

RB14 - Introduction à la programmation

Florence Zara

LIRIS - Université Lyon 1

<http://liris.cnrs.fr/florence.zara>

E-mail: florence.zara@liris.cnrs.fr

Objectif du cours

- Savoir ce que le terme programmation veut dire
- Comment on crée des applications ?
- Quels sont les langages de programmation ?
- Comment on écrit un programme ?

Il y a de la programmation partout...

- Exemples
 - Réveil
 - programmer l'heure de la sonnerie
 - puce contenue dans le réveil permet de stocker des instructions très simples
 - Box
 - programmer un enregistrement
 - Porte d'entrée
 - composer le code et ouvrir la porte

Programmation et informatique

- On utilise les ordinateurs au travers de programmes
 - le système d'exploitation
 - les applications utilisateur
 - des programmes que l'on fait soi-même
 - macros suite bureautique (macro VBA dans Excel)
 - automatisation des tâches
 - des programmes plus « importants »

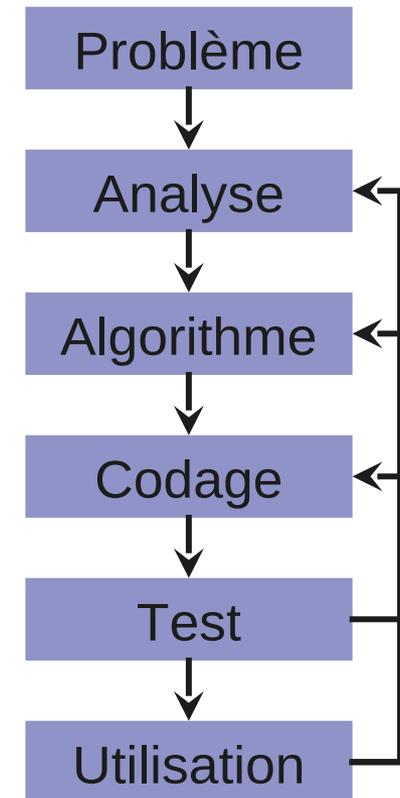
} programmés
par d'autres

Programmeur et utilisateur

- Programmeur
 - conçoit et fabrique un programme
 - qui rendra des services à un utilisateur
 - en fonction d'une commande (les besoins exprimés)
- Utilisateur
 - utilise un programme informatique au cours de son activité
 - ne l'utilise jamais exactement comme le concepteur l'a prévu
- Remarque
 - un programmeur est un utilisateur d'un programme informatique destiné à aider à la conception de programmes informatiques
 - Visual C++ (Windows), KDevelop (linux), Xcode (Mac)

Du problème au programme

- **Problème**
- **Analyse**
 - données d'entrée
 - résultats
 - traitements
 - cas critiques
- **Algorithme**
 - indépendant du langage de programmation
- **Codage**
- **Tests**
 - simulations : vérification des cas critiques
- En cas d'erreur, on retourne en arrière



Algorithme

• Première définition

- décrit comment un humain ou une machine peuvent réaliser un objectif
 - suivre une recette de cuisine (objectif : fabriquer une recette)
 - décomposer un numéro de Sécurité Sociale (objectif : extraire des informations sur le possesseur d'un numéro de SS)
 - utiliser les transports en commun (objectif : venir à l'Université)

• Deuxième définition

- suite d'actions
 - chaque action est décrite par une ou plusieurs instructions
- à appliquer à des données
 - indépendamment de leurs valeurs
- pour obtenir un résultat
 - en un nombre fini d'étapes
(doit s'arrêter après un certain temps)

• Remarque

- devrait prévoir tous les cas possibles

Algorithme

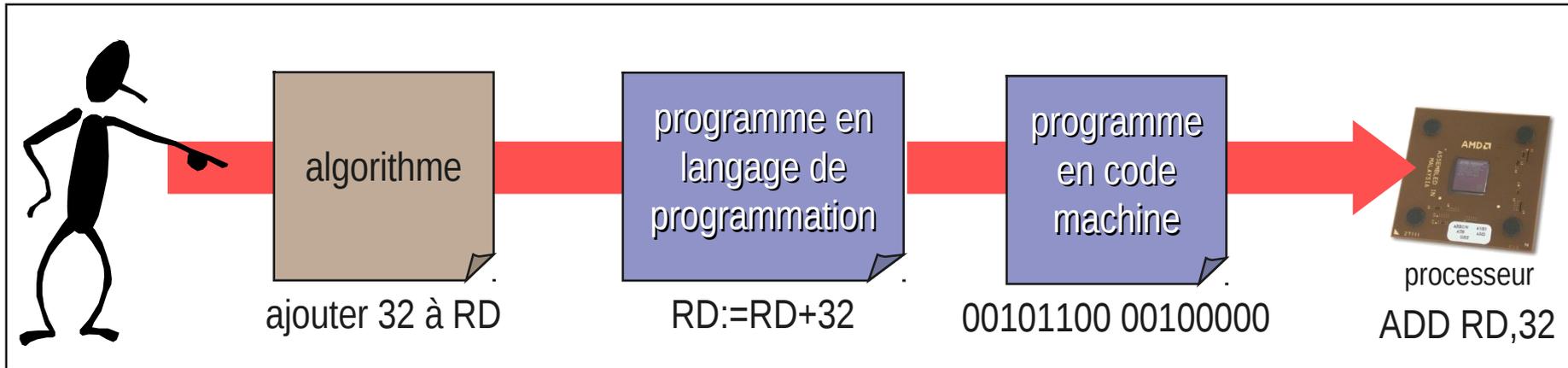
- Description « en français » des instructions à faire pour résoudre un problème
- Ensuite traduction de cet algorithme dans un langage de programmation
- Programme écrit dans un langage : écriture du « code source »
- Ce programme sera ensuite compris par la machine

