Projet POO en Java
Jeu de Questions / Réponses

1) Instructions générales

Ce projet est à réaliser entièrement en langage Java et en mode console.

a) Documentation

- Supports de cours
- Des ressources disponibles sur Internet, notamment sur le site Oracle ou sur la bibliothèque Scholarvox offerte sur myEffrei.

b) Modules, APIs et concepts concernés par ce programme

La réalisation de ce projet implique l'utilisation des :

- Concepts généraux de Java (encapsulation, surcharge de méthodes. héritage, polymorphisme)
- Packages, interfaces
- Collections de type List<T>
- Input/Output avec des fichiers éventuellement
- des notions de Java non étudiées peuvent être utilisées à la condition sine qua non qu'elles soit comprise par tous les membres du groupe et dûment justifié dans le rapport (enum, classe Arrays, classes internes, interface iterable<T>, collections Map<T>, Set<T>…, lambda expressions.

Deux packages application et testsapplication sont demandés.

c) Planning

- Mise en ligne du projet : **04/11/2021**
- **Revue de code : semaine du 15/11/2021 ou du 22/11/2021**
    - **1er dépôt du code le 14/11/2021 avant minuit**
    - Une présentation de l'état d'avancement de chaque équipe ainsi que les perspectives tracées avant la remise du projet final. Elle se déroulera sur une durée d'environ 15 minutes.

- semaine du **06/12/2021**
    - **2ème dépôt du code le 5/12/2021 avant minuit**
    - soutenance du projet avec temps de parole équitable parmi tous les membres d'une équipe.
    - Exécution de tests significatifs
    - **Chaque membre d'une équipe doit s'être approprié l'intégralité du code rendu**
- semaine du **13/12/2021**
    - **3ème dépôt du code et du rapport le 17 décembre avant 20h**

d) Consignes d'organisation

- Le projet est à réaliser en équipes de 4 membres.
- La liste définitive des équipes est à fournir à vos intervenants respectifs au plus tard le **10/11/2021**. Un malus sera appliqué pour tout retard.
- La note de projet comptera pour 30% de la note finale du cours POO en Java.

e) Rendus attendus

- Les projets sont à rendre au plus tard le **17/12/2021 à 20h.**
- Les livrables doivent inclure un fichier **.zip** ou **.jar** (bien faire attention à ce que les classes **.java** y soient présentes) et un rapport au format pdf.
- Les livrables seront soumis dans des zones de dépôt dédiées sur Moodle.
- Le rapport doit synthétiser la conception technique de votre projet ainsi que vos choix de solutions justifiés.

2) Cahier des charges

L'objectif de ce projet est d'implémenter un jeu de Questions / Réponses entre **au moins 4 joueurs** au début du jeu. Les joueurs sont éliminés à travers les différentes phases du jeu jusqu'à ce qu'il ne reste plus qu'un seul gagnant.

Le jeu se base sur une liste de questions auxquelles les candidats doivent répondre à tour de rôle. En fonction des réponses, un score est cumulé pour chaque joueur et les candidats ayant les meilleurs scores sont sélectionnés pour les phases suivantes.

Les questions traitent **au moins 10 thèmes** et sont de 3 types différents :

- QCM : Le joueur peut choisir une réponse entre plusieurs
- VF : Le joueur peut répondre uniquement par vrai / faux
- RC : Le joueur peut saisir une réponse courte.

De plus, les questions ont 3 niveaux de difficulté :

- 1 : Niveau facile
- 2 : Niveau moyen
- 3 : Niveau difficile

La réalisation de cette application s'appuie sur les spécifications fonctionnelles détaillées dans la section qui suit.

a) Description fonctionnelle

Pour réaliser le jeu de Question / Réponse, il convient d'implémenter les classes suivantes :

Les classes des types des questions

Une question de type QCM est décrite par un texte, 3 variables de réponses proposées et une variable bonne réponse.

Une question de type VF est décrite par son texte et sa bonne réponse de type booléen.

Une question de type RC est décrite par son texte et une variable bonne réponse de type chaîne de caractères.

Les questions sont numérotées de manière séquentielle indépendamment de leurs thèmes.

