**Enoncé :**

Deux équipes vont s’affronter lors d’un jeu de connaissance ayant comme thème les capitales européennes.

Il s’agit d’un quizz.

Voici un exemple de question :

*Quelle est la capitale de la Belgique ?*

1. *Amsterdam*
2. *Bruxelles*
3. *Paris*

Pour gagner, une équipe doit obtenir 3 points.

Pour chaque question, les 2 équipes sélectionne chacune un candidat.

Le candidat qui buzze en premier, donne sa réponse. Si la réponse n’est pas bonne, il est éliminé, sinon, il fait gagner un point à son équipe et peut la rejoindre.

L’autre candidat rejoint son équipe.

Au sein de chaque équipe, les candidats sont placés dans une liste. L’ordre y a de l’importance. C’est le premier candidat de la liste qui sera sélectionné pour répondre à une question. S’il est remis en jeu, il se remettra en fin de liste.

Les questions sont tirées au hasard dans un questionnaire. Elles ne sont pas réutilisées sauf si toutes les questions ont été posées.

Si toutes les questions ont été utilisées, on n’arrête pas le jeu et on réutilise le même questionnaire.

Le jeu se termine lorsqu’une équipe a remporté 3 points ou parce qu’une équipe n’a plus de candidat.

**Implémentation :**

**La classe *Equipe* est à compléter.**

Chaque équipe possède un numéro

Elle possède des points.

Au départ, l’équipe a 0 point.

La méthode ajouter1Point() permet d’augmenter de 1 ses points..

Un candidat est reconnu par son nom (*String*).

Pour la liste des candidats, nous avons choisi une table : tableCandidats.

Les candidats encore en jeu sont placés en début de table.

La classe possède l’attribut nombreCandidatsEnJeu. Cet attribut permet de connaître le nombre de candidats en jeu, mais aussi de savoir quelle partie de la table est réellement occupée.

La méthode toString() donnée renvoie une chaîne de caractères avec uniquement les « nombreCandidatsEnJeu » premières cases de la table.

La classe possède un constructeur qui reçoit en paramètre une table avec les candidats au départ et le numéro de l’équipe.

Elle possède également les méthodes « get » habituelles.

La classe possède un autre constructeur qui va servir pour les tests. Ne le modifiez pas !

**Voici les 2 méthodes à compléter : selectionnerCandidat()et remettreEnJeu().**

Respectez bien la *JavaDoc* et l’implémentation choisie.

**La classe *TestEquipe* permet de tester les 2 méthodes demandées.**

**La classe *QuestionCM* est donnée.**

Tout objet de la classe *QuestionCM* possède comme attributs une question, 3 réponses et le numéro de la bonne réponse.

Cette classe possède un constructeur et les méthodes « get » habituelles.

**La classe *Questionnaire* est à compléter.**

Le questionnaire possède, comme attributs, une table de questions à choix multiples (*QuestionCM*) : tableQuestions et le nombre de questions restantes : nombreQuestionsRestantes.

La table, tout au long du jeu, contiendra toujours toutes les questions proposées au départ mais l’ordre de celles-ci vont varier.

Les questions qui n’ont pas encore été tirées se trouvent au début de la table, ensuite on retrouve les autres.

Voici un exemple :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| QA | QI | QB | QE | QF |

(4 questions restantes)

Dans cet exemple, la table contient 5 questions.

La question QF (capitale de la France) a déjà été tirée.

Le nombre de questions restantes est de 4.

Le constructeur s’occupe d’initialiser le questionnaire grâce à une table de questions à choix multiples passées en paramètres.

La méthode reinitialiser()réinitialise le questionnaire. Aucune des questions n’ayant disparu de la table, cette méthode se contente de remettre l’attribut nombreQuestionsRestantes au nombre de questions contenues dans la table.

**Il reste une méthode à compléter : fournirQuestion().**

Cette méthode renvoie une question au hasard parmi les questions restantes.

Pour que la table contienne bien d’abord les questions non tirées et ensuite celles déjà tirées, **une** permutation sera nécessaire.

La question tirée au hasard sera permutée avec la dernière question non encore tirée.

Dans l’exemple précédent, voici quel sera le contenu de la table, si la question tirée au hasard est QI :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| QA | QE | QB | QI | QF |

(3 questions restantes)

Pour tirer au hasard une question, passer par les indices des questions dans la table (QI était à l’indice 1). La méthode unEntierAuHasardEntre()est donnée.

Respectez bien la *JavaDoc* et l’implémentation choisie.

**La classe *TestQuestionnaire* permet de tester la classe *Questionnaire*.**

Attention, compte tenu du côté aléatoire de la méthode fournirResultat(), vous devez déduire vous-même si votre méthode est correcte ou pas via les affichages !

**La classe *Buzz* est à compléter.**

**La méthode main() de la classe *Buzz* permet de simuler un jeu. Elle doit être complétée.**

Pour faciliter les tests, les candidats et les questions ont été *hardcodés*.

Il faut pouvoir suivre le jeu grâce à des affichages appropriés.

Voici ce que pourrait être l’affichage d’une « question » :

*Quizz - Capitales des pays de l'union européenne*

*Voici les candidats en jeu de l'equipe 1 :*

*Lea Marie Chloe Julie*

*Voici les candidats en jeu de l'equipe 2 :*

*Pierre Sam Tom Hugo*

*Nouvelle question*

*Selectionnez les 2 candidats*

*Le candidat selectionne de l'equipe 1 est Lea*

*Le candidat selectionne de l'equipe 2 est Pierre*

*Voici la question :*

*PAYS : Allemagne*

*1 --> Amsterdam*

*2 --> Dublin*

*3 --> Berlin*

*Entrez le numero de l'equipe qui a buzze (1 ou 2) : 2*

*Entrez son choix (1, 2 ou 3) : 2*

*La reponse n'est pas bonne*

*Pierre est elimine*

*l'equipe 1 a 0 point(s)*

*l'equipe 2 a 0 point(s)*

*Voici les candidats en jeu de l'equipe 1 :*

*Marie Chloe Julie Lea*

*Voici les candidats en jeu de l'equipe 2 :*

*Sam Tom Hugo*

*Nouvelle question*

*Selectionnez les 2 candidats*

*Le candidat selectionne de l'equipe 1 est Marie*

*Le candidat selectionne de l'equipe 2 est Sam*

*Voici la question :*

*PAYS : France*

*1 --> Paris*

*2 --> Prague*

*3 --> Vilnius*

*Entrez le numero de l'equipe qui a buzze (1 ou 2) : 1*

*Entrez son choix (1, 2 ou 3) : 1*

*Bravo! La reponse est bonne*

*l'equipe 1 a gagne 1 point*

*l'equipe 1 a 1 point(s)*

*l'equipe 2 a 0 point(s)*

etc …

Voici ce que pourrait être l’affichage à la fin du jeu :

*Le jeu est termine*

*l'equipe 2 n'a plus de candidat*