Cours Introduction à la Programmation
Java IV
(IP1 Java)

Arnaud Sangnier
sangnier@irif.fr

Mercredi 25 Octobre 2017
INFO et MATHS-INFO

Dans les épisodes précédents

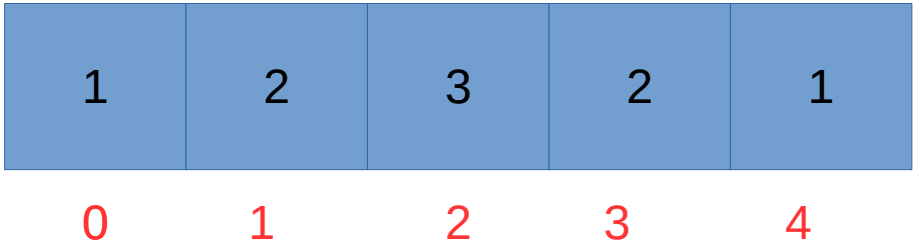
- Présentation de certains aspects de Java
- Ce que nous avons vu :
  - Les données et leur type : int , String et boolean
  - Les variables : affectation, lecture, modification
  - Toutes les instructions :
    - manipulation de variables
    - boucles
    - tests
    - appel de fonctions
  - Définition de fonctions

Stocker des données d'une autre façon

- Imaginons qu'un programme veuille calculer la moyenne d'age d'une population de 1000 personnnes
- Pour cela il pourrait utiliser une fonction moyenne qui prendrait en paramètres 1000 entiers et renverrait la moyenne
- Deux questions :
  - Combien de variables faudrait-il ? Du coup 1000
  - Est ce que la fonction qui prendrait en paramètres 5000 entiers et renverrait la moyenne serait très différente ? Non
- On peut résoudre ces problèmes en utilisant une liste pour stocker les données

Les tableaux

- Un tableau peut être vue comme un ensemble de cases mémoires consécutives contenant des données
- Par exemple : int[] t = {1, 2, 3, 2, 1} est un tableau que l'on pourrait représenter sous la forme



- 0, 1, 2,3 et 4 sont les indices du tableau
- C'est comme si on avait 5 variables t[0], t[1], t[2],t[3] et t[4]

Manipulation de tableaux

- Pour créer un tableau :
  - On peut faire int [] t = {5,6,7} ;
  - Mais aussi int[] t2 = new int[1000] ← Crée un tableau de taille 1000
  - Attention avec int[] t2 = new int[1000] , on ne sait pas les valeurs contenues dans le tableau
- Pour modifier la i-ème case d'un tableau t, on peut faire t[i]=3
- Attention si i dépasse **la taille du tableau -1** on a un problème
- Taille du tableau : nombre de cases du tableau
- Les indices vont de 0 à la taille du tableau -1
- La taille d'un tableau t est donné par t.length

Les tableaux → un nouveau type

- Quand on crée un tableau int[] t = {5,6,7}, on obtient un nouveau type de données → int[]
- Qu'est ce que cela implique :
  - On a des variables indiquant des tableaux
  - On peut donner des tableaux en argument de fonctions
  - Les fonctions peuvent retourner des tableaux
- De plus, les éléments dans un tableaux peuvent être des données d'autres types
- Donc, on peut avoir:
  - Des tableaux d'entiers (int[])
  - Des tableaux de chaînes de caractères (String[])
  - Des tableaux de booléens (boolean[])
  - mais aussi des tableaux de tableaux d'entiers (int[][])
  - des tableaux de tableaux de tableaux d'entiers (int[][][])
  - Etc
- **Les données d'un même tableau sont d'un même type**

Des concepts importants sur les tableaux

- On va vouloir créer des méthodes qui marchent pour des tableaux dont on ne connaît pas la taille
- Pour cela le .length qui donne la taille d'un tableau sera très utile
- Par exemple:
  - Afficher tous les éléments d'un tableau
  - Chercher le plus petit élément d'un tableau
  - Chercher un élément d'un tableau
  - etc
- L'utilisation de **boucles** pour parcourir le tableau sera nécessaire

