# Programmation avancée en Java

Atelier 1

#### anatomie d'un programme java.

```
Créer un dossier nommé votrenomTP1, ouvrir l'éditeur de texte et y saisir le code suivant:
```

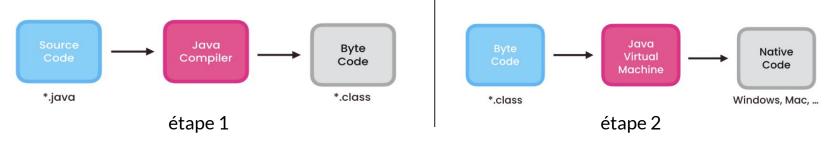
```
class Maclasse {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

enregistrer ce fichier en <u>prog1.java</u> puis dans la fenêtre Terminal exécuter ces 4 commandes:

- java -version
- javac prog1.java
- Is votrenomTP1 (ou dir si vous êtes sous windows)
- java Maclasse

Que remarquez vous?

### compilation et interprétation



Java Runtime Environment (JRE)

## les types de variable: primitive types

#### **Primitive Types**

Туре	Bytes	Range
byte	1	[-128, 127]
short	2	[-32K, 32K]
int	4	[-2B, 2B]
long	8	
float	4	
double	8	
char	2	A, B, C,
boolean		

```
byte i = 0;
short m = 14;
int x = 0;
long xl = 123456798L;
float reel = 10.5F;
double doublereel = 1235698712035.65598;
char letter = 'A';
boolean estriche = false; // prend false ou
true.
```

#### les types de variable: reference types

- commence par une majuscule: **String, Date..**
- déclaré avec new
- possède des méthodes propres: "hello".toUpperCase();
- nécessite généralement d'importer une bibliothèque: import java.util.Date;

```
// reference types
String phrase = "ma phrase est bien écrite";
Date now = new Date();
System.out.println(now);
System.out.println(phrase + "!!!");
System.out.println(now);
```

#### les tableaux: arrays (reference types)

• <u>déclaration</u>:

```
type[] name = new type[taille];
ou
type[] name = {2, 5, 6, -9, 4};
```

- est un type référence.
- a une taille fixe.
- activité à faire:taper ce code et déduire la fonction de chaque ligne.

```
int[] nombres = new int[5];
nombres[0] = 12;
nombres[1] = -9;
nombres[2] = 103;
System.out.println(nombres);
System.out.println(Arrays.toString(nombres));
System.out.println(nombres.length);
Arrays. sort (nombres);
System.out.println(Arrays.toString(nombres));
```

### Les matrices: 2D Arrays

```
int[][] nombres = new int[2][3];
nombres[0][0] = 12;
nombres[0][1] = -9;
nombres[1][0] = 103;
System.out.println(Arrays.toString(nombres));
System.out.println(Arrays.deepToString(nombres));
```

### Applications.

- 1. Écrire un programme qui affiche un tableau de 20 entiers contenant les valeurs de 0 à 19.
- 2. Ecrire un programme qui construit une matrice dont les éléments sont la somme des compteurs i+j:

```
exemple: matrix[1][3] = 1+3 = 4.
```

- 3. compléter le programme précédent pour afficher combien de 5 il y a dans la matrice.
- 4. soit double n1, n2, écrire un programme qui affiche le maximum entre ces deux réels..

#### lire des variables: Scanner.

```
String name; byte age;
   Scanner scanner = new Scanner(System.in);
   System.out.println("comment vous appelez vous ?");
  name = scanner.next();
   System.out.println("votre age?");
   age = scanner.nextByte();
   System.out.println("Bonjour " + name + " , vous
avez " + age + " ans");
```

- 1. executer ce programme avec juste un prénom
- 2. exécuter avec un nom complet, qu'est ce qui se passe?

sol: remplacer next par nextLine

#### les fonctions: exemples.

```
public class Main {
  public static void main(String[] args) {
      mabrouk("john", 2023);
      mabrouk("ali", 2023);
      mabrouk("hammadi", 2023);
  public static void mabrouk(String name, int annee) {
      System.out.println("bonne année" + annee + " " + name);
      System.out.println("à l'année prochaine\n----");
```

- 1. essayer ce programme et voir l'output.
- 2. réaliser une fonction qui affiche le factoriel d'un nombre donné.