Feuille de TD n°1

Exercice 1. Évaluation d'expressions arithmétiques

On considère les déclarations suivantes :

```
byte b1 = 5, b2 = 30 ;
short s = 300 ;
int n = 600 ;
long q = 1500 ;
float x = 1.5E2f ;
double y = 5.0E3 ;
```

Donner le type et la valeur des expressions arithmétiques suivantes :

```
b1 + b2
s + b1
q + s * (b1 + b2)
x + q * n
b1 * q/x
b1 * q * 2. /x
b1 * q * 2.f /y
```

Exercice 2. Quelles valeurs contiennent les variables `n`, `p` et `q` après évaluation de chaque expression ?

```
int n = 3, p = 5, q = 7;
n = p = q = 10;
n += p += q;
q = n < p ? n++ : p++;
q = n < p ? ++n : ++p;
q = n > p ? n++ : p++;
n = q == p ? ++p : --q;
n = n == q ? p++ : --q;
n = ( p==n+1)||(q==n+1)? p%2 : q%2;
```

Écrire l'expression `q = n < p ? n++ : p++` avec un `if-else` et sans l'opérateur d'incrémentation.

Écrire l'expression `q = n < p ? ++n : ++p` avec un `if-else` et sans l'opérateur d'incrémentation.

Exercice 3. Ecrire le code permettant de tester si un entier `n` donné est pair et qui affiche un message indiquant le résultat : "l'entier est pair" ou "l'entier est impair".
Rappel : Pour afficher un message, on utilise `System.out.println("leMessage");`

Exercice 4. Ecrire un code permettant de vérifier si une année `a` donnée est bissextile. Une année est bissextile si :

•c'est un multiple de 4 mais pas de 100 ou
•c'est un multiple de 400

(Autre version : Si `a` n'est pas un multiple de 4, l'année n'est pas bissextile. Si `a` est un multiple de 4, l'année est bissextile sauf si `a` est divisible par 100 et pas par 400.)

Exercice 5. Ecrire un code permettant de retourner le plus petit de trois entiers donnés.

Exercice 6. On considère le code ci-dessous. Quel résultat obtient-on?

```
int i=10, j=5 ;
if (i<5 && j++<6) {
     System.out.println ("Premier && -> vrai") ;
}
else {
      System.out.println ("Premier && -> faux") ;
}
System.out.println ("i = " + i + " j = " + j) ;
if (i<15 && j++<6) {
     System.out.println ("Deuxième && -> vrai") ;
}
else {
     System.out.println ("Deuxième && -> faux") ;
}
System.out.println ("i = " + i + " j = " + j) ;
if (i<15 || j++<6) {
     System.out.println ("|| vrai") ;
}
else {
     System.out.println ("|| faux") ;
}
System.out.println ("i = " + i + " j = " + j) ;
```

Exercice 7. Traduction Python vers Java

On considère le code Python suivant qui prend en entrée deux valeurs positives `nb1` et `nb2` correspondant à une note sous la forme d'un nombre de points. Par exemple, 45 points sur 70 ou 25 points sur 50. Décrire ce que fait le programme et réécrire le code en Java.

```
note=nb1/nb2
if note>=0.8:
    print("Excellent, vous avez eu un A")
elif note>=0.6:
    print("Bien, vous avez eu un B")
elif note>=0.5:
    print("Moyen, vous avez eu un C")
elif note>=0.4:
    print("Passable, vous avez eu un D")
else:
    print("Insuffisant, vous avez eu un E")
```

Exercice 8. Sélecteur `switch`

On considère un entier `n` dont la valeur sera lue au clavier. Quel sera l'affichage à l'issue du switch

pour chacune des valeurs saisies suivantes: 0, 1, 2, 3, 4, 10, -5 ?

```
switch (n) {
     case 0 : System.out.print ("Bonjour, ");
     case 1 :
     case 2 : System.out.println ("ça va?");
              break;
     case 3 : break;
     case 4 :
     case 5 : System.out.println ("Hello!");
     default : System.out.println ("Au revoir");
}
```

