

## Module 3 – Aide à la Réussite : EP 2 Soutien en Programmation

### Travaux Dirigés (5), Licence 1ère Année

### Manipulation des chaînes de caractères

#### Exercice 1 Jeu du pendu

Le but du jeu du Pendu est de retrouver un mot caché en moins de 10 essais, en proposant à chaque fois une lettre. Au départ, on affiche la 1ère et la dernière lettre du mot, on met entre les deux le nombre d'étoiles correspondant au nombre de lettres manquantes. Un utilisateur saisira le mot caché, on "cachera" le mot saisi en affichant 50 lignes vides, un autre essaiera de trouver le mot en proposant des lettres au programme. Si la lettre se trouve dans le mot caché, on l'affiche autant de fois qu'elle apparaît dans le mot, on laisse les étoiles sur ce qui n'a pas été trouvé. Si la lettre n'est pas dans le mot, on affiche le dessin du pendu, au stade où il est.

- 1.1. Écrire une méthode **void affichePendu(int n)** qui affiche le dessin du pendu correspondant au nombre **n** d'essais déjà effectués : on commence par le tiret du bas, puis les 4 traits verticaux, puis les 5 tirets du haut, puis le trait oblique de la potence, puis la corde, puis la tête, puis le corps, puis les deux bras, puis chaque jambe, ce qui fait bien **10** essais en tout.

**Note** : utiliser **switch()**.

Dessin complet :

```

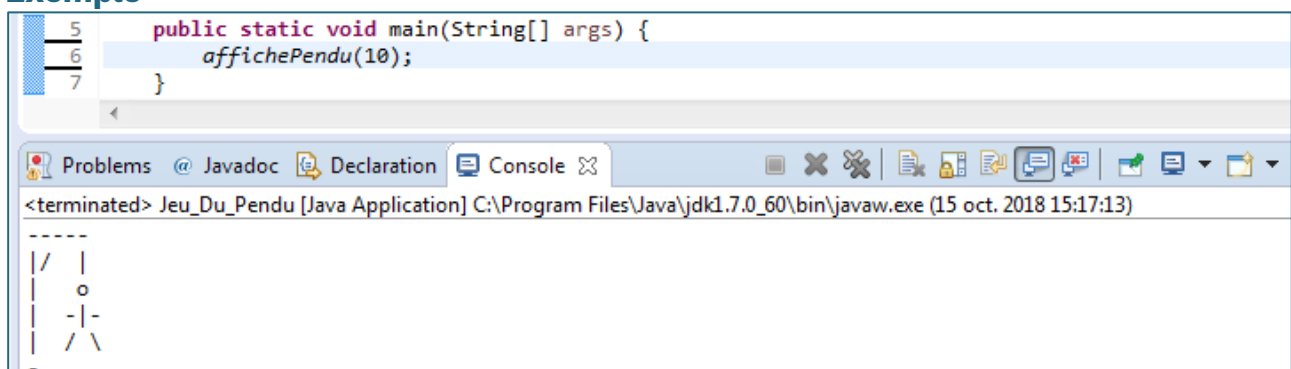
-----
|/  |
|   o
|  -|-
|  / \
-

```

On affiche le dessin à chaque fois que l'utilisateur se trompe de lettre. Lorsque le dessin est complet, le jeu s'arrête et on affiche "**Perdu !**". Si l'utilisateur saisit une lettre se trouvant à l'intérieur du mot, on remplace toutes les étoiles correspondant à la lettre (par la lettre) et on affiche le mot avec les étoiles correspondant au nouveau stade de recherche.

Tester la méthode **affichePendu()** en appelant dans le `main()` : `affichePendu(3)`, `affichePendu(7)`, `affichePendu(10)`, ...

#### Exemple



```

5 public static void main(String[] args) {
6     affichePendu(10);
7 }

```

Problems @ Javadoc Declaration Console

<terminated> Jeu\_Du\_Pendu [Java Application] C:\Program Files\Java\jdk1.7.0\_60\bin\javaw.exe (15 oct. 2018 15:17:13)

```

-----
|/  |
|   o
|  -|-
|  / \
-

```

- 1.2. Écrire une méthode **String initMot(String mot)**, de paramètre une chaîne de caractères **mot**, qui retourne le mot (chaîne de caractères) correspondant à la 1ère et à la dernière lettre de la chaîne mot avec des étoiles entre.

