University of Science and Technology Houari Boumediene

Programmation Python – Travaux Pratiques

ENG-3 cyber security

Teacher: **MOUHOUN SAID**
Email: s.mouhoun@gmail.com
X (tweeter): https://x.com/saidmouhoun
Linkedin: https://www.linkedin.com/in/said-mouhoun/

Lab Exam: Develop a Multiple-Choice Questionnaire (MCQ) Application for Computer Science Students

Objective

Create a Python application that:

- Allows users to answer MCQs.
- Evaluates their answers and displays a score.
- Provides a user-friendly console interface.
- Manages user data, tracking their history of MCQs and previous scores.

Required Features

1. User Management

- When a user starts the application for the first time, they must enter an identifier (name or ID). The application will check if they already exist (stored in a JSON or CSV file).
  - If the user exists, their MCQ history, including previous scores and test dates, will be displayed.
  - If the user is new, they will create a profile, and their MCQ history will be recorded starting with the first session.
- After each test, the application will save the score and test date in the user's profile.
- The application will display the full MCQ history for each user, including dates and scores.

2. Question and Answer Management

- Questions and options should be stored in a file or dictionary.
- Each question must have one correct answer.
- Users will select an answer for each question. Responses will be validated, and a final score will be calculated.

3. Evaluation and Feedback

- After completing the test, the application will display the user's final score.
- Feedback will be provided for each question, indicating whether the answer was correct or incorrect. If incorrect, the application will display the correct answer.

4. Advanced Modes (Optional for Advanced Teams)

- Implement a timer to limit the response time per question or for the entire test.
- Enable exporting results to a text or CSV file to log user scores.
- Add question categories (e.g., Python, Networking, Algorithms) and allow users to choose a category before starting the MCQ.

Project Steps

Week 1: Design and Basics

Analysis:

- Teams will define the core features of the application.
- Plan the data structure (e.g., how to organize questions, files to store user data and scores).

First Version:

- Load questions from a file (JSON or CSV recommended).
- Implement a user management system to record and track MCQ history and scores.
- Pose questions to the user and record their responses.
- Save and display the user's MCQ history.

Week 2: Advanced Features and Finalization

Improvements:

- Add answer validation and score calculation.
- Implement feedback for each question, indicating if the answer was correct or not.
- Add a timer for individual questions or the entire MCQ session.
- Enable exporting results to a text or CSV file.
- Allow users to select question categories.

Testing and Deployment:

- Test the application with various users to ensure it works correctly.
- Finalize and prepare the project presentation.

Technologies and Concepts Involved

- **Data Structures:** Use lists or dictionaries to organize questions and answers. Store user data and scores in a JSON or CSV file.
- **File Management:** Load and save data from files (JSON or CSV recommended).
- **Loops and Conditionals:** Use loops to present questions, validate answers, and display scores.
- **Functions:** Divide the application into modules for simplicity (e.g., a function to load questions, another to evaluate answers).
- **Input Validation:** Ensure valid user inputs and correct option selection.

Expected Output Example

Console

```
Welcome to the Computer Science MCQ!

Enter your username: Alice

Alice's History:
- Date: 2024-12-15, Score: 18/20
- Date: 2024-11-20, Score: 15/20

Question 1: What is the data type in Python used to represent text?
a) int
b) str
c) list
Answer: b
Correct!

Question 2: What is the average time complexity of searching in a sorted array?
a) O(1)
b) O(log n)
c) O(n)
Answer: b
Correct!

Your final score: 2/2
```

Team Deliverables

1. **Source Code:**
    - Store the code in a GitHub or GitLab repository with a commit history.

    - The team leader will manage the repository, but all members must collaborate using the same repository.
2. **Usage Instructions:**

    - Include a README file explaining how to use the application, with execution examples.
3. **Report:**

    - A 5-page maximum report explaining technical choices, challenges encountered, and solutions.
4. **Feature Demonstration:**

    - Demonstrate the main features of the application, including user management, MCQs, score display, and history tracking.