


*Éléments d'informatique – Cours 3*  
*La programmation structurée en langage C*  
*L'instruction de contrôle if*

Pierre Boudes

12 octobre 2011



- Éléments d'architecture des ordinateurs (+mini-assembleur) 
- Éléments de systèmes d'exploitation
- Programmation structurée impérative (éléments de langage C)
  - Structure d'un programme C
  - Variables : déclaration (et initialisation), affectation
  - Évaluation d'expressions
  - Instructions de contrôle : if, for, while
  - Types de données : entiers, caractères, réels, tableaux, enregistrements
  - Fonctions d'entrées/sorties (scanf/printf)
  - Écriture et appel de fonctions
  - Débogage
- Notions de compilation
  - Analyse lexicale, analyse syntaxique, analyse sémantique
  - préprocesseur du compilateur C (include, define)
  - Édition de lien
- Algorithmes élémentaires
- Méthodologie de résolution, manipulation sous linux

## *Liens utiles*

- ma page : <http://www-lipn.univ-paris13.fr/~boudes/>
- Un livre de la BU : *Le livre du C, premier langage (pour les vrais débutants en programmation)*, Claude Delannoy.
- <http://www.siteduzero.com/> (chercher langage C)
- <http://www.developpez.com/> (chercher langage C)
- le cours de Anne Canteaut :  
[http://www-roc.inria.fr/secret/Anne.Canteaut/COURS\\_C/](http://www-roc.inria.fr/secret/Anne.Canteaut/COURS_C/)
- le cours de Bernard Cassagne :  
[http://clips.imag.fr/commun/bernard.cassagne/Introduction\\_ANSI\\_C.html](http://clips.imag.fr/commun/bernard.cassagne/Introduction_ANSI_C.html)
- le cours de Henri Garreta :  
<http://www.dil.univ-mrs.fr/~garreta/generique/>
- codeblocks : <http://www.codeblocks.org/>
- ubuntu : <http://www.ubuntu-fr.org/>
- virtualbox : <http://www.virtualbox.org/>

## *La programmation structurée*

### *Programmation en C*

Structure d'un programme C

Variables impératives et traduction de l'affectation

L'instruction de contrôle if

### *Demos*

### *Divers*

# *La programmation structurée*

## *Definition (Programmation structurée)*

Programmer par *blocs* d'instructions en combinant ces blocs de trois manières :

## *La programmation structurée*

### *Definition (Programmation structurée)*

Programmer par *blocs* d'instructions en combinant ces blocs de trois manières :

1. exécuter les blocs les uns à la suite des autres (*séquence*)

## *La programmation structurée*

### *Definition (Programmation structurée)*

Programmer par *blocs* d'instructions en combinant ces blocs de trois manières :

1. exécuter les blocs les uns à la suite des autres (*séquence*)
2. si une certaine condition est vraie, exécuter un bloc sinon en exécuter un autre (*sélection*)

## *La programmation structurée*

### *Definition (Programmation structurée)*

Programmer par *blocs* d'instructions en combinant ces blocs de trois manières :

1. exécuter les blocs les uns à la suite des autres (*séquence*)
2. si une certaine condition est vraie, exécuter un bloc sinon en exécuter un autre (*sélection*)
3. recommencer l'exécution d'un bloc tant qu'une certaine condition est vraie (*répétition*).



## *La programmation structurée*

### *Definition (Programmation structurée)*

Programmer par *blocs* d'instructions en combinant ces blocs de trois manières :

1. exécuter les blocs les uns à la suite des autres (*séquence*)
2. si une certaine condition est vraie, exécuter un bloc sinon en exécuter un autre (*sélection*)
3. recommencer l'exécution d'un bloc tant qu'une certaine condition est vraie (*répétition*).

Un bloc peut lui-même contenir une combinaison de blocs.

## *La programmation structurée*

### *Definition (Programmation structurée)*

Programmer par *blocs* d'instructions en combinant ces blocs de trois manières :

1. exécuter les blocs les uns à la suite des autres (*séquence*)
2. si une certaine condition est vraie, exécuter un bloc sinon en exécuter un autre (*sélection*)
3. recommencer l'exécution d'un bloc tant qu'une certaine condition est vraie (*répétition*).