Codage de l'algorithme par une suite d'instructions

- En programmation, une instruction
 - décrit une action élémentaire
 - est spécifiée par un mot-clé
 - soit fourni par le langage
 - soit défini par le programmeur
 - peut avoir des paramètres
 - Sur quelles données est appliquée une instruction ?
- Exemples
 - trouver la troisième lettre d'une chaîne de caractères
 - prendre un nombre au hasard
 - calculer l'arrondi d'une valeur
 - compter de 1 à 100

Programme et langage de programmation

- Programme
 - c'est la traduction d'un algorithme dans un langage informatique
 - éventuellement découpé en modules (sous-programmes)
- Langage de programmation
 - langage intermédiaire entre l'humain et le processeur
 - permet d'exprimer les instructions algorithmiques dans un langage rigoureux
- Programme en code machine
 - description binaire du programme, adaptée au système et au microprocesseur



Choix du langage de programmation

- Critères à prendre en compte
 - Portabilité : passage d'un type d'ordinateur à un autre (système d'exploitation, carte graphique)
 - Stabilité : langage ancien ou récent
 - Performance : rapidité d'exécution
 - Sécurité : robustesse face aux attaques
 - ...

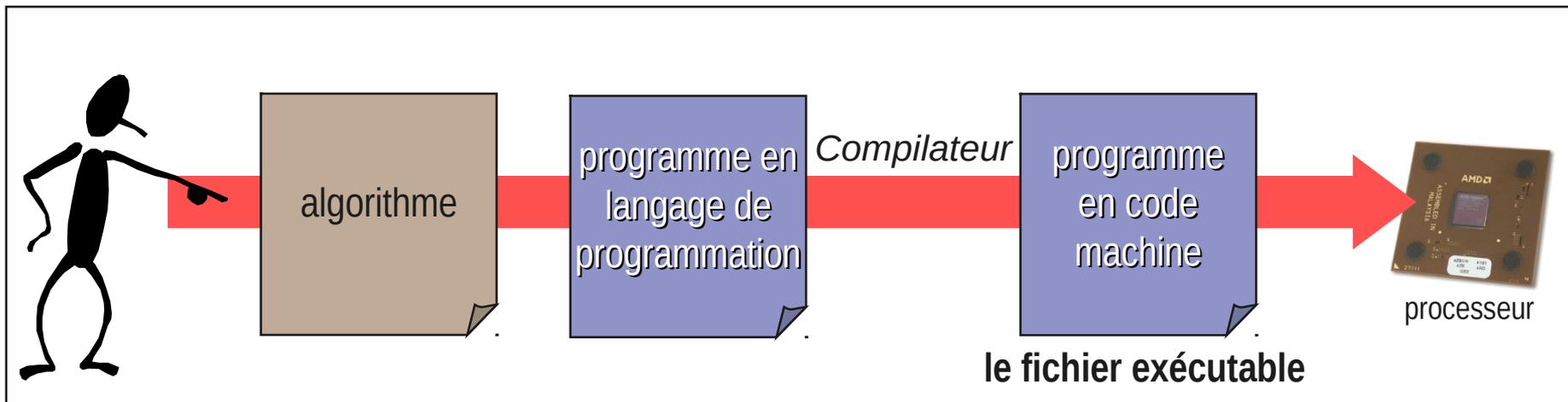
Classement des langages de programmation en fonction de leur utilisation (plus de 700 langages de programmation) :

http://www.tiobe.com/tiobe_index?page=index

Familles de langages de programmation (1)

• Langages compilés

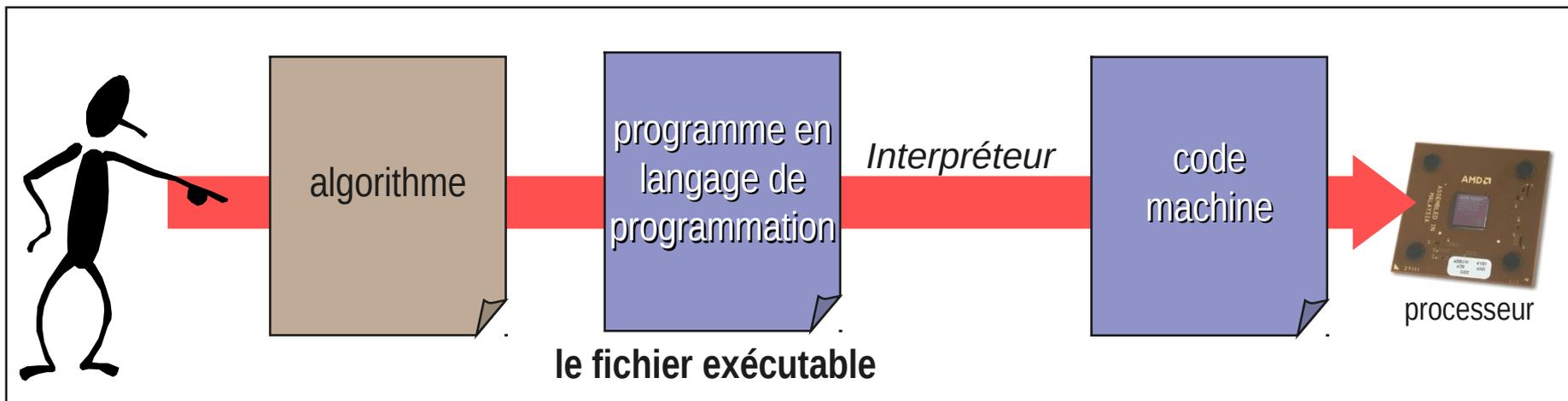
- le programme décrit dans le langage de programmation (code source) est compilé (traduit) en code machine
- cette traduction se fait une seule fois, avant l'exécution du programme
- le programme est stocké sous deux formes
 - il faut le recompiler pour l'exécuter sur un système/machine différent
- exemples de langages
 - Cobol, Fortran, Pascal, SmallTalk, C, C++, Delphi, Visual Basic...



