**Chapitre 5 :Structurez votre code avec les conditions**

Nous avons vu comment créer des variables, nous allons maintenant nous intéresser aux **conditions**. Elles permettent de tester des variables. Sans conditions, nos programmes informatiques feraient toujours la même chose !

**Utilisez des conditions if... else**

On peut par exemple dire :

"si la variable machin est égale à 50, fais ceci."

Mais ce serait dommage de ne pouvoir tester que l'égalité ! Il faudrait aussi pouvoir tester si la variable est inférieure à 50, inférieure ou égale à 50, supérieure, supérieure ou égale… Ne vous inquiétez pas, le C a tout prévu !

Avant de voir comment on écrit une condition de type if... else  en C, il faut donc que vous connaissiez les symboles de base. Voici un tableau récapitulatif **à connaître par cœur**:

| **Symbole** | **Signification** |
| --- | --- |
| == | est égal à |
| > | est supérieur à |
| < | est inférieur à |
| >= | est supérieur ou égal à |
| <= | est inférieur ou égal à |
| != | est différent de |

Il y a bien deux symboles ==  pour tester l'égalité. Une erreur courante que font les débutants est de ne mettre qu'un symbole =  , ce qui n'a pas la même signification en C.

**Ouvrez une condition avec un if**

Nous allons faire un test simple, qui va dire à l'ordinateur :

"SI la variable vaut ça, ALORS fais ceci".

En anglais, le mot "si" se traduit par if.

**On l'utilise en langage C pour introduire une condition :**

1. Commencez par écrire if.
2. Ouvrez des parenthèses (à l'intérieur de ces parenthèses on écrit une condition).
3. Ouvrez une accolade {  et fermez-la un peu plus loin}. Tout ce qui se trouve à l'intérieur des accolades sera exécuté uniquement si la condition est vérifiée.

Cela nous donne :

**if (/\* Votre condition \*/)**

**{**

 **// Instructions à exécuter si la condition est vraie**

**}**

À la place de mon commentaire "Votre condition", on va écrire une condition pour tester une variable. Pour s'entraîner, on va tester si vous êtes majeur avec une variable  age  , c'est-à-dire "vous êtes majeur **si votre âge est supérieur ou égal à 18**" :

**if (age >= 18)**

**{**

 **printf ("Vous etes majeur !");**

**}**

Rappel : le symbole >=  signifie « supérieur ou égal ».

S'il n'y a qu'une instruction entre les accolades (comme c'est le cas ici), alors celles-ci deviennent facultatives. Je recommande néanmoins de toujours mettre des accolades pour des raisons de clarté.

Si vous voulez tester les codes précédents pour voir comment le if fonctionne, il faudra placer le if à l'intérieur d'une fonction main  , et ne pas oublier de déclarer une variable age à laquelle on donnera la valeur de notre choix :

**#include <stdio.h>**

**#include <stdlib.h>**

**int main()**

**{**

 **int age = 20;**

 **if (age >= 18)**

 **{**

 **printf ("Vous etes majeur !\n");**

 **}**

 **return 0;**

**}**

Ici, la variable age vaut 20, donc le "Vous êtes majeur !" s'affichera.

Essayez de changer la valeur initiale de la variable pour voir. Mettez par exemple 15 : la condition sera fausse, et donc "Vous êtes majeur !" ne s'affichera pas cette fois.

**Exécutez une instruction alternative avec else**

Maintenant, allons plus loin : si le test n'a pas marché (il est faux), on va dire à l'ordinateur d'exécuter d'autres instructions.

En français, nous allons donc écrire quelque chose qui ressemble à :

"SI la variable vaut ça, ALORS fais ceci,

SINON fais cela".

Il suffit de rajouter else après l'accolade fermante du if :

**if (age >= 18) // Si l'âge est supérieur ou égal à 18**

**{**

 **printf ("Vous etes majeur !");**

**}**

**else // Sinon...**

**{**

 **printf ("Ah c'est bete, vous etes mineur !");**

**}**

Si la variable age est supérieure ou égale à 18, on affiche le message "Vous êtes majeur !", sinon on affiche "Vous êtes mineur".

**Conditionnez l'instruction alternative avec else if**

Il est possible de faire un autre test si le premier test n'a pas marché.  On demande alors à l'ordinateur :

"SI la variable vaut ça ALORS fais ceci,

SINON SI la variable vaut ça ALORS fais ça,

SINON fais cela".

else if  ( ou "sinon si") se place entre le if et le else :

**if (age >= 18) // Si l'âge est supérieur ou égal à 18**

**{**

 **printf ("Vous etes majeur !");**

**}**

**else if ( age > 4 ) // Sinon, si l'âge est au moins supérieur à 4**

**{**

 **printf ("Bon t'es pas trop jeune quand meme...");**

**}**

**else // Sinon...**

**{**

 **printf ("Aga gaa aga gaaa"); // Langage bébé, vous pouvez pas comprendre**

**}**

**L'ordinateur** fait les tests dans l'ordre :

1. D'abord il teste le premier if: si la condition est vraie, alors il exécute ce qui se trouve entre les premières accolades.
2. Sinon, il va au « sinon si » et fait à nouveau un test : si ce test est vrai, alors il exécute les instructions correspondantes entre accolades.
3. Enