

# Langage C (les bases)



Mise à jour - @September 27, 2021



tips and tricks



à connaître



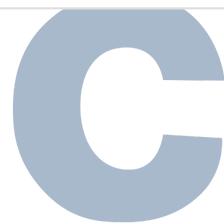
attention

C est un langage de programmation impératif généraliste, de bas niveau. Inventé au début des années 1970 pour réécrire UNIX, C est devenu un des langages les plus utilisés, encore de nos jours.

## C (langage)

C Date de première version Paradigme Impératif, procédural, structuré Auteur Dennis Ritchie Développeur Dennis Ritchie, Bell Labs Typage Statique, faible Normes ANSI X3.159-1989

W [https://fr.wikipedia.org/wiki/C\\_\(langage\)](https://fr.wikipedia.org/wiki/C_(langage))



PROGRAMMING

Vous pouvez faire avec tout ce que vous faites avec Scratch même beaucoup plus.

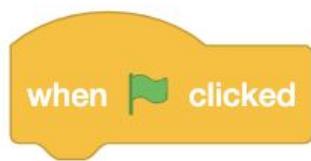
Nous avons vu comment dire `Hello World` en scratch et ci dessous exactement la même chose en langage C.



```
#include <stdio.h>

int main(void)
{
    printf("hello, world");
}
```

- Fonctions
- Conditions
- Expressions Booléennes
- Boucles
- ...



```
int main(void)
{
}
```

Ceci permet d'initier le programme tout ce que l'on va écrire aujourd'hui devra donc être entre ces accolades plutôt que sous une pièce de puzzle comme nous le faisions dans Scratch.

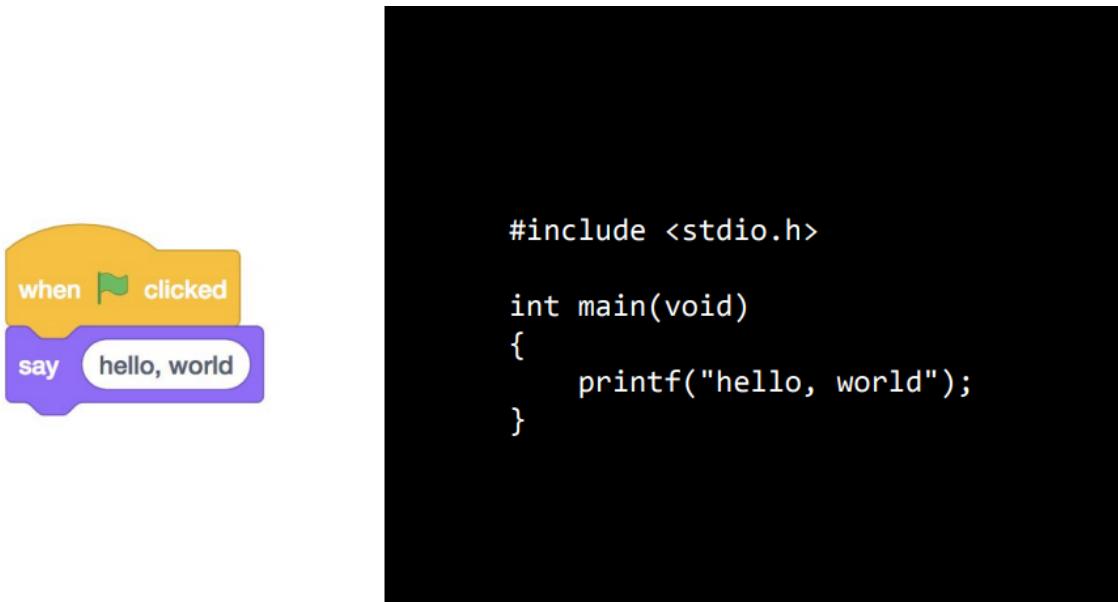
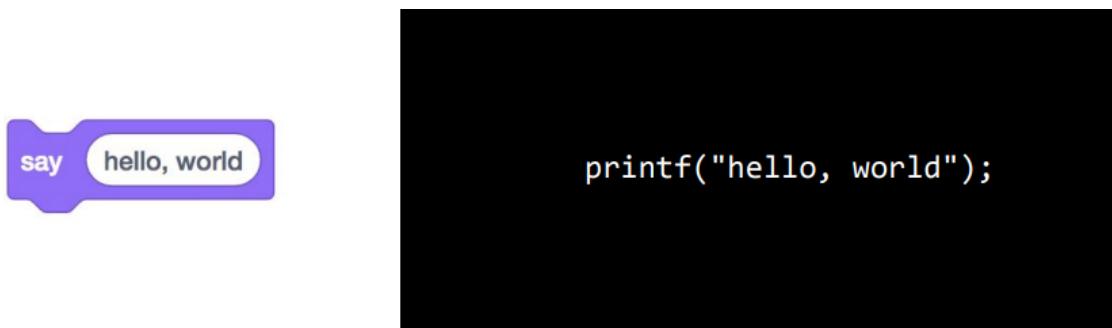
Il faut être conscient que ces deux choses sont exactement équivalentes.

