DUT MMI – IUT de Marne-la-Vallée
03/11/2016
M1202 - Algorithmique

Cours 3
Tableaux et boucles

UNIVERSITÉ PARIS-EST MARNE-LA-VALLÉE

iut Marne-la-Vallée

Philippe Gambette

Sources

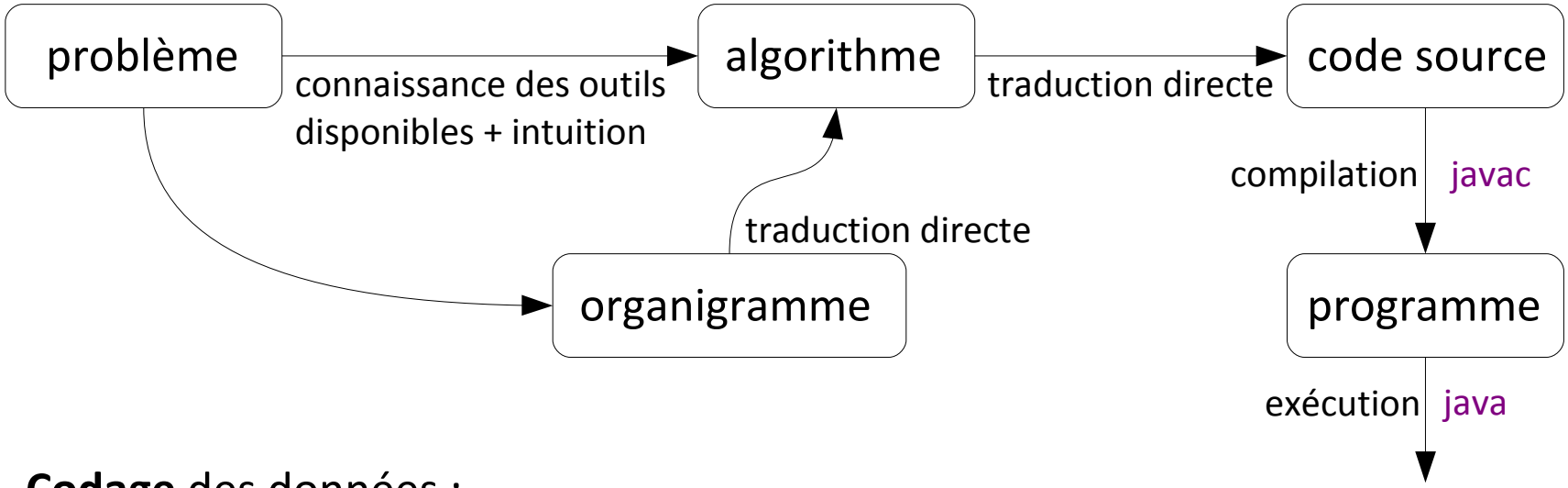
- *Le livre de Java premier langage*, d'A. Tasso
- Cours INF120 de J.-G. Luque
- Cours FLIN102 de l'Université Montpellier 2
- Cours de J. Henriet : http://julienhenriet.olympe-network.com/Algo.html
- http://xkcd.com, http://xkcd.free.fr

Plan du cours 3 – Tableaux et boucles

- Résumé des épisodes précédents
- Les tableaux
- Affichage du contenu d'un tableau
- La boucle “for” / “pour tout”
- Dessin d'un graphique à partir du contenu d'un tableau

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 : +, -, *, /, %, &&, ||, !, ...)

Plan du cours 3 – Tableaux et boucles

- Résumé des épisodes précédents
- Les tableaux
- Affichage du contenu d'un tableau
- La boucle “for” / “pour tout”
- Dessin d'un graphique à partir du contenu d'un tableau

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** :

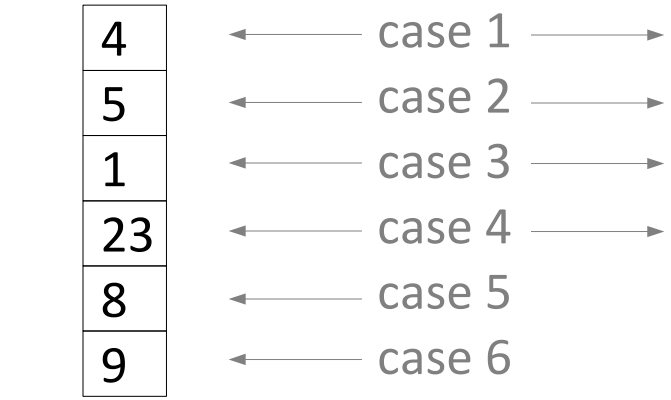
“chaine1”
“chaine2”
“blabla”
“toto”

Les tableaux

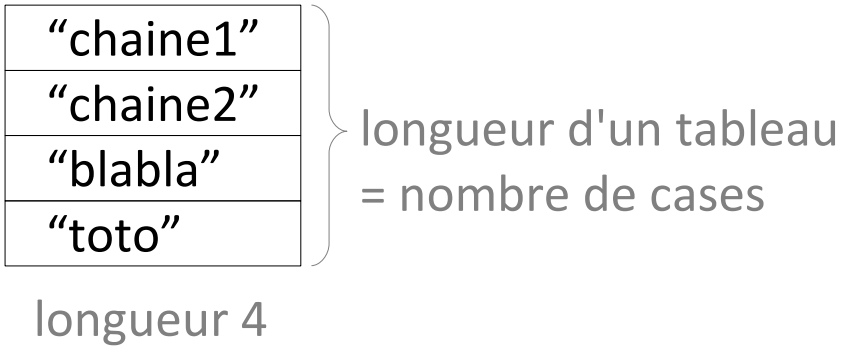
Les tableaux sont des variables qui contiennent **plusieurs variables de même type**, stockées chacune dans une des cases du tableau.