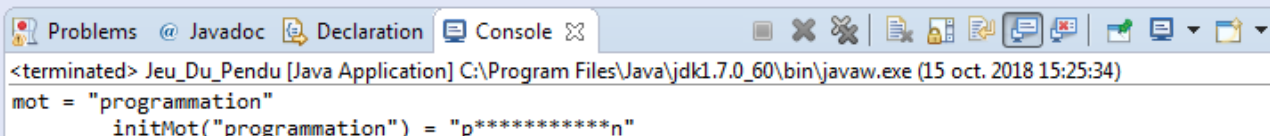
Tester la méthode **initMot()** dans le main() en demandant à l'utilisateur de saisir une chaîne et en appelant **initMot()** ensuite...

### Exemple

```

5   public static void main(String[] args) {
6       String mot = "programmation";
7       System.out.println("mot = \""+mot+"\"");
8       System.out.println("\tinitMot(\""+mot+"\") = \"\" + initMot(mot)+"\"");
9   }

```



```

<terminated> Jeu_Du_Pendu [Java Application] C:\Program Files\Java\jdk1.7.0_60\bin\javaw.exe (15 oct. 2018 15:25:34)
mot = "programmation"
    initMot("programmation") = "p*****n"

```

- 1.3. Écrire une méthode **String interieurMot(String mot)**, de paramètre une chaîne de caractères **mot**, qui retourne (chaîne de caractères) l'intérieur de la chaîne **mot** (cela nous servira pour chercher le caractère ensuite).

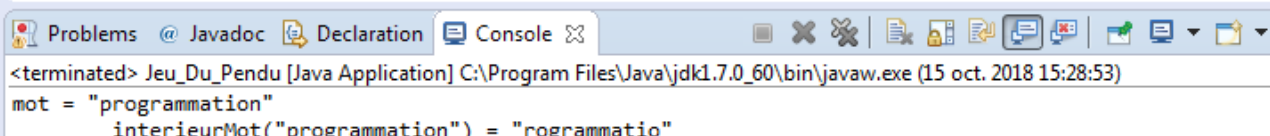
Tester la méthode **interieurMot()**.

### Exemple

```

5   public static void main(String[] args) {
6       String mot = "programmation";
7       System.out.println("mot = \""+mot+"\"");
8       System.out.println("\tinterieurMot(\""+mot+"\") = \"\" + interieurMot(mot)+"\"");
9   }

```



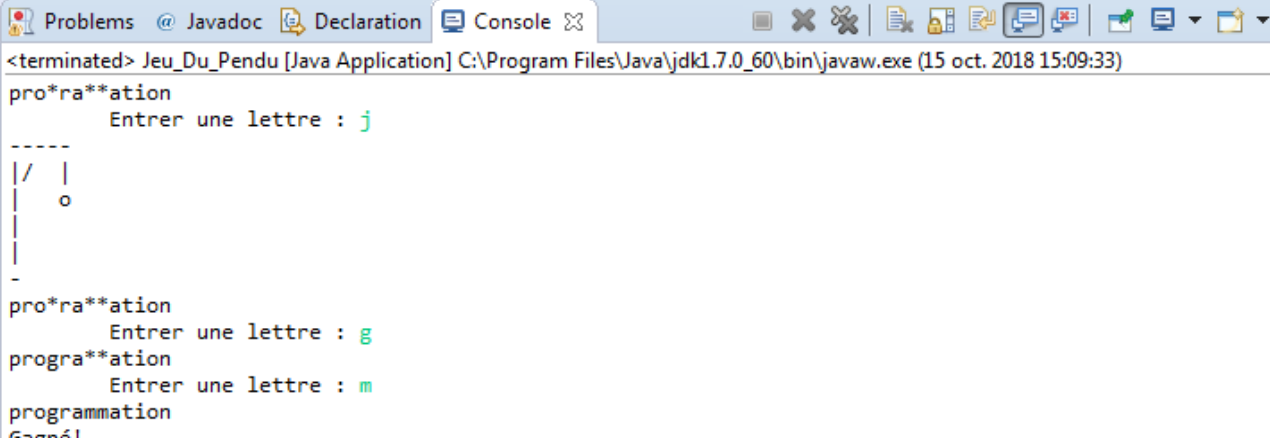
```

<terminated> Jeu_Du_Pendu [Java Application] C:\Program Files\Java\jdk1.7.0_60\bin\javaw.exe (15 oct. 2018 15:28:53)
mot = "programmation"
    interieurMot("programmation") = "rogrammatio"

```

- 1.4. Écrire une méthode **void joue(String mot)**, de paramètre une chaîne de caractères **mot**, sans retour, qui réalise le jeu du pendu...

