TD4 les collections

Exercice 1 Ecrire une collection de personnes dans laquelle on peut ajouter des employés ou des étudiants

Vous pouvez reprendre les classes Personne, Etudiant et Employe des exercices précédents.

1) Créer une méthode qui trie les personnes par nom.
2) Ecrire une méthode qui sépare les employés des étudiants
3) Parcourir les 2 nouvelles listes avec la méthode .forEach pour afficher leur contenu
4) Créer une fonction qui extrait la personne de plus petite taille de la liste en utilisant une boucle ou un stream
5) Créer une méthode qui appelle la précédente et qui ajoute la personne de plus petite taille à la liste triée puis affichez la liste triée avec la méthode de votre choix

Exercice 2 Trier la liste des personnes dans un set en utilisant l'interface comparable.

1) Les personnes sont triées par nom puis par prénom dans l'ordre alphabétique.
2) Parcourir la liste de personnes pour ne conserver uniquement les employés qui ont plus de 3 ans d'ancienneté et les ajouter dans une liste secondaire.
3) Même question mais en utilisant un stream et la méthode filter

Exercice 3 Jeu pierre feuille ciseaux

1) Créer un enum Coup (Pierre, Feuille, Ciseaux) qui représentera un coup.

2) Ecrire une classe qui représente une manche

La classe Manche contient :

- Le coup du joueur 1
- Le coup du joueur 2
- Un résultat (joueur 1 gagne,égalité,joueur 2 gagne)
- Une date de début de manche (LocalDateTime)
- Une date de fin de manche (LocalDateTime)
- Une méthode qui compare les coups des joueurs et retourne le résultat
    - La pierre bat les ciseaux
    - La feuille bat la pierre
    - Les ciseaux battent la feuille
- Une méthode statique coupAleatoire qui génère et retourne un coup.

3) Créer une classe qui représente une partie

Elle contient

- Pseudo Joueur 1
- Pseudo Joueur 2
- Une liste de manches
- Une date de début de partie (LocalDateTime)
- Une date de fin de partie (LocalDateTime)
- Une méthode qui compte le nombre de points du joueur passé en paramètre
- Une méthode qui affiche les 2 joueurs, les manches réalisées et le résultat total ainsi que la date et heure de début de partie et la durée totale de la partie
- Créer une méthode jouerManche qui récupère le coup du joueur 1 et le coup du joueur 2 (si un seul joueur, elle génère un coup aléatoire)
  La méthode instancie une Manche, ajoute la manche à la liste
- Une méthode creerPartie qui demande le nombre de joueur (1 ou 2) le pseudo du joueur 1 puis du joueur 2 (si 2 joueurs)
- Créer une méthode jouer qui appelle la méthode jouer manche jusqu'à ce qu'un joueur atteigne 3 points. Une fois les 3 points atteints, elle affiche les résultats et termine la partie.

4) Créer une classe PierreFeuilleCiseauTest

Elle contient la méthode main qui permet de démarrer une partie

A la fin de chaque partie, vous pouvez soit quitter la partie soit rejouer.

5) Comment peut-on rendre la méthode qui compare les coups des joueurs plus générique ?

Vérifier votre solution au près de l'enseignant avant de l'implémenter

Indice : il s'agit d'utiliser une structure de données adaptée qui nous évite de comparer chacune des combinaisons possibles et de simplifier l'ajout de nouveaux coups

6) Ajouter le puits

- Le puits bat la pierre
- Le puits bat les ciseaux
- La feuille bat le puits

PS : Libre à vous d'ajouter de nouvelle méthodes ou fonctionnalités, nous sommes là pour apprendre !