Parcours de tableaux

- L'idée d'un parcours de tableaux et de faire un programme qui va voir les éléments du tableau un par un
- Par exemple,
  - si on a un tableau indiquée par une variable tab
  - on sait que la taille du tableau est tab.length
  - On va d'abord regarder l'élément tab[0], puis tab[1], puis tab[2],…,jusqu'à tab[tab.length-1]

```
for (int i=0;i<tab.length;i=i+1) {
  System.out.println(tab[i]);
}
```

Première fonction avec tableaux

- Écrire une procédure affiche qui affiche ligne par ligne les éléments d'un tableau d'entiers
  - Cette fonction prend comme argument un tableau d'entiers, appelons-le tab
  - Elle ne renvoie rien
  - Elle ne fait qu'afficher

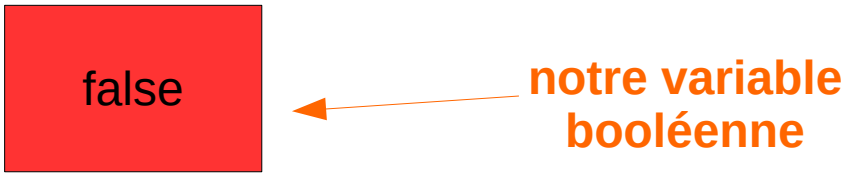
```
public static void affiche(int [] tab) {
  for (int i=0;i<tab.length;i=i+1) {
    System.out.println(tab[i]);
  }
}
```

Recherche d'éléments

- On veut chercher si un élément, par exemple un entier stocké dans une variable a est dans un tableau tab
- Comment faire ?
  - Il faut parcourir le tableau
  - Tester si un des tab[i] pour i allant de 0 à tab.length-1 est égal à a
  - Mais que fait-on si c'est égal ou différent
    - Si c'est différent, il faut continuer le parcours
    - Si c'est égal, on peut continuer le parcours mais il faut se rappeler que l'on a vu l'élément, comment faire ?
    - On va utiliser une variable booléenne qui sera fausse jusqu'à ce que l'on rencontre l'élément

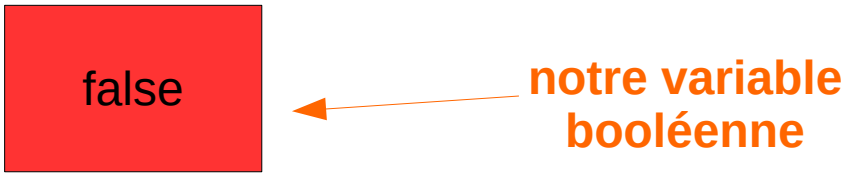
Intuition

- On cherche 3 dans le tableau ci-dessous



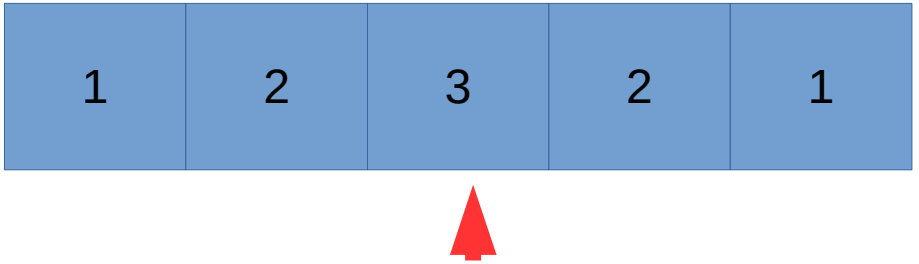
Intuition

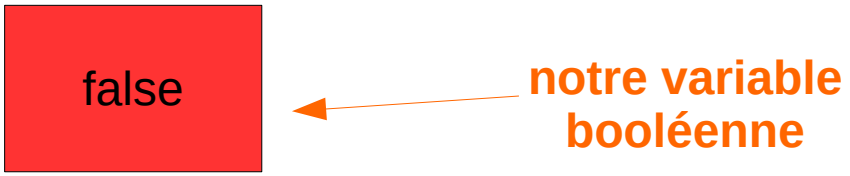
- On cherche 3 dans le tableau ci-dessous