Donner l'implémentation **des questions de telle sorte que pour chaque question** on ait :

- Un constructeur
- Une méthode **qui retourne une représentation textuelle** d'une question décrite par un numéro, un thème, un niveau de difficulté, un énoncé et ses champs de réponse en fonction de son type
- Une méthode **de saisie** qui permet la saisie d'une question d'un type donné.
  Cette méthode de saisie peut se limiter à écrire les données en dur dans le programme ou bien dans un fichier. Aucune saisie interactive en mode console n'est permise.

Themes

Un thème est désigné par une chaîne de caractères (Sciences, Sport, Histoire, ..).

On dispose d'**au moins 10 thèmes différents**.

Lorsqu'un thème est sélectionné, un indicateur est positionné sur ce thème pour ne pas le sélectionner lors des sélections suivantes.

Donner l'implémentation de la classe **Themes** comportant:

- Un constructeur
- Une méthode **de sélection d'un thème** qui demande à l'utilisateur un thème et renvoie son indice
- Une méthode **de sélection des thèmes**
- Une méthode **qui retourne une représentation textuelle** de tous les thèmes et la valeur de leur indicateur.

Questions

Pour chaque thème, on dispose d'un **ensemble de questions dont le nombre et les types sont variables**. L'ensemble des questions relatives à un thème est stocké dans une liste chaînée de questions. A chaque fois, un indicateur de la question sélectionnée est mis à jour.

Le nombre de questions de chaque thème est donné aléatoirement entre 5 et 10.

Donner l'implémentation de la classe **Questions** contenant :

- Un constructeur
- Une méthode **d'ajout d'une question**
- Une méthode **de suppression d'une question par son rang dans la liste**
- Une (ou plusieurs) méthode(s) **de sélection d'une question** qui sélectionne une question pour un joueur selon une politique qui sera définie dans les différentes phases du jeu
- Une méthode **qui retourne une représentation textuelle** de tous les thèmes et de leurs questions.

Phase

Une partie de jeu se déroule en plusieurs phases.

A chaque phase, un nombre de joueurs ayant les plus faibles scores est éliminé et les autres sont sélectionnés pour la phase qui suit.

Le nombre de joueurs éliminés ne doit pas conduire à une impossibilité de poursuivre le jeu dans les phases suivantes, par exemple faute de joueurs en nombre suffisant.

Définir une interface **Phase** comportant les méthodes permettant de :

- sélectionner des joueurs pour la phase suivante
- dérouler une phase de jeu

NB : Il est possible d'ajouter toute autre classe jugée nécessaire pour implémenter une partie de jeu à partir de l'interface **Phase**.

Joueur

Un joueur est décrit par un numéro, un nom, un score et un état (sélectionné, gagnant, super gagnant, éliminé ou en attente). Pour chaque réponse correcte, le score est incrémenté d'une valeur dépendant de la phase du jeu. Le numéro du joueur est un numéro qui commence à 100 et est incrémenté de 10 à chaque fois (100, 110, 120, 130, ...).

Donner l'implémentation de la classe **Joueur** comportant :

- Un constructeur
- Une méthode **de saisie des informations d'un joueur.**
  Cette méthode de saisie peut se limiter à écrire les données en dur dans le programme ou bien dans un fichier. Aucune saisie interactive en mode console n'est permise.
-
- Une méthode **qui retourne une représentation textuelle**
- Une méthode **de mise à jour du score**
- Une méthode **de changement de l'état**

Joueurs

L'ensemble des joueurs est stocké dans un **tableau 20 joueur**s candidats dont **au moins 4** pourront participer au jeu.

La modélisation à l'aide d'un tableau est imposée et ne peut être modifiée.

Donner l'implémentation de la classe **Joueurs** comportant :

- Un constructeur
- Une méthode de sélection aléatoire d'un joueur du tableau
- Une méthode générant aléatoirement l'ensemble des joueurs participant au jeu
- Une méthode **qui retourne une représentation textuelle de** l'ensemble des joueurs participant au jeu

Remarques :

1. Pour des besoins de tests, il faut prévoir à l'intérieur des classes des constructeurs supplémentaires permettant de remplir par programme les structures nécessaires au déroulement du test afin d'éviter la saisie de données longue et fastidieuse.
2. Il faudra remplir avec des données correctes et sensées.
3. Pour les noms des joueurs, utiliser les lettres de l'alphabet A, B, C, …, Z.
4. **S'il y a lieu surcharger et/ou redéfinir les méthodes et/ou créer des sous-classes.**

b) Application à réaliser

Phases du jeu

Le jeu de Questions/Réponses à réaliser comprendra **trois phases** où **au moins 4 joueurs** s'affrontent autour de questions portant sur **au moins 10 thèmes:**

Phase I :

Dans cette phase, le jeu se déroule entre au moins 4 joueurs choisis aléatoirement parmi les 20 joueurs/

Un thème parmi au moins 10 thèmes est sélectionné automatiquement dans un ordre séquentiel (un indicateur du dernier thème sélectionné est mis à jour, après le choix du dernier thème, on revient au premier).