En C, il n'y a pas de fonction Say donc nous allons utiliser `printf`

Nous pouvons décomposer cette fonction en `print` + `f`, print signifie imprimer et f signifie formater, donc littéralement imprimer/afficher un texte formaté.

Les parenthèses en C peuvent nous faire penser à la forme ovale de Scratch. Mais en C, il faut vraiment encapsuler le texte en ajoutant également des doubles guillemets.

Il faut également toujours finir par un point virgule, chose qui est souvent oublié.



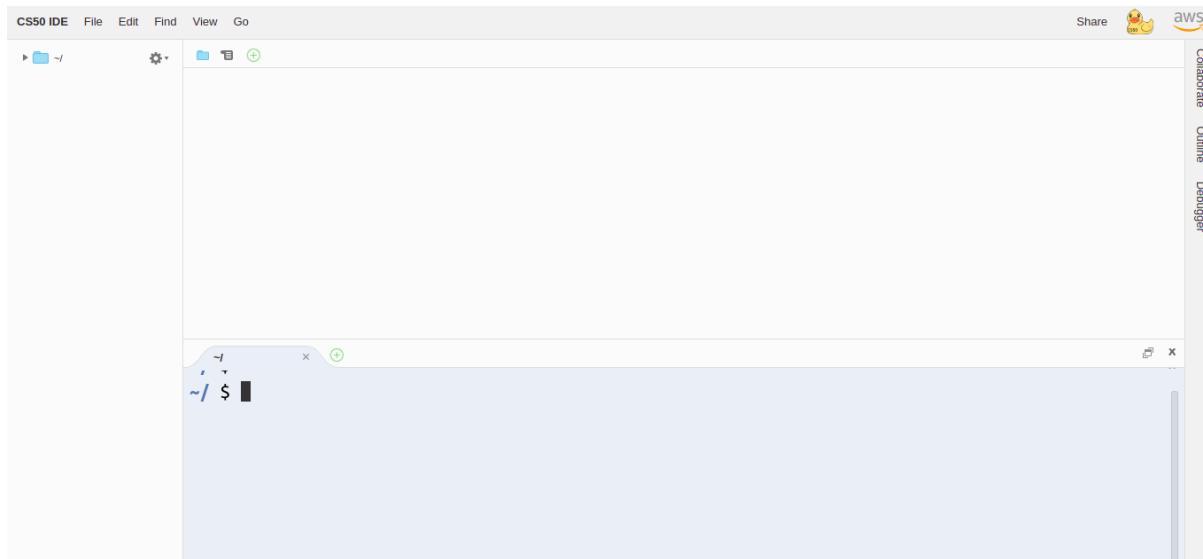
Vous devez informer à l'avance à votre ordinateur où cette fonction est implémentée, où elle a été enregistrée, il faut donc ajouter `#include <stdio.h>`

Nous allons travailler avec l'IDE de CS50 afin de tous travailler dans le même environnement comme nous le faisions avec Scratch.