Intuition

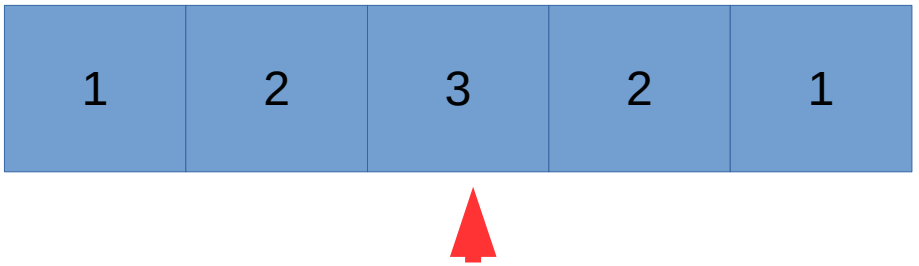
- On cherche 3 dans le tableau ci-dessous
- On le trouve, la variable booléenne change

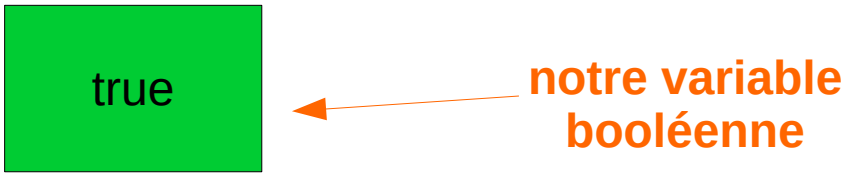




Intuition

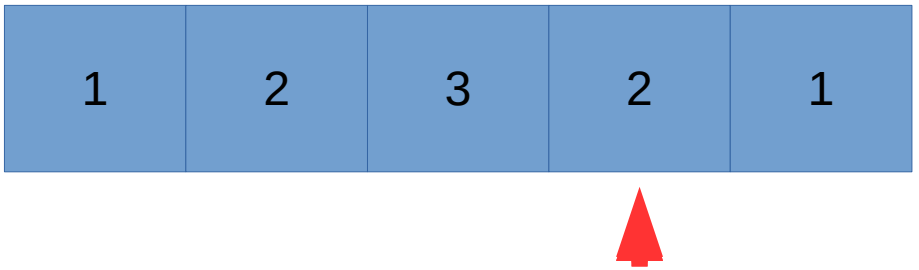
- On cherche 3 dans le tableau ci-dessous
- On le trouve, la variable booléenne change

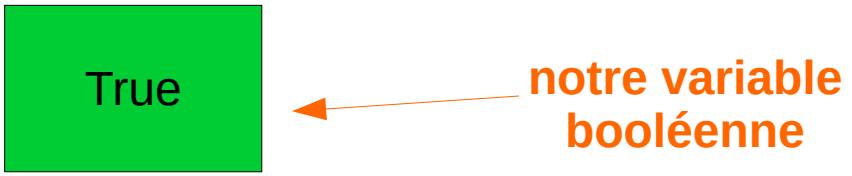




Intuition

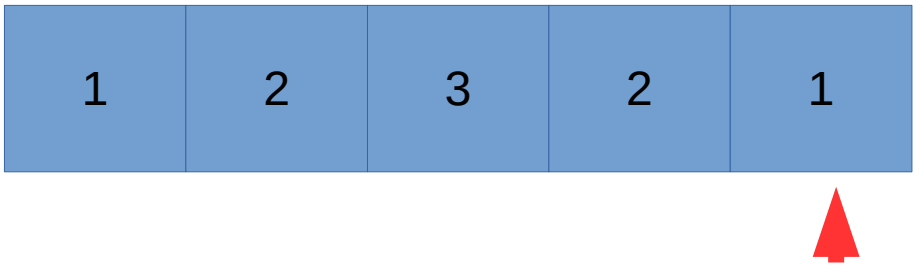
- On cherche 3 dans le tableau ci-dessous

