

# TP 12 Programmes Java

Informations techniques PC Suse :

- Pour démarrer une session : utilisateur **licencep** et mot de passe **7002n\***. Vous trouverez :
  - un gestionnaire de fichiers en haut à gauche placé dans le dossier personnel HOME
  - une icône lézard  en haut à droite pour accéder au menu.
- Pour démarrer une *console* : clic sur l'icône lézard en haut à droite → Terminal → Konsole.
- Pour ouvrir un gestionnaire/navigateur de fichiers : clic sur l'icône lézard → Utilitaires → Dolphin.
- Pour modifier un fichier, clic droit sur le fichier → Ouvrir avec → Kate (ou autre éditeur de votre choix).

**Exercice 1** Démarrer un terminal et taper les commandes suivantes pour commencer à écrire des fichiers Java dans un dossier tp12. À la fin de la séance, vous allez pouvoir copier le dossier tp12 sur une clé USB.

```
cd                #se placer dans le dossier personnel
mkdir tp12        #créer un nouveau dossier
cd tp12           #se placer dans le nouveau dossier tp12
touch Exo1.java   #créer nouveau fichier
kate Exo1.java&  #editer le fichier, n'oublier pas le '&' sinon le terminal reste bloqué
```

Éditer le fichier Exo1.java pour écrire un programme Java qui affiche *Ciao Bella*. Pour des exemples, n'hésitez pas à regarder la première diapo du Cours 12 sur le Java, disponible sur le site web

[cedric.cnam.fr/~porumbed/vari1](http://cedric.cnam.fr/~porumbed/vari1)

Pour compiler et exécuter, il faut utiliser `javac` et `java`, voir les indications fournies dans les diapos du cours 12 ou au TP6, TP7, TP8, TP9 et TP10.

**Exercice 2** Modifier le programme précédent et écrire une classe Exo2 dans un fichier Exo2.java. Ajouter une fonction statique `puissance4(float x)` qui renvoie  $x^4 = x \times x \times x \times x$ . Appeler cette fonction dans le programme principal (c. à. d. dans la fonction `main(...)`) pour afficher  $2^4$ ,  $3^4$ ,  $4^4$  et  $5^4$ .

Vous pourriez avoir besoin de faire Kate afficher les numéros de lignes pour corriger les erreurs : taper **F11**, ou Configuration → Configurer Kate → Apparence → Onglet Bordures → Afficher les numéros de lignes.

**Exercice 3** Écrire une classe Exo3 dans un fichier Exo3.java. Définir une fonction statique

```
static double impots(double revenuFiscalRef)
```

qui renvoie le montant des impôts pour un revenu fiscal référence `revenuFiscalRef`. Faire fonctionner le code ci-après qui permet à l'utilisateur de saisir son revenu fiscal. Il s'agit d'un impôt progressif, par tranches, basé sur les règles (simplifiées) suivantes définies par la loi :

- Pas d'impôt pour la tranche 0 – 9710 euros.
- 14% pour la tranche de 9710 à 26818 euros
- 30% pour tout revenu supérieur à 26818 euros.

```
class Exo3{
    static double impots(double ...) {
        ...
    }
    public static void main(String [] args){
        java.util.Scanner scan = new java.util.Scanner(System.in);
        System.out.println(" Entrer votre revenu fiscal de référence:");
        double revenuNet = scan.nextDouble(); //lecture clavier
        double mesImpots = impots(revenuNet);
        System.out.println(" Vos impôts=" + mesImpots);
    }
}
```

Attention au coté progressif de l'impôt. Tester votre programme sur les revenus suivants :

- `impots(10000) = 0.14 × (10000 – 9710) = 40.6;`
- `impots(30000) = 0.14 × (26818 – 9710) + 0.3 × (30000 – 26818) = 3349.72.`

**Exercice 4** Écrire une classe `Exo4` dans un fichier `Exo4.java` pour afficher l'indice de masse corporelle (le poids divisé par la taille en mètres au carré) pour un poids entier et une taille entière saisies par l'utilisateur. Faire fonctionner le code ci-après.

**Note1 :** vous allez avoir besoin de transformer les `cm` en mètres, utiliser `double m=(double)cm/100` au début de la fonction `imc(...)`.