## CS50 IDE

<https://ide.cs50.io/>

Vous pouvez voir trois parties, en haut la partie où sera écrit le code, la partie du bas sera la partie Terminal et d'exécution du code. Pour finir à gauche vous trouverez vos dossiers et vos fichiers de code.



Nous créons donc un nouveau fichier en cliquant sur le et nous allons appeler ce fichier `hello.c` le point C signifie par convention que c'est un programme en langage C.

A screenshot of the code editor in the CS50 IDE. The file 'hello.c' is open, showing the following C code:

```
1 #include <stdio.h>
2
3 int main(void)
4 {
5     printf("hello, world\n");
6 }
7
```

Nous venons d'écrire notre premier programme en C avec une total de six lignes de code.

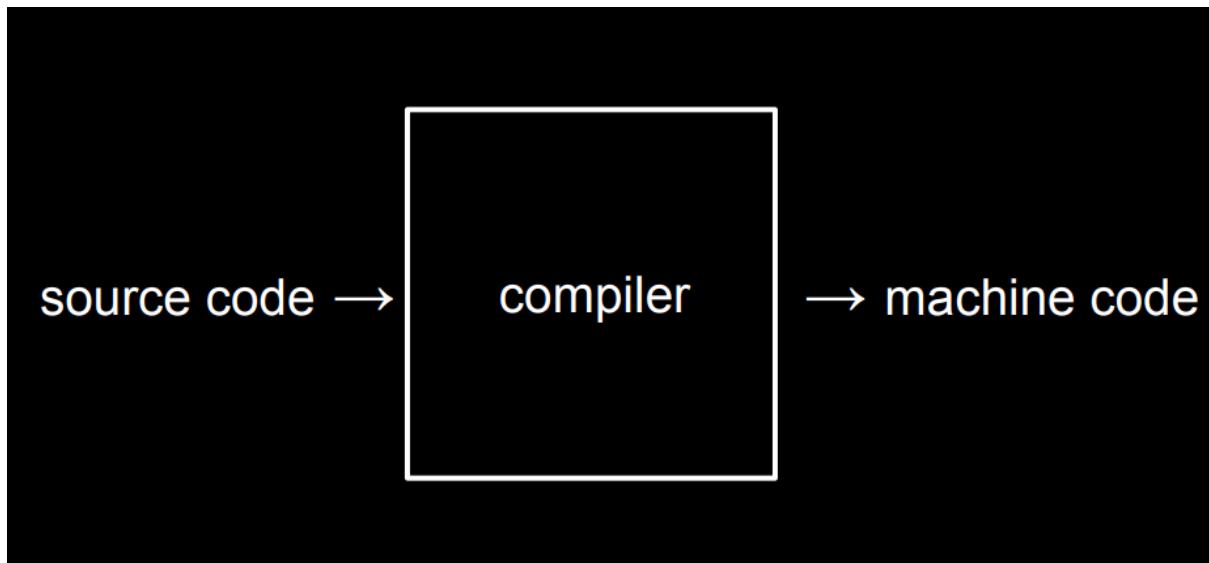
Maintenant nous allons exécuter ce code.

