TP 14 Programmes Java

Avant de démarrer, taper dans un terminal les commandes Linux qui permettent de créer un dossier `tp14`. Vous allez pouvoir écrire des fichiers `Java` dans ce dossier. À la fin de la séance, vous allez pouvoir copier le dossier `tp14` sur une clé USB. Pour les commandes Linux à taper, voir le début du TP12 ou TP13.

Exercice 1 Corriger les 2 lignes indiquées dans le code ci-après pour le faire afficher la valeur minimale de x.

```
class Exo1{
    public static void main(String[] args){
        int[] x = {5, 7, 3, 14, 21, 83, 9, -4, -3, 12};
        int min = 0;
        for(int i=0;i<x.length;i++)
            if(tab[i]<min) min = tab[i];
        System.out.println("Min="+min);
    }
}
```

Exercice 2 Corriger 3 lignes dans le code ci-après pour le faire afficher la valeur minimale du tableau.

```
class Exo2{
    public static void main(String[] args){
        double[] x = {5, 7, 3, 14, 21, 83, 9, -4.2, -3, 12};
        int double min = x[0];
        for(int i=0;i<x.length;i++);
            if(x[i]<min)
                x[i] = min;
        System.out.println("Min="+min);
    }
}
```

Exercice 3 Le jeu *Fizz buzz* a été utilisé par des enfants pour apprendre la division. Les enfants sont rangés en cercle et chaque enfant parle après avoir écouté son voisin à gauche. En commençant par 1, chaque enfant dit un nombre qui augmente le nombre précédent d'une unité. En fait, si un enfant arrive à un multiple de 3, **il doit dire** « `Fizz` » **au lieu de dire le nombre**. S'il arrive à un multiple de 5, il doit dire « `Buzz` ». Pour un multiple de 15, il faut dire « `Fizz Buzz` ». Voici un exemple de jeu :

`1, 2, Fizz, 4, Buzz, Fizz, 7, 8, Fizz, Buzz, 11, Fizz, 13, 14, Fizz Buzz, 16, 17, Fizz, 19, Buzz`

Ce jeu a été souvent utilisé dans des entretiens d'embauche dans le monde anglo-saxon. Il est facile d'afficher les 100 premiers tours du jeu. Cela permet d'éliminer les candidats avec des compétences insuffisantes en programmation. [1]

Écrire un programme `Exo3.java` qui permet de **générer et afficher les 100 premiers tours du jeu**. Pour savoir si un entier x est divisible par 3, on utilise l'opérateur %, ex, $(x\%3)$ vaut 0 pour $x \in \{3, 6, 9, 12, 15, \ldots\}$.

Exercice 4 Écrire un programme `Exo4.java` qui doit demander à l'utilisateur de saisir le nom d'un fichier (utiliser un objet `java.util.Scanner` et la méthode `nextLine()`, voir le Cours 14 ou le TP 13). Le programme devrait faire la somme des 3 nombres situées sur les 3 premiers lignes du fichier indiqué. Vous allez avoir besoin d'un objet de classe `java.io.BufferedReader`, voir le Cours 14 ou l'exercice 7 du TP 13.

1. L'article Wikipedia dit : "Fizz buzz has been used as an interview screening device for computer programmers. Creating a list of the first 100 Fizz buzz numbers is a trivial problem for any would-be computer programmer, so interviewers can easily sort out those with insufficient programming ability.", voir `en.wikipedia.org/wiki/Fizz_buzz`

Exercice 5 Écrire une fonction

```
static double f(double x)
```

qui renvoie $\sqrt{6x}$, n'hésitez pas à utiliser `Math.sqrt(...)`. Écrire une autre fonction

```
static double g(int n)
```

qui renvoie la valeur

$$\frac{1}{1^2}+\frac{1}{2^2}+\frac{1}{3^2}+\frac{1}{4^2}+\cdots\frac{1}{n^2}$$

Afficher `f(g(20000))` dans le programme principal. Vous obtenez le plus fameux nombre en mathématiques ?