Familles de langages de programmation (2)

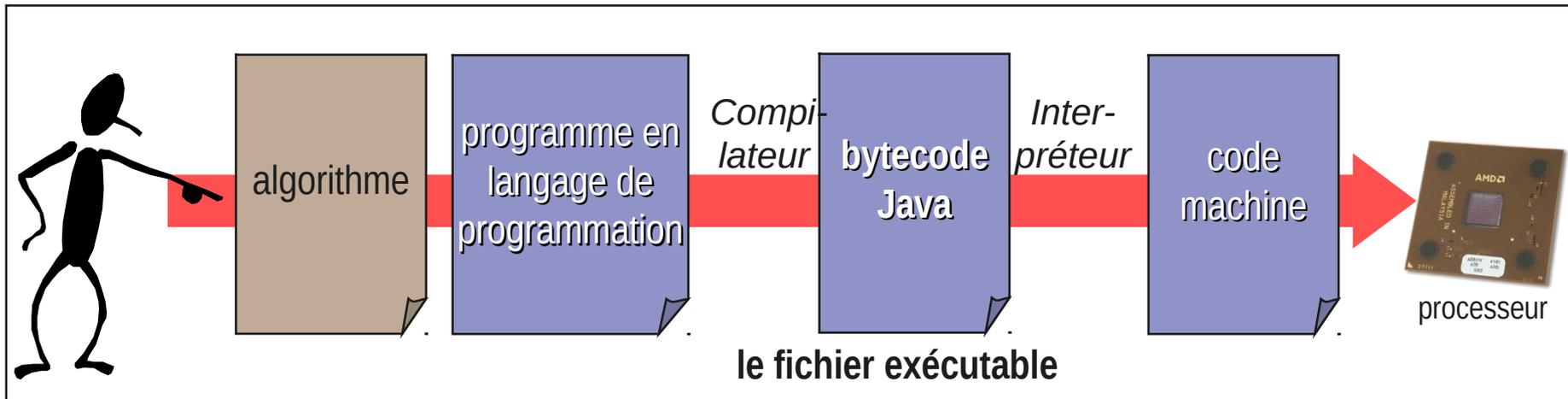
Langages interprétés

- la traduction en code machine se fait à chaque exécution du programme
- le programme n'est stocké que sous une seule forme, qui est le fichier exécutable
 - il pourra être utilisé tel quel sur plusieurs systèmes/machines différents, si chacun dispose d'un interpréteur
- exemples de langages
 - commandes DOS, shell Unix, Javascript, Perl, PHP, Python, Visual Basic for Applications (VBA)



Familles de langages de programmation (3)

- Le langage Java
 - le programme en langage de programmation est traduit (compilé) en bytecode Java (code machine indépendant du processeur)
 - le bytecode Java est exécuté (interprété) par une machine virtuelle Java
 - la machine virtuelle est dépendante du système/machine sur lequel elle s'exécute
 - la compilation se fait avant l'exécution du programme, et le programme est stocké sous deux formes



Autres classifications des langages

• Programmation impérative

- Très proche des instructions du processeur (affectation de variables, lecture, condition, boucle, etc.)
- C, Pascal, Delphi, VBA, Visual Basic

• Programmation fonctionnelle

- On ne peut décrire que des fonctions
- Un programme = 1 expression avec des fonctions
- On ne peut pas changer la valeur de variable en cours d'exécution
- Scheme, Lisp

• Programmation objet

- Smalltalk, C++, Java, Delphi

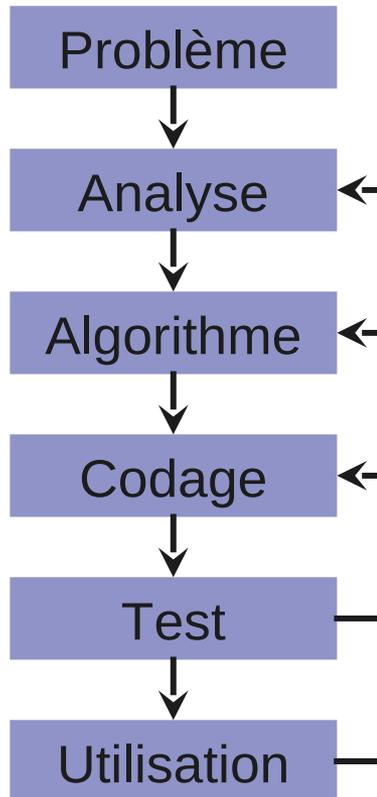
• Programmation événementielle

- Delphi, Visual Basic, Javascript

• Programmation logique

- Prolog

Du problème au programme : un exemple



Phase 1 : problème

- Calculer le montant d'un placement sur un compte rémunéré après un certain nombre d'années
- Exemple d'écran d'interaction
 - communication avec l'utilisateur du programme

