Table de hachage, generics, implantations de liste

Exercice 1 - Traitement de texte

On souhaite créer un petit traitement de texte permettant de faire des transformations sur des lignes de texte.

Ecrire un petit petit programme qui prend un nom de fichier contenant un texte et affiche celui-ci ligne à ligne sur la sortie standard.
Vous utiliserez pour cela le `Scanner`.
Transformer votre programme pour qu'il affiche les lignes en muscule ou en minuscule suivant l'option de la ligne de commande

`-upper` met la ligne en majuscule
`-lower` met la ligne en minuscule
si la ligne de commande ne contient ni `-upper` ni `-lower` affiche la ligne identique

Eviter d'écrire des if ... else pour associer à une option l'action à effectuer.
Ajouter l'option `-reverse` qui met une ligne à l'envers.
Ajouter l'option `-replace` qui remplace les "!" par des "*".

Exercice 2 - Générification

Le but de cet exercice est de générifier les classes `fr.umlv.datas.LinkedLink` et `fr.umlv.datas.Link`

1 Paramétré la classe `fr.umlv.datas.LinkedLink` pour que celle-ci soit générique.
2 Modifier la classe `fr.umlv.datas.main.Main` en conséquence.

Exercice 3 - Performance sur les listes

Le but de cet exercice est de tester les différences de performance entre les classes `ArrayList` et `LinkedList` sur différents algorithmes.

1 Nous allons dans un premier temps chronométrer le temps d'un parcours d'une `ArrayList` contenant un million `(1 000 000)` d'entiers en utilisant un `Iterator` (pour le parcours).
Utilisez la méthode `System.nanoTime()` pour effectuer une mesure de temps.
2 Modifier le code pour pouvoir facilement chronométrer le parcours dans le cas d'une `ArrayList` ou d'une `LinkedList`.
3 Dans le but de faire d'autres tests de performance, imaginer un patron de conception (**design pattern**) permettant d'effectuer plusieurs tests sur plusieurs types de `List`.
On appelle patron de conception, une façon d'arranger les objets dans un but précis, ici pour effectuer des tests sur des listes.
4 Refactoriser le code existant en utilisant le patron de conception ainsi défini.

Utiliser le patron de conception pour effectuer les tests suivants sur les deux implémentations de `List` :

- en insérant un milier d'entiers en première position dans une liste vide au départ (comme pour une file).
- parcours de la liste d'un million d'entiers par un itérateur
- parcours de la liste d'un million d'entiers par un itérateur en sens inverse

- parcours de la liste d''un million d'entiers par un index