Foundations of Computer Science
Midterm Project

Start: November 9th, 2023 **Time:** 8:00 PM
End: November 12th, 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 links (Replit & GitHub) to your respective instructor (**Fred**: frederick@sefactory.io, **Georgio**: georgio@sefactory.io) via email no later than **November 12th, 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.
- You should perform several commits during the implementation of the project. **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/>

Advanced Browser Tabs Simulation (100 points)

You are tasked with developing an advanced browser tabs simulation in Python for a cutting-edge browser. The goal is to create a feature-rich system that allows users to perform various operations on tabs using a menu-based interface. The implementation should include features such as opening, closing, and switching between tabs, handling nested tabs, and additional functionalities.

The program starts by greeting the user and displaying the following menu:

1. **Open Tab**
2. **Close Tab**
3. **Switch Tab**
4. **Display All Tabs**
5. **Open Nested Tab**
6. **Clear All Tabs**
7. **Save Tabs**
8. **Import Tabs**
9. **Exit**

- If the admin chooses (1), the system should allow the user to add a new tab by asking for the **Title** and the **URL** of the website.
- If the admin chooses (2), the system should permit the user to input the **index** of the tab they wish to close. If no index is provided, the system will close the last opened tab.
- If the admin chooses (3), the system should enable the user to enter the index of the tab for **displaying its content**. If no index is provided, the system will display the content of the last opened tab. Note: 'Displaying' in this context refers to printing the HTML content of the URL associated with the tab. Conduct some research on **web scraping** for more insights.
- If the admin chooses (4), the system should **print the titles** of all open tabs. If there are nested tabs, display them hierarchically.
- If the admin chooses (5), the system should enable users to create nested tabs by specifying the index of the **parent tab** where they want to insert additional tabs. After entering the index, the system should prompt the user to input the titles and contents for the new tabs.
- If the admin chooses (6), the system should allow users to **clear all opened tabs.**
- If the admin chooses (7), the system should prompt the user to provide a file path as a parameter to save the current state of open tabs. Each tab's information, including **title, content, and any nested tabs**, should be written to the file in JSON format. Conduct some research for additional insights on JSON.

- If the admin chooses (8), the system should prompt the user to input a file path as a parameter to **load tabs from the specified file**.
- If the user chooses (9), the program should exit.

Instructions:

- Use dictionaries to represent each tab and a list to maintain the order of open tabs.
- Implement functions
- Test your implementation with various scenarios, including opening, closing, and switching between tabs, as well as using the additional features.
- We know that JSON and File Systems are not explained in class. Your research is part of the exam.

Note:

- Your code should be modular, well-commented, and adhere to good coding practices.
- You are not required to implement a graphical user interface; focus on creating a functional text-based menu system.
- You should have a minimum of 30 commits for your project to be graded

<Good luck/>