Ce programme calcule le montant d'un placement sur un compte rémunéré

Donnez le montant du placement : 100

Donnez le taux d'intérêt (ex : 3 pour 3%) : 4

Donnez la durée en années : 4

Après 4 ans, le montant sera de : 112,55 euros

Phase 2 : analyse

- **Données d'entrée fournies**
 - un nombre représentant la valeur placée
 - un nombre représentant le taux d'intérêt (pour 10% : 10)
 - un nombre représentant une durée
- **Résultat souhaité**
 - un nombre représentant le montant après versement des intérêts, après une certaine durée
- **Démarche à adopter**
 - prendre connaissance de la somme initiale, du taux d'intérêt et de la durée
 - calculer le résultat :
 - calculer $1 + \text{taux}/100$
 - mettre le résultat à la puissance durée
 - multiplier le résultat par la somme initiale
 - afficher le nouveau montant ainsi obtenu

Phase 3 : algorithme

Algorithme *CalculDeRémunération*

Variables *MontantInitial*, *NouveauMontant*, *Durée*, *Taux* : réels
début

/ Saisie des données */*

Afficher "Ce programme calcule le montant d'un placement après un an sur un compte rémunéré "

Afficher "Donnez le montant du placement"

Saisir *MontantInitial*

Afficher "Donnez le taux d'intérêt (ex : 3 pour 3%)"

Saisir *Taux*

Afficher "Donnez la durée en années"

Saisir *Durée*

/ Calcul à effectuer */*

$NouveauMontant \leftarrow MontantInitial \times (1 + Taux / 100)^{Durée}$

/ Affichage du résultat */*

Afficher "Après" & *Durée* "ans, le montant sera de : " & *NouveauMontant* & "Euros"

fin

Phase 4 : codage

- Par exemple en Javascript

```
...
<script>
function placement() {
montant = parseFloat ( window.prompt("Entrez le montant initial : " ) );
taux = parseFloat ( window.prompt("Entrez le taux en pourcents : " ) );
duree = parseInt ( window.prompt("Entrez la durée en années : " ) );
mntfinal = montant * Math.pow ( 1+taux/100,duree);
alert ("Après " + duree + " an(s), le montant sera de : " +mntfinal);
}
</script>
<h1>Placement</h1>
<p>Ce programme calcule...<p>
<form name="placementform">
<input type="button" value="Calculer" onClick="placement()">
</form>
...
```

Phase 5 : simulation de fonctionnement

	Simulation 1				
<i>MontantInitial</i>	-	100	100	100	100
<i>Taux</i>	-	-	5	5	5
<i>Durée</i>	-	-	-	1	1
<i>NouveauMontant</i>	-	-	-	-	105
<i>Affichage</i>					... 105 €

	Simulation 2				
<i>MontantInitial</i>	-	200	200	200	200
<i>Taux</i>	-	-	10	10	10
<i>Durée</i>	-	-	-	5	5
<i>NouveauMontant</i>	-	-	-	-	322,102
<i>Affichage</i>					... 322,102 €

Variables

- D'un point de vue matériel
 - zone de stockage en mémoire centrale
 - définie par son nom et son type (entier, réel, caractère, ...)

B	12
A	1
MontantInitial	100
Taux	5
UneLettre	a
NouveauMontant	105

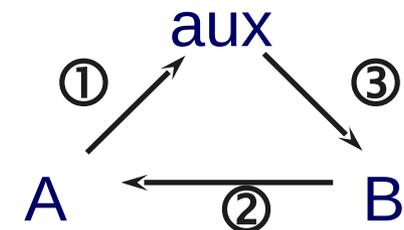
Tableau	1,75	0
	1,50	1
	2,01	2
	1,84	3
	1,61	4
	1,55	5
	1,78	6

Affectation et initialisation des variables

- Le programmeur peut
 - remplir la zone mémoire en lui attribuant une valeur
 - modifier à tout moment le contenu de la zone mémoire en changeant de valeur
 - consulter la valeur contenue dans la zone mémoire (uniquement si elle est remplie)
- Affectation
 - c'est le processus par lequel on attribue une valeur à une variable
- Initialisation
 - c'est le processus par lequel on attribue une première valeur à une variable

Affectation

- On donne une valeur à une variable
 - le contenu de la variable est modifié
 - la valeur précédente est définitivement perdue
- Exemples
 - $\text{variable} \leftarrow \text{valeur}$
 - $\text{variable} \leftarrow \text{variable}$
 - $\text{variable} \leftarrow \text{résultat du calcul}$
 - incrémentation
 - $\text{compteur} \leftarrow \text{compteur} + 1$
 - permutation du contenu de 2 variables A et B
 - besoin d'une variable auxiliaire (aux)



Catégories de variables

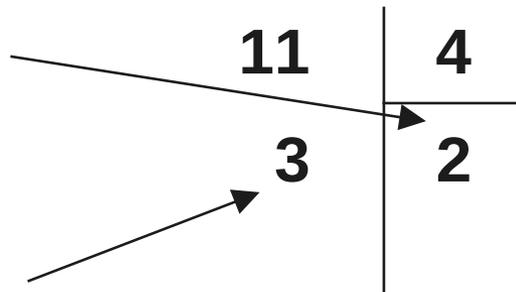
- 3 catégories de variables
 - les données (entrée)
 - les résultats (sortie)
 - les utilitaires (données intermédiaires)
- Constantes
 - variables dont la valeur est fixe (pi, taux de TVA...)
 - définies dès le début du programme
 - ne peuvent être modifiées pendant l'exécution du programme

Types de variables

- entier
 - 23 ; 0 ; 3
- réel
 - -104,324 ; 0,25
- caractère
 - 'a' ; 'A' ; '1' ; '?'
- chaîne de caractères
 - "caractère" ; "c" ; ""
- booléen
 - vrai ; faux
- la taille de la zone de stockage dépend du type de la variable