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”}

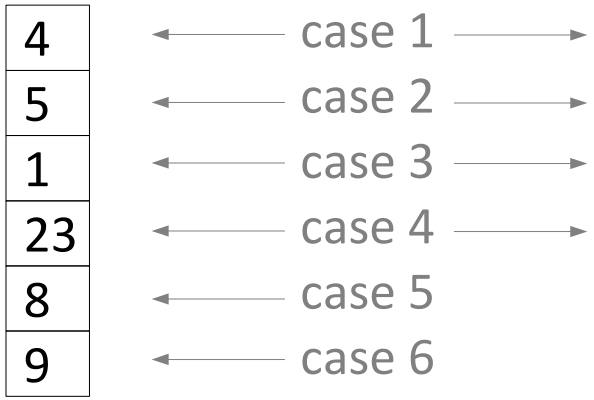
Les tableaux

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;`

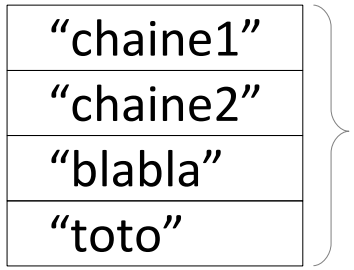
Un **tableau d'entiers** :



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

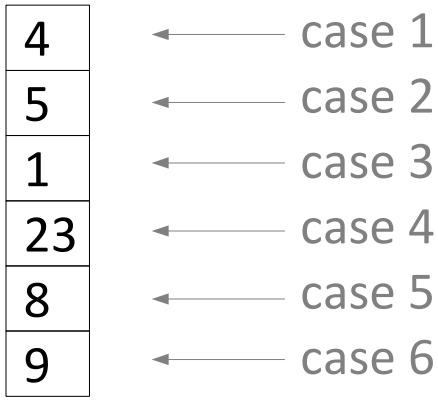
en Java

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;
```

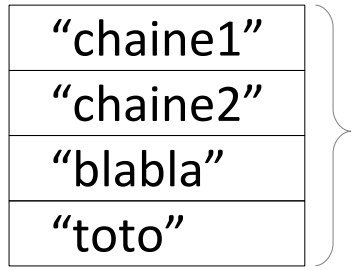
Un **tableau d'entiers** :



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.

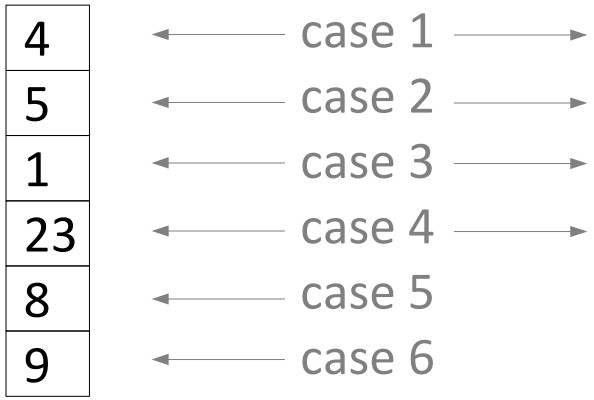
Les tableaux

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;`

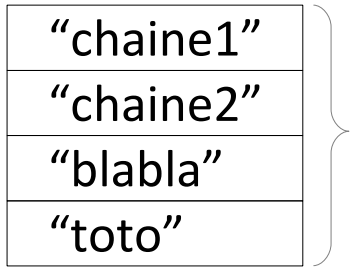
Un **tableau d'entiers** :



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
- Les tableaux
- Affichage du contenu d'un tableau
- La boucle “for” / “pour tout”
- Dessin d'un graphique à partir du contenu d'un tableau

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
- Les tableaux
- Affichage du contenu d'un tableau
- La boucle “for” / “pour tout”
- Dessin d'un graphique à partir du contenu d'un tableau

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”

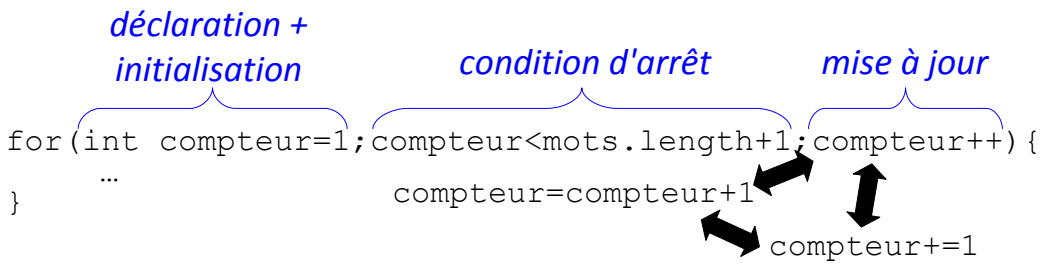
Une boucle pour **parcourir tous les entiers entre deux valeurs entières**.

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        copie(           ){




}
```

