DUT MMI – IUT de Marne-la-Vallée
18/11/2015
M1202 - Algorithmique

Cours 3
Méthodologie
Tableaux et boucles

UNIVERSITÉ PARIS-EST MARNE-LA-VALLÉE

iut Marne-la-Vallée

Philippe Gambette

Sources

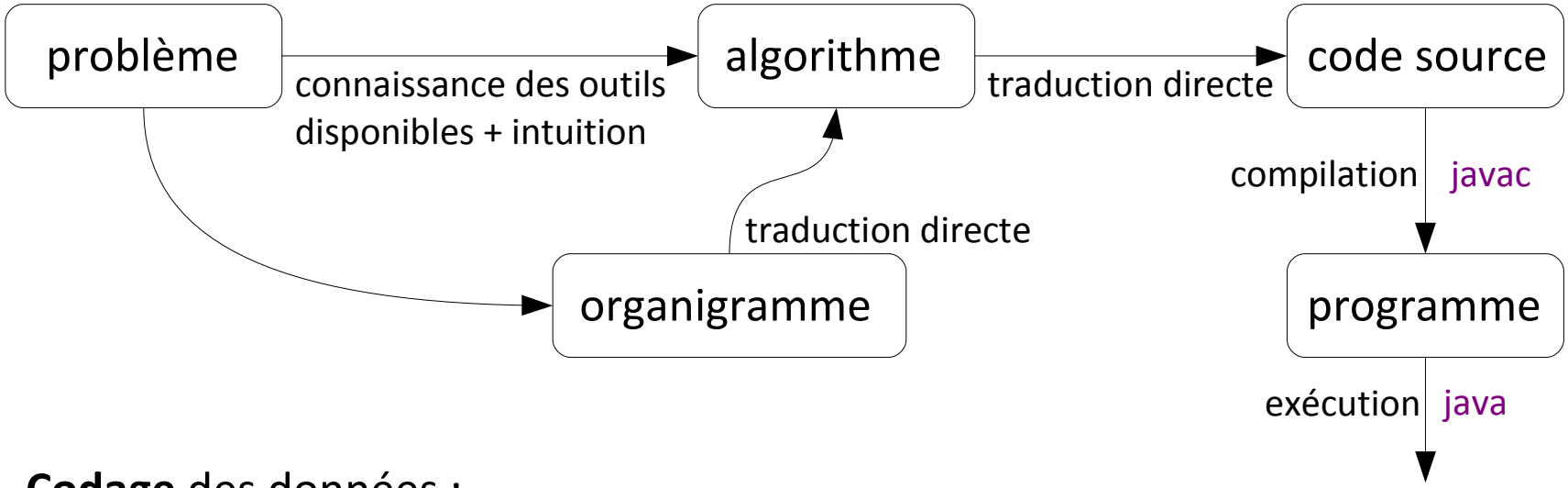
- *Le livre de Java premier langage*, d'A. Tasso
- Cours INF120 de J.-G. Luque
- Cours FLIN102 de l'Université Montpellier 2
- Cours de J. Henriet : http://julienhenriet.olympe-network.com/Algo.html
- http://xkcd.com, http://xkcd.free.fr

Plan du cours 3 – Tableaux et boucles

- Résumé des épisodes précédents
- Méthodologie : lire et comprendre un algorithme
- Méthodologie : concevoir un algorithme
- Les tableaux
- Lecture du contenu d'un tableau
- La boucle “for” / “pour tout”

Résumé des épisodes précédents

Du problème au programme pour le résoudre :



Codage des données :

• Pour chaque **type** de **variable** (entiers, flottants, chaînes de caractères, couleurs, booléens), une méthode de **codage** en binaire est choisie
(en Java : int, float, double, String, Color, boolean, ...)

• Définition d'**opérations de base** pour chaque type de données
(en Java : +, -, *, /, %, &&, ||, !, ...)

Méthodo : Lire et comprendre un algorithme

Premiers éléments à identifier :
- qu'est-ce que l'algorithme **prend en entrée** ? **Combien** de variables, de quel **type** ?
- qu'est-ce que l'algorithme **renvoie en sortie** ? **Rien** ? Ou bien **une** variable ? De quel **type** ?