Opérateurs

- Addition
- Soustraction
- Multiplication
- Division réelle
 - $11/4 \rightarrow 2,75$
- Division entière (euclidienne)
 - sur des entiers
 - $11 \text{ DIV } 4 \rightarrow 2$
- Reste de la division entière
 - sur des entiers
 - $11 \text{ RESTE } 4 \rightarrow 3$



Relations d'ordre

- Relations d'ordre
 - égal =
 - différent \neq ($\langle \rangle$, $!=$)
 - supérieur $>$
 - supérieur ou égal \geq ($>=$)
 - inférieur $<$
 - inférieur ou égal \leq
- Attention
 - on ne compare que des éléments de types compatibles

Opérateurs logiques

- Opérateurs logiques
 - ET
 - OU
 - NON
 - exemple : $a=b$ ou $a=c$
- Tables de vérité
 - X et Y, 2 variables booléennes

X	Y	X ET Y	X OU Y	NON X
V	V	V	V	F
V	F	F	V	F
F	V	F	V	V
F	F	F	F	V

- Lois de De Morgan
 - $\text{NON}(A \text{ ET } B) \Leftrightarrow (\text{NON } A) \text{ OU } (\text{NON } B)$
 - $\text{NON}(A \text{ OU } B) \Leftrightarrow (\text{NON } A) \text{ ET } (\text{NON } B)$

Conditionnelle : si...alors

si condition(s) alors

Condition respectée

instruction 1

instruction n

finsi

instruction

Condition non respectée

Conditionnelle : si...alors

si condition(s) alors

Condition respectée

instruction 1

instruction n

Condition = expression
booléenne valant vrai ou faux

finsi

instruction

Condition non respectée

Conditionnelle : si...alors...sinon

si condition(s) alors

instruction 1

|

instruction n

Condition respectée

Condition non respectée

sinon

instruction 1

|

instruction n

finsi

Conditionnelle : exemple

```
/* Saisie des données */
```

```
Afficher "Calcul du résultat de la division de 2 entiers"
```

```
Afficher " Donnez le premier entier "
```

```
Saisir A
```

```
Afficher " Donnez le deuxième entier "
```

```
Saisir B
```

```
/* Calcul et affichage du résultat */
```

```
si B  $\neq$  0 alors
```

```
    | Resultat  $\leftarrow$  A / B
```

```
    | Afficher Resultat
```

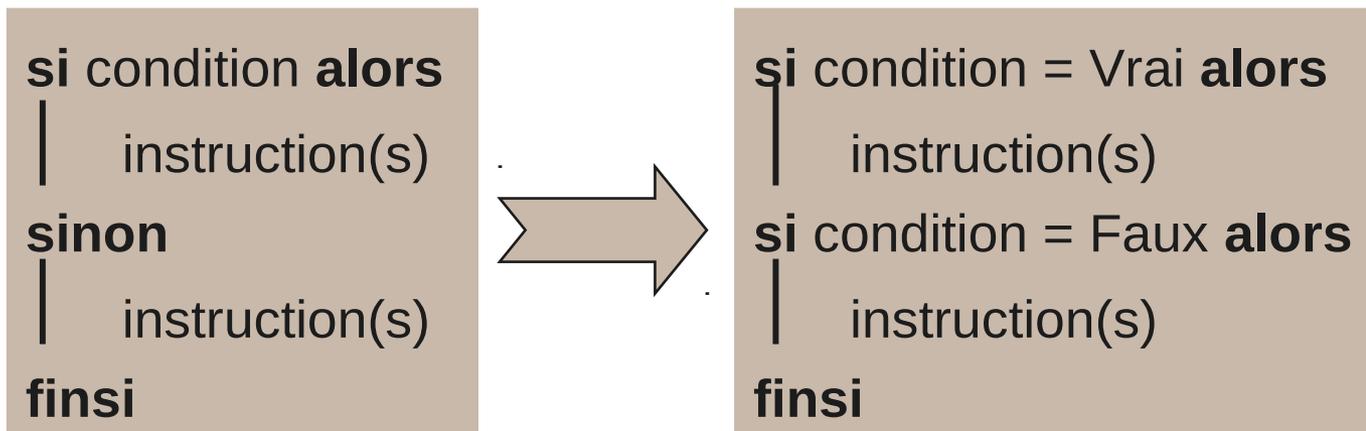
```
sinon
```

```
    | Afficher " Impossible de diviser un nombre par 0"
```