**Note2 :** pour saisir deux valeurs au clavier, il faut construire un objet `java.util.Scanner` une seule fois avec `new`.

```
class Exo4{
    static double imc(int kg, int cm){
        double m = ...
        ... ..
    }
    public ... void main(String [] args){
        int kg, cm;
        //faire l'utilisateur saisir kg,cm
        double indice=imc(kg,cm);
        S_....out.println("IMC="+indice);
        if(indice < 20)
            System....println("maigre");
        if(indice > 30)
            System....println("surpoids");
    }
}
```

**Exercice 5** Écrire une fonction

```
void trouveMinMax(double[] tabd)
```

qui enregistre la valeur minimale et la valeur maximale du tableau dans les variables globales `min` et `max`. Afficher `min` et `max` dans le programme principal pour `double[] t={1.5, 9,0.3,8, 12.4,13}`

**Exercice 6** Écrire une classe `Exo6` dans un fichier `Exo6.java` pour tester si un entier saisi par l'utilisateur appartient à un tableau. Le tableau est initialisé au début de la fonction `main()` ainsi :

```
int[] tab = {12, 15, 13, 10, 8, 9, 13, 14};
```

**Indication :** Utiliser une variable booléenne `valSaisieExiste` qui sera au début `false`. Mais si on trouve la valeur saisie, on met `valSaisieExiste=true`. Ainsi, à la dernière ligne du programme on vérifie si `valSaisieExiste==false`. Si c'est le cas, on affiche "La valeur saisie n'existe pas".

```
valSaisieExiste = false
for i = 0 to n-1
    if valSaisie=tab[i]
        valSaisieExiste =true
if valSaisieExiste then print "trouvé" else..
```

**Exercice 7** Écrire une fonction

```
static boolean premier(int n)
```

qui renvoie `true` si `n` est premier ou `false` sinon.

**Rappel :** `n` est premier si on ne trouve **aucun** nombre `i` dans la liste `2, 3, ... n - 1` qui divise `n`, c. à. d.  $n \% i \neq 0 \forall i \in \{2, 3, \dots n-1\}$ . Tester la fonction `premier(...)` sur les nombres `5, 19, 51` dans le `main(...)`.

**Exercice 8** Écrire un fonction

```
static int[] concatener(int[] a, int[] b)
```

qui concatène les tableau `a` et `b` qui peuvent éventuellement être vides. Le programme principal ci-après devrait afficher `[4,7,9,2,19,8,9]`.

```
int [] t = {4,7,9};
int [] u = {2,19,8,9};
int [] v = concatener(t,u);
System.out.println(
    java.util.Arrays.toString(v));
```

**Exercice 9** Écrire une fonction qui affiche une matrice (tableau de tableaux). Appeler cette fonction en lui passant une table de multiplication  $n \times n$  comme argument. Le programme saisit `n` et affiche par exemple pour `n = 4` une matrice **bien alignée** :

```
1 2 3 4          1 2 3 4
2 4 6 8          ET NON PAS: 2 4 6 8
3 6 9 10         3 6 9 10
4 8 10 16        4 8 10 16
```

**Exercice 10** Écrire un programme qui affiche un carré composé de caractères `*`, dont la taille est entrée par l'utilisateur. Par exemple, pour une taille `4`, ce sera :

```
****
****
****
****
```

**Exercice 11** Dans la lignée de l'exercice précédent, dessiner cette fois un triangle avec la pointe tournée vers le haut. Le nombre de lignes est fixé à `n = 4`, mais vous pouvez tester d'autres valeurs. Pour chaque ligne, il faudra afficher un certain nombre d'espaces sans saut de ligne (via `System.out.print(" ")`), puis un certain nombre d'étoiles, puis un saut de ligne (via `System.out.println("")`). Il faut déterminer le nombre d'espaces et le nombre d'étoiles pour chaque numéro de ligne  $i \in \{0, 1, \dots n-1\}$  (ex,  $n-i$  espaces ?)

```
*          #//3 espaces, 1 étoile
***        #//2 espaces, 3 étoiles
*****    #//1 espace , 5 étoiles
*****    #//0 espaces, 7 étoiles
```