Ensuite :
- quels sont les autres **algorithmes appelés** par l'algorithme ?

Enfin :
- faire la **trace** de l'algorithme, c'est-à-dire l'essayer sur un **exemple**
(… ou plusieurs pour passer au moins une fois par toutes les instructions de l'algorithme)
et voir ce que valent **toutes les variables à chaque étape** (et noter ces valeurs dans un tableau contenant une ligne par variable et une colonne par étape),
- noter en particulier le **résultat obtenu en sortie** pour une **entrée testée.**

Méthodo : Concevoir un algorithme

Premiers éléments à identifier :
- quels sont les **outils à disposition ?** (pour ces outils : données en entrée, type de données en entrée, résultat en sortie, type de résultat en sortie, résultat attendu sur un exemple...)
- quel est le **comportement attendu** pour mon algorithme ? (données en entrée, type de données en entrée, résultat en sortie, type de résultat en sortie, résultat attendu sur un exemple...)

Ensuite, résoudre le problème en utilisant ces outils :
- comment résoudre le problème **étape par étape** ? (essayer sur l'exemple testé)
- est-ce que les **outils à disposition** sont **utilisables** pour réaliser chaque étape ?

Enfin :
- comment **structurer** l'utilisation des outils à disposition ? (**combinaison** des différents outils à l'intérieur de structure de **boucles**, de **tests**, utilisation d'un **organigramme**...)
- comment **décomposer** le problème ? (et **reformuler** chaque sous-problème pour le résoudre avec les outils à disposition, écrire un algorithme par sous-problème)

+ fiche méthodo : Vérifier un algorithme !

Plan du cours 3 – Tableaux et boucles

- Résumé des épisodes précédents
- Méthodologie : lire et comprendre un algorithme
- Méthodologie : concevoir un algorithme
- Les tableaux
- Lecture du contenu d'un tableau
- La boucle “for” / “pour tout”

Les tableaux

Les tableaux sont des variables qui contiennent **plusieurs variables de même type**, stockées chacune dans une des cases du tableau.

Par exemple,

Un **tableau d'entiers** :

4
5
1
23
8
9

Un **tableau de chaînes de caractères** :

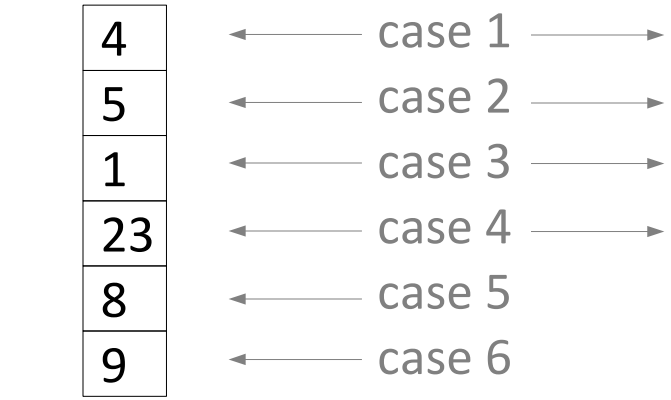
“chaine1”
“chaine2”
“blabla”
“toto”

Les tableaux

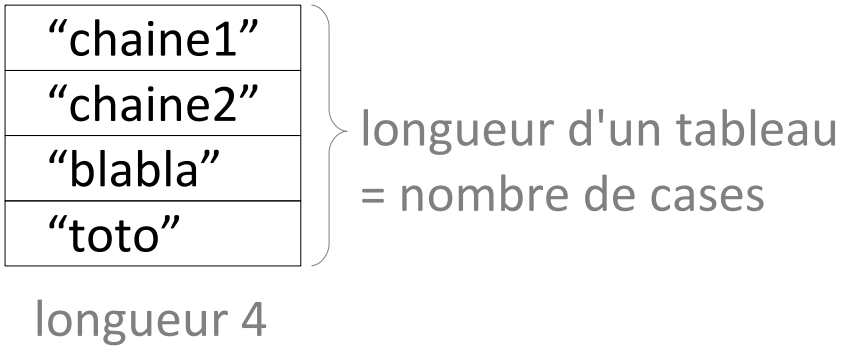
Les tableaux sont des variables qui contiennent **plusieurs variables de même type**, stockées chacune dans une des cases du tableau.