Pour rappel, un ordinateur ne comprend pas directement des lignes de texte / code il ne comprend que le binaire des et des .

il faut donc pouvoir convertir ce code en binaire.

```
01111111 01000101 01001100 01000110 00000010 00000001 00000001 00000000  
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000  
00000010 00000000 00111110 00000000 00000001 00000000 00000000 00000000  
10110000 00000101 01000000 00000000 00000000 00000000 00000000 00000000  
01000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000  
11010000 00010011 00000000 00000000 00000000 00000000 00000000 00000000  
00000000 00000000 00000000 00000000 01000000 00000000 00111000 00000000  
00001001 00000000 01000000 00000000 00100100 00000000 00100001 00000000  
00000110 00000000 00000000 00000000 00000101 00000000 00000000 00000000  
01000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000  
01000000 00000000 01000000 00000000 00000000 00000000 00000000 00000000  
11111000 00000001 00000000 00000000 00000000 00000000 00000000 00000000  
11111000 00000001 00000000 00000000 00000000 00000000 00000000 00000000  
00001000 00000000 00000000 00000000 00000000 00000000 00000000 00000000  
00000011 00000000 00000000 00000000 00000100 00000000 00000000 00000000  
00111000 00000010 00000000 00000000 00000000 00000000 00000000 00000000
```

Nous devons donc passer par une étape intermédiaire pour faire la conversion.



Pour passer du code source au code machine, nous passons le code par un logiciel qui s'appelle un compilateur et on compile le code.

Dans le terminal, Interface de Ligne de Commande, CLI, nous allons donc taper la premier commande `make hello`, c'est le nom d'un programme qui existe pour compiler du code.

La commande `make` est une commande disponible nativement sur linux et mac OS elle va d'elle-même chercher le fichier en `.c` pour le compiler. Maintenant nous utiliserons donc la commande `make`.



Ensuite nous pouvons lancer notre programme avec la commande `./hello`



Maintenant faisons quelque chose de plus complexe.

A Scratch script and its corresponding C code. The Scratch script consists of two blocks: 'ask [What's your name?] and wait' and 'say [join [hello,] [answer]]'. To the right, the C code is shown: 

```
string answer = get_string("What's your name?\n");
printf("hello, %s\n", answer);
```

la fonction `ask` dans Scratch sera la fonction `get_string` dans le langage C, c'est la fonction la plus proche.

```
get_string("what....");
```

il faut que la réponse aille dans une variable afin qu'elle soit stocké si nous souhaitons pouvoir la réutiliser.

```
answer = get_string("what....");
```

En C, c'est la vielle école il faut donc être très précis dans ce que l'on indique en variable et préciser le type de variable ici c'est un `string`.

`string answer = get_string("what....");` ça indique à l'ordinateur le type de valeur que je vais lui demander de stocker. On lit le code de droite à gauche.

Le signe `=` ne signifie pas exactement égal, ça veut dire assigner ou opérateur d'affectation, ainsi que déplacer quelque chose de la droite vers la gauche.

Nous voulons pouvoir afficher de manière dynamique la réponse.



```
#include <cs50.h>
#include <stdio.h>

int main(void)
{
    string answer = get_string("comment t'appelles-tu?\n");
    printf("Hello, %s\n", answer);
}
```

---

## Compteur

Maintenant voyons comment faire un compteur.

A Scratch script consisting of a single orange "set [counter v] to [0]" control block.

```
int counter = 0;
```

A Scratch script consisting of a single orange "change [counter v] by [1]" control block.

```
counter = counter + 1;
```

Cette ligne suppose que le compteur existe et comme il a été déclaré précédemment, il n'est pas nécessaire de le refaire ici nous n'avons donc pas besoin d'indiquer de nouveau `int`.

Ci-dessous deux autres versions pour faire un compteur, c'est littéralement la même chose avec la première version qui est `counter = counter + 1;`

A Scratch script consisting of a single orange "change [counter v] by [1]" control block.

```
counter += 1;
```

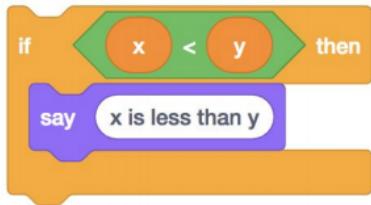
A Scratch script consisting of a single orange "change [counter v] by [1]" control block.

```
counter++;
```

Par convention un compteur sera plutôt appelé `i` dans le cas d'un premier compteur et `j` dans le cas d'un second.