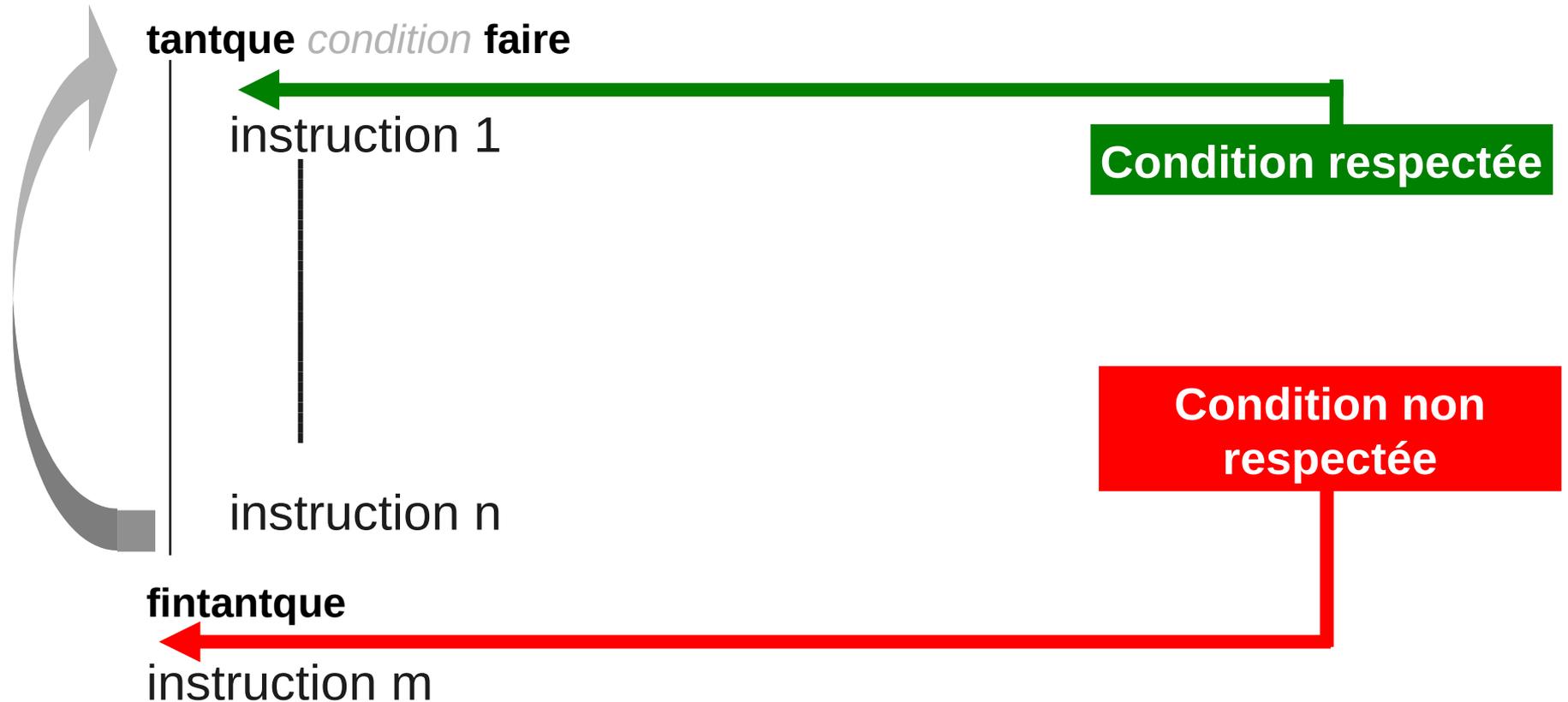
```
finsi
```

Conditions et expressions booléennes

- La condition est une expression booléenne
- Elle renvoie une valeur booléenne
 - Vrai
 - Faux



Instruction de répétition : tantque



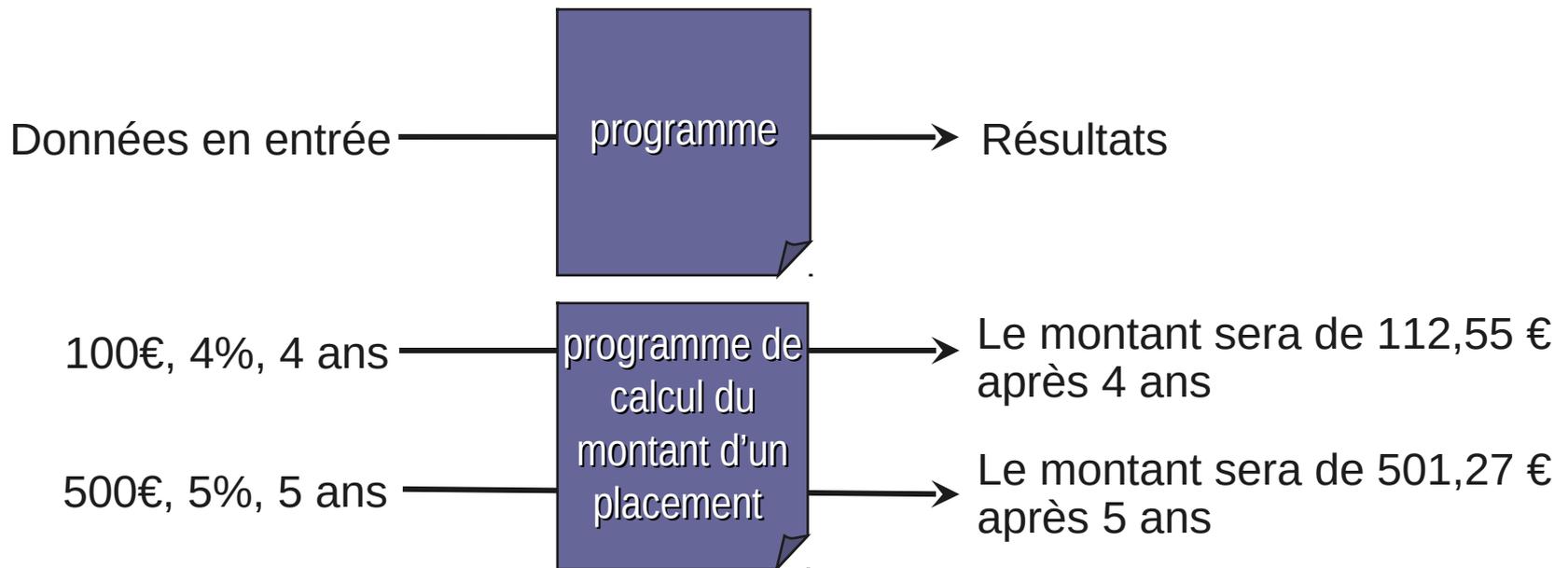
Attention à prévoir la sortie de la boucle (boucle infinie)

