

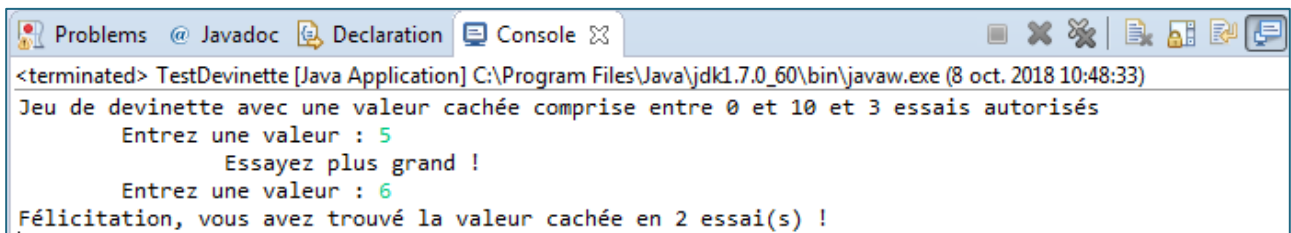
Module 1 – Informatique – Algorithmique et Programmation Objet

Travaux Pratiques (9), Licence 1ère Année

Classes et Méthodes

Exercice 1 Jeu de devinettes

On s'intéresse dans cet exercice à écrire un petit jeu de devinette. L'ordinateur doit générer une valeur entière aléatoire comprise entre **0** et une **valeur maximale** passée en argument et l'utilisateur doit ensuite essayer de retrouver cette valeur cachée. Pour cela, on va utiliser une classe Devinette. L'illustration suivante montre un déroulé possible de partie.



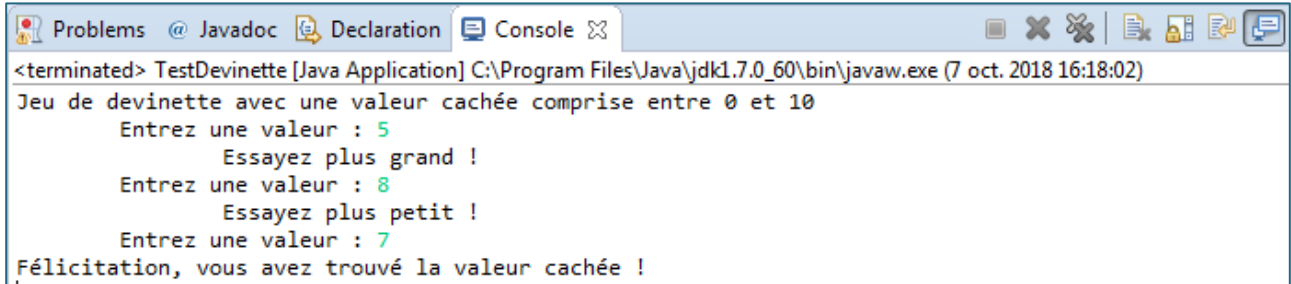
```
<terminated> TestDevinette [Java Application] C:\Program Files\Java\jdk1.7.0_60\bin\javaw.exe (8 oct. 2018 10:48:33)
Jeu de devinette avec une valeur cachée comprise entre 0 et 10 et 3 essais autorisés
  Entrez une valeur : 5
    Essayez plus grand !
  Entrez une valeur : 6
Félicitation, vous avez trouvé la valeur cachée en 2 essai(s) !
```

- 1.1. Écrire la classe **Devinette** qui contient les champs suivants :
 - un entier **cache** qui contient la valeur cachée,
 - un entier **maxVal** qui représente la valeur maximale que peut prendre la valeur cachée
- 1.2. Ajouter un **constructeur** à votre classe **Devinette** qui prend en argument la borne supérieure à partir de laquelle la valeur cachée sera générée. Ce constructeur doit initialiser le champ **maxVal** à la valeur de l'argument et le champs **cache** à l'aide de **Math.random()**.
- 1.3. Écrire une méthode accesseur **int getCache()** qui retourne la valeur du champ **cache**.
- 1.4. Écrire une méthode accesseur **int getMaxVal()** qui retourne la valeur du champ **maxVal**.
- 1.5. Écrire une méthode **String toString()** qui retourne une chaîne de description de la devinette "**Jeu de devinette avec une valeur cachée entre 0 et maxVal**". Dans la chaîne précédente, le champ **maxVal** doit être remplacé par sa valeur entière.
- 1.6. Proposer enfin une méthode **void jouer()** qui construit un objet Scanner pour saisir les entrées clavier de l'utilisateur et réitère la demande de saisie d'une valeur tant que la valeur cachée n'est pas découverte.

