AP - Application de quiz

1 Contexte

M.Kedelec, directeur de l'agence de colonies de vacances 123Cambrousse basée dans les Côtes d'Armor, vous a mandaté pour la réalisation d'une application. Il souhaite une application Console de quiz (type QCM) destinée aux enfants de 4 à 16 ans. Celle-ci sera pilotée par des animateurs de colonie, et devra être entièrement paramétrable sans nécessiter de grandes connaissances en informatique. Il faudra accompagner l'application d'une documentation utilisateur permettant aux animateurs de comprendre comment créer et modifier eux-mêmes leurs quiz.

2 Objectif

Par groupes de 2, vous travaillerez à la réalisation de l'application console. L'accent sera mis sur les étapes fonctionnelles de recette et documentation utilisateur, ainsi que sur l'organisation du travail en équipe via Trello. La recette aura pour but de rendre votre application robuste, les éventuelles erreurs de paramétrage devront être gérées. L'atelier sera découpé en itérations. Pour chaque itération, vous procéderez par étapes dont vous respecterez l'ordre :

- Rédaction de différents cas de recette, cas normal et cas limite (3 minimum par étape)
- Identification et répartition des tâches dans le binôme via Trello.
- Rédaction/actualisation de la documentation utilisateur pour coller à l'état courant du projet.

Le travail de recherche/codage/documentation se fera sur 5 semaines

La semaine 6 sera la semaine de recette (Mardi 28 février 2023) :

Un étudiant du groupe sera tiré au sort pour présenter les choix, méthodes, … et faire la démo sans faire intervenir l'autre mais l'appréciation obtenue est attribuée au binôme..

Les contraintes du nouveau jeu sont les suivantes :

- Le thème du jeu est libre.
- Il n'y a pas de limite pour le nombre de questions.
- Le nombre d'essai sera **paramétrable**.
- Les questions peuvent avoir plusieurs réponses possibles sans limite mais le joueur ne peut saisir qu'une seule réponse par question.
- Les écrans du jeu (le jeu par lui-même et l'affichage des résultats) garderont un aspect semblable aux écrans du jeu de géographie. (Annexe 2)
- Les points attribués à chaque question seront **paramétrables**.
- Lors de l'écriture de la solution, il faut rendre le jeu complètement **paramétrable** (*ce qui n'est pas le cas dans le jeu ci dessous*.) c'est-à-dire que la feuille de code développé doit s'adapter aux changements **sans modification de code**. Ainsi, le *thème*, les *questions*, les *réponses*, le *nombre d'essai* et les *points* seront donc définis dans un fichier séparé du code pour s'adapter au jeu (*fichier .**JSON** par exemple*)

  Lors de la phase de recette et lors de la démo, le jeu doit être capable de bien réagir aux données du recetteur. (Annexe 3)
- Le code sera documenté pour le comprendre du mieux possible.
- Il faudra concevoir une notice utilisateur indiquant le mode opératoire pour adapter le jeu aux besoins de l'utilisateur.

3 Les itérations

3.1 Première itération

Pour la première itération, nous allons paramétrer le thème du questionnaire et le nombre de questions qui seront posées.
Pour cela, vous implémenterez les objectifs suivants :

- La création d'un fichier de configuration JSON qui définit le thème du questionnaire et le nombre de questions posées.
- La création d'une fonction Python qui va charger les données du fichier de configuration et gérer l'initialisation du jeu.

3.2 Deuxième itération

Pour cette itération, nous souhaitons charger les questions du quiz, tirer au hasard les questions qui seront posées et les afficher.
Pour cela, vous implémenterez les objectifs suivants :

- L'ajout de questions de test dans le fichier de configuration JSON.
- Le tirage aléatoire de n questions (n étant le nombre de questions paramétré dans le fichier de configuration). Les questions ne doivent pas apparaître plusieurs fois dans une même session de quiz.
- L'affichage d'une question avec ses réponses possibles.
- La possibilité de valider une réponse et afficher la question suivante (nous omettrons l'évaluation de la réponse dans cette étape)

3.3 Troisième itération

Pour cette troisième itération, nous voulons évaluer les réponses, donner des points, et afficher le score final à la fin du quiz.
Pour cela, vous implémenterez les objectifs suivants :

- L'ajout des bonnes réponses aux questions dans le fichier de configuration JSON.
- La vérification d'une réponse lors de la validation par l'utilisateur, et l'attribution ou non de points.
- Une conclusion propre du quiz après la validation de la dernière réponse, l'affichage du score final, et la possibilité de relancer un quiz.