Par exemple,

Un **tableau d'entiers** :



Un **tableau de chaînes de caractères** :



Les tableaux

en pseudo-code

Les tableaux sont des variables qui contiennent **plusieurs variables de même type**, stockées chacune dans une des cases du tableau.

Par exemple, Variables : *tableau1*, un tableau d'entiers,
tableau2, un tableau de chaînes de caractères

Un **tableau d'entiers** :

4	case 1
5	case 2
1	case 3
23	case 4
8	case 5
9	case 6

longueur 6

tableau1←**NouveauTableau**(6)
Case(*tableau1*,1)←4
Case(*tableau1*,2)←5
Case(*tableau1*,3)←1
Case(*tableau1*,4)←23
Case(*tableau1*,5)←8
Case(*tableau1*,6)←9

Un **tableau de chaînes de caractères** :

“chaine1”
“chaine2”
“blabla”
“toto”

longueur 4

longueur d'un tableau
= nombre de cases
Longueur(*tableau2*)

tableau2←**NouveauTableau**(4)
Case(*tableau2*,1)←”chaine1”
Case(*tableau2*,2)←”chaine2”
Case(*tableau2*,3)←”blabla”
Case(*tableau2*,4)←”toto”

Plus court :
tableau1←{4,5,1,23,8,9}
tableau2←{”chaine1”,”chaine2”,”blabla”,”toto”}

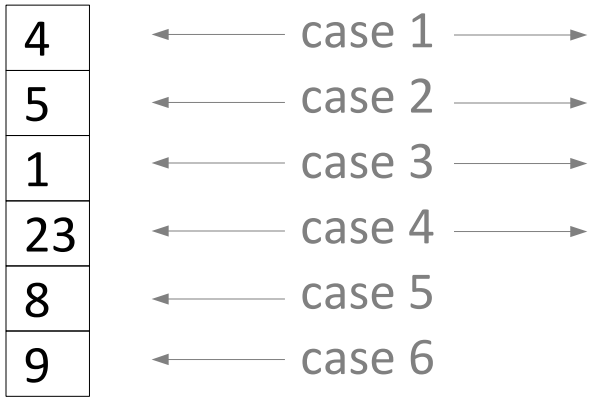
Les tableaux

en Java

Les tableaux sont des variables qui contiennent **plusieurs variables de même type**, stockées chacune dans une des cases du tableau.

Par exemple, `int[] tableau1; String[] tableau2;`

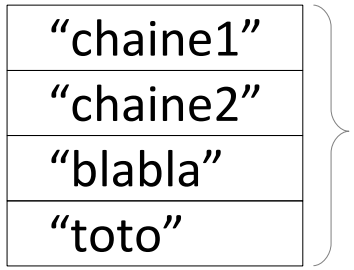
Un **tableau d'entiers** :



longueur 6

```
tableau1=new int[6];
tableau1[0]=4;
tableau1[1]=5;
tableau1[2]=1;
tableau1[3]=23;
tableau1[4]=8;
tableau1[5]=9;
```

Un **tableau de chaînes de caractères** :



longueur d'un tableau
= nombre de cases
`tableau2.length`

longueur 4

```
tableau2=new String[4];
tableau2[0]=”chaine1”;
tableau2[1]=”chaine2”;
tableau2[2]=”blabla”;
tableau2[3]=”toto”;
```

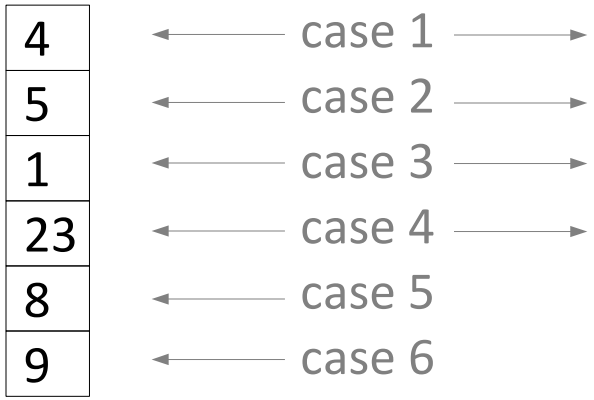
Les tableaux

en Java

Les tableaux sont des variables qui contiennent **plusieurs variables de même type**, stockées chacune dans une des cases du tableau.

