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/

Examen TP : Développer une Application de QCM pour Étudiants en Informatique

Objectif

Créer une application Python qui :

1. Permet aux utlisateurs de répondre à des QCM.
2. Évalue leurs réponses et affiche un score.
3. Offre une interface conviviale en console.
4. Gère les utilisateurs, permettant de suivre l'historique des QCM et des scores précédents.

Fonctionnalités Requises

1. Gestion des Utilisateurs

- Lorsqu'un utilisateur démarre l'application pour la première fois, il doit entrer un identifiant (nom ou ID). L'application vérifiera s'il existe déjà (stockée dans un fichier JSON ou CSV).
- Si l'utilisateur existe déjà, l'application affichera son historique de QCM, incluant les scores précédents et les dates des tests.
- Si l'utilisateur est nouveau, l'application lui permettra de créer un profil et commencera un nouvel enregistrement de QCM.
- Après chaque test, l'application enregistrera le score et la date du QCM dans le profil de l'utilisateur.
- L'application affichera également l'historique complet des QCM passés pour chaque utilisateur, avec la date et le score.

2. Gestion des Questions et Réponses

- Les questions et leurs options doivent être stockées dans un fichier ou un dictionnaire.
- Chaque question aura une seule bonne réponse.
- Les utilisateurs choisiront une réponse pour chaque question. Les réponses seront vérifiées et un score final sera calculé.

3. Évaluation et Feedback

- L'application affichera le score final de l'utilisateur après le test.
- Un feedback sera donné pour chaque question, avec un message indiquant si la réponse était correcte ou incorrecte. En cas de mauvaise réponse, l'application affichera la bonne réponse.

4. Modes Avancés (optionnels pour équipes avancées)

- Implémenter un **chronomètre** qui limite le temps de réponse par question ou pour tout le test.
- Permettre l'**exportation des résultats** dans un fichier texte ou CSV pour enregistrer les scores des utilisateurs.
- Ajouter des **catégories de questions** (par exemple : Python, Réseaux, Algorithmes), et permettre à l'utilisateur de choisir une catégorie avant de commencer le QCM.

Étapes du Projet

Semaine 1 : Conception et Bases

1. **Analyse :**
    - Les équipes doivent définir les fonctionnalités principales de l'application.
    - Elles devront planifier la structure des données (comment organiser les questions, les fichiers nécessaires pour stocker les utilisateurs et leurs scores).
2. **Première Version :**
    - Charger les questions depuis un fichier (JSON ou CSV recommandé).
    - Implémenter un système de gestion des utilisateurs permettant d'enregistrer et de suivre l'historique des QCM et des scores.
    - Poser les questions à l'utilisateur et enregistrer les réponses.
    - Enregistrer et afficher l'historique des QCM pour un utilisateur.

Semaine 2 : Fonctionnalités Avancées et Finalisation

1. **Améliorations :**
    - Ajouter la **vérification des réponses** et le calcul du score.
    - Implémenter des **commentaires de feedback** pour chaque question, indiquant si la réponse était correcte ou non.
    - Ajouter la fonctionnalité de chronomètre par question ou pour l'ensemble du QCM.
    - Implémenter l'exportation des résultats dans un fichier texte ou CSV.
    - Ajouter la possibilité de sélectionner des **catégories de questions**.
2. **Tests et Déploiement :**
    - Tester l'application sur différents cas pour s'assurer qu'elle fonctionne correctement avec des utilisateurs différents.
    - Finaliser et préparer la présentation du projet.

Technologies et Concepts Mobilisés

- **Structures de données :** Utiliser des listes ou des dictionnaires pour organiser les questions et réponses. Les utilisateurs et leurs scores doivent être stockés dans un fichier JSON ou CSV.

- **Gestion des fichiers :** Charger et sauvegarder des données depuis un fichier (JSON ou CSV recommandé).

- **Boucles et conditionnelles :** Utiliser des boucles pour poser les questions, vérifier les réponses et afficher les scores.

- **Fonctions :** Diviser l'application en modules pour simplifier le code (par exemple, une fonction pour charger les questions, une autre pour évaluer les réponses).

- **Validation des entrées :** Vérifier que l'utilisateur entre des réponses valides et qu'il sélectionne correctement ses options.

Exemple de Résultat Attendu

Console

Bienvenue au QCM Informatique !

Entrez votre nom d'utilisateur : Alice

Historique de Alice :
- Date: 2024-12-15, Score: 18/20
- Date: 2024-11-20, Score: 15/20

Question 1 : Quel est le type de données en Python pour représenter du texte ?
a) int
b) str
c) list
Réponse : b
Bonne réponse !

Question 2 : Quelle est la complexité moyenne de recherche dans un tableau trié ?
a) O(1)
b) O(log n)
c) O(n)
Réponse : b
Bonne réponse !

Votre score final : 2/2

Livrables des Équipes

1. **Code source de l’application** : Le code doit être stocké dans un dépôt GitHub ou GitLab, avec un historique de commits. Le team leader sera responsable du dépôt, mais tous les membres doivent collaborer via le même repository.
2. **Instructions d’utilisation** : Inclure un fichier README expliquant comment utiliser l’application, avec des exemples d'exécution.
3. **Rapport** : Un rapport de maximum 5 pages expliquant les choix techniques, les défis rencontrés et les solutions apportées.
4. **Démonstration des fonctionnalités principales** : Une démonstration de l'application, avec les principales fonctionnalités comme la gestion des utilisateurs, les QCM, l'affichage du score et l'historique.