3.4 Quatrième itération

La quatrième itération introduira la possibilité d'avoir un nombre différent de réponses pour chaque question. Il faudra pouvoir décider du nombre de réponses à afficher pour chaque question, et ces réponses devront être tirées au sort parmi une sélection.
Pour cela, vous implémenterez les objectifs suivants :

- L'adaptation du fichier de configuration JSON pour coller aux nouvelles exigences.
- Le tirage aléatoire de n réponses pour une question donnée (n étant précisé pour chaque question dans le fichier de configuration JSON) Les réponses ne doivent pas apparaître plusieurs fois dans une même question.

3.5 Cinquième itération

Pour la cinquième itération, nous souhaitons rendre possible qu'une question ait plusieurs bonnes réponses à cocher. Il faut que la question soit validée uniquement si toutes les bonnes réponses, et seulement celles-ci sont cochées.
Pour cela, vous implémenterez les objectifs suivants :

- L'adaptation du fichier de configuration JSON pour coller aux nouvelles exigences.
- L'adaptation du code afin de prendre en compte la possibilité que de telles questions soient tirées.

3.6 Sixième itération

Enfin, cette dernière itération permettra à l'utilisateur de choisir un quiz parmi une liste de choix.
Tous les quiz seront paramétrés dans le même fichier JSON. Pour cela, vous implémenterez les objectifs suivants :

- L'adaptation du fichier de configuration JSON pour coller aux nouvelles exigences.
- L'adaptation de l'écran de fin de quiz permettant de lancer un nouveau quiz, ou de relancer le même quiz.

ANNEXE 1 : Code Python initial

```
# Simple quiz en utilisant les dictionnaires
def main():
    questions={'Quel est la capitale du Pérou ? ':'Lima',
               'Comment s\'appelle le petit du lion':'lionceau',
               'Comment dit on plage en espagnol ':'playa',
               'Quel est la longueur du canal de Nantes à Brest':'260',
               }
    print ("*** Début du Quiz ***\n")
    nom = input (" Entrez votre nom: ").title()
    print ()
    print("\nBien joué {0}, vous avez repondu à {1} de {2} questions.".format(nom,
quiz(questions), len(questions)))

def quiz(qs):
    points = 0
    for qu,an in qs.items():
        if input(qu).lower() == an.lower():
            points += 1
            print("Juste.")
        else:
            print("Oups, la bonne réponse est \"{}\".".format(an))
    return points

if __name__ == "__main__":
    main()
```

Annexe 2 : Ecran d'exemple

JEU DE GEOGRAPHIE

QUELLE EST LA CAPITALE DU PEROU <chaine>

DANS QUEL DEPARTEMENT SE TROUVE GLENAC <chaine>

DONNER LE NOM DU DEPARTEMENT DE N° 86 <chaine>

B R A V O !!!

TOUT EST EXACT, VOUS ETES UN AS

VOTRE SCORE EST <entier>/<entier>

VOUS AVEZ AU MOINS UNE MAUVAISE REPONSE !

LES SOLUTIONS SONT LES SUIVANTES :

QUELLE EST LA CAPITALE DU PEROU LIMA

DANS QUEL DEPARTEMENT SE TROUVE GLENAC 56 ou MORBIHAN DONNER LE NOM DU DEPARTEMENT DE N° 86 VIENNE

VOTRE SCORE EST <entier>/<entier>

<entier> signifie qu'il sera affiché en lieu et place une valeur entière. Ceci vaut pour n'importe quel type.