Un bloc peut lui-même contenir une combinaison de blocs.

Cette idée simple conduisit à l'introduction de langages dits de haut niveau tel le langage C.

## La programmation structurée

### Definition (Programmation structurée)

Programmer par *blocs* d'instructions en combinant ces blocs de trois manières :

1. exécuter les blocs les uns à la suite des autres (*séquence*)
2. si une certaine condition est vraie, exécuter un bloc sinon en exécuter un autre (*sélection*)
3. recommencer l'exécution d'un bloc tant qu'une certaine condition est vraie (*répétition*).

Un bloc peut lui-même contenir une combinaison de blocs.

Cette idée simple conduisit à l'introduction de langages dits de haut niveau tel le langage C.

Ces langages nécessitent une **traduction en langage machine**, par un compilateur ou bien un interprète.

## La programmation structurée

### Definition (Programmation structurée)

Programmer par *blocs* d'instructions en combinant ces blocs de trois manières :

1. exécuter les blocs les uns à la suite des autres (*séquence*)
2. si une certaine condition est vraie, exécuter un bloc sinon en exécuter un autre (*sélection*)
3. recommencer l'exécution d'un bloc tant qu'une certaine condition est vraie (*répétition*).

Un bloc peut lui-même contenir une combinaison de blocs.

Cette idée simple conduisit à l'introduction de langages dits de haut niveau tel le langage C.

Ces langages nécessitent une **traduction en langage machine**, par un compilateur ou bien un interprète.

Aujourd'hui nous allons voir la sélection en langage C, le **if else**.



## Structure d'un programme C

### Code source

```
/* Declaration de fonctionnalites supplementaires */
#include <stdlib.h> /* EXIT_SUCCESS */

/* Declaration des constantes et types utilisateurs */

/* Declaration des fonctions utilisateurs */

/* Fonction principale */
int main()
{
    /* Declaration et initialisation des variables */
    ...
    /* valeur fonction */
    return EXIT_SUCCESS;
}

/* Definitions des fonctions utilisateurs */
```

Les commentaires sont ignorés lors de la traduction en langage machine.



## *Traduction de l'affectation (rappel du TD 2)*

### *Code source*

```
...  
/* Fonction principale */  
int main()  
{  
    /* Declaration et initialisation des variables */  
    int x = 5;  
  
    x = 3 * x + 1;  
    ...  
}
```

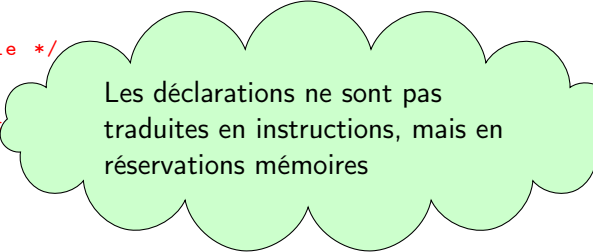


## Traduction de l'affectation (rappel du TD 2)

### Code source

```
...  
/* Fonction principale */  
int main()  
{  
    /* Declaration et  
    int x = 5;  
  
    x = 3 * x + 1;  
    ...
```

### Schéma de traduction



Les déclarations ne sont pas traduites en instructions, mais en réservations mémoires



## Traduction de l'affectation (rappel du TD 2)

### Code source

```
...  
/* Fonction principale */  
int main()  
{  
    /* Declaration et initialisation  
    int x = 5;
```

```
    x = 3 * x + 1;  
    ...
```

### Schéma de traduction

*x est une case mémoire de type entier qui contient initialement 5*





## Traduction de l'affectation (rappel du TD 2)

### Code source

```
...  
/* Fonction principale */  
int main()  
{  
    /* Declaration et initialisation  
    int x = 5;
```

```
    x = 3 * x + 1;  
    ...  
}
```

### Schéma de traduction

*x est une case mémoire de type entier qui contient initialement 5*

l'affectation est traduite par  
l'évaluation d'une expression et  
une écriture en mémoire