Par exemple, `int[] tableau1; String[] tableau2;`

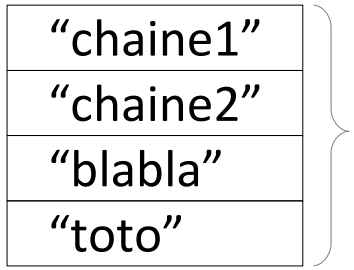
Un **tableau d'entiers** :



longueur 6

```
tableau1=new int[6];
tableau1[0]=4;
tableau1[1]=5;
tableau1[2]=1;
tableau1[3]=23;
tableau1[4]=8;
tableau1[5]=9;
```

Un **tableau de chaînes de caractères** :



longueur d'un tableau
= nombre de cases
`tableau2.length`

longueur 4

```
tableau2=new String[4];
tableau2[0]=”chaine1”;
tableau2[1]=”chaine2”;
tableau2[2]=”blabla”;
tableau2[3]=”toto”;
```

Attention, cases du tableau `t` numérotées de `0` à `t.length-1` en Java.

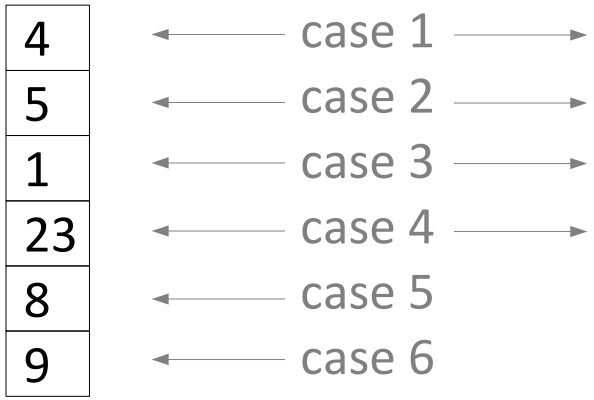
Les tableaux

en Java

Les tableaux sont des variables qui contiennent **plusieurs variables de même type**, stockées chacune dans une des cases du tableau.

Par exemple, `int[] tableau1; String[] tableau2;`

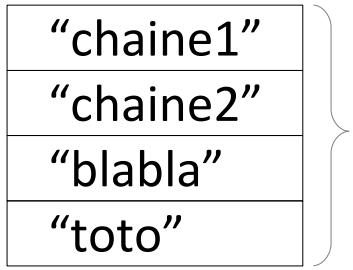
Un **tableau d'entiers** :



longueur 6

```
tableau1=new int[6];
tableau1[0]=4;
tableau1[1]=5;
tableau1[2]=1;
tableau1[3]=23;
tableau1[4]=8;
tableau1[5]=9;
```

Un **tableau de chaînes de caractères** :



longueur d'un tableau
= nombre de cases
`tableau2.length`

longueur 4

```
tableau2=new String[4];
tableau2[0]=”chaine1”;
tableau2[1]=”chaine2”;
tableau2[2]=”blabla”;
tableau2[3]=”toto”;
```

Plus court (**déclaration + initialisation**) :

```
int[] tableau1 = {4,5,1,23,8,9};
String[] tableau2 = {”chaine1”,”chaine2”,”blabla”,”toto”};
```

Les tableaux

Pour lire le contenu d'un tableau...
il faut une **boucle pour aller lire chaque case** !

Si le tableau a été prévu trop court au début, **impossible de changer sa longueur**... il faut une boucle pour le recopier dans un tableau plus grand !

Possibilité de créer des **tableaux de tableaux**...

Manipulation et expériences en TD/TP...

