Foundations of Computer Science
Midterm Project

Start: July 28th, 2023 **Time:** 10:00 AM
End: August 1st, 2023 **Time:** 11:59 PM

Instructions:

This exam consists of one project worth a total of 100 points. Please adhere to the following directions:

- You are allowed to use all and any online resources during the test, but if you happen to find any methods/functions online, **you must report your findings**.
- If you utilize any code-snippets from online sources, **add the reference URL** as comments in the code.
- Plagiarism and cheating are serious offenses and will result in a lifetime ban from SE Factory.
- Sharing code will lead to penalties for all parties involved. Proving that you were the original author of the code will not be considered a defense.
- The use of AI tools, such as **ChatGPT**, is strictly prohibited.
- Seeking external help for the project is strictly forbidden.
- The project should be implemented in **Python**.
- Ensure that you use **Git** and push the project to **GitHub** during implementation.
- Once you finish the implementation, copy your code on **Replit** (http://www.replit.com) and submit the project's link to your respective instructor (**Frederick**: fred@sefactory.io or **Georgio**: georgio@sefactory.io) via email no later than **August 1st, 11:59 PM.**
- The subject of your email submission should be "**FirstName Last Name <> Midterm Solution.**"
- The time of your submission will be based on the time we receive your email.
- **Commits are not allowed after the deadline**. Instructors will inspect the time of the last commit.
- Failure to comply with these rules will result in your immediate termination from the program.

<Best of Luck/>

Corrupted Ticketing System (100 points)

You are tasked with writing a program that simulates a corrupted ticketing system. The project serves two main purposes: to provide practice in writing programs that manipulate strings, data structures, and files, and to apply sorting and searching algorithms. The simulated system allows admins to change the priority of user admissions based on ticket priority.

The program will first display a login form with two user types: **Admin** and **Users**. If "admin" is entered as the username and "admin123123" as the password, the system will show the admin menu. If a normal user logs in without a password (empty password), a different menu will be shown. For an incorrect admin login, the system should display "Incorrect Username and/or Password" and restrict the admin to a maximum of 5 attempts.

The System

Your job is to write a script that handles the user interaction component of the system. To solve the problem, your program must be able to:

- Read one text file and upload all the tickets into the special queue of the system
- Implement the basic control structure and manage the details

The text file includes several tickets in the following format:

tick101, ev003, fred, 20230802, 0
tick102, ev002, gio, 20230803, 2
….

This file represents the list of tickets which were issued. Each line contains 5 words: ticket id, event id, username, timestamp (YYYYMMDD), and the last word represents the priority (integer). This file should be imported when the program starts.

The Flow of The System:

A. Import tickets from the text file into the Special Queue without user intervention.
B. Greet the user and ask for their username and password (if the password was left empty, then it’s a normal user)

For **admin** users, display the following menu options:

1. Display Statistics
2. Book a Ticket
3. Display all Tickets
4. Change Ticket's Priority
5. Disable Ticket
6. Run Events
7. Exit

- **If the admin chooses (1)**, show the event ID with the highest number of tickets.
- **If the admin chooses (2)**, allow the admin to book a new ticket for an event by specifying the username, event ID, and priority (the ticket ID auto-incremented, date automatically taken from the computer).
- **If the admin chooses (3)**, show all tickets registered in the system, ordered by date and event ID (Today, Tomorrow, etc.). Old tickets should not be shown.
- **If the admin chooses (4)**, allow the admin to change the priority of a ticket by specifying the ticket ID and priority. If the ticket is not found, an appropriate message will be shown.
- **If the admin chooses (5)**, allow the admin to remove a ticket from the system by providing the ticket ID. If the ticket is not found, an appropriate message will be shown.
- **If the admin chooses (6)**, display today's events found in the queue, sorted by priority, and remove them from the queue.
- **If the user chooses (7)**, exit the program without saving.

For **normal** users, display the following menu options:

1. Book a ticket
2. Exit

- **If the user chooses (1),** add a new ticket to the system by specifying the event ID, using the username from the login system, the current date of the system, and setting the default priority to 0 (ticket ID automatically incremented).
- **If the user chooses (2),** save any newly added tickets and terminate the program.

After each option, the menu should be displayed again, allowing the user to use the system multiple times. It's essential to use good programming practices while developing this project.

Best of luck with your implementation! ☺