Série B : Les chaînes de caractères

Table des matières

Exercice B.1: La classe String

Dans la suite, nous voulons écrire quelques méthodes qui vont faciliter le traitement de chaînes de caractères. Ce serait logique de définir une classe qui étend la classe String.

Ouvrez la page JavaDoc de la classe **`String`** et analysez l'en-tête de la classe. Que constatez-vous ? Quelle est la conséquence pour nous, c.-à-d. quelle est alors la meilleure possibilité pour définir nos méthodes utilitaires pour la classe **`String`**?

Définir (en Unimozer) une nouvelle classe pour les méthodes que nous allons définir dans la suite. Appelez la classe **`GiString`**.

Notions requises : définition de classes

Exercice B.2: Ajout de fonctions de traitement de chaînes de caractères

Définir les méthodes suivantes (n'oubliez pas les préfixes pour les méthodes) :

`String delete(String target, int beginIndex, int endIndex)`

qui supprime de la chaîne **`target`** la sous-chaîne qui se trouve entre **`beginIndex`** et **`endIndex`**. La lettre à la position **`endIndex`** n'est pas touchée. La chaîne modifiée est retournée comme résultat.

`String insert (String target, String str, int index)`

qui insère la chaîne **`str`** dans la chaîne **`target`** à la position **`index`**. La chaîne modifiée est retournée comme résultat.

`int count(String target, String str)`

qui compte combien de fois la sous-chaîne **`str`** apparaît dans **`target`**.

`String reverse (String target)`

qui retourne l'ordre des caractères dans la chaîne **`target`**. La chaîne modifiée est retournée comme résultat.

`boolean palindromeWord(String target)`

qui retourne **`true`** si la chaîne **`target`** est un palindrome (c.-à-d. un mot qui reste exactement le même si on le lit de la fin vers le début).

Exemples :

'*Salut*' → non '*RAD AR*' → non

'*RADAR*' → oui '*Radar*' → non

'*racecar*' → oui

`String removeAccents(String target)`

qui remplace toutes les lettres spéciales ou accentuées (âàä,îï,ç,...) par les lettres non accentuées correspondantes (a,i,c,...). Les autres caractères de la chaîne restent inchangés. La chaîne modifiée est retournée comme résultat.

`String removeNonAlphabetic (String target)`

qui supprime tous les espaces et autres caractères spéciaux, ne laissant que les lettres alphabétiques (majuscules ou minuscules). La chaîne modifiée est retournée comme résultat.

`boolean palindromeSentence(String target)`

qui retourne **`true`** si la chaîne **`target`** est un palindrome. Cette méthode ignore tous les caractères spéciaux, la casse (majuscule, minuscule), et considère les lettres accentuées comme des lettres non accentuées.

Exemples de phrases palindromes :

Step on no pets !

Panic in a Titanic, I nap.

A man, a plan, a canal -- Panama!

La mère Gide digère mal.

Engage le jeu que je le gagne.

Voir aussi :

http://www.derf.net/palindromes/old.palindrome.html

http://www.gnudung.de/kram/sprache/palindrom.htm

http://fr.wikipedia.org/wiki/Liste_de_palindromes_français

`String extract(String target, String separator, int n)`

Soit **`target`** une chaîne qui contient un certain nombre de sous-chaînes, qui sont séparées par un séparateur **`separator`**. La méthode **`extractString`** retourne la **`n`**-ième sous-chaîne de la chaîne **`target`**.

Si le numéro **`n`** donné est supérieur au nombre de sous-chaînes, la fonction retourne une chaîne vide.

Exemples:

```
extract("Jim-Bob Walton#John Deere#Jack Daniels#Joe Cartwright", "#", 2)  → "John Deere"
extract("Jim-Bob Walton#John Deere#Jack Daniels#Joe Cartwright", "#", 4)  → "Joe Cartwright"
extract("Jim-Bob Walton#John Deere#Jack Daniels#Joe Cartwright", "#", 6)  → ""
extract("Jim-Bob Walton#John Deere#Jack Daniels#Joe Cartwright", " ", 3)  → "Deere#Jack"
```

`ArrayList<String> extractAll(String target, String separator)`

Employez la méthode **`extract`** pour séparer toutes les sous-chaînes de **`target`** en les ajoutant une à une dans une **`ArrayList`**.

Notions requises : Utility Class
Classes requises : String, ArrayList

Exercice B.3: Encryptage

1. Cryptographie

Le mot «cryptographie» vient du grec *kryptos*, «caché», et *graphein*, «écrire». Il désigne l'ensemble des procédés permettant de transformer un message écrit, dit clair, en un autre message, dit chiffré ou crypté, par le moyen de certains codages convenus d'avance; le texte reçu n'est donc compréhensible que pour celui qui en connaît la clé.

Chiffrage: Changer les données en une forme non compréhensive en appliquant un algorithme de chiffrage.

Déchiffrage: Reconstituer les données à leur forme initiale.

2. Le chiffre de César *(Shift by 3)*

La cryptographie existe déjà depuis longtemps. Le chiffrage le plus connu (mais pas le plus ancien) est celui de Jules César.