### Exemple



```

<terminated> Jeu_Du_Pendu [Java Application] C:\Program Files\Java\jdk1.7.0_60\bin\javaw.exe (15 oct. 2018 15:09:33)
pro*ra**ation
    Entrer une lettre : j
-----
|/ |
|  o
|
|
|
-
pro*ra**ation
    Entrer une lettre : g
progra**ation
    Entrer une lettre : m
programmation
Gagné!

```

## Exercice 2 Jeu de l'Alphabet inversé

On se propose d'écrire un programme qui affiche à l'utilisateur tour à tour chacune des lettres de l'alphabet, dans un ordre quelconque, et à ce dernier d'introduire le plus rapidement **la lettre qui la précède alphabétiquement**. Le but du jeu est d'introduire chacune des lettres demandées en un minimum de temps. Initialement, on dispose des chaînes de caractères suivantes :

```
String alpha = "abcdefghijklmnopqrstuvwxyz";
String trouve = "";
String reste = alpha;
```

- **alpha** est la chaîne comprenant toutes les lettres de l'alphabet,
- **trouve** contiendra les lettres trouvées par l'utilisateur, initialement vide,
- **reste** contiendra les lettres qui n'ont pas encore été proposées à l'utilisateur ou qui ont été réintroduites après une réponse incorrecte, initialement identique à **alpha**.

Si le programme propose la lettre 'y' et que l'utilisateur introduit la lettre 'x', alors la réponse est correcte ('x' précède alphabétiquement 'y'), 'x' est alors ajouté à **trouve** et 'y' est retiré de **reste**.

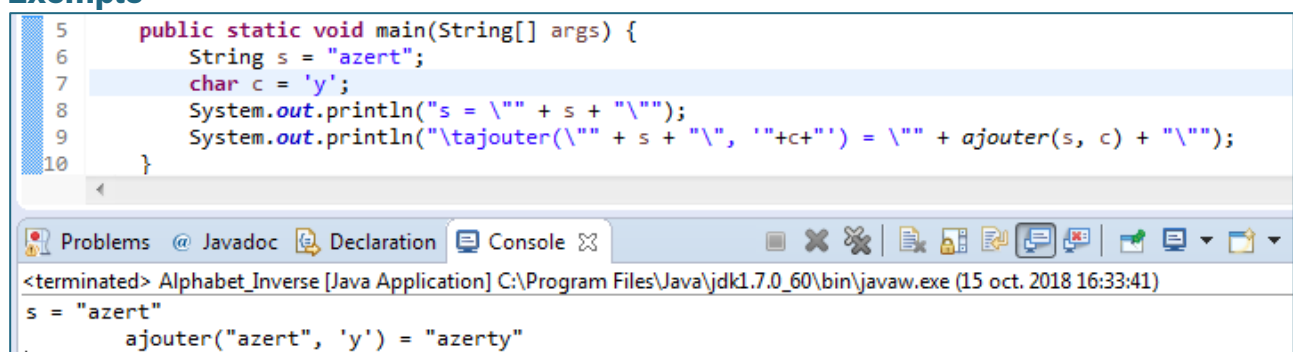
Si la réponse de l'utilisateur est incorrecte, 'y' n'est pas retiré de **reste** (elle sera reproposée ultérieurement) et une sanction est appliquée à l'utilisateur : elle consiste à retirer une lettre choisie aléatoirement parmi celles de **trouve** (déjà trouvées par l'utilisateur). Par conséquent, **la lettre qui la suit alphabétiquement** est ajoutée à **reste**.

**Note** : on considère que 'z' précède alphabétiquement 'a'.

- 2.1. Écrire une méthode **String ajouter(String s, char c)**, de paramètres une chaîne **s** et un caractère **c**, qui ajoute le caractère **c** à la fin de la chaîne **s** et retourne cette dernière comme résultat.