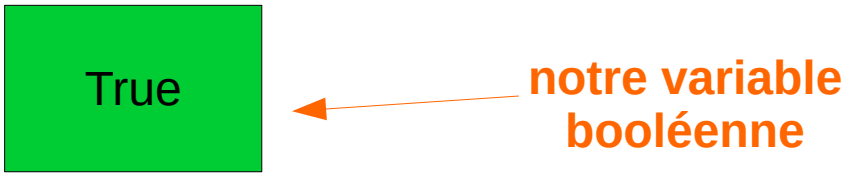




Intuition

- On cherche 3 dans le tableau ci-dessous





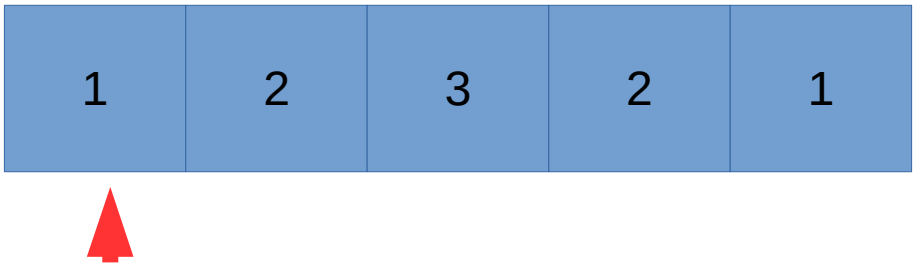
Recherche d'éléments

- Écrire une fonction qui renvoie true si un entier a est présent dans un tableau d'entiers tab
  - Cette fonction prend deux arguments: un tableau d'entier tab et un entier a
  - Elle renvoie un booléen
  - On va l'appeler cherche

```
public static boolean cherche(int [] tab, int a) {
  boolean r=false;
  for (int i=0;i<tab.length;i=i+1) {
    if(tab[i]==a){
       r= true ;
    }
  }
  return r;
}
```

Erreur classique

- On cherche 3 dans le tableau



false

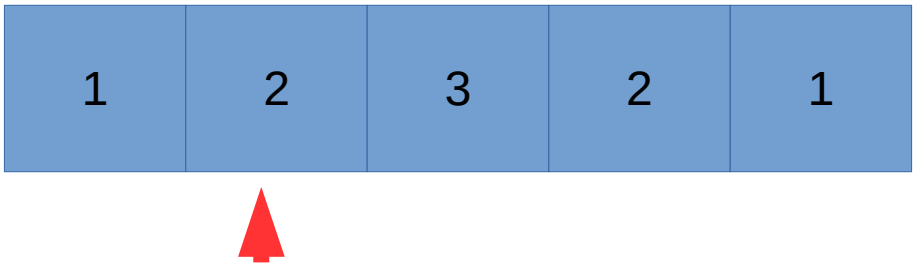
notre variable booléenne r

fausse solution !

```
public static boolean cherche(int [] tab, int a) {
  boolean r=false;
  for (int i=0;i<tab.length;i=i+1) {
    if(tab[i]==a){
       r= true ;
    }
    else {
      r=false ;
    }
  }
  return r;
}
```

Erreur classique

- On cherche 3 dans le tableau



false

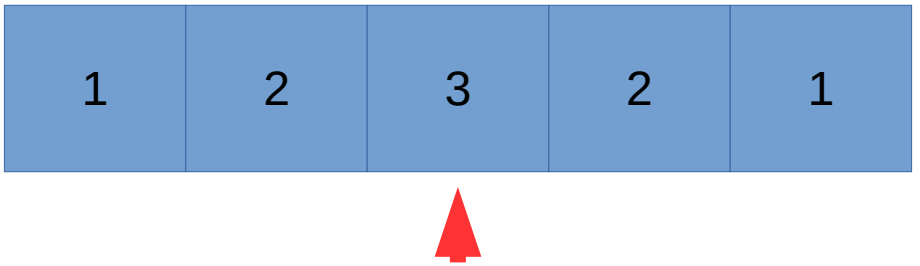
notre variable booléenne r

fausse solution !

```
public static boolean cherche(int [] tab, int a) {
  boolean r=false;
  for (int i=0;i<tab.length;i=i+1) {
    if(tab[i]==a){
       r= true ;
    }
    else {
      r=false ;
    }
  }
  return r;
}
```

Erreur classique

- On cherche 3 dans le tableau



false

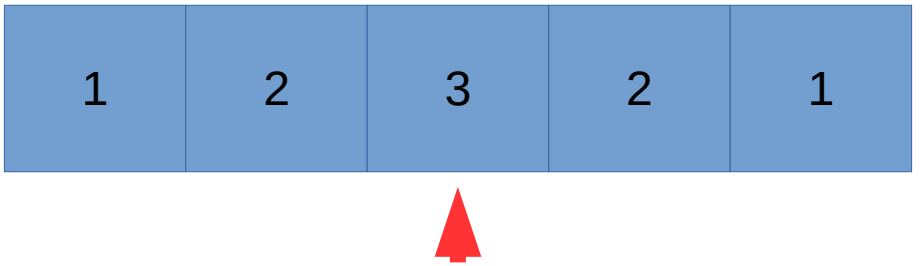
notre variable booléenne r

fausse solution !

```
public static boolean cherche(int [] tab, int a) {
  boolean r=false;
  for (int i=0;i<tab.length;i=i+1) {
    if(tab[i]==a){
       r= true ;
    }
    else {
      r=false ;
    }
  }
  return r;
}
```

