R101 — Introduction au développement

Initiation à la programmation en langage C
Structures de contrôle

Clément Pagès
clement.pages@u-pec.fr

Université Paris-Est Créteil
BUT Informatique 1re année

2025-2026

Licence CC-BY-SA

Sommaire

1 Structures conditionnelles

2 Structures itératives

Licence CC-BY-SA

Structure `if`

```c
int n = 47;
if(n%5==0) { // Accolade ouvrante
    printf("%d est un multiple de 5.\n");
} // Accolade fermante
```


Licence CC-BY-SA

Structure if

```c
int n = 47;
if(n%5==0) { // Accolade ouvrante
    printf("%d est un multiple de 5.\n");
} // Accolade fermante
```

```c
int heure = 14;
if(heure>=8 && heure<=20) { // Accolade ouvrante
    printf("Il fait jour !\n");
    if(heure>=14 && heure <=17)
        printf("C'est l'heure de la sieste !\n");
} // Accolade fermante
```


Licence CC-BY-SA

Structure `if`

```c
int n = 47;
if(n%5==0) { // Accolade ouvrante
    printf("%d est un multiple de 5.\n");
} // Accolade fermante
```

```c
int heure = 14;
if(heure>=8 && heure<=20) { // Accolade ouvrante
    printf("Il fait jour !\n");
    if(heure>=14 && heure <=17)
        printf("C'est l'heure de la sieste !\n");
} // Accolade fermante
```

- Les accolades délimitent le début et la fin du bloc.
- Elles sont facultatives lorsque le bloc ne contient qu'une seule instruction.
- L'intendation améliore la lisibilité du code source.

Licence CC-BY-SA

Structure `if else`

```c
int n = 47;
if(n%5 == 0)
    printf("%d est un multiple de 5.\n");
else
    printf("%d n'est pas un multiple de 5.\n");
```


Licence CC-BY-SA

Structure `if else`

```c
int n = 47;
if(n%5 == 0)
    printf("%d est un multiple de 5.\n");
else
    printf("%d n'est pas un multiple de 5.\n");
```

```c
int heure = 14;
if(heure>=8 && heure<=20) { // Accolade ouvrante
    printf("Il fait jour !\n");
    if(heure>=14 && heure <=17)
        printf("C'est l'heure de la sieste !\n");
    else
        printf("La journée s'annonce longue…")
} // Accolade fermante
else {
    printf("Il fait nuit.");
    if(heure==23)
        printf("Les étudiant.es déposent leur devoir à rendre…");
}
```


Licence CC-BY-SA

Opérateur ? :

Une alternative synthétique à `if else`

Les trois codes suivants sont équivalents.

```c
int a=2, b=3, max;

if(a>b) {
    max = a;
}
else {
    max = b;
}
```


Licence CC-BY-SA

Opérateur ? :

Une alternative synthétique à `if else`

Les trois codes suivants sont équivalents.

```c
int a=2, b=3, max;

if(a>b) {
    max = a;
}
else {
    max = b;
}
```

```c
int a=2, b=3, max;

if(a>b) max = a;
else    max = b;
```

```c
int a=2, b=3, max;

max = (a>b)? a : b;
```


Licence CC-BY-SA

Structure `if else if else`

```c
int heure = 11;

if (heure > 0 && heure < 7)
    printf("Sommeil paradoxal… \n");
else if (heure >= 7 && heure < 12)
    printf("C'est le matin !\n");
else if (heure == 12)
    printf("Il est midi !\n");
else if (heure > 12 && heure < 18)
    printf("C'est l'après-midi !\n");
else if (heure >= 18 && heure < 24)
    printf("C'est le soir !\n");
else if (heure == 24 || heure == 0)
    printf("Il est minuit.\n");
else
    printf("Il y a une erreur quelque part !\n");
```


Licence CC-BY-SA

Structure `switch` : comparaison à des constantes entières I

```c
int moyenDeTransport = 0;

printf("Comment venez-vous à l'IUT?\n");
printf("\t 1. Transports en commun.\n"); // \t :  tabulation
printf("\t 2. Marche à pied.\n");
printf("\t 3. Vélo.\n");
printf("\t 4. Voiture.\n");
printf("\t 5. Autre\n");

printf("Réponse : ");
scanf("%d", &moyenDeTransport); // Valeur saisie au clavier

// Suite page suivante…
```


