Dans ce TP, nous continuerons d'utiliser des listes chaînées, mais cette fois-ci en prêtant attention à l'ordre des éléments. Nous vous rappelons qu'il est très fortement conseillé de lire tout le TP une fois avant de le commencer.
Nous allons poursuive le TP4, en rajoutant des fonctionnalités. Nous vous conseillons de faire une copie de tout le dossier du TP4 avant de commencer le TP5.
Nous vous rappelons que vous êtes censé tester régulièrement votre code. Il est normal que vous n'ayez pas toujours le résultat voulu du premier coup, donc n'hésitez pas à faire des essais. Et *surtout*, si vous ne comprenez pas pourquoi un test ne fonctionne pas comme vous vous y attendiez, pensez à utiliser les fonctionnalités de `jGrasp` vues au TP1 pour suivre ce que votre code fait effectivement.

Exercice 1 [Vérification du TP4] Vous pouvez faire le TP 5 même si le 4 n'est pas fini. Mais vous devez avoir au minimum :

1. une classe Employe avec les attributs
   `private final String nom;`
   `private int salaire;`
   `private int benefice;`
   Des getters et setters pour ces variables, ainsi qu'une méthode d'affichage et un constructeur
2. une classe Cellule avec les atributs
   `Employe emp`
   `Cellule suivant`
   ainsi qu'un constructeur `public Cellule(Employe emp)`
3. une classe Entreprise qui contient l'unique attribut
   `private Cellule chef`
   un constructeur des méthodes :
   `public void affiche()` pour afficher l'entreprise
   `public void ajout(Employe emp)` pour rajouter un employé
   `public void demission(String nom)`, retirer un employé de la liste [1].

Exercice 2 [Quelques méthodes en plus] Si les questions 3 ou 2 paraissent trop compliqué à coder, passer à l'exercice suivant.

1. Créez un méthode `public boolean augmente(String nom, int montant)` qui augmente l'employé `nom`, s'il existe, d'un `montant` strictement positif. La méthode renvoie `false` si une des condition n'est pas respecté.
2. Créez une méthode `public ArrayList<Employe> choixSalaire(int min, int max)` qui renvoie la liste des Employes dont le salaire est compris entre `min` et `max`. Reportez vous à la document ou au tp3 si vous ne connaissez plus les ArrayList.
3. Refléchissez à la manière de créer une méthode `public void afficheSalaire()` qui affiche la liste des Employes par ordre de salaire croissant.

Exercice 3 [Liste ordonnée] Nous allons maintenant abandonner l'idée que le chef est en tête de liste, à la place nous rajoutons la condition que les Employes seront classés par salaire.

1. Mettez la méthode `changerChef()` et `recrute(Employe emp)` en commentaire, elles ne serviront plus.
2. Modifier les méthodes pour tenir compte de ce nouveau critère.

[1] On suppose qu'il n'y a pas deux employés qui ont le même nom.

(a) `ajout(Employe emp)` doit placer l'employé à son emplacement précis dans la liste, après un employé qui gagne moins et avant un qui gagne plus. Ou alors en début (resp. ou fin) de liste si personne ne gagne moins (resp. ne gagne plus)

(b) `augmente(String nom, int montant)` doit faire avancer l'employé `nom` dans la liste, pour le mettre après ceux qui gagnaient plus que lui et qui maintenant gagnent moins, pour que l'ordre soit de nouveau respecté,

(c) et `acquisition(Entreprise ent)`, la méthode qui ajoute une entreprise `ent` à l'entreprise courante, doit placer tous les employé de l'entreprise acquise dans le bon ordre.

3. Y a t-il d'autres méthodes que vous pouvez modifier pour tenir compte de ce nouveau critère ? Que vous pouvez simplifier, ou rendre plus rapide ?
4. Créer une méthode `public boolean correct()` qui renvoie `true` si et seulement si les salaires sont croissant dans la liste.
5. Si cette question est trop complexe pour l'instant, passez à l'exercice suivant :

   Réfléchissez à comment vous coderiez une méthode `public void afficheNom()` qui affiche les Employes par ordre alphabétique et finit par une ligne de tirets.

Exercice 4 [Listes ordonnées] Nous allons maintenant rajouter en plus l'ordre alphabétique à la cellule. Cet exercice va vous demander de modifier des méthodes que vous avez déjà écrites, donc avant de commencer cet exercice, faite une copie de sauvegarde de votre dossier dans l'état dans lequel il est en ce moment, pour être sûr de ne rien perdre.

1. Modifiez la classe `Cellule` pour remplacer l'attribut `suivante` par deux attributs, `suivanteNom` et `suivanteSalaire`,
2. Dans la classe `Entreprise`, remplacez l'attribut `Cellule cellule` par `Cellule petitSalaire` et `Cellule petitNom`, qui contiendront la cellule du Employe le moins bien payé et du premier Employe par ordre alphabétique.
3. Tout comme dans la question 3-2 Modifier les différentes méthodes pour que la liste reste en permanence trié et par le salaire et par le nom.

   En particulier, pensez à regarder la documentation de String pour trouver comment comparer les noms.
4. Est-ce que `afficheNom` est plus facile à coder ?