Erreur classique

- On cherche 3 dans le tableau



true

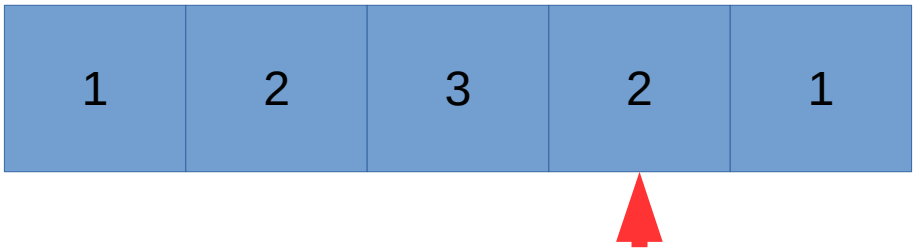
notre variable booléenne r

fausse solution !

```
public static boolean cherche(int [] tab, int a) {
  boolean r=false;
  for (int i=0;i<tab.length;i=i+1) {
    if(tab[i]==a){
       r= true ;
    }
    else {
      r=false ;
    }
  }
  return r;
}
```

Erreur classique

- On cherche 3 dans le tableau



true

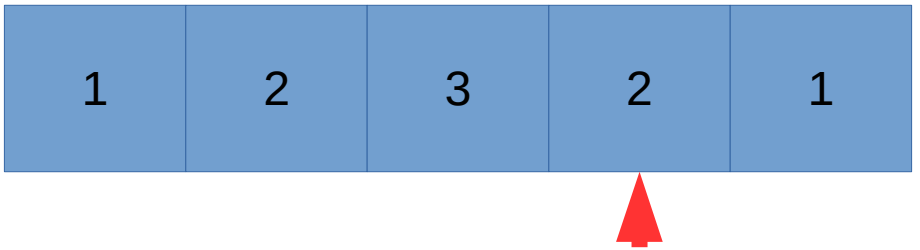
notre variable booléenne r

fausse solution !

```
public static boolean cherche(int [] tab, int a) {
  boolean r=false;
  for (int i=0;i<tab.length;i=i+1) {
    if(tab[i]==a){
       r= true ;
    }
    else {
      r=false ;
    }
  }
  return r;
}
```

Erreur classique

- On cherche 3 dans le tableau



false

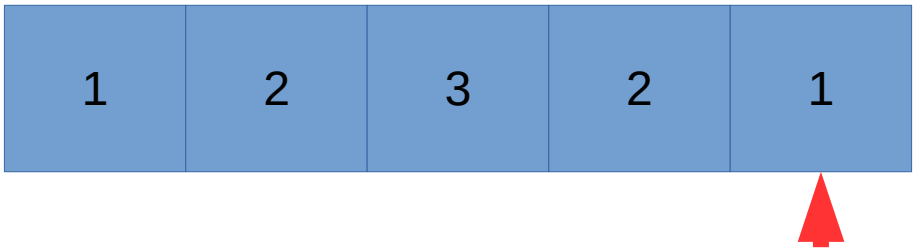
notre variable booléenne r

fausse solution !

```
public static boolean cherche(int [] tab, int a) {
  boolean r=false;
  for (int i=0;i<tab.length;i=i+1) {
    if(tab[i]==a){
       r= true ;
    }
    else {
      r=false ;
    }
  }
  return r;
}
```

Erreur classique

- On cherche 3 dans le tableau



false

notre variable booléenne r

fausse solution !

```
public static boolean cherche(int [] tab, int a) {
  boolean r=false;
  for (int i=0;i<tab.length;i=i+1) {
    if(tab[i]==a){
       r= true ;
    }
    else {
      r=false ;
    }
  }
  return r;
}
```

Création de tableaux

- On utilise aussi des fonctions pour créer des tableaux
- Par exemple, une fonction seq qui crée un tableau de taille n de la forme : {1,2,3,4,5,6,7, …,n}
- Cette fonction prend un argument n et renvoie un tableau
- Il y a deux étapes

  1) On crée le tableau int [] t= new int[n]

  2) On la remplit correctement avec un parcours qui change chaque élément

Intuition

- Que fait seq(5) ?

