

## Travaux pratiques N° 1

### Introduction et éléments de base du Java (Méthodes statiques et implémentations algorithmiques)

#### Exercice 1 :

Ecrire un programme permettant d'appliquer une opération arithmétique sur deux valeurs numériques. L'opérateur ainsi que les valeurs doivent être saisis à partir de la console en suivant le format suivant :

```
java Exemple operateur valeur1 valeur2
```

Un minimum de quatre opérateurs doit être fourni : add (addition), sous (soustraction), mul (multiplication) et div (division) (utiliser le nouveau switch sous forme d'une expression à renvoyer).

#### Exercice 2 :

Ecrire un programme permettant de lire la largeur et la hauteur d'un rectangle.

Ce programme devra afficher, à la demande, le périmètre ou la surface du rectangle.

Le programme devra être bien modularisé de sorte à ce que la méthode main se présente comme suit :

```
public static void main(String args[]) {  
    double largeur = lireDonnee("largeur");  
    double hauteur = lireDonnee("hauteur");  
    boolean donneesOk = testerDonnees(largeur, hauteur);  
    if (donneesOk) {  
        calculer(largeur, hauteur);  
    } else {  
        afficherErreur();  
    }  
}
```

Exemples d'exécutions:

```
Entrez la largeur: 15  
Entrez la hauteur: 13,8  
Surface ('s/S') ou périmètre ('p/P')?: s  
La surface est 207.0  
ou encore:  
Entrez la largeur: 9  
Entrez la hauteur: -4  
Erreur: vous avez introduit une largeur ou une hauteur négative!
```

#### Exercice 3 :

Écrire un programme permettant d'évaluer un polynôme du 3ème degré de la forme :

$$\left(\frac{a+b}{2}\right)x^3 + (a+b)^2 x^2 + a + b + c$$

Si  $(a+b)=0$  il faut afficher un message d'erreur.

Exemple d'exécution:

```
Entrez a (int) : 1  
Entrez b (int) : 2  
Entrez c (int) : 3  
Entrez x (double) : 3.5  
La valeur du polynôme est : 180.5625
```

#### Exercice 4 :

Ecrire un programme qui permet de lire un entier positif **N** et de déterminer les nombres premiers inférieurs à **N**.

Utiliser une méthode booléenne "premier" qui retourne **Vrai** si le nombre passé en paramètre est premier.

#### Exercice 5 :

Ecrire un programme permettant de calculer le factoriel d'un entier saisi au clavier en utilisant une méthode récursive puis une méthode itérative.

#### Exercice 6 :

Deux entiers sont dits amiables si chacun d'eux est égal à la somme des diviseurs de l'autre (par exemple 220 et 284 sont amiables).

Ecrire un programme qui permet de lire un entier positif **N** et de déterminer et afficher toutes les paires de nombres amiables inférieurs à **N**.

Utiliser :

- Une méthode "sommeDiviseurs" qui retourne la somme des diviseurs d'un entier passé en paramètre.
- Une méthode booléenne "amiable" qui reçoit deux entiers et qui retourne **Vrai** si les deux entiers sont amiables ; **Faux** sinon.

#### Exercice 7 :

Écrire un programme récursif et itératif qui calcule le  $n^{\text{ième}}$  terme de la suite de Fibonacci :

$$F_0 = 0 ; F_1 = 1 ;$$

$$F_n = F_{n-1} + F_{n-2} \quad \text{pour } n \geq 2$$

### Exercice 8 :

Les égyptiens de l'antiquité savaient :

- additionner deux entiers strictement positifs.
- soustraire 1 à un entier strictement positif.
- multiplier par 1 et 2 tout entier strictement positif.
- diviser par 2 un entier strictement positif pair.

Ils se basent sur ces opérations pour calculer le produit de deux entiers strictement positifs

Voici un exemple de calcul du produit  $14 \times 13$ , en utilisant uniquement ces opérations :

$$\begin{aligned}
 14 \times 13 &= 14 + 14 \times (13 - 1) = 14 + 14 \times 12 \\
 &= 14 + (14 \times 2) \times (12 / 2) = 14 + 28 \times 6 \\
 &= 14 + (28 \times 2) \times (6 / 2) = 14 + 56 \times 3 \\
 &= 14 + 56 + 56 \times (3 - 1) = 70 + 56 \times 2 \\
 &= 70 + (56 \times 2) \times (2 / 2) = 70 + 112 \times 1 \\
 &= 70 + 112 = 182
 \end{aligned}$$

Donner le corps de la méthode **multiplicationEgyptienne** qui calcule le produit de a par b.

### Exercice 9 :

La suite de **Syracuse** est définie selon une condition de parité comme suit :

$$u_0 \in \mathbb{N}^*, \quad u_{n+1} = \begin{cases} u_n / 2 & \text{si } u_n \text{ est pair} \\ 3u_n + 1 & \text{si } u_n \text{ est impair} \end{cases}$$

La « conjecture tchèque » énonce que pour toute valeur initiale  $u_0 \in \mathbb{N}^*$  il existe un rang **n** pour lequel  $u_n = 1$

Par exemple, si  $u_0 = 6$  alors **n** = 8

n	0	1	2	3	4	5	6	7	8	9	10	...
$u_n$	6	3	10	5	16	8	4	2	1	4	2	....

Ecrire un programme qui demande à l'utilisateur la saisie de la valeur initiale  $u_0$  et qui détermine et affiche la plus petite valeur de **n** vérifiant  $u_n = 1$ .

### Exercice 10 :

Soit la suite  $(X_n)_{n \in \mathbb{N}}$  suivante

$$\begin{cases} X_0 = A \\ X_n = \left( X_{n-1} + \frac{A}{X_{n-1}} \right) / 2 \quad n \geq 1 \end{cases}$$

A est un nombre réel positif.

- Implémenter la suite suivante en utilisant les deux méthodes
  - La première méthode est récursive.
  - La deuxième est itérative.
- Que calculent ces méthodes ?
- Le point d'arrêt des itérations est  $|X_n - X_{n-1}| < 10^{-9}$

### Exercice 11 :

On considère l'ensemble  $H_a$  suivant :

$$H_a = \{n \in \mathbb{N} / 2^n > a\} ; a \in \mathbb{N}$$

- 1- Ecrire le programme de la méthode **minEnsemble** qui permet de déterminer le minimum de l'ensemble  $H_a$  :

```
Public static int minEnsemble (int a)
```

- 2- Utiliser le résultat de la méthode **minEnsemble** pour écrire le programme de la méthode **decimalBinaire** qui permet de convertir un entier de la base décimale à la base binaire (le résultat renvoyé est stocké dans un tableau) :

```
Public static int [] decimalBinaire (int a)
```

### Exercice 12 :

Ecrire un programme Java basé sur une méthode récursive appelée : « **inverserTableau** » qui permet de réarranger les éléments d'un tableau en ordre inverse.

### Exercice 13 :

Ecrire un algorithme et le programme correspondant en langage Java qui permet :

- d'additionner deux matrices.
- de multiplier une matrice par un réel.
- de déterminer la transposée d'une matrice.
- de multiplier deux matrices.