Au moins une question de niveau facile est sélectionnée pour chacun des joueurs selon une politique Round-Robin (i.e de manière circulaire). Les joueurs répondent à leur(s) question(s) séparément, le score est incrémenté de 2 si la réponse donnée est correcte.

Le nombre de joueurs éliminés, le nombre de questions, et le nombre de thèmes ne doivent pas conduire à une impossibilité de poursuivre le jeu dans les phases suivantes, par exemple faute de joueurs et/ou de questions et/ou de thèmes en nombre suffisant.

Phase II :

Le jeu se déroule entre les joueurs gagnants de la phase I.

Cette phase propose au moins deux questions de niveau moyen par thème pour chaque joueur.

A ce niveau, les questions porteront sur au moins 6 thèmes choisis aléatoirement.

A tour de rôle et de manière alternée, chaque joueur choisit un thème (ensuite un deuxième) dont les indicateurs sont mis à jour.

Les questions de niveau moyen, dans le thème choisi, sont sélectionnées selon la politique Round-Robin et présentées au joueur.

Le score du joueur donnant la bonne réponse est incrémenté de 3.

Le nombre de joueurs en lice, le nombre de questions et le nombre de thèmes ne doivent pas conduire à une impossibilité de poursuivre le jeu dans la phase suivante, par exemple faute de joueurs et/ou de questions et/ou de thèmes en nombre suffisant.

Seul les 2 joueurs ayant le meilleur score sont sélectionnés pour la phase 3.

Phase III :

Dans cette phase, le jeu se déroule entre les deux joueurs gagnants de la phase II.

Au moins deux questions de niveau difficile porteront sur trois thèmes choisis par le concepteur du jeu.

Une question de niveau difficile, dans le thème choisi, est sélectionnée selon la politique Round-Robin et présentée au joueur.

Le score du joueur donnant la bonne réponse est incrémenté de 5.

Le gagnant du jeu est le joueur dont le score est le plus élevé à l'issue de la phase 3.

Description de l'application

L'application doit, de manière générale, permettre de réaliser les actions suivantes :

1. Afficher les thèmes choisis
2. Créer une liste de questions pour chaque thème
3. Afficher toutes les questions d'un niveau donné par thème
4. Ajouter une question à la liste pour un thème donné
5. Supprimer une question de numéro donné de la liste pour un thème donné
6. Créer le tableau de joueurs et afficher leurs états
7. Lancer une partie du jeu avec au moins 4 joueurs choisis en affichant toutes les étapes du déroulement du jeu
8. Quitter le jeu
9. D'autres actions peuvent être ajoutées si nécessaire.

3) Fonctionnalités optionnelles BONUS

Gestion de conflits

Pour chaque joueur un timer régi par un Thread est associé. Il démarre lorsque son joueur obtient la main pour répondre à la question, et s'arrête dès que la réponse est fournie.

En cas d'égalité de scores entre les joueurs à une phase donnée, ils seront départagés grâce aux valeurs des timers. Ainsi, les joueurs ayant été les plus rapides seront ceux qui se qualifient à la phase d'après.

En cas d'égalité des scores et des timers, au moins trois questions supplémentaires pour les départager. Après cela, faire une sélection aléatoire pour passer à l'étape suivante.

Le grand jeu

Faire en sorte que l'application permette de considérer 3 groupes d'*a*u moins 4 joueurs

- Implémenter une partie de jeu pour chaque groupe de joueurs
- Récupérer les numéros des trois joueurs gagnants
- Implémenter le grand jeu entre les trois joueurs gagnants, la même politique est reprise commençant à la phase II.
- Afficher toutes les étapes du déroulement du jeu.

Implémenter une IA

Utiliser les threads pour jouer le rôle de joueurs et répondre aléatoirement aux questions dans le cadre d'une partie de jeu complète.