## Traduction de l'affectation (rappel du TD 2)

### Code source

```
...  
/* Fonction principale */  
int main()  
{  
    /* Declaration et initialisation  
    int x = 5;
```

```
    x = 3 * x + 1;  
    ...
```

### Schéma de traduction

*x est une case mémoire de type entier qui contient initialement 5*

*évaluation de l'expression  $3x + 1$  dans un registre*



## Traduction de l'affectation (rappel du TD 2)

### Code source

```
...  
/* Fonction principale */  
int main()  
{  
    /* Declaration et initialisation  
    int x = 5;
```

```
    x = 3 * x + 1;  
    ...
```

### Schéma de traduction

*x est une case mémoire de type entier qui contient initialement 5*

*évaluation de l'expression  $3x + 1$  dans un registre*

*écriture du résultat à l'adresse de x*



## *L'instruction de contrôle if*

*Syntaxe :* `if (condition) { bloc1 } else { bloc2 }.`



## *L'instruction de contrôle if*

*Syntaxe* : `if (condition) { bloc1 } else { bloc2 }.`

*Code source*

```
/* avant */
if (age < 18)
{
    permis = 0;
}
else
{
    permis = 1;
}
/* après */
```



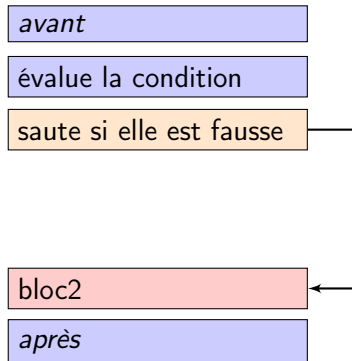
## *L'instruction de contrôle if*

*Syntaxe* : `if (condition) { bloc1 } else { bloc2 }.`

### *Code source*

```
/* avant */  
if (age < 18)  
{  
    permis = 0;  
}  
else  
{  
    permis = 1;  
}  
/* après */
```

### *Schéma de traduction*





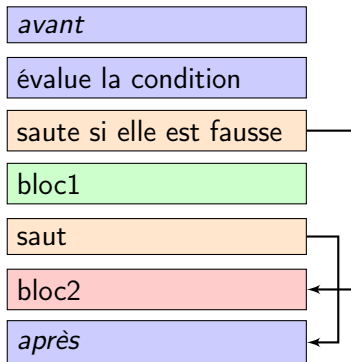
## *L'instruction de contrôle if*

*Syntaxe* : `if (condition) { bloc1 } else { bloc2 }.`

### *Code source*

```
/* avant */  
if (age < 18)  
{  
    permis = 0;  
}  
else  
{  
    permis = 1;  
}  
/* après */
```

### *Schéma de traduction*





# *Démos*



## *Divers*

- À partir de cette semaine, il faudra préparer le TP à l'avance, et penser à prendre des notes manuscrites pendant la séance de TP.

## *Divers*

- À partir de cette semaine, il faudra préparer le TP à l'avance, et penser à prendre des notes manuscrites pendant la séance de TP.
- Il y a une méthodologie de résolution qu'il faudra appliquer (faire des exemples, écrire un algorithme, traduire en langage C, tester). Le contrôle de TP aura sans doute lieu dans deux semaines.

## *Divers*

- À partir de cette semaine, il faudra préparer le TP à l'avance, et penser à prendre des notes manuscrites pendant la séance de TP.
- Il y a une méthodologie de résolution qu'il faudra appliquer (faire des exemples, écrire un algorithme, traduire en langage C, tester). Le contrôle de TP aura sans doute lieu dans deux semaines.
- Pour vous entraîner à programmer en langage C sur un ordinateur personnel, il y a par exemple la solution codeblocks.

## *Divers*

- À partir de cette semaine, il faudra préparer le TP à l'avance, et penser à prendre des notes manuscrites pendant la séance de TP.
- Il y a une méthodologie de résolution qu'il faudra appliquer (faire des exemples, écrire un algorithme, traduire en langage C, tester). Le contrôle de TP aura sans doute lieu dans deux semaines.
- Pour vous entraîner à programmer en langage C sur un ordinateur personnel, il y a par exemple la solution codeblocks.
- Alternative plus difficile : virtual box et une ubuntu.