Note : Pour aider l'utilisateur, la méthode affichera « Essayez plus grand » ou « Essayez plus petit » lorsque l'utilisateur a introduit respectivement une valeur plus petite ou plus grande par rapport à la valeur cachée. Quand l'utilisateur trouve la valeur cachée, un message de félicitation doit être affiché et le jeu de la devinette se termine.

- 1.7.  crire un programme principal dans une nouvelle classe **TestDevinette** qui construit un objet *Devinette*, l'affiche   l'aide de `System.out.println()` et lance une partie.

Exemple



```

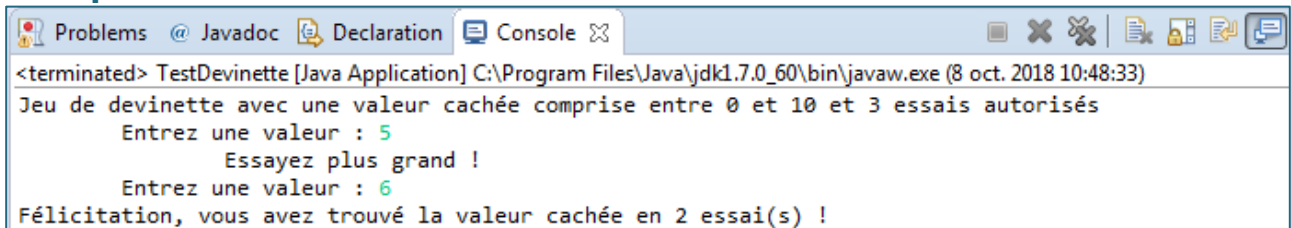
<terminated> TestDevinette [Java Application] C:\Program Files\Java\jdk1.7.0_60\bin\javaw.exe (7 oct. 2018 16:18:02)
Jeu de devinette avec une valeur cach e comprise entre 0 et 10
  Entrez une valeur : 5
    Essayez plus grand !
  Entrez une valeur : 8
    Essayez plus petit !
  Entrez une valeur : 7
F licitation, vous avez trouv  la valeur cach e !
  
```

- 1.8. Modifier votre classe *Devinette* pour ajouter un champ **maxEssais** qui repr sente le nombre maximal d'essais autoris s avant de perdre et son accesseur **int getMaxEssais()** qui retourne sa valeur. Vous proposerez un nouveau constructeur ayant deux arguments : le maximum de la valeur cach e et le nombre maximal d'essais. Votre nouveau constructeur devra appeler le constructeur pr c dent   l'aide du mot-cl  **this**.

Note : vous modifierez la m thode **toString()** pour qu'elle affiche  galement le nombre d'essais autoris s.

- 1.9. Modifier la m thode **void jouer()** pour prendre en compte un nombre maximal d'essais lors de la partie de devinette.

Exemple



```

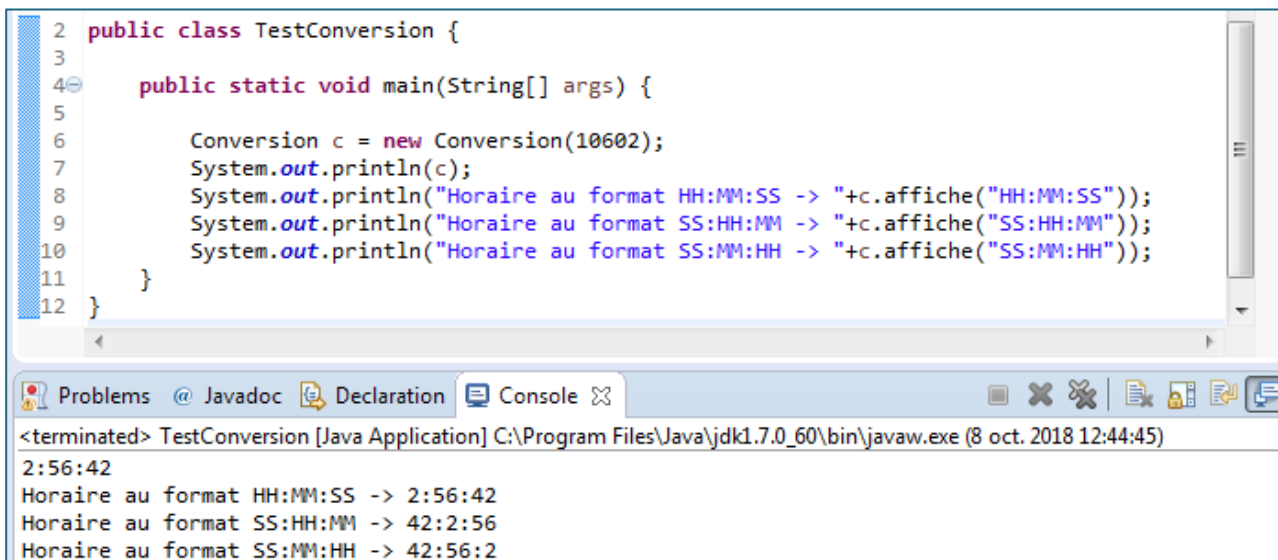
<terminated> TestDevinette [Java Application] C:\Program Files\Java\jdk1.7.0_60\bin\javaw.exe (8 oct. 2018 10:48:33)
Jeu de devinette avec une valeur cach e comprise entre 0 et 10 et 3 essais autoris s
  Entrez une valeur : 5
    Essayez plus grand !
  Entrez une valeur : 6
F licitation, vous avez trouv  la valeur cach e en 2 essai(s) !
  