La copie d'un tableau

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

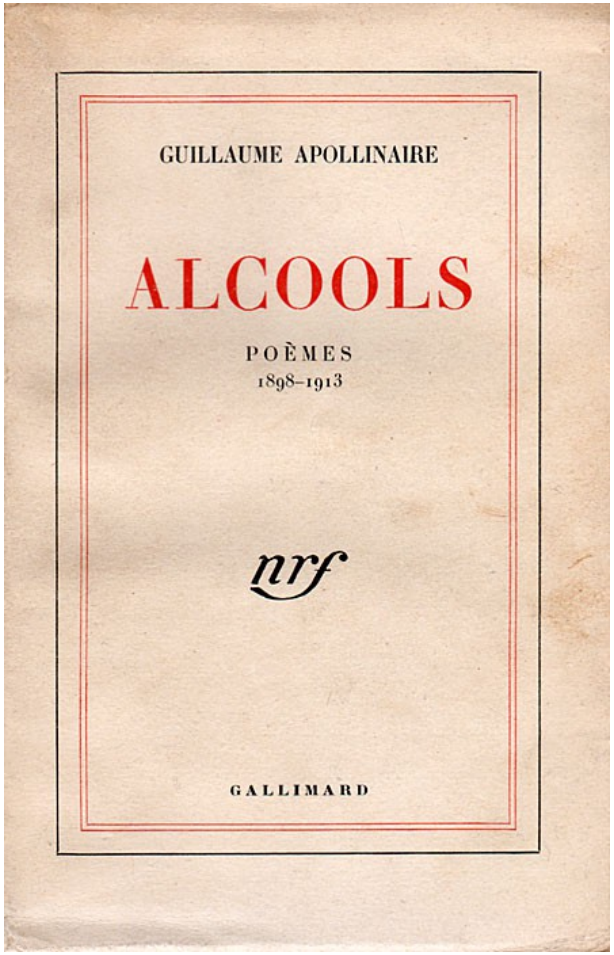
En Java :

```
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;
}
```

Plan du cours 3 – Tableaux et boucles

- Résumé des épisodes précédents
- Les tableaux
- Affichage du contenu d'un tableau
- La boucle “for” / “pour tout”
- Dessin d'un graphique à partir du contenu d'un tableau

Graphique du nombre d'apparitions des mots d'un texte



J'ai cueilli ce brin de bruyère
L'automne est morte souviens-t'en
Nous ne nous verrons plus sur terre
Odeur du temps brin de bruyère
Et souviens-toi que je t'attends

Graphique du nombre d'apparitions des mots d'un texte

Un tableau de chaînes de caractères *mots*

j	1
ai	1
cueilli	1
ce	1
brin	2
de	2
bruyère	2
l	1
automne	1
est	1
morte	1
souviens	2
t	2
en	1
nous	2
ne	1
verrons	1
plus	1
sur	1
terre	1
odeur	1
du	1
temps	1
et	1
toi	1
que	1
je	1
attends	1

J'ai cueilli ce brin de bruyère
L'automne est morte souviens-t'en
Nous ne nous verrons plus sur terre
Odeur du temps brin de bruyère
Et souviens-toi que je t'attends

Un tableau d'entiers *nbApparitions*

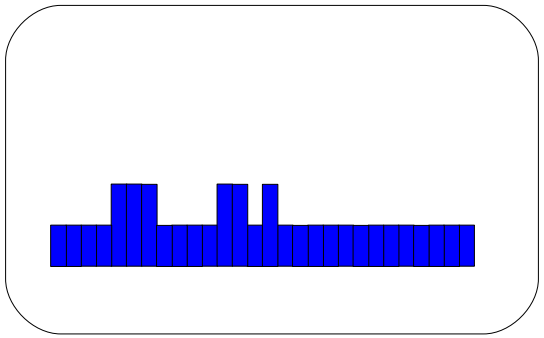
Graphique du nombre d'apparitions des mots d'un texte

Un tableau de chaînes de caractères *mots*

j	1
ai	1
cueilli	1
ce	1
brin	2
de	2
bruyère	2
l	1
automne	1
est	1
morte	1
souviens	2
t	2
en	1
nous	2
ne	1
verrons	1
plus	1
sur	1
terre	1
odeur	1
du	1
temps	1
et	1
toi	1
que	1
je	1
attends	1

Un tableau d'entiers *nbApparitions*

Résultat voulu :



