduction to Object Oriented Programming (CENG 1004), I take academic integrity very seriously and ask you to do as well. That's why, this page is dedicated to some clear statements that defines the policies of this assignment, and hence, will be in force. Before reading this assignment booklet, please first read the following rules to avoid any possible violation on academic integrity.

- This assignment must be done individually unless stated otherwise.
- You are encouraged to discuss with your classmates about the given assignments, but these discussions should be carried out in an abstract way. That is, you cannot copy code (in whole or in part) of someone else, cannot share your code (in whole or in part) with someone else either.
- The previous rule also holds for the material found on the web as everything on the web has been written by someone else.
- You must not look at solution sets or program code from other years.
- You cannot share or leave your code (in whole or in part) in publicly accessible areas.
- You have to be prepared to explain the idea behind the solution of this assignment you submit.
- Finally, you must make a copy of your solution of this assignment and keep it until the end of this semester.

I have carefully read every of the statements regarding this assignment and also the related part of the official disciplinary regulations of Muğla Sıtkı Koçman University and the Council of Higher Education. By signing this document, I hereby declare that I shall abide by the rules of this assignment to prevent any violation on academic integrity.

Signature .....

[1]This page should be filled and signed by your handwriting. Make it a cover page of your report.

A) Foundation

First of all, I've created a class that will include our main method. Since this task was based on reading text files, I was supposed to read the lines in text files that are given as parameters. After reading "users.txt", I've used the data inside to create new users by calling User object, which was also created by me.

Inside the User class, I was supposed to assign every user object's attributes one by one. For that, I had to declare attributes as private without initializing them. Right after that, I was supposed to write a constructor. This constructor will be called every time a new user object is being created and assign it's attributes to that object specifically. After that, I wrote setters and getters for these attributes since all of them are private and I need setter/getter methods to reach the needed information if necessary. I will be explaining the rest of User class in Section C, Subsection I.

Since I've created all users that were given to me, all I had to do was to read "commands.txt" and implement every possible command and scenario in my code. I didn't declare or initialize the variables before writing while loop. They were needed in the loop in more than one case, so I had to declare them right outside of the loop to be safe as I was writing inner parts.

I needed a loop since there were probably more than one lines that was supposed to be read and executed as a command. Then, I've used switch/case conditional to act according to which command is given in the beginning of the line. I wrote corresponding codes for possible scenarios in the cases, which I will be explaining in Section B.

B) MySocialBook

I've mostly explained what happens in this class except the inside of the while loop. Firstly, I've splitted the line and assigned it's first word as the command. Then, I've printed the command since it was demanded from me. Finally, I've used switch/case conditional to execute according to commands.

I've marked the commands with "(*check)" that has to be executed only if a user is signed in, which are checking if that's the situation before starting to try and execute the command. If that's not the case and no user is logged in, it prints out "Error: Please sign in and try again.".

I. Adding New User

Date was given to as in String, but we had to keep it in a date type. I've written a method named "stringToDate" for that and changed the date's type. Then, I created new users by calling User class with currently processed line's data as parameters.

II. Removing User

I had to check if there is any user in userList arraylist, so that there wouldn't be any errors. In the case of there is at least one user in userList and his/her userID matches with the given id, user is removed from userList.

III. Sign In

I've checked whether if there's a user already logged in or not, and if not, looked for any username and password matches with any user in the userList. (Used a boolean to keep track of status)

IV. Showing the Users List (*check)

I've used a for loop to iterate through the userList and printed every User in the list.

V. Updating Profile (*check)

I've changed the active user's name, date of birth and school that user went via setters.

VI. Changing Password (*check)

I've changed the actively signed in user's password using setter.

VII. Adding People to Friends List (*check)

I've created an arraylist to be able to keep track of friends, which is called friends. I've checked if the user is already friends with this person or not. If not, this person is added to friends. Of course there is the possibility of there is no such user.

VIII. Removing People from Friends List (*check)

I've checked if the user is already friends with this person or not. If so, this person is removed from friends. Of course there is the possibility of there is no such user.

IX. Showing the Friends List (*check)

I've checked if the user has any friends or not. If he/she has, these friends are printed with their attributes. Of course there is the possibility of there is no friends yet.

X. Adding Text Post to Your Profile (*check)

I've created an arraylist to be able to keep track of posts in Post class. I've checked if the person user wants to tag is user's friend or not. If so, a text post is added with the other user tagged with location and a text.

XI. Adding Image Post to Your Profile (*check)

I've checked if the person user wants to tag is user's friend or not. If so, an image post is added with the other user tagged with location, a text and a image name with it's resolution.

XII. Adding Video Post to Your Profile (*check)

I've checked if the person user wants to tag is user's friend or not. If so, also if the video duration is smaller than 10 mins, a video post is added with the other user tagged with location, a text and a videp name with it's duration.

XIII. Removing Last Post from Your Profile (*check)

I've checked if the user has any posts or not. If there are any, the last post is deleted from the arraylist named posts.

XIV. Showing a User's Posts

I've printed every post the given user has in his/her profile.

XV. Blocking a User (*check)

I've checked if the given user is in active user's blocked list or not, which is also an arraylist. If not, then I've checked if the user exists or not. If such a user exists, he/she is added to blocked list of our active user.

XVI. Showing Blocked Friends (*check)

I've checked if the active user has any user in his/her blocked list or not. If there are any users in blocked list (who also needs to be in active user's friends list as well), I've printed them.

XVII. Showing Blocked Users (*check)

I've checked if the active user has any user in his/her blocked list or not. If there are any users in blocked list, I've printed all of them.

XVIII. Unblocking Users (*check)

First, I've checked if the given user exists or not. If the user exists and is in blocked list of active user, he/she is removed from blocked list regardless of that person being active user's friend or not.

XIX. Signing Out (*check)

All I did was to change the status to false (meaning signed out) and release the active user since the user has signed out.

C) Classes and Methods

I. Users

I've declared attributes of every user object and assign them the parameters every time user class is called.

II. Posts

I've declared attributes of every post object has in common and assign them the parameters every time post class is called.

-TextPost:

This class extends from Post Class and has the same attributes in Post class.

-ImagePost:

This class extends from Post Class and has every attribute in Post class and even more, such as image name and resolution.

-VideoPost:

This class extends from Post Class and has every attribute in Post class and even more, such as video name and duration.

III. Location

This class has only 2 attributes, latitude and longitude.

D) UML Diagram

And here's the UML Diagram that shows the hierarchy between objects.(I'm guessing it's not readable much, so I'll be also sending it as a .png file)