Intuition

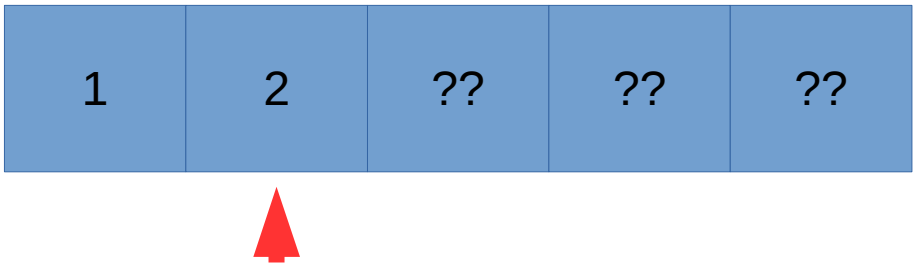
- Que fait seq(5) ?

Intuition

- Que fait seq(5) ?

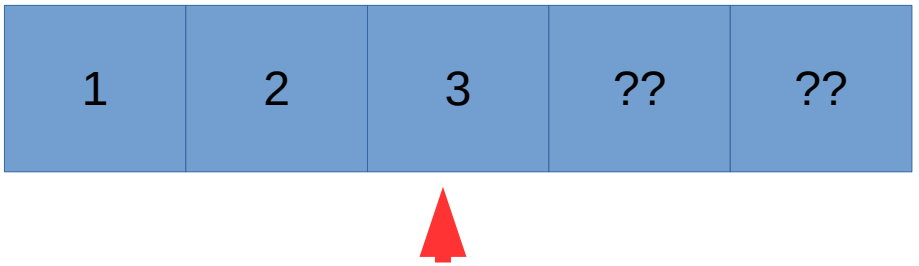
Intuition

- Que fait seq(5) ?



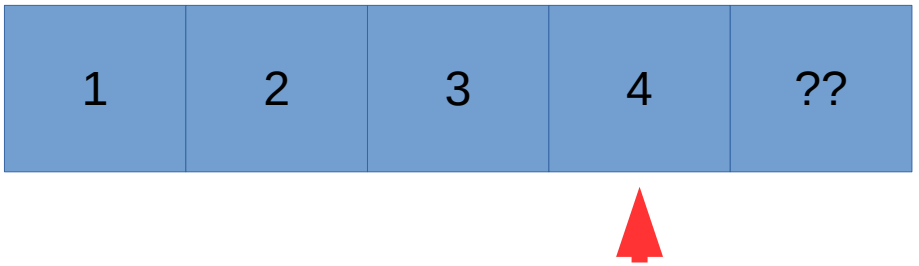
Intuition

- Que fait seq(5) ?



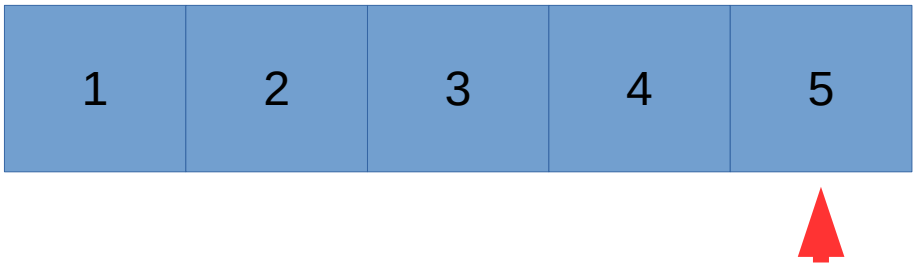
Intuition

- Que fait seq(5) ?



Intuition

- Que fait seq(5) ?



Solution

```
public static int[] seq(int n){
  int []t=new int[n];
  for (int i=0;i<t.lentgth;i++){
    t[i]=i+1 ;
  }
  return t ;
}
```