Exercice 6 Écrire une fonction

```
static double ff(double x)
```

qui renvoie $\sqrt{8x}$. Écrire une deuxième fonction

```
static double gg(int n)
```

qui renvoie la valeur

$$\frac{1}{1^2}+\frac{1}{3^2}+\frac{1}{5^2}+\frac{1}{7^2}+\cdots\frac{1}{(2n+1)^2}$$

Afficher `ff(gg(20000))` dans le programme principal. Vous obtenez le même nombre ?

Exercice 7 Afficher la valeur suivante pour $n = 1000$.

$$\frac{4}{1}-\frac{4}{3}+\frac{4}{5}-\frac{4}{7}+\frac{4}{9}-\frac{4}{11}\cdots.$$

Quel programme (Exo 5, Exo 6 ou Exo 7) trouve la valeur la plus exacte de π pour $n = 20000$?

Exercice 8 On appelle factorielle de n le produit

$$n! = 1\times 2\times 3\times 4\cdots\times n.$$

Le morceau de code suivant calcule $0! = 1$, $1! = 1$, $2! = 2$, $3! = 6$, $4! = 24$, $5! = 120$, $6! = 720$, ... $10!$ et stocke ces valeurs dans le tableau `fact`.

```
int n       = 10;
int[] fact = new int[n];
for(int p=0;p<n;p++){
    int factVal = 1;
    for(int j=1;j<=p;j++)
        factVal = factVal * j;
    fact[p] = factVal;
}
System.out.println(
    java.util.Arrays.toString(fact));
```

Attention : Pour chaque valeur de p, le programme calcule $p-1$ multiplications, voir ligne jaune. Pour arriver à $p = 9$, le programme fait 1+2+3+4+5+6+7+8 = 36 multiplications.

Modifier le programme pour le faire calculer le tableau avec une seule multiplication pour chaque valeur de p. L'idée est de démarrer avec `fact[0]=0` et d'utiliser `fact[p] = fact[p-1]*p` pour chaque valeur de $p = 2, 3, \ldots n-1$.

Exercice 9 Continuer le programme précédent. Après avoir rempli le tableau `fact`, calculer :

$$S = \frac{1}{0!}+\frac{1}{1!}+\frac{1}{2!}+\frac{1}{3!}+\cdots+\frac{1}{n!}$$

Afficher cette valeur pour $n = 14$. Vous trouverez la nombre mathématique e.

Exercice 10 Utiliser la fonction

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

de l'exercice 7 du TP 12 (voir la correction) pour afficher tous les nombres premiers inférieurs à $n = 100$.

Exercice 11 Le crible d'Ératosthène est un procédé qui permet de trouver tous les nombres premiers inférieurs à un certain n. Il procède par élimination : il s'agit de supprimer d'une table des entiers de 2 à n tous les multiples d'un entier. En supprimant tous les multiples, à la fin il ne restera que les entiers qui ne sont multiples d'aucun entier, et qui sont donc les nombres premiers. On commence par rayer les multiples de $p = 2$. Pour illustration, pour "rayer" les multiples de $p = 2$, on pourrait utiliser un code comme :

```
p=2
int multiple_p=p+p
while (multiple_p<n){
    tab[multiple_p]= 0
    multiple_p=multiple_p+p
}
```

Ensuite, à chaque fois, on raye les multiples du plus petit entier restant. Après $p = 2$, on passe à $p = 3$, puis à $p = 5$. **Utiliser cet algorithme pour afficher tous les nombres premiers inférieurs à $n = 100$.**

Exercice 12 Finir les 2 derniers exercices du TP 13.

Exercice 13 Écrire un programme qui lit 9 valeurs entières stockées sur 9 lignes d'un fichier. Ces entiers représentent les valeurs d'une matrice 3×3. Déterminer si cette matrice représente un carré magique, c. à. d. de même somme sur chaque ligne et chaque colonne (chercher plus de détails sur carré magique Wikipédia).

Modifier le programme pour lire un fichier qui fournit que 4 valeurs : `m[0][0]`, `m[0][1]`, `m[0][2]` et `m[1][0]`. Déterminer les 5 autres cases de m pour construire un carré magique (avec que des valeurs positives).