Plan du cours 3 – Tableaux et boucles

- Résumé des épisodes précédents
- Méthodologie : lire et comprendre un algorithme
- Méthodologie : concevoir un algorithme
- Les tableaux
- Lecture du contenu d'un tableau
- La boucle “for” / “pour tout”

Affichage du contenu d'un tableau d'entiers

Algorithme **AfficheTableau**

```
public static       AfficheTableau(         tableau1){
```

Affichage du contenu d'un tableau d'entiers

Algorithme **AfficheTableau**
Variable d'entrée : tableau d'entiers *tableau1*
Variable : entier *i*
Début
  i ← 1
  Tant que *i* < **Longueur**(*tableau1*)+1 faire :
    Affiche(**Case**(*tableau1*,*i*))
    i ← *i*+1
  Fin TantQue
Fin

```
public static void AfficheTableau(int[] tableau1){
  //Afficher les cases du tableau tableau1
  int i;
  i = 0;
  while (i<tableau1.length){
    System.out.println(tableau1[i]);
    i = i+1;
  }
}
```

Plan du cours 3 – Tableaux et boucles

- Résumé des épisodes précédents
- Méthodologie : lire et comprendre un algorithme
- Méthodologie : concevoir un algorithme
- Les tableaux
- Lecture du contenu d'un tableau
- La boucle “for” / “pour tout”

La boucle “for” / “Pour tout...”

Pour **parcourir tous les entiers entre deux valeurs entières**.

En pseudo-code :

Pour tout entier *i* de 1 à 42 faire :

    …

Fin Pour

En Java :

```
for(int i=1;i<43;i++){
    …
}
```

La boucle “for” / “Pour tout...”

Exemple : **parcours des cases d'un tableau**

En pseudo-code avec Tant que :

Variables : tableau d'entiers *tab*, entier *i*
i ← 1
Tant que *i* < **Longueur**(*tab*)+1 faire :
    [des choses avec la *i*-ième case du tableau **Case**(*tab*,*i*)...]
    i ← *i*+1
Fin Tant que

En pseudo-code avec Pour :

Variables : tableau d'entiers *tab*, entier *i*
Pour *i* de 1 à **Longueur**(*tab*) faire :
    [des choses avec la *i*-ième case du tableau **Case**(*tab*,*i*)...]
Fin Pour

La boucle “for” / “Pour tout...”

Pour **parcourir tous les entiers entre deux valeurs entières**.

En pseudo-code :

Pour tout entier *i* de 1 à 42 faire :

    …

Fin Pour

En Java :

```
for(int i=1;i<43;i++){
    …
}
```

En Java avec while :

```
int i=1;
while(i<43){
…
i++;
}
```

La boucle “for” / “Pour tout...”

Pour **parcourir tous les entiers entre deux valeurs entières**.

En pseudo-code :

Pour tout entier *i* de 1 à 42 faire :

…

Fin Pour

En Java :

```
for(int i=1;i<43;i++){
    …
}
```

En Java avec while :

```
int i=1;
while(i<43){
…
i++;
}
```

La boucle “for” / “Pour tout...”

La boucle “for” / “Pour tout”

Une boucle pour **parcourir tous les entiers entre deux valeurs entières**.

Algorithme **DessineHistogramme**
Entrée : tableau de chaînes de caractères *Mots* et tableau d'entiers *NbApparitions*.
Variable : entier *compteur*
Début
compteur ← 1
Tant que *compteur* < **Longueur**(*Mots*)+1 faire :
    dessineRectanglePlein(*compteur**4-4,
        50-10***Case**(*NbApparitions*,*compteur*),
        4,10***Case**(*NbApparitions*,*compteur*),
        couleurRGB(0,0,255))
    compteur ← 1 + *compteur*
Fin TantQue
Fin

Algorithme **DessineHistogramme**
Entrée : tableau de chaînes de caractères *Mots* et tableau d'entiers *NbApparitions*.
Variable : entier *compteur*
Début
Pour *compteur* de 1 à **Longueur**(*Mots*) faire :
    dessineRectanglePlein(*compteur**4-4,
        50-10***Case**(*NbApparitions*,*compteur*),
        4,10***Case**(*NbApparitions*,*compteur*),
        couleurRGB(0,0,255))
Fin Pour
Fin

La boucle “for” / “Pour tout...”

La boucle “for” / “Pour tout”

Une boucle pour **parcourir tous les entiers entre deux valeurs entières**.