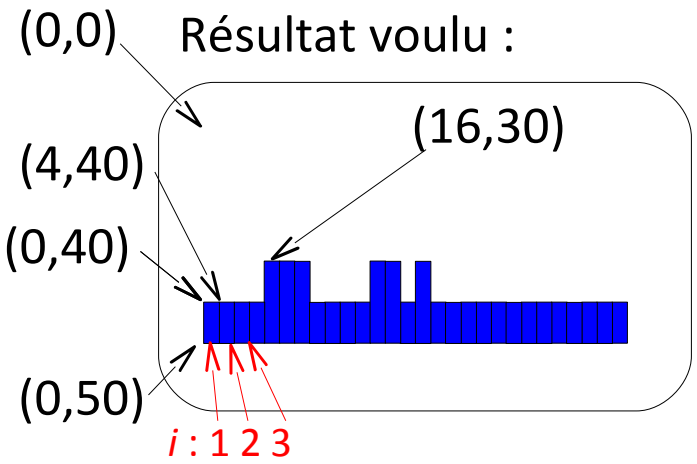
Graphique du nombre d'apparitions des mots d'un texte

Un tableau de chaînes de caractères *mots*

j	1
ai	1
cueilli	1
ce	1
brin	2
de	2
bruyère	2
l	1
automne	1
est	1
morte	1
souviens	2
t	2
en	1
nous	2
ne	1
verrons	1
plus	1
sur	1
terre	1
odeur	1
du	1
temps	1
et	1
toi	1
que	1
je	1
attends	1

Un tableau d'entiers *nbApparitions*

Résultat voulu :



Algorithme **dessineHistogramme**

Entrée : tableau de chaînes de caractères *mots* et tableau d'entiers *nbApparitions*.

Variable : entier *i*

Début

i ← 1

Tant que ... faire :

Fin Tant que

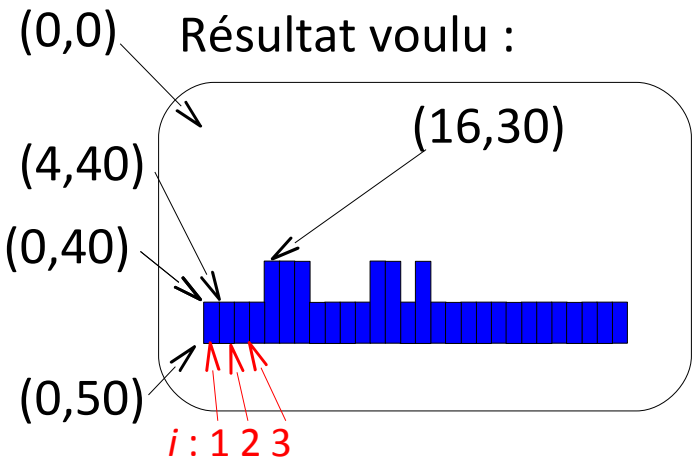
Fin

Graphique du nombre d'apparitions des mots d'un texte

Un tableau de chaînes de caractères *mots*

j	1
ai	1
cueilli	1
ce	1
brin	2
de	2
bruyère	2
l	1
automne	1
est	1
morte	1
souviens	2
t	2
en	1
nous	2
ne	1
verrons	1
plus	1
sur	1
terre	1
odeur	1
du	1
temps	1
et	1
toi	1
que	1
je	1
attends	1

Un tableau d'entiers *nbApparitions*



Algorithme **dessineHistogramme**

Entrée : tableau de chaînes de caractères *mots* et tableau d'entiers *nbApparitions*.

Variable : entier *i*

Début

i ← 1

Tant que *i* < **longueur**(*mots*)+1 faire :

Fin Tant que

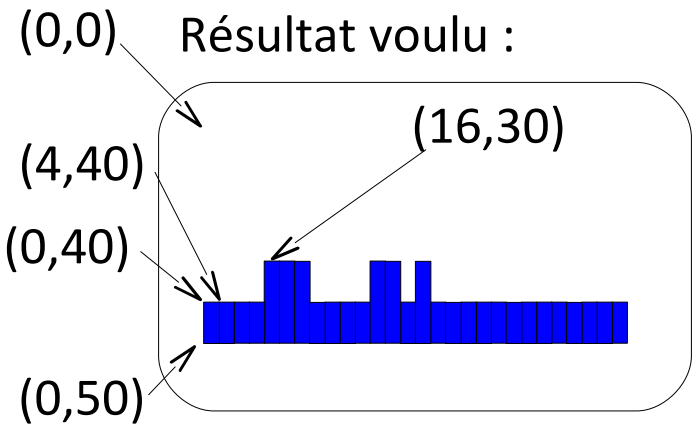
Fin

Graphique du nombre d'apparitions des mots d'un texte

Un tableau de chaînes de caractères *mots*

j	1
ai	1
cueilli	1
ce	1
brin	2
de	2
bruyère	2
l	1
automne	1
est	1
morte	1
souviens	2
t	2
en	1
nous	2
ne	1
verrons	1
plus	1
sur	1
terre	1
odeur	1
du	1
temps	1
et	1
toi	1
que	1
je	1
attends	1

Un tableau d'entiers *nbApparitions*



Algorithme **dessineHistogramme**
Entrée : tableau de chaînes de caractères *mots* et tableau d'entiers *nbApparitions*.
Variable : entier *i*
Début
i ← 1
Tant que *i* < **longueur**(*mots*)+1 faire : ...

```
        dessineRectanglePlein(0,40,4,10,
                couleurRGB(0,0,255))
...
        dessineRectanglePlein(4,40,4,10,
                couleurRGB(0,0,255))
```

