VII. Les tableaux

- En Java, ce sont des objets ;
- Il faut donc utiliser des méthodes pour y accéder ;
- Le 1er élément a pour indice 0 ;

1. Déclaration

- type nom-du-tableau [ ] ;

  type [ ] nom-du-tableau ;

- Les éléments d'un tableau peuvent être de type primitif ou de type objet ;
- Exemples : int tab[ ] ;
  Point tabPoints[ ] ;

On crée un tableau comme on crée un objet : avec l'opérateur "new"...

- int tab[] ;
  tab = new int [5] ;

- On peut exécuter les 2 instructions en 1 fois :

  int tab[ ] = new int [5] ;

2. Déclaration avec initialisation

- int tab1[] = {1, 8, 5, 6} ;
- int tab2[] = {n, n+2} ;
- char tab3[] = {'a', 'b', 'c'} ;

La notation {...} ne peut être utilisée que lors de la déclaration, par la suite il faut initialiser les éléments un par un...

3. Utilisation d'un tableau

```
tab[0] = 15 ;

tab[2]++ ;

System.out.print (tab[4]) ;
```

Remarque : Si la valeur de l'indice est incorrecte, on obtient une erreur d'exécution ;

4. La taille d'un tableau

- La variable 'length' retourne le nombre d'éléments d'un tableau ;
- Exemple : Parcours d'un tableau

```
for (i = 0 ; i < tableau.length ; i++)  { ... }
```

- Remarque : on écrit tableau.length sans parenthèses car length est un champs public de l'objet tableau et non une méthode...

Partie labo : Rappel concernant la Notation UML

Nom de la classe
Attributs classés par visibilité (public +, protected #, private -)
Méthodes classées par visibilité (public +, protected #, private -)

Syntaxe d'un attribut :

visibilité nom : type = valeur_initiale (facultatif)

Syntaxe d'une méthode :

visibilité nom (paramètres): type_retour

Remarques sur le passage des arguments

a) Lorsqu'on transmet une variable de type primitif en argument, on envoie en fait une copie de la valeur de la variable (=passage par valeur) :

```
public class TransmPrimitif

   public static void main(String[] args)
   {  int i = 5 ;
      modifVariable(i) ;
      System.out.println("Après la procédure : " + i) ;
   }
   static void modifVariable(int k)
   {  System.out.println("Avant la modif : " + k) ;
       k = 10 ;
      System.out.println("Après la modif : " + k) ;
   }
```

L'affichage donnera :

Avant la modif : 5

Après la modif : 10

Après la procédure : 5

b) Lorsqu'on transmet un tableau en argument, on envoie la référence du tableau (= son adresse), la méthode agit alors directement sur le tableau concerné et non sur une copie (=passage par référence) ...

```
public class TransmReference
{ public static void main(String[] args)
  {
    int t[] = {4,8} ;
    modifTab(t);
    System.out.println(t[0]) ;
  }
  static void modifTab (int tab[])
  { tab[0] = 3 ;
    System.out.println(tab[0]) ;
  }
}
```

5. Exercices

Que donne l'exécution du code suivant ?

```
public class Affectation
{   public static void main (String args[])
    {
        int i ;
        int t1[] = {1, 2, 3} ;
        int t2[] = new int[5] ;
        for (i=0 ; i<5 ; i++ ) t2[i] = 2*i ;
        t2 = t1 ;
        for (i=0 ; i<t2.length ; i++) System.out.println (t2[i]) ;
    }
}
```


Quelles sont les erreurs commises ici ?

```
public class Erreurs
{
    public static void main (String args[])
    {
        final int n = 10 ;
        int t1[] = { 1, 3, 5 } ;
        int t2[] ; ?? = new int [3]
        t2 = { 5, 8, 3 };
        float t3[] = { 2, 3.5, 4 } ;
        double t4[] = { 2, 3.5 , 4, n++ } ;
    }
}
```