## La forme conditionnelle

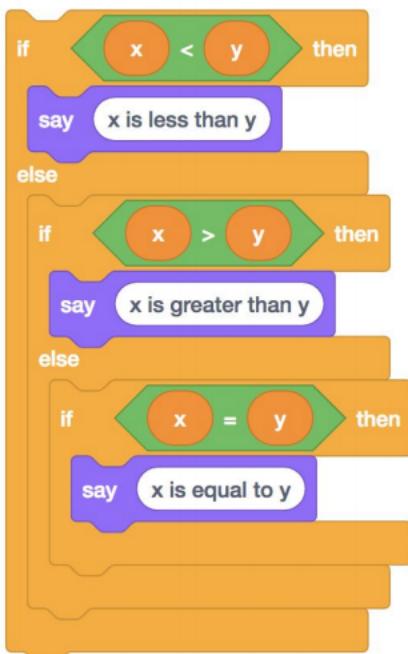
Voyons maintenant les conditions `if` , `else if` et `else` .



```
if (x < y)
{
    printf("x is less than y\n");
}
```



```
if (x < y)
{
    printf("x is less than y\n");
}
else
{
    printf("x is not less than y\n");
}
```

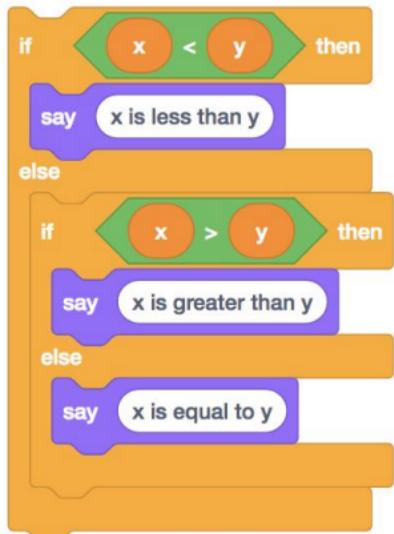


```

if (x < y)
{
    printf("x is less than y\n");
}
else if (x > y)
{
    printf("x is greater than y\n");
}
else if (x == y)
{
    printf("x is equal to y\n");
}

```

Il n'est pas nécessaire de mettre à la fin `else if (x == y)` mais juste `else`.



```

if (x < y)
{
    printf("x is less than y\n");
}
else if (x > y)
{
    printf("x is greater than y\n");
}
else
{
    printf("x is equal to y\n");
}

```

## Le pseudo code

Rappelez-vous l'exemple de pseudo code que nous avons réalisé ensemble.



```
Je déclare une variable qui est un entier et qui s'appelle i dont  
la valeur est égale à 20  
Je déclare une variable qui est un entier et qui s'appelle j dont  
la valeur est égale à 10  
  
SI la variable i est inférieure à la variable j alors j'écris  
SØÑØN j'écris "NOK"  
AUTREMENT i = j  
  
if ( variable i est inférieure à ma variable j)  
  
    if(i < j) {  
        écrire Ok  
    }autrement{  
        écrire NOK  
    }  
  
    if (i < j) {  
        printf ("Ok")}  
    else {  
        printf ("NOK")  
    }
```

## Exemples

Voici deux nouveaux exemples avec des variables `int` et `float`.



```
#include <cs50.h>
#include <stdio.h>

int main(void)
{
    int age = get_int("Quel est ton âge?\n");
    int jours = age * 365;
    printf("ton âge en jours est de %i jours.\n", jours);
}
```



```
#include <cs50.h>
#include <stdio.h>

int main(void)
{
    float prix = get_int("Quel est le prix?\n");
    printf("Le prix total est de %.2f euros.\n", prix * 1.2);
}
```

## Les boucles

La boucle `while` (tant que), ici dans l'exemple, tant que `i` est inférieur à 50 la boucle imprimera le message `hello, world`.



```
int i = 0;
while (i < 50)
{
    printf("hello, world\n");
    i++;
}
```

C'est la boucle la plus simple.