Fin Tant que
Fin

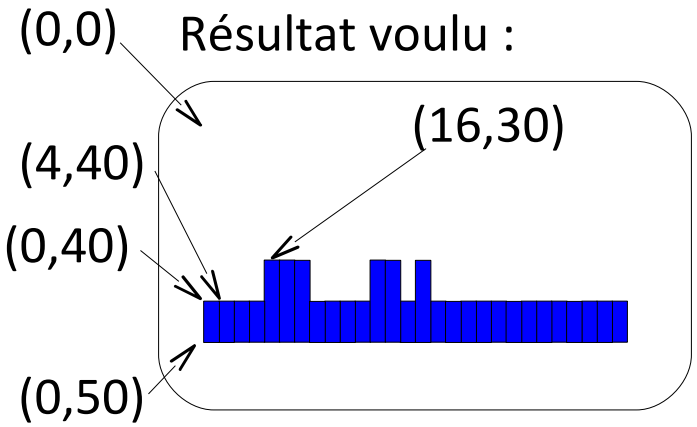
Graphique du nombre d'apparitions des mots d'un texte

Un tableau de chaînes de caractères *mots*

j
ai
cueilli
ce
brin
de
bruyère
l
automne
est
morte
souviens
t
en
nous
ne
verrons
plus
sur
terre
odeur
du
temps
et
toi
que
je
attends

1
1
1
1
2
2
2
1
1
1
1
2
2
1
2
1
1
1
1
1
1
1
1
1
1
1
1
1
1

i	*x*	4*i*
1	0	4
2	4	8
3	8	12
4	12	16
...		



Algorithme **dessineHistogramme**
Entrée : tableau de chaînes de caractères *mots* et tableau d'entiers *nbApparitions*.
Variable : entier *i*
Début
i ← 1
Tant que *i* < **longueur**(*mots*)+1 faire : ...

x

dessineRectanglePlein(0,40,4,10, **couleurRGB**(0,0,255))
...
dessineRectanglePlein(4,40,4,10, **couleurRGB**(0,0,255))

Fin Tant que
Fin

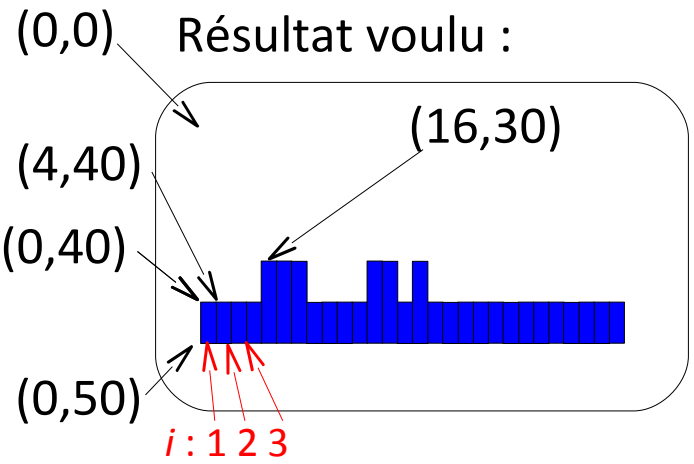
Un tableau d'entiers *nbApparitions*

Graphique du nombre d'apparitions des mots d'un texte

Un tableau de chaînes de caractères *mots*

j	1
ai	1
cueilli	1
ce	1
brin	2
de	2
bruyère	2
l	1
automne	1
est	1
morte	1
souviens	2
t	2
en	1
nous	2
ne	1
verrons	1
plus	1
sur	1
terre	1
odeur	1
du	1
temps	1
et	1
toi	1
que	1
je	1
attends	1

Un tableau d'entiers *nbApparitions*



Algorithme **dessineHistogramme**

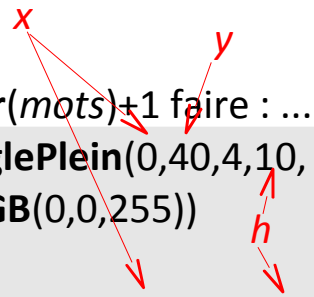
Entrée : tableau de chaînes de caractères *mots* et tableau d'entiers *nbApparitions*.

Variable : entier *i*

Début

i ← 1

Tant que *i* < **longueur**(*mots*)+1 faire : ...



dessineRectanglePlein(0,40,4,10,
couleurRGB(0,0,255))

...

dessineRectanglePlein(4,40,4,10,
couleurRGB(0,0,255))