César a crypté ses messages en employant une méthode qu'on a appelée ensuite le «Chiffre de César». Le chiffre de César consistait simplement à décaler les lettres de l'alphabet de quelques positions vers la droite ou la gauche.

On trouve une description du chiffre de César dans les ***Vies des douze Césars*** de **Suétone** (historien antique qui vivait autour de 100 après J.-C.) :

... Extant et ad Ciceronem, item ad familiares domesticis de rebus, in quibus, si qua occultius perferenda erant, per notas scripsit, id est sic structo litterarum ordine, ut nullum verbum effici posset: quae si qui investigare et persequi velit, quartam elementorum litteram, id est D pro A et perinde reliquas commutet. ... *(Livre premier, César, LVI)*	... On a conservé en outre ses lettres à Cicéron, et celles qu'il adressait à ses familiers sur ses affaires domestiques; quand il avait à leur faire quelque communication secrète, il usait d'un chiffre, c'est-à-dire qu'il brouillait les lettres de telle façon qu'on ne pût reconstituer aucun mot: si l'on veut en découvrir le sens et les déchiffrer, il faut substituer à chaque lettre la troisième qui la suit dans l'alphabet, c'est-à-dire le D à l'A, et ainsi de suite. ...

L'empereur Auguste utilisait un chiffre similaire, mais il décalait les lettres d'un rang seulement. ***Exemple:***

Décalons par exemple les lettres de 3 rangs vers la gauche, comme le faisait Jules César:

Clair	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Chiffré	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Ainsi, le message : `AVE CAESAR MORITURI TE SALUTANT`

devient : `DYH FDHVDU PRULWXUL WH VDOXWDQW`

Malgré (ou à cause de) sa simplicité, cet encryptage chiffre fut encore employé par des officiers sudistes pendant la guerre de Sécession et par l'armée russe en 1915.

Le chiffre ROT13, apparu en 1984 dans un programme permettant de lire les "News" de USENET, décale les lettres de 13 rangs.

Travail à réaliser:

Développer une classe **Cryptor** qui permet de chiffrer et de déchiffrer des textes par la méthode '***Shift by N***' (c.-à-d. un déplacement des lettres de l'alphabet de **n** positions.).

Version pure: Convertir d'abord le texte à chiffrer en majuscules. Chiffrer/Déchiffrer uniquement les lettres majuscules de l'alphabet en effectuant une rotation de **n** positions entre les lettres de l'alphabet. Les autres caractères restent inchangés. Limitez n à une valeur entre 1 et 26.

Version élargie: Chiffrer/Déchiffrer tous les caractères en effectuant une rotation de **n** positions (**n** étant un entier quelconque).

3. Chiffre de Vigenère

Agé de vingt-six ans, **Blaise de Vigenère (1523-1596)**, diplomate français, passa deux années en mission diplomatique à Rome, où il se familiarisa avec les écrits sur la cryptographie de Alberti, Trithème et Porta. Au début, son intérêt pour la cryptographie était purement pratique et lié à son activité diplomatique. Une dizaine d'années plus tard, vers 1560, Vigenère considéra qu'il avait mis de côté assez d'argent pour abandonner sa carrière et se consacrer à l'étude. C'est seulement à ce moment-là qu'il examina en détail les idées de ses prédécesseurs, tramant grâce à elles un nouveau chiffre, cohérent et puissant. Bien qu'Alberti, Trithème, Bellaso et Porta en aient fourni les bases du chiffre, c'est du nom de Vigenère que ce nouveau chiffre fut baptisé, en l'honneur de l'homme qui lui donna sa forme finale.

Le **chiffre de Vigenère** est une amélioration décisive du chiffre de César, trop facile à casser. Ce chiffre utilise une **clé** qui définit le décalage pour chaque lettre du message (A: décalage de 1 position, B: 2 positions, C: 3 positions, ..., Z: 26 positions).

Exemple: codons le texte "CHIFFRE DE VIGENERE" avec la clé "BACHELIER".

Clair	C	H	I	F	F	R	E	D	E	V	I	G	E	N	E	R	E
Clé	B	A	C	H	E	L	I	E	R	B	A	C	H	E	L	I	E
Décalage	2	1	3	8	5	12	9	5	18	2	1	3	8	5	12	9	5
Chiffré	E	I	L	N	K	D	N	I	W	X	J	J	M	S	Q	A	J

La grande force du chiffre de Vigenère est que la même lettre sera chiffrée de différentes manières. Par exemple le E du texte clair ci-dessus a été chiffré successivement M V L P I, ce qui rend inutilisable l'analyse des fréquences classique.

Travail à réaliser:

Copiez les méthodes développées dans la partie 2 et modifiez-les de façon à ce qu'elles permettent de chiffrer et de déchiffrer des messages par le chiffre de Vigenère. La clé n'est pas mémorisée dans la classe ! Elle peut contenir tous les caractères possibles. Toujours est-il qu'une lettre 'A' dans la clé doit provoquer une rotation d'une unité.

Informations supplémentaires:

- **http://www.apprendre-en-ligne.net/crypto**
- **http://www.simonsingh.net/The_Black_Chamber/vigenere_cracking.html**