Instruction de répétition : exemple

```
je monte dans le tram
tant que (arrêt ≠ "Université Lyon 1") faire
    | je me tiens à une barre
    | je surveille les arrêts
fin tantque
je descends du tram
```

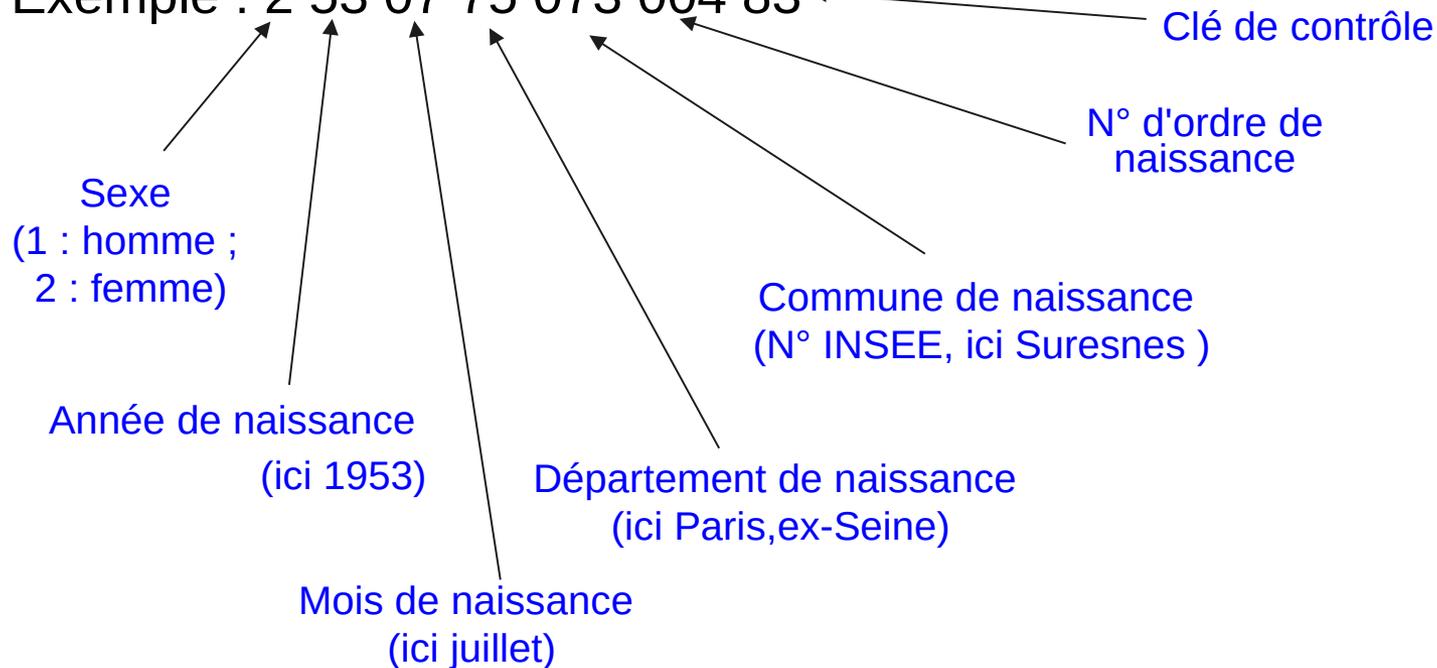
Appel de programme et paramètres

- On peut appliquer un programme à des données différentes
 - les paramètres
 - exemples
 - calcul du montant d'un placement sur un compte rémunéré



Exemple : numéro de sécurité sociale

- Aussi appelé NIR (Numéro d'Inscription au Registre)
- Numéro unique composé de 13 caractères suivis d'une clé de contrôle
- Exemple : 2 53 07 75 073 004 83



Exemple : décomposition du numéro de SS

- Problème
 - décomposer mon numéro de Sécurité Sociale
- Données en entrée
 - numéro de Sécurité Sociale
- Résultat
 - le sexe, l'année et le mois de naissance, l'âge de l'assuré
- Méthode
 - trouver le sexe associé au premier chiffre du numéro
 - trouver l'année associée au deux chiffres suivants
 - trouver le mois associé au deux chiffres suivants
 - calculer l'âge de l'assuré

Exemple : algorithme

demander numeroSS

sexe ← 1er caractère de numeroSS

annee ← 2ème et 3ème caractères de numeroSS

mois ← 4ème et 5ème caractères de numeroSS

departement ← 6ème et 7ème caractères de numeroSS

afficher "Vous êtes un"

si sexe="1" **alors**

afficher "homme"

sinon

afficher "e femme"

finsi

afficher "né(e) en"

selon mois

01 : afficher "janvier"

02 : afficher "février"

03 : afficher "mars"

04 : afficher "avril"

05 : afficher "mai"

06 : afficher "juin"

07 : afficher "juillet"

08 : afficher "août"

09 : afficher "septembre"

10 : afficher "octobre"

11 : afficher "novembre"

12 : afficher "décembre"

autre : afficher "erreur sur le mois"

finselon

anneeNaissance ← "19" & annee

afficher anneeNaissance

age ← 0

anneeCourante ← 2010

tantque (age + anneeNaissance < anneeCourante) **faire**

 age ← age + 1

finTantque

afficher "Vous avez " & age & "ans"

- 1. Introduction
 - 2. Définitions
 - 3. Du problème au programme
 - 4. Pour programmer
- 1. Variables et opérateurs
 - 2. Conditionnelles et répétitions
 - 3. Appel avec paramètres
 - 4. Exemples