Fin Tant que

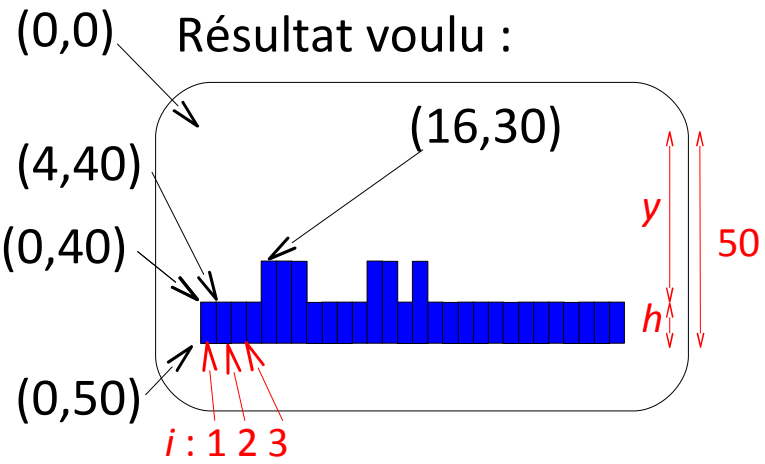
Fin

Graphique du nombre d'apparitions des mots d'un texte

Un tableau de chaînes de caractères *mots*

j	1
ai	1
cueilli	1
ce	1
brin	2
de	2
bruyère	2
l	1
automne	1
est	1
morte	1
souviens	2
t	2
en	1
nous	2
ne	1
verrons	1
plus	1
sur	1
terre	1
odeur	1
du	1
temps	1
et	1
toi	1
que	1
je	1
attends	1

Résultat voulu :



Algorithme **dessineHistogramme**

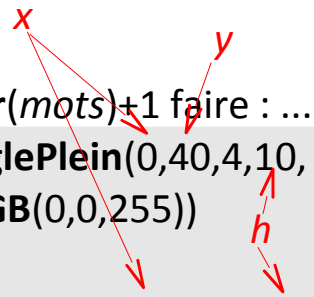
Entrée : tableau de chaînes de caractères *mots* et tableau d'entiers *nbApparitions*.

Variable : entier *i*

Début

i ← 1

Tant que *i* < **longueur**(*mots*)+1 faire : ...



dessineRectanglePlein(0,40,4,10,
couleurRGB(0,0,255))

...

dessineRectanglePlein(4,40,4,10,
couleurRGB(0,0,255))

Fin Tant que

Fin

Un tableau d'entiers *nbApparitions*

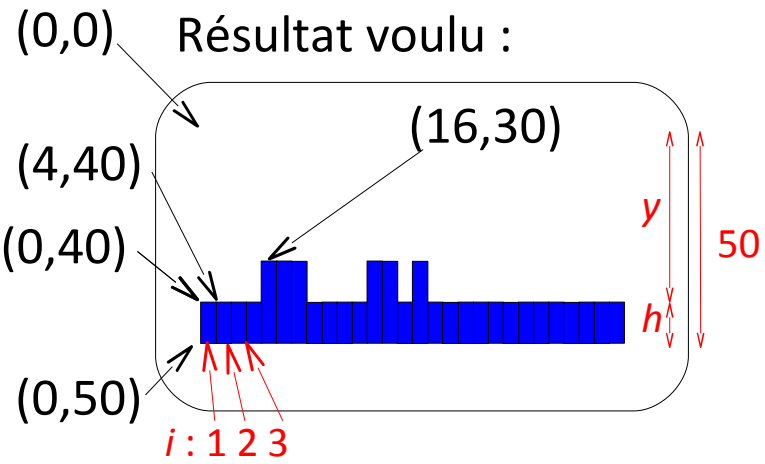
Graphique du nombre d'apparitions des mots d'un texte

Un tableau de chaînes de caractères *mots*

j	1
ai	1
cueilli	1
ce	1
brin	2
de	2
bruyère	2
l	1
automne	1
est	1
morte	1
souviens	2
t	2
en	1
nous	2
ne	1
verrons	1
plus	1
sur	1
terre	1
odeur	1
du	1
temps	1
et	1
toi	1
que	1
je	1
attends	1

Un tableau d'entiers *nbApparitions*

Résultat voulu :



Algorithme **dessineHistogramme**

Entrée : tableau de chaînes de caractères *mots* et tableau d'entiers *nbApparitions*.

Variable : entier *i*

Début

i ← 1

Tant que *i* < **longueur**(*mots*)+1 faire : ...

x = 4(*i*-1)

h = 10***case** (*NbApparitions*,*i*)

dessineRectanglePlein(0,40,4,10, **couleurRGB**(0,0,255))

...

dessineRectanglePlein(4,40,4,10, **couleurRGB**(0,0,255))

