TD N°07 JAVA

Exercise 1 :

Écrivez la méthode `main` d'une classe `TestChaine` qui crée un chaîne de caractères et qui affiche

- les 3 premiers caractères ;
- les 3 derniers ;
- le (ou les 2) caractère placé au milieu (un peu de calcul à faire...).
- la chaîne en majuscules, puis en minuscules.

Testez l'égalité de 2 chaînes à la casse près. Par exemple, "FAIRE", "FaiRe" et "faire" devraient être égaux.

Comparez des chaînes (dites quelle est la première par ordre alphabétique). Par exemple, testez avec "13" et "123". Testez aussi avec des chaînes qui contiennent des caractères accentués ; par exemple, "écrire" et "Ecrire"

Exercise 2 :

1. Dans la classe `TestNomFichier` écrivez une méthode de classe "`isJavaSource`" qui teste si un nom de fichier se termine par ".java".

2. Tapez un nom absolu de fichier en paramètre de la ligne de commande ; par exemple "/u/rep1/rep2/repfinal/nomfichier.png". La méthode `main` affiche le nom final du fichier ("nomfichier.png"), ainsi que le répertoire racine ("/u") et le répertoire sous lequel le fichier est placé, avec son nom absolu ("/u/rep1/rep2/repfinal") et son nom terminal ("repfinal"). Elle affiche ensuite si le fichier est un fichier source Java.

Exercise 3 :

1. Dans une classe `Chaine` écrivez une méthode `eclater` qui prend en paramètre une chaîne de caractères et un séparateur et qui renvoie un tableau de chaînes de caractères *complètement rempli* dont les éléments sont les sous-chaînes de la chaîne passée en paramètre, séparés des autres éléments par le séparateur. Par exemple, `eclater("vincent, françois, paul, et les autres", ",")` renverra un tableau de taille 4, dont les éléments sont "vincent", " françois" " paul" " et les autres" (remarquez les espaces au début des 3 derniers éléments). Pour simplifier vous pourrez supposer qu'il y a moins de 100 séparateurs. N'oubliez pas de tester le cas où le séparateur est formé de plus d'un caractère. N'utilisez que les méthodes `substring` et `indexOf` de la classe `String` pour effectuer cet éclatement.
2. Surchargez la méthode pour qu'elle accepte un troisième paramètre de type booléen qui indique si les espaces qui entourent les éléments doivent être ignorés ou non. Cette fois-ce utilisez la méthode `split`, et éventuellement d'autres méthodes (parcourez la javadoc pour chercher une autre méthode qui pourrait vous être utile), de la classe `String` (mais sans utiliser les facilités des expressions régulières) pour éclater la chaîne et enlever les espaces. Pour l'exemple de la question précédente, les espaces pourraient alors être enlevés au début des éléments.

Exercise 4 :

Dans la méthode `main` d'une classe `Concatenation`, concaténez tous les paramètres de la ligne de commande en une seule chaîne que vous afficherez (tapez au moins 6 paramètres). Chaque paramètre sera séparé du précédent par ";" dans la concaténation. Utilisez d'abord le type `String` puis le type `StringBuilder`.

Exécutez chaque opération 100.000 fois (adapter le nombre en fonction de la puissance de votre ordinateur) et calculer le temps mis pour `String` et pour `StringBuilder` grâce à la méthode `System.currentTimeMillis()` qui renvoie le temps actuel en millisecondes.