MOOC Init Prog Java
Exercices semaine 5

Exercice 15 : Échauffement avec les tableaux dynamiques (tableaux dynamiques)

En vous aidant si nécessaire d'un programme, répondez aux questions suivantes :

A : Quelles valeurs contient le tableau `tab` après l'exécution du programme suivant? Expliquez.

```
import java.util.ArrayList;

class DynamicArray1
{
    public static void main(String[] args)
        {
            final int TAILLE = 10;
            ArrayList<Integer> tab = new ArrayList<Integer>();
            for (int i = 0; i < TAILLE; ++i) {
                tab.add(tab.size());
            }
        }

}
```

B : Quelles valeurs contient le tableau `tab2` après l'exécution du programme suivant? Expliquez.

```
import java.util.ArrayList;

class DynamicArray2
{
    public static void main(String[] args)
        {
            ArrayList<Integer> tab1 = new ArrayList<Integer>();
            tab1.add(99);
            tab1.add(1);
            tab1.add(0);

            ArrayList<Integer> tab2 = new ArrayList<Integer>();
            for(int i=0; i < tab1.size(); ++i) {
                tab2.add(tab1.get(0));
            }
        }
}
```

Exercice 16 : Nombres premiers (tableaux dynamiques)

Rappel : Pour utiliser le type `ArrayList`, il faut importer les classes ie définissant ce type, au moyen de la directive :

```
import java.util.ArrayList
```

Écrivez le programme `Premiers.java` qui stocke dans un tableau dynamique l'ensemble des nombres premiers compris entre 2 et 100 et affiche le contenu de ce tableau.

Test de la primalité d'un nombre n (algorithme) :

1. Vérifier si le nombre **n** est pair (si oui, il n'est pas premier sauf si c'est 2).
2. Pour tous les nombres impairs inférieurs ou égaux à la racine carrée de **n**, vérifier s'ils divisent **n**. Si ce n'est pas le cas, alors **n** est premier.

Votre programme devrait produire un affichage ressemblan à ceci :

```
Les nombres premiers compris entre 2 et 100 sont les suivants :
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
53
59
61
67
71
73
79
83
89
97
```

Exercice 17 : Palindrômes (chaînes de caractères)

Un palindrome est un mot que l'on peut lire dans les deux sens. La distinction entre majuscules/minuscules n'a aucune importance pour la lecture d'un palindrome. Si on ne tient pas compte des caractères non alphabétiques (i.e. `' '`, `','`, `'-'` et `'\''`), une phrase complète peut aussi être considérée comme un palindrome.

Exemples de palindromes:

```
Otto
Elu par cette crapule
Esope reste ici et se repose
Tu l'as trop ecrase, Cesar, ce Port-Salut
A man, a plan, a canal, Panama
```

Exemples de non-palindromes:

```
Cours de Java
Le pont de la rivière Kwai
```

Ecrivez un programme `Palindrome.java` qui :

1. lit une chaine de caractères du clavier;
2. l'épure (ou plutôt en épure une copie) des caractères non alphabétiques;
3. et teste si la chaîne ainsi épurée est un palindrome.

Exemple d'exécution:

```
Entrez un mot ou une phrase : Otto
C'est un palindrome !
```

Pour ce programme, il convient d'utiliser plusieurs méthodes prédéfinies de la classe `String`, comme par exemple `charAt()`, `toLowerCase()` et `length()`.

- `chaine.toLowerCase()` permet de convertir tous les caratères de l'objet `chaine` en minuscule.

Indication : l'appel `Character.isLetter(c)`, où `c` est un caractère, permet de tester si `c` est alphabétique (`Character.isLetter` est à écrire tel quel.. nous verrons un peu plus tard ce que sont les méthodes statiques qui s'utilisent de cette façon).