Fin Tant que

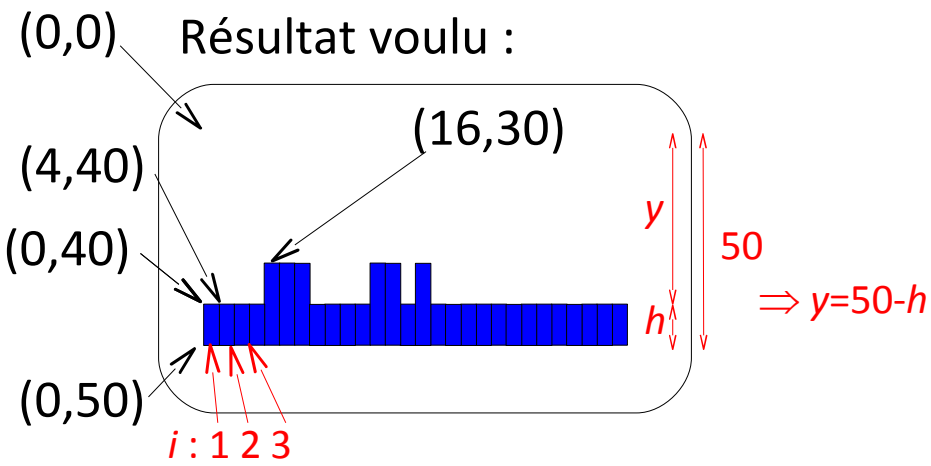
Fin

Graphique du nombre d'apparitions des mots d'un texte

Un tableau de chaînes de caractères *mots*

j	1
ai	1
cueilli	1
ce	1
brin	2
de	2
bruyère	2
l	1
automne	1
est	1
morte	1
souviens	2
t	2
en	1
nous	2
ne	1
verrons	1
plus	1
sur	1
terre	1
odeur	1
du	1
temps	1
et	1
toi	1
que	1
je	1
attends	1

Un tableau d'entiers *nbApparitions*



Algorithme **dessineHistogramme**

Entrée : tableau de chaînes de caractères *mots* et tableau d'entiers *nbApparitions*.

Variable : entier *i*

Début

i ← 1

Tant que *i* < **longueur**(*mots*)+1 faire :

x = 4(*i*-1)

y = 50-10***case** (*nbApparitions*,*i*)

h = 10***case** (*nbApparitions*,*i*)

```
    dessineRectanglePlein(0,40,4,10,
        couleurRGB(0,0,255))
...
    dessineRectanglePlein(4,40,4,10,
        couleurRGB(0,0,255))
```

Fin Tant que

Fin

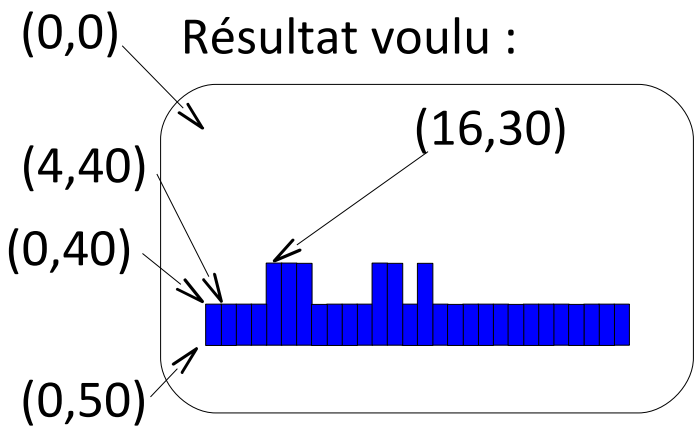
Graphique du nombre d'apparitions des mots d'un texte

Un tableau de chaînes de caractères *mots*

j	1
ai	1
cueilli	1
ce	1
brin	2
de	2
bruyère	2
l	1
automne	1
est	1
morte	1
souviens	2
t	2
en	1
nous	2
ne	1
verrons	1
plus	1
sur	1
terre	1
odeur	1
du	1
temps	1
et	1
toi	1
que	1
je	1
attends	1

Un tableau d'entiers *nbApparitions*

Résultat voulu :



Algorithme **dessineHistogramme**
Entrée : tableau de chaînes de caractères *mots* et tableau d'entiers *nbApparitions*.
Variable : entier *i*
Début
i ← 1
Tant que *i* < **longueur**(*mots*)+1 faire :
    dessineRectanglePlein(*i**4-4,
        50-10***case**(*nbApparitions*,*i*),
        4,10***case**(*nbApparitions*,*i*),
        couleurRGB(0,0,255))

Fin Tant que
Fin

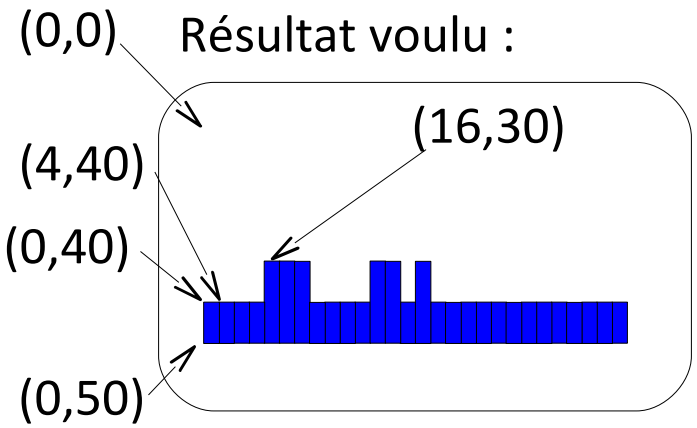
Graphique du nombre d'apparitions des mots d'un texte

Un tableau de chaînes de caractères *mots*

j
ai
cueilli
ce
brin
de
bruyère
l
automne
est
morte
souviens
t
en
nous
ne
verrons
plus
sur
terre
odeur
du
temps
et
toi
que
je
attends

1
1
1
1
2
2
2
1
1
1
1
2
2
1
2
1
1
1
1
1
1
1
1
1
1
1
1
1
1

Résultat voulu :