Tester la méthode **ajouter()**.

### Exemple



```

5 public static void main(String[] args) {
6     String s = "azert";
7     char c = 'y';
8     System.out.println("s = \"" + s + "\"");
9     System.out.println("\tajouter(\"\" + s + "\", '\"+c+\"') = \"" + ajouter(s, c) + "\"");
10 }

```

Problems @ Javadoc Declaration Console

<terminated> Alphabet\_Inverse [Java Application] C:\Program Files\Java\jdk1.7.0\_60\bin\javaw.exe (15 oct. 2018 16:33:41)

```

s = "azert"
ajouter("azert", 'y') = "azerty"

```

- 2.2. Écrire une méthode **String retirer(String s, char c)**, de paramètre une chaîne **s** et un caractère **c**, qui supprime le caractère **c** de la chaîne **s** et retourne cette dernière comme résultat.

Tester la méthode **retirer()**.

### Exemple

```

5   public static void main(String[] args) {
6       String s = "azerty";
7       char c = 't';
8       System.out.println("s = \"\" + s + \"\"");
9       System.out.println("\tretirer(\"\" + s + "\", '\" + c + '\"') = \"\" + retirer(s, c) + \"\"");
10  }

```

Problems @ Javadoc Declaration Console

<terminated> Alphabet\_Inverse [Java Application] C:\Program Files\Java\jdk1.7.0\_60\bin\javaw.exe (15 oct. 2018 16:38:32)

```

s = "azerty"
retirer("azerty", 't') = "azery"

```

**2.3.** Écrire une méthode **char choisir(String s)**, de paramètre une chaîne **s**, qui retourne un caractère choisi aléatoirement parmi ceux de la chaîne **s**. Tester la méthode **choisir()**.

### Exemple

```

5   public static void main(String[] args) {
6       String s = "azerty";
7       System.out.println("s = \"\" + s + \"\"");
8       System.out.println("\tchoisir(\"\" + s + \"\") a retourné aléatoirement le caractère '\"+
9           choisir(s)+\"\"");
10      System.out.println("\tchoisir(\"\" + s + \"\") a retourné aléatoirement le caractère '\"+
11          choisir(s)+\"\"");
12  }

```

Problems @ Javadoc Declaration Console

<terminated> Alphabet\_Inverse [Java Application] C:\Program Files\Java\jdk1.7.0\_60\bin\javaw.exe (15 oct. 2018 16:44:55)

```

s = "azerty"
choisir("azerty") a retourné aléatoirement le caractère 'a'
choisir("azerty") a retourné aléatoirement le caractère 'e'

```

**2.4.** Écrire une méthode **void joue()** qui fait appel aux méthodes précédentes et qui réalise le jeu tel que décrit ci-dessus.

**Remarque :** pour calculer le temps mis par l'utilisateur pour terminer le jeu, utiliser la méthode **System.currentTimeMillis()** qui retourne l'heure courante en millisecondes.

Mettre en début de la méthode **joue()** : **Long debut = System.currentTimeMillis();** mettre en fin de la méthode **joue()** : **println("Durée : "+((System.currentTimeMillis()-debut)/1000)+" secondes !");** et vous aurez le temps écoulé en secondes.

### Exemple d'exécution

Problems @ Javadoc Declaration Console

<terminated> Alphabet\_Inverse [Java Application] C:\Program Files\Java\jdk1.7.0\_60\bin\javaw.exe (15 oct. 2018 16:46:28)

```

Réponse correcte !
Lettre trouvées : "ypxuaemgdvtiwhqfrokbnjcs"
Donner la lettre avant : a
e
Réponse incorrecte !
Lettre trouvées : "ypxuemgdvtiwhqfrokbnjcs"
Donner la lettre avant : a
z
Réponse correcte !
Lettre trouvées : "ypxuemgdvtiwhqfrokbnjcsz"
Donner la lettre avant : b
a
Réponse correcte !
Lettre trouvées : "ypxuemgdvtiwhqfrokbnjcsza"
Donner la lettre avant : m
l
Réponse correcte !
Lettre trouvées : "ypxuemgdvtiwhqfrokbnjcszal"
Toutes les lettres ont été trouvées en 117 secondes !

```