```

- 1.10.  crire finalement une m thode **boolean equals()** qui permet de comparer deux devinettes.

Note : Deux devinettes sont  gales si leur valeur cach e, leur valeur maximale et leur nombre d'essais sont identiques.

Exercice 2 Conversion

On s'intéresse désormais à la programmation d'un objet de conversion automatique d'une durée en secondes en différents formats de temps comme illustré dans le programme principal suivant.



```

2 public class TestConversion {
3
4     public static void main(String[] args) {
5
6         Conversion c = new Conversion(10602);
7         System.out.println(c);
8         System.out.println("Horaire au format HH:MM:SS -> "+c.affiche("HH:MM:SS"));
9         System.out.println("Horaire au format SS:HH:MM -> "+c.affiche("SS:HH:MM"));
10        System.out.println("Horaire au format SS:MM:HH -> "+c.affiche("SS:MM:HH"));
11    }
12 }
  
```

Console output:
 <terminated> TestConversion [Java Application] C:\Program Files\Java\jdk1.7.0_60\bin\javaw.exe (8 oct. 2018 12:44:45)
 2:56:42
 Horaire au format HH:MM:SS -> 2:56:42
 Horaire au format SS:HH:MM -> 42:2:56
 Horaire au format SS:MM:HH -> 42:56:2

- 2.1. Écrire une classe **Conversion** qui contient un champ privé entier **secondes** qui indique le nombre de secondes à partir duquel on souhaite réaliser la conversion ainsi qu'un **constructeur** qui permet d'initialiser ce champ.
- 2.2. Écrire une méthode accesseur **int getSeconds()** qui retourne la valeur du champ **seconds**.
- 2.3. Écrire des méthodes **int getHours()**, **int getMinutes()** et **int getSeconds()** qui traduisent la valeur **secondes** en heures, minutes et secondes et retourne chacune de ces valeurs.
- 2.4. Proposer une méthode **String toString()** qui retourne l'heure correspondant à **secondes** au format Heures:Minutes:Seconds.
- 2.5. (**Difficile**) Écrire une méthode **String affiche(String format)** qui prend en entrée une chaîne de format contenant les valeurs "HH", "MM" et "SS" dans un ordre quelconque et séparées par des ":" et retourne l'affichage de l'heure dans l'ordre de ce qui est indiqué dans la chaîne format.

Exemple : le format "HH:SS:MM" permet d'afficher l'heure correspondant à **secondes** avec en premier les heures, puis les secondes et enfin les minutes. L'illustration au début de l'exercice fournit d'autres exemples.

Note : On pourra utiliser la méthode `split` des chaînes de caractères qui retourne un tableau de chaînes. On accède aux éléments d'un tableau en indiquant l'indice de l'élément (qui commence à 0) entre crochets "[" et "]".

Exercice 3 Manipulation de chaînes de caractères

Le but de cet exercice est d'écrire une nouvelle classe de chaînes de caractères qui s'appelle **MaChaine** qui permet de contenir un champ qui est une chaîne de caractères **chaine**, composée uniquement de **lettres minuscules** et d'espaces, et de manipuler son contenu à travers plusieurs méthodes déjà étudiées lors des TP précédents :

3.1. Ajoutez à votre class **MaChaine** un constructeur qui permet d'initialiser la chaîne.

Note : la chaîne **chaine** ne sera initialisée que si la chaîne de caractères en paramètre du constructeur ne comporte que des lettres minuscules et des espaces.