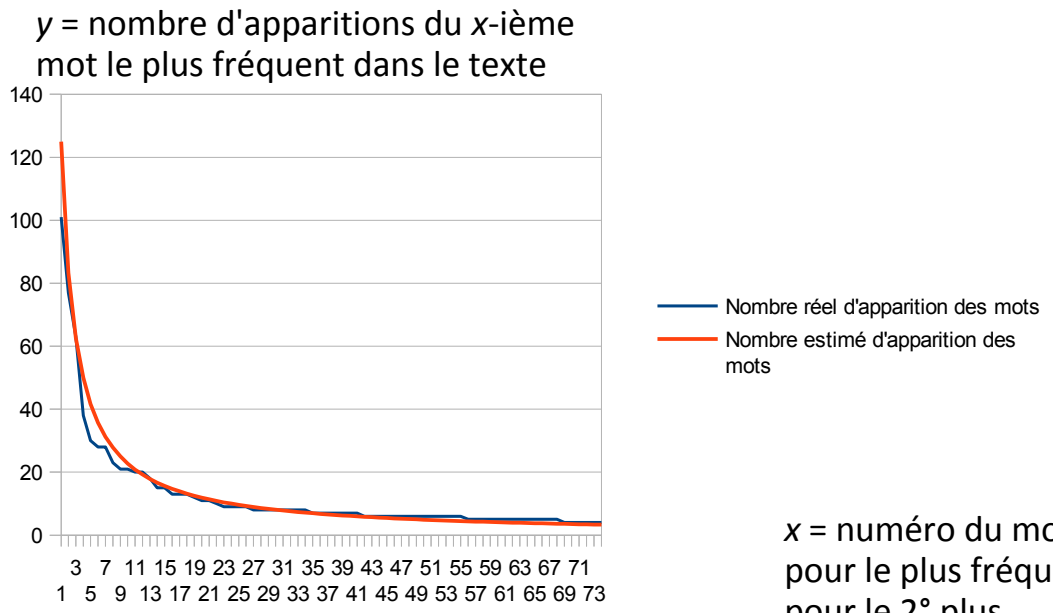
Algorithme **dessineHistogramme**
Entrée : tableau de chaînes de caractères *mots* et tableau d'entiers *nbApparitions*.
Variable : entier *i*
Début
i ← 1
Tant que *i* < **longueur**(*mots*)+1 faire :
    dessineRectanglePlein(*i**4-4,
        50-10***case**(*nbApparitions*,*i*),
        4,10***case**(*nbApparitions*,*i*),
        couleurRGB(0,0,255))
    i ← 1 + *i*
Fin Tant que
Fin

Un tableau d'entiers *nbApparitions*

Le nombre d'apparitions d'un mot dans un texte

La “minute mathématique”

La loi de Zipf prédit la courbe du nombre d'apparitions des mots les plus fréquents d'un texte.

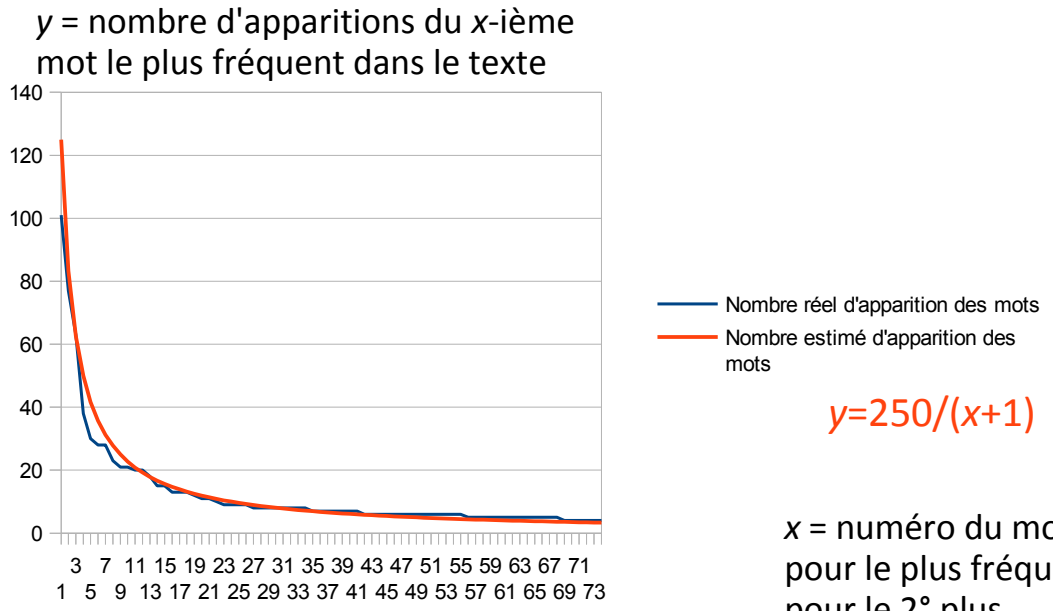


x = numéro du mot (1 pour le plus fréquent, 2 pour le 2° plus fréquent...)

Le nombre d'apparitions d'un mot dans un texte

La “minute mathématique”

La loi de Zipf prédit la courbe du nombre d'apparitions des mots les plus fréquents d'un texte.



Fonctionne pour n'importe quel texte assez long...