TP 1 : Les premiers pas en Java

L'objectif de ce TP est d'écrire, compiler et exécuter les premiers programmes Java.

ECRITURE	COMPILATION		EXECUTION	
Code	..... ▶	Codebyte	..... ▶	Fenêtre Dos
MaClasse.java	*javac MaClasse.java*	MaClasse.class	*java* MaClasse	Résultats affichés

EXERCICE 1 : classe Bonjour

1) Dans l'éditeur de texte «bloc note », saisir le code suivant de la classe Bonjour :

```
public class Bonjour
{
public static voidmain (String args[])
{
System.out.println("Bonjour Tout Le Monde !");
}}
```

2) Créer un nouveau répertoire et enregistrer le texte ci-dessus dans ce répertoire. Le fichier contenant ce texte porte le nom **Bonjour.java**.
3) Compiler votre fichier à l'aide de la commande **javac Bonjour.java**. Si tout va bien, vous obtenez le fichier **Bonjour.class** dans le même répertoire.
4) Exécuter le programme à l'aide de la commande **java Bonjour**.
5) Reprendre le même programme sous NetBeans. Compiler et Exécuter.

Quelques commentaires :

- ✓ **public class Bonjour :** est la **signature** de la classe.
- ✓ **public static void main (String args[]) :** est la **signature** de la méthode **main().**
- ✓ **public :** signifie que la classe Bonjour n'est pas protégée, elle pourrait être utilisée par une classe d'un autre paquetage.
- ✓ **void :** signifie que la méthode main() ne renvoie pas de valeur.
- ✓ **String args[] : args** est un paramètre de type tableau de chaînes de caractères qui permet de récupérer des arguments de lignes de commandes.

EXERCICE 2 : Tableau args

Soit le programme java suivant:

```
public class Array {
public static void main(String args[]) {
String jour_semaine[]=new String [7];
int jour=Integer.parseInt(args[0]) ; // Integer est une classe enveloppe
jour_semaine[0]="dimanche" ;
jour_semaine[1]= "lundi" ;
```

```
jour_semaine[2]= "mardi" ;
jour_semaine[3]= "mercredi" ;
jour_semaine[4]= "jeudi" ;
jour_semaine[5]= "vendredi" ;
jour_semaine[6]= "samedi" ;
System.out.println("Moi, je prefere le "+jour_semaine[jour]) ; }}
```

1) Commenter le programme
2) Que se passera-t-il si on lance les commandes en expliquant :
   java Array 2
   java Array 7

EXERCICE 3 : Premier programme

Écrire un programme qui calcule la somme des nombres entiers compris entre a et b (limites incluses). Les valeurs de a et b seront lues au clavier (utiliser la classe Scanner). Si a > b inverser le rôle des deux nombres.

Note sur la classe Scanner

Elle simplifie la lecture de données sur l'entrée standard (clavier) ou dans un fichier.
Pour utiliser la classe Scanner, il faut d'abord l'importer : import java.util.Scanner;
Ensuite il faut créer un objet de la classe Scanner : Scanner sc = new Scanner(System.in);
Pour récupérer les données, il faut faire appel sur l'objet sc aux méthodes décrites ci-après.
Ces méthodes parcourent la donnée suivante lue sur l'entrée et la retourne :

- String next() : donnée de type String qui forme un mot,
- String nextLine() donnée de type String qui forme une ligne,
- boolean nextBoolean() donnée booléenne,
- int nextInt() donnée entière de type int,
- double nextDouble() donnée réelle de type double.

➔ Il peut être utile de vérifier le type d'une donnée avant de la lire :

- boolean hasNext() renvoie true s'il y a une donnée à lire
- boolean hasNextLine() renvoie true s'il y a une ligne à lire
- boolean hasNextBoolean() renvoie true s'il y a un booléen à lire
- boolean hasNextInt() renvoie true s'il y a un entier à lire
- boolean hasNextDouble() renvoie true s'il y a un double à lire.

EXERCICE 4 : La classe String

Note sur la classe String

Elle décrit des objets qui contiennent une chaîne de caractères *constante* (cette chaîne ne peut pas être modifiée). La classe String possède de nombreuses méthodes, voici quelques exemples :

- ✓ **int length()** Retourne le nombre de caractères compris dans la chaîne.
- ✓ **int indexOf(char c, int i)** Retourne la position du caractère c en partant de la position i
- ✓ **String substring(int i, int j)** Retourne une chaîne extraite de la chaîne sur laquelle est appliquée cette méthode, en partant de la position i à la position j
- ✓ **boolean equals (String s)** comparaison sémantique des chaînes.

Méthodes de la classe String

1) Écrivez une méthode main dans la classe **TestChaine** qui crée une chaîne de caractères et qui affiche :
   - ✓ les 3 premiers caractères ;
   - ✓ les 3 derniers ;
   - ✓ le (ou les 2) caractère(s) placé(s) au milieu (un peu de calcul à faire...).
   - ✓ la chaîne en majuscules, puis en minuscules.
2) Testez l'égalité de 2 chaînes à la casse près. Par exemple, "FAIRE", "FaiRe" et "faire" devraient être égaux.
3) 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"

Eclatement d'une chaine de caractère avec la classe String

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("ali, sonia, wael, et les autres", ",") renverra un tableau de taille 4, dont les éléments sont "ali", " sonia" " wael" " 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-ci 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.

Si vous avez fini avant les autres ...

Ecrire une 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.