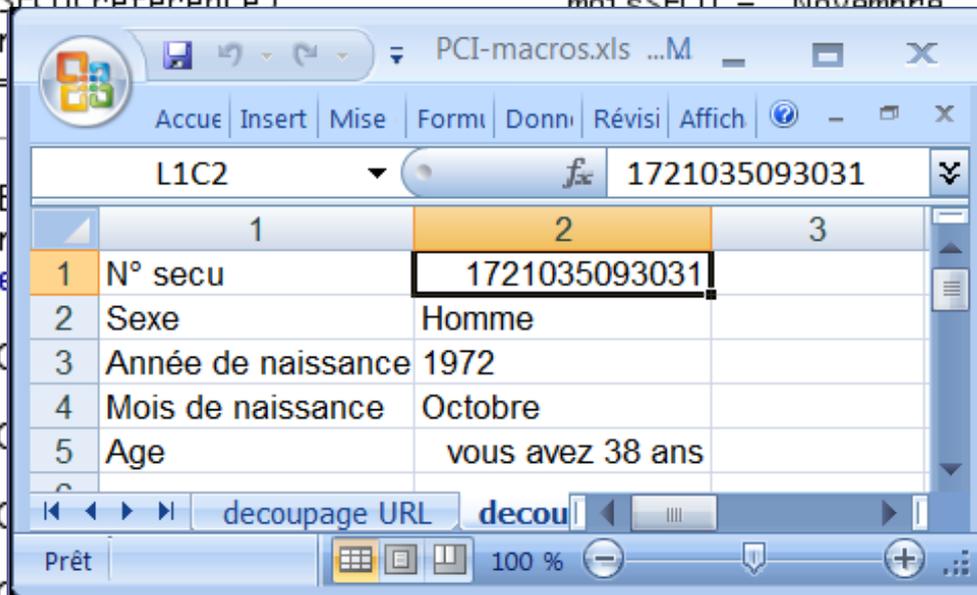
Exemple : codage avec Excel

```
Function sexeSECU(reference)
    ch = CStr(reference.Value)
    If Left(ch, 1) = "1" Then
        sexeSECU = "Homme"
    ElseIf Left(ch, 1) = "2" Then
        sexeSECU = "Femme"
    Else
        sexeSECU = "Indéfini"
    End If
End Function
```

```
Function anneeSECU(reference)
    ch = CStr(reference.Value)
    anneeSECU = ...
End Function
```

```
Function moisSECU(reference)
    ch = CStr(reference.Value)
    Select Case ch
    Case "1"
        moisSECU = "Janvier"
    Case "2"
        moisSECU = "Février"
    Case "3"
        moisSECU = "Mars"
    Case "4"
        moisSECU = "Avril"
    Case "5"
        moisSECU = "Mai"
    Case "6"
        moisSECU = "Juin"
    Case "7"
        moisSECU = "Juillet"
    Case "8"
        moisSECU = "Août"
    Case "9"
        moisSECU = "Septembre"
    Case "10"
        moisSECU = "Octobre"
    Case "11"
        moisSECU = "Novembre"
    Case "12"
        moisSECU = "Décembre"
    End Select
End Function
```

```
Case "6"
    moisSECU = "Juin"
Case "7"
    moisSECU = "Juillet"
Case "8"
    moisSECU = "Août"
Case "9"
    moisSECU = "Septembre"
Case "10"
    moisSECU = "Octobre"
Case "11"
    moisSECU = "Novembre"
```



```
ageSECU = "vous avez " & age & " ans"
End Function
```

Exemple : codage en JavaScript

Démonstration

The screenshot shows a web browser window with the address bar containing `http://pci.univ-lyon1.fr/uploads`. The page title is "Analyse clé sécu". The main content of the page is titled "N° Sécurité sociale" and "Analyse d'un numéro de Sécurité Sociale". It instructs the user to enter their social security number in a text input field and click the "Calculer" button. The input field contains the number "1861255454".

An alert dialog box is displayed in the foreground, titled "Alerte http://pci.univ-lyon1.fr/". The message inside the dialog reads: "Vous êtes un homme, né en décembre 1986" and "Vous avez 24 ans". There is an "OK" button at the bottom right of the dialog.

```
...
<script language="javascript">
function decoupage () {
numSECU =
document.forms["saisie"].elements["secu"].value
sexe = parseInt (numSECU.substr(0,1));
annee = parseInt(numSECU.substr (1,2));
mois = parseInt(numSECU.substr(3,2));
dept = parseInt(numSECU.substr (5,2));

texte = "Vous êtes un";
if (sexe ==1) texte = texte + " homme, né en ";
else texte = texte + "e femme, née en ";

switch (mois) {
case 1: texte = texte + "janvier"; break;
case 2: texte = texte + "février"; break;
case 3: texte = texte + "mars"; break;
case 4: texte = texte + "avril"; break;
case 5: texte = texte + "mai"; break;
case 6: texte = texte + "juin"; break;
case 7: texte = texte + "juillet"; break;
case 8: texte = texte + "août"; break;
case 9: texte = texte + "septembre"; break;
case 10: texte = texte + "octobre"; break;
case 11: texte = texte + "novembre"; break;
case 12: texte = texte + "décembre"; break;
}
}
```

```
annee = annee + 1900;
texte = texte + " " + annee;

age = 0;
while (annee < 2010 ) { age ++; annee ++; }
texte = texte + "\nVous avez " + age + " ans";

alert(texte);
}
</script>
```

```
<h1>N° Sécu</h1>
<p>Ce programme calcule...</p>
<form name="saisie">
<input type="text" name="secu" value="">
<input type="button" value="Calculer"
onClick="decoupage()">
</form>
```

```
...
```