Algorithme **DessineHistogramme**
Entrée : tableau de chaînes de caractères *Mots* et tableau d'entiers *NbApparitions*.
Variable : entier *compteur*
Début
compteur ← 1
Tant que *compteur* < **Longueur**(*Mots*)+1 faire :
    dessineRectanglePlein(*compteur**4-4,
        50-10***Case**(*NbApparitions*,*compteur*),
        4,10***Case**(*NbApparitions*,*compteur*),
        couleurRGB(0,0,255))
    compteur ← 1 + *compteur*
Fin TantQue
Fin

En Java :

```
int compteur;
compteur=1;
while(compteur<mots.length+1) {
    …
}
```

Algorithme **DessineHistogramme**
Entrée : tableau de chaînes de caractères *Mots* et tableau d'entiers *NbApparitions*.
Variable : entier *compteur*
Début
Pour *compteur* de 1 à **Longueur**(*Mots*) faire :
    dessineRectanglePlein(*compteur**4-4,
        50-10***Case**(*NbApparitions*,*compteur*),
        4,10***Case**(*NbApparitions*,*compteur*),
        couleurRGB(0,0,255))
Fin Pour
Fin

```
for(int compteur=1;compteur<mots.length+1;compteur++){
    …
}
```

La boucle “for” / “Pour tout…”

La boucle “for” / “Pour tout”

Une boucle pour **parcourir tous les entiers entre deux valeurs entières**.

Algorithme **DessineHistogramme**
Entrée : tableau de chaînes de caractères *Mots* et tableau d'entiers *NbApparitions*.
Variable : entier *compteur*
Début
compteur ← 1
Tant que *compteur* < **Longueur**(*Mots*)+1 faire :
    dessineRectanglePlein(*compteur**4-4,
        50-10***Case**(*NbApparitions*,*compteur*),
        4,10***Case**(*NbApparitions*,*compteur*),
        couleurRGB(0,0,255))
    compteur ← 1 + *compteur*
Fin TantQue
Fin

En Java :

```
int compteur;
compteur=1;
while(compteur<mots.length+1) {
    …
}
```

Algorithme **DessineHistogramme**
Entrée : tableau de chaînes de caractères *Mots* et tableau d'entiers *NbApparitions*.
Variable : entier *compteur*
Début
Pour *compteur* de 1 à **Longueur**(*Mots*) faire :
    dessineRectanglePlein(*compteur**4-4,
        50-10***Case**(*NbApparitions*,*compteur*),
        4,10***Case**(*NbApparitions*,*compteur*),
        couleurRGB(0,0,255))
Fin Pour
Fin

déclaration + initialisation

```
for(int compteur=1;compteur<mots.length+1;compteur++){
    …
}
```

La boucle “for” / “Pour tout...”

La boucle “for” / “Pour tout”

Une boucle pour **parcourir tous les entiers entre deux valeurs entières**.

Algorithme **DessineHistogramme**
Entrée : tableau de chaînes de caractères *Mots* et tableau d'entiers *NbApparitions*.
Variable : entier *compteur*
Début
compteur ← 1
Tant que *compteur* < **Longueur**(*Mots*)+1 faire :
    dessineRectanglePlein(*compteur**4-4,
        50-10***Case**(*NbApparitions*,*compteur*),
        4,10***Case**(*NbApparitions*,*compteur*),
        couleurRGB(0,0,255))
    compteur ← 1 + *compteur*
Fin TantQue
Fin

En Java :

```
int compteur;
compteur=1;
while(compteur<mots.length+1) {
    …
}
```

Algorithme **DessineHistogramme**
Entrée : tableau de chaînes de caractères *Mots* et tableau d'entiers *NbApparitions*.
Variable : entier *compteur*
Début
Pour *compteur* de 1 à **Longueur**(*Mots*) faire :
    dessineRectanglePlein(*compteur**4-4,
        50-10***Case**(*NbApparitions*,*compteur*),
        4,10***Case**(*NbApparitions*,*compteur*),
        couleurRGB(0,0,255))
Fin Pour
Fin

déclaration + initialisation *condition d'arrêt*

```
for(int compteur=1;compteur<mots.length+1;compteur++){
    …
}
```

La boucle “for” / “Pour tout...”

La boucle “for” / “Pour tout”

Une boucle pour **parcourir tous les entiers entre deux valeurs entières**.

