Indications : Il vous faut respecter la présentation des listes faites en cours. C'est à dire que la modélisation doit se faire en utilisant 3 classes :

- une pour le contenu,
- une pour les cellules, où se trouve l'essentiel de la gestion du chaînage,
- et une principale, qui modélise la liste dans sa globalité, et qui implémente les méthodes classiques de l'interface de liste.

Dans ces exercices nous nommons les classes `Contenu`, `Cellule`, `Liste` d'une autre façon, mais vous les reconnaîtrez sans difficultés.

Cadre : On souhaite réaliser une chorale de robots. Pour cela on les regroupe en file indienne, où le premier est désigné chef. L'ensemble est représenté par une structure de liste chaînée.

Rythme : Il ne serait pas raisonnable de passer plus de 30 minutes sur les deux premiers exercices.

Exercice 1

Voici le début de la classe `Robot` :

```
public class Robot{
    private final char nom;//lettre entre 'a' et 'z'
    private int energie;
    private final String texte; //ce qu'il doit dire

    public Robot(char nom, String paroles){
        this.nom = nom;
        // on donne une énergie entre 10 et 20
        energie = 10 + (int)(Math.random() * 11);
        texte = paroles;
    }
}
```

1. Ajoutez une méthode `description` qui retournera une chaîne de caractères décrivant le robot sur le modèle `"Robot <...> dit <...> quand il parle et a <...> points d'énergie"`
2. Ajoutez une méthode `boolean nomCorrect` qui vérifie qu'un robot a bien comme nom une lettre de l'alphabet minuscule.
3. Ajoutez des accesseurs permettant de récupérer les valeurs des différents attributs de `Robot`.

Exercice 2 - Constructions

On considère les classes `Groupe` et `Cellule` suivantes :

```
public class Cellule{
    private Robot rob;
    private Cellule suivant;
}
public class Groupe{
    private Cellule chefDeFile;
 }
```

Ajoutez y les éléments suivants :

1. Un constructeur qui crée un groupe vide
2. Un constructeur de `Cellule` qui prend un argument de type `Robot` et un argument de type `Cellule`
3. Un constructeur de `Cellule` qui prend seulement un argument de type `Robot`, et qui s'écrit en faisant appel au constructeur précédent.
4. Écrivez une méthode `void prendreTete(Robot r)` qui teste si le robot `r` a un nom correct et si oui le place en position d'être leader du groupe. L'ancien chef de file, s'il existait, est maintenant juste après lui.

Exercice 3 - Parcours simples

1. Écrivez une méthode `affiche()` qui affiche la description de tout le groupe. Vérifiez votre affichage dans le cas d'un groupe vide.
2. Écrivez une méthode `ajouteNouveau` qui ajoute un robot en fin de groupe, l'ajout n'a lieu que si le nom du robot est correct.
3. On rappelle que le type `char` peut être converti facilement en entiers. Par exemple l'expression `'b'-'a'` s'évalue à 1. Vous pouvez donc associer la première lettre de l'alphabet à la valeur 1 et la dernière à 26. Écrivez une méthode `numerologie` qui donnera pour résultat la somme des valeurs lettres des noms des robots, le tout modulo 9.
4. Écrivez une méthode `bandName` qui donnera le nom du groupe constitué de la concaténation de tous les noms des robots pris dans l'ordre.
5. Les robots vont chanter leur texte et ils vont le répéter 2 fois si leur nom est 'b', 3 fois si c'est 'c' etc... Un robot qui s'appelle 'd' dont le texte est "pomme", chantera donc "pomme-pommepommepommepomme"

   Ecrivez une méthode `chante` dans `Robot` qui affiche ce résultat. Après avoir chanté il perdra 10 points d'énergie (sans pouvoir passer en dessous de zéro).
6. Écrivez maintenant une méthode `chantez` qui un à un fait chanter tous les robots du groupe.

Exercice 4 : Opérations chirurgicales

1. Écrivez une méthode de la classe `Groupe` dont la signature est :
   `Groupe couperAPartirDe(char nom)`
   Elle élimine tous les robots qui sont derrière le premier robot dont le nom est précisé. On retournera ceux qui ont été enlevés sous la forme d'un `Groupe`.

   Indications : Comme en cours, on écrira dans la classe `Groupe` simplement ce qu'il est nécessaire pour traiter des cas très particuliers, et l'essentiel du travail sera fait via une méthode auxiliaire de la classe `Cellule`. Probablement de signature :
   `Cellule couperAPartirDe(char nom)`
   Si vous avez besoin ensuite d'un nouveau constructeur, il faudra réfléchir à le déclarer privé ou public.
2. (***) Ecrivez une méthode de signature : `Groupe prendrePause()` qui s'applique à un groupe et en extrait les robots dont l'énergie est passée à zéro. Si vous arrivez à la réaliser pendant la séance de TP, montrez là à votre enseignant qui pourra la prendre en compte dans votre notation de TP.

Exercice 5 : Petites variations (si vous avez le temps)

1. Définissez une classe qui capture plusieurs groupes dans un tableau.

2. Écrivez un constructeur qui crée un tel ensemble, le nombre de groupes étant donné en argument.
3. Écrivez une méthode `effectifs` qui retournera dans un tableau d'entiers les effectifs de chaque groupe d'un objet de cette classe.