Licence CC-BY-SA

Structure `switch` : comparaison à des constantes entières II

```c
// Suite de la page précédente…

switch(moyenDeTransport) {
    case 1: // Transports en commun
        printf("Félicitations, c'est bon pour la planète !");
        break; // Pour sortir du switch.
    case 2: // Marche à pied
    case 3: // Vélo
        // On gère les cas 2 et 3 en même temps
        printf("Bravo, c'est bon pour la santé et la planète.");
        printf("Le vélo est le plus rapide en ville");
        break; // Pour sortir du switch.
    case 4:
        printf("N'avez-vous pas d'autre alternative ?\);
        break; // Pour sortir du switch.
    case 5:
        printf("Vous venez en tapis volant ?!");
        break;
    default: // N'importe quel autre valeur : erreur de saisie.
        printf("Il doit y avoir une erreur…");
```


Licence CC-BY-SA

Sommaire

1 Structures conditionnelles

2 Structures itératives

Licence CC-BY-SA

Les boucles

- On a souvent besoin de répéter des instructions plusieurs fois.
- Exemples de problèmes-types :
  - Calculer la somme des entiers de 1 à 3 000 :

$$1 + 2 + 3 + \cdots + 2\,998 + 2\,999 + 3\,000 = ?$$

  - Écrire les tables de multiplication jusqu'à 50.
- Il existe trois structures itératives en C :
  - `while(condition) { instructions; }`
    Répète les instructions tant que la condition est réalisée.
  - `do { instructions; while(condition)}`
    Exécute les instructions *puis* les répète tant que la condition est réalisée.
  - `for(i = 1 ; i<=10 ; i++) { instructions; }`
    Répète les instructions pour toutes les valeurs de `i` entre 1 et 10.

Licence CC-BY-SA

Les boucles

La boucle `while`

- Analogue de la boucle `TANT QUE … FAIRE` (cf. R113).

Licence CC-BY-SA

Les boucles

La boucle `while`

- Analogue de la boucle `TANT QUE ... FAIRE` (cf. R113).

C

```c
int i = 0;
while (i<5) {
    printf("i = %d\n", i);
    i = i+1;
}
printf("Fin");
```

Console

```
i = 0
i = 1
i = 2
i = 3
i = 4
Fin
```


Licence CC-BY-SA

Les boucles

La boucle `while`

- Analogue de la boucle `TANT QUE ... FAIRE` (cf. R113).

```c
int i = 0;
while (i<5) {
    printf("i = %d\n", i);
    i = i+1;
}
printf("Fin");
```

```
i = 0
i = 1
i = 2
i = 3
i = 4
Fin
```

```c
int i=6;
while (i<6) {
    printf("i = %d\n", i);
    i = i+1;
}
printf("Fin\n");
```

```
Fin
```


Licence CC-BY-SA

Les boucles

La boucle `do while`

- Analogue de la boucle `FAIRE ... TANT QUE` (cf. R113).

Licence CC-BY-SA

Les boucles

La boucle `do while`

- Analogue de la boucle `FAIRE ... TANT QUE` (cf. R113).

C

```c
int i=0;
do {
    printf("i = %d\n", i);
    i = i+1;
} while(i<6);
printf("Fin\n");
```

Console

```
i = 0
i = 1
i = 2
i = 3
i = 4
i = 5
Fin
```


Licence CC-BY-SA

Les boucles

La boucle `do while`

- Analogue de la boucle `FAIRE ... TANT QUE` (cf. R113).

```c
int i=0;
do {
    printf("i = %d\n", i);
    i = i+1;
} while(i<6);
printf("Fin\n");
```

```
i = 0
i = 1
i = 2
i = 3
i = 4
i = 5
Fin
```

```c
int i=7;
do {
    printf("i = %d\n", i);
    i = i+1;
} while (i<6);
printf("Fin\n");
```

```
i = 7
Fin
```


Licence CC-BY-SA

Les boucles

La boucle `for`

- Analogue de la boucle `POUR` (cf. R113).

Licence CC-BY-SA

Les boucles

La boucle `for`

- Analogue de la boucle POUR (cf. R113).

C

```c
for(int i=0 ; i<6 ; i++) {
    printf("i = %d", i);
}
```

Console

```
i = 0
i = 1
i = 2
i = 3
i = 4
i = 5
```


Licence CC-BY-SA

Les boucles

La boucle `for`

- Analogue de la boucle POUR (cf. R113).

```c
for(int i=0 ; i<6 ; i++) {
    printf("i = %d", i);
}
```

```
i = 0
i = 1
i = 2
i = 3
i = 4
i = 5
```

Boucles imbriquées

Que fait ce programme ?

```c
for(int i=0 ; i<=50 ; i++) {
    for(int j=0 ; j<=10 ; j++) {
        printf("%d×%d = %d", i, j, i*j);
    }
}
```


Licence CC-BY-SA

Retour au problème-type

Exercice

Écrire un programme qui calcule des entiers de 1 à 3 000, sans utiliser la formule mathématique correspondante (si vous la connaissez) :

$$1 + 2 + 3 + \cdots + 2\,998 + 2\,999 + 3\,000$$

Licence CC-BY-SA