3.2. Ajoutez un accesseur **String getChaine()** qui retourne le contenu du champ **chaine**.

Exemple

```

85 public static void main(String[] args) {
86     MaChaine chaine1 = new MaChaine(" b!rav%o le^s ?cham()pions du Jafa ");
87     System.out.println("Le contenu de chaine1 est \" + chaine1.getChaine() + "\"");
88     MaChaine chaine2 = new MaChaine(" bravo les champions du java ");
89     System.out.println("Le contenu de chaine2 est \" + chaine2.getChaine() + "\"");

```

3.3. La méthode **void inversion()** permet d'inverser les caractères de la chaîne

Exemple

```

73 public static void main(String[] args) {
74     MaChaine chaine = new MaChaine("wjiejpm");
75     System.out.println("Le contenu de chaine est \" + chaine.getChaine() + "\"");
76     chaine.inversion();
77     System.out.println("Le nouveau contenu de chaine est \" + chaine.getChaine() + "\"");
78 }

```

3.4. La méthode **boolean palindrome()** qui retourne **true** si et seulement si le contenu de **MaChaine** représente un palindrome.

3.5. La méthode **void enleveEspace()** qui modifie le contenu de la chaîne pour enlever les espaces en trop.

Exemple

```

85 public static void main(String[] args) {
86     MaChaine chaine = new MaChaine(" bravo les champions du java ");
87     System.out.println("Le contenu de chaine est \"" + chaine.getChaine() + "\"");
88     chaine.enleveEspace();
89     System.out.println("Le nouveau contenu de chaine sans espaces en trop est \"" +
90         chaine.getChaine() + "\"");
91 }

```

Problems @ Javadoc Declaration Console

<terminated> MaChaine [Java Application] C:\Program Files\Java\jdk1.7.0_60\bin\javaw.exe (9 oct. 2018 20:12:50)
 Le contenu de chaine est " bravo les champions du java "
 Le nouveau contenu de chaine sans espaces en trop est "bravo les champions du java"

3.6. La méthode **void crypte(int d)** qui modifie le contenu de la chaîne en remplaçant chacun de ses caractères par celui se trouvant à **d** positions dans l'ordre croissant de l'alphabet.

Exemple : si **chaine** contient **"abc"**, avec un décalage **d = 3**, on obtient le contenu **"def"** et avec le contenu **"xyz"** et un décalage **d = 1**, on obtient le contenu **"yza"** ('z' devient 'a' quand on décale d'une position).

Note : un décalage **d = 26** transforme la chaîne en elle-même.

Exemple

```

85 public static void main(String[] args) {
86     MaChaine chaine = new MaChaine("bravo les champions du java");
87     System.out.println("Le contenu de chaine est \"" + chaine.getChaine() + "\"");
88     int d = 3;
89     chaine.crypte(d);
90     System.out.println("Le contenu de chaine après cryptage avec d = " + d + " " +
91         " est \"" + chaine.getChaine() + "\"");
92 }

```

Problems @ Javadoc Declaration Console

<terminated> MaChaine [Java Application] C:\Program Files\Java\jdk1.7.0_60\bin\javaw.exe (9 oct. 2018 20:16:51)
 Le contenu de chaine est "bravo les champions du java"
 Le contenu de chaine après cryptage avec d = '3' est "eudyr ohv fkdpslrqv gx mdyd"

3.7. La méthode **void decrypte(int d)** qui modifie le contenu de la chaîne en restituant son contenu avant qu'elle ait été cryptée par **void crypte (int d)**.

Exemple

```

85 public static void main(String[] args) {
86     MaChaine chaine = new MaChaine("eudyr ohv fkdpslrqv gx mdyd");
87     System.out.println("Le contenu de chaine est \"" + chaine.getChaine() + "\"");
88     int d = 3;
89     chaine.decrypte(d);
90     System.out.println("Le contenu de chaine après décryptage avec d = " + d + " " +
91         " est \"" + chaine.getChaine() + "\"");
92 }

```

Problems @ Javadoc Declaration Console

<terminated> MaChaine [Java Application] C:\Program Files\Java\jdk1.7.0_60\bin\javaw.exe (9 oct. 2018 20:21:40)
 Le contenu de chaine est "eudyr ohv fkdpslrqv gx mdyd"
 Le contenu de chaine après décryptage avec d = '3' est "bravo les champions du java"