Algorithme **DessineHistogramme**
Entrée : tableau de chaînes de caractères *Mots* et tableau d'entiers *NbApparitions*.
Variable : entier *compteur*
Début
compteur ← 1
Tant que *compteur* < **Longueur**(*Mots*)+1 faire :
    dessineRectanglePlein(*compteur**4-4,
        50-10***Case**(*NbApparitions*,*compteur*),
        4,10***Case**(*NbApparitions*,*compteur*),
        couleurRGB(0,0,255))
    compteur ← 1 + *compteur*
Fin TantQue
Fin

En Java :

```
int compteur;
compteur=1;
while(compteur<mots.length+1) {
    …
}
```

Algorithme **DessineHistogramme**
Entrée : tableau de chaînes de caractères *Mots* et tableau d'entiers *NbApparitions*.
Variable : entier *compteur*
Début
Pour *compteur* de 1 à **Longueur**(*Mots*) faire :
    dessineRectanglePlein(*compteur**4-4,
        50-10***Case**(*NbApparitions*,*compteur*),
        4,10***Case**(*NbApparitions*,*compteur*),
        couleurRGB(0,0,255))
Fin Pour
Fin

déclaration + initialisation — *condition d'arrêt* — *mise à jour*

```
for(int compteur=1;compteur<mots.length+1;compteur++){
    …
}
```

La boucle “for” / “Pour tout...”

La boucle “for” / “Pour tout”

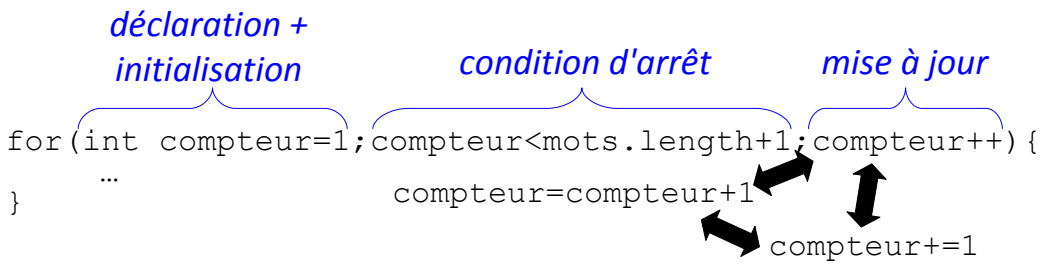
Une boucle pour **parcourir tous les entiers entre deux valeurs entières**.

Algorithme **DessineHistogramme**
Entrée : tableau de chaînes de caractères *Mots* et tableau d'entiers *NbApparitions*.
Variable : entier *compteur*
Début
compteur ← 1
Tant que *compteur* < **Longueur**(*Mots*)+1 faire :
    dessineRectanglePlein(*compteur**4-4,
        50-10***Case**(*NbApparitions*,*compteur*),
        4,10***Case**(*NbApparitions*,*compteur*),
        couleurRGB(0,0,255))
    compteur ← 1 + *compteur*
Fin TantQue
Fin

En Java :

```
int compteur;
compteur=1;
while(compteur<mots.length+1){
    …
}
```

Algorithme **DessineHistogramme**
Entrée : tableau de chaînes de caractères *Mots* et tableau d'entiers *NbApparitions*.
Variable : entier *compteur*
Début
Pour *compteur* de 1 à **Longueur**(*Mots*) faire :
    dessineRectanglePlein(*compteur**4-4,
        50-10***Case**(*NbApparitions*,*compteur*),
        4,10***Case**(*NbApparitions*,*compteur*),
        couleurRGB(0,0,255))
Fin Pour
Fin



La copie d'un tableau

Pour copier le contenu d'un tableau d'entiers *t1* dans un nouveau tableau d'entiers *t2*.

En Java :

La copie d'un tableau

Pour copier le contenu d'un tableau d'entiers *t1* dans un nouveau tableau d'entiers *t2*.

En Java :

```
public static int[] Copie(int[] t1){
   int[] t2;
   t2=new int[t1.length];
   int i=0;
   while(i<t1.length){
     t2[i]=t1[i];
     i=i+1;
   }
   return t2;
}
```