Exercice 9. Instructions itératives

a) DivisionEuclidienne

On se donne deux entiers positifs `a` et `b, b‡0.`

Ecrire le code permettant d'effectuer la division euclidienne de `a` par `b` en utilisant l'algorithme suivant :

```
int q = 0, r = a;
Tant que r >= b
     remplacer q par q + 1
     remplacer r par r - b
```

En sortie, `q` est le quotient et `r` le reste de la division euclidienne de `a` par `b`.

Afficher `q` et `r`.

b) Calcul de la valeur d'une série

On se donne un entier positif `n.`

Ecrire le code permettant de calculer la somme des `n` premiers termes de la série harmonique : `1 + 1/2 + 1/3 + ... + 1/n.`

Le résultat sera stocké dans une variable de type `float`.

Exercice 10.

```
public class Test1Tableau {

     public static void main(String[] args) {
          int[] t1 = {3, 7, 24, 30} ;
          int[] t2= new int[10];
          for ( int i = 0; i<10; i++) {
               t2[i]=5*i;
          }
          t2 = t1;
          for ( int i = 0; i<10; i++) {
               System.out.println(t2[i]);
          }
     }

}
```

Qu'obtient-on lorsqu'on exécute ce programme ?

Exercice 11.

```
public class Test2Tableau {

     public static void main(String[] args) {
```

```
        int[][] t = new int[3][];
        for ( int i = 0; i<3; i++) {
             t[i] = new int[i+1];
             for ( int j = 0; j<t[i].length; j++)
                  t[i][j] = i+j;
        }
        for ( int i = 0; i<3; i++) {
             for ( int j = 0; j<t[i].length; j++)
                  System.out.print(t[i][j]);
              System.out.println( );
        }
    }
}
```

Qu'obtient-on lorsqu'on exécute ce programme ?

Exercice 12. On considère un tableau rempli avec les notes d'une classe :
`double[] lesNotes = {12.5, 15, 8, 20, 19, 13.5, 11.5, 5};`
•Ecrire le code permettant de calculer la moyenne de ces notes.
•Ecrire le code permettant de comptabiliser le nombre de notes supérieures ou égales à la moyenne.
Rappel : `lesNotes.length` permet de connaître le nombre de cases du tableau.

Exercice 13. On considère un tableau d'entiers : `int[] tab.`
Ecrire le code permettant de tester si ce tableau est un palindrome, c'est à dire qu'il se lit de la même manière dans les deux sens.

Exercice 14. Tri par sélection
On considère un tableau d'entiers : `int[] tab.`
•Ecrire le code permettant de rechercher le premier indice `min` du plus petit élément du tableau.
•Compléter le code précédent sachant que pour effectuer un tri par sélection, on recherche l'indice `min` du plus petit élément du tableau puis on échange tab[0] avec tab[min] si min est différent de 0. On recommence ces deux opérations avec les éléments du tableau à partir de la l'indice `1`. Et ainsi de suite.

Exercice 15. Tri par insertion
On considère un tableau d'entiers : `int[] tab.`
•Ecrire le code permettant d'effectuer un tri par insertion. Le tri par insertion procède de la même façon qu'un joueur de cartes pour trier ses cartes. On prend une carte, puis 2 et on les met dans l'ordre si nécessaire, puis 3 et on met la 3ème carte à sa place dans les 2 premières, ... De manière générale, on suppose les i−1 premières cartes triées. On prend la i ème carte, et on essaie de la mettre à sa place dans les i − 1 cartes déjà triées.
Pour classer le i ème élément du tableau `tab`, on regarde successivement en marche arrière à partir du i − 1 ième. On décale les éléments visités vers la droite pour pouvoir mettre tab[i] à sa juste place.