Maintenant voyons un autre type de construction de boucle, la boucle `for` c'est une boucle plus souvent utilisée que la précédente. La boucle `for` prend 3 entrées (initialisation du compteur, expression booléenne, mise à jour d'une ou plusieurs variables)



```
for (int i = 0; i < 50; i++)
{
    printf("hello, world\n");
}
```

Même si la boucle `for` fonctionne un peu de manière différente, le résultat est identique à la boucle `while` mais avec une syntaxe différente, ici elle est plus succincte.

## Le type de variables

Il existe en C plusieurs types de variables possibles, comme :

- `bool` (expression booléenne dont la valeur est vrai ou fausse / true ou false)
- `char` (stocker une lettre / un seul caractère, pas plus) - `%c`
- `double` (nombre réel qui peut avoir encore plus de chiffres) - `%f`
- `float` (valeur à virgule flottante, nombre réel) - `%f`

- int (stocker un entier / integer, il a en général un certaine taille, vous pouvez compter jusqu'à 4 milliards et ce n'est pas assez grand pour certaines applications, exemple Google, Facebook,...) - `%i`
  - long (utilise plus de bits donc peuvent compter encore plus) - `%li`
  - string (une chaîne de caractère, un ou plusieurs caractère(s) entre double guillemets) - `%s`
  - ...
- 

## Autres fonctionnalités avec la librairie CS50

- get\_string (que nous avons déjà vu ensemble)
  - get\_char
  - get\_double
  - get\_float
  - get\_int
  - get\_long
  - ...
- 

## La suite...

Maintenant vous avez les bases pour commencer à expérimenter par vous-mêmes. L'idée est surtout d'explorer les concepts et de développer votre mémoire à utiliser ces principes.

**Parité :** Pair ou impair, on pourrait le faire de manière fastidieuse avec un `if` mais nous allons utiliser `%` le modulo - le reste.



```
#include <cs50.h>
#include <stdio.h>

int main(void)
{
    int n = get_int("n: ");

    if (n % 2 == 0)
    {
        printf("pair");
    }else{
        printf("impair");
    }
}
```

**Commencez les bonnes pratiques et commentez votre code.**



```
//un commentaire sur une seule ligne

/*un commentaire sur
plusieurs lignes*/
```

**Une bibliothèque/librairie** n'est qu'un fichier de code développer par un tiers et que nous utilisons pour nous faciliter le codage

## Créer sa propre fonction

```
//solution sans fonction et sans boucle
#include <cs50.h>
#include <stdio.h>

int main(void)
{
    printf("miaou\n");
    printf("miaou\n");
    printf("miaou\n");
}
```

```
//solution sans fonction et avec une boucle for
#include <cs50.h>
#include <stdio.h>

// Better design

int main(void)
{
    for (int i = 0; i < 3; i++)
    {
        printf("miaou\n");
    }
}
```

```
//solution avec fonction
#include <cs50.h>
#include <stdio.h>

// Abstraction

void miaou(void); //dire à l'ordinateur que la fonction existe

int main(void)
{
    for (int i = 0; i < 3; i++)
    {
        miaou(); //appel de la fonction
    }
}

// création de la fonction
void miaou(void)
{
    printf("miaou\n");
}
```

```
#include <cs50.h>
#include <stdio.h>

// Abstraction with parameterization

void miaou(int n);

int main(void)
{
    miaou(3);
}

// création d'une fonction générique via n
void miaou(int n)
{
    for (int i = 0; i < n; i++)
    {
        printf("miaou\n");
    }
}
```

---

## L'outil style50

Il permet de vous aider à améliorer votre style, l'aspect esthétique de votre code en respectant les bonnes pratiques.

<https://cs50.readthedocs.io/style50/>

---

## La boucle Do-While