Affichage d'un tableaux de taille fixe

Le code suivant :

```
double[] t1 = {1.1, 2.2, 3.4};
System.out.println(t1);
```

affiche la référence au tableau `t1`, donc une adresse.

Si l'on veut faire afficher les entrées du tableau référencé par `t1`, il faut prévoir une boucle (itération) !

Accès aux éléments d'un tableau (1)

Très souvent, on voudra accéder aux éléments d'un tableau en effectuant une *itération* sur ce tableau.

Il existe en fait au moins *trois* façons d'itérer sur un tableau :

- avec les itérations sur ensemble de valeurs

  `for(Type element : tableau)`

  ☞ *Type* est le type des éléments du tableau
- avec une itération `for` « classique » :

  `for(int i=0; i < TAILLE; ++i)`

  ☞ *TAILLE* ? ? voir plus loin
- avec des itérateurs (non présenté dans ce cours)

Accès aux éléments d'un tableau (2)

Attention, les itérations sur ensemble de valeurs :

`for(Type element : tableau)`

est très simple et élégant mais :

- ne permet pas modifier le contenu du tableau
- ne permet d'itérer que sur un seul tableau à la fois : il n'est pas possible de traverser en une passe deux tableaux pour les comparer par exemple
- ne permet l'accès qu'à un seul élément : on ne peut pas par exemple comparer un élément du tableau et son suivant
- itère d'un pas en avant seulement.

Nombre d'éléments d'un tableau

Pour connaître la **taille d'un tableau** :

- `nomTableau.length`

Exemple :

```
int[] scores = {1000, 1500, 2490, 6450};
System.out.println(scores.length); // 4
boolean[] bs = {true, false};
System.out.println(bs.length);     // 2
```

Attention ! `length` donne le **nombre possible** d'éléments. Le remplissage effectif du tableau n'a pas d'importance !

Exemple :

```
int[] scores = new int[2];
System.out.println(scores.length); // 2
```

Erreurs courantes avec les tableaux

1. Problème d'indice
2. Accès avant la construction du tableau

(3.) Accès à un élément non initialisé (objets, valeur `null`)

(4.) Confusion syntaxique tableau/objet

Les deux derniers types d'erreurs seront exposés après introduction de la notion d'objet

Erreur : problème d'indice

Attention !

- L'indice est toujours un `int`
- Il faut respecter les bornes du tableau :
  - ☞ Toujours énumération de `[0]` à `[T-1]` (où `T` est la taille du tableau)

Exemples :

```
int[] entiers = new int[250];
entiers[1.0] = 1; // erreur type
entiers[-13] = 2; // erreur borne
entiers[250] = 4; // erreur borne
```

Erreur : accès avant construction

Il est impossible en Java d'accéder à un élément si le tableau n'a pas encore été construit.

La construction se fait :

- Soit en indiquant les valeurs directement dans l'instruction de déclaration
- Soit en spécifiant la taille avec `new type[taille]`

Exemple :

```
int[] entiers1 = {1, 2, 3}; // Déclaration-initialisation
entiers1[0] = 4;            // OK
int[] entiers2;             // Déclaration
entiers2[0] = 4;            // Erreur !
```

Erreur : accès avant construction

Il est impossible en Java d'accéder à un élément si le tableau n'a pas encore été construit.

La construction se fait :

- Soit en indiquant les valeurs directement dans l'instruction de déclaration
- Soit en spécifiant la taille avec `new type[taille]`

Exemple :

```
int[] entiers1 = {1, 2, 3}; // Déclaration-initialisation
entiers1[0] = 4;            // OK
int[] entiers2;             // Déclaration
entiers2 = new int[10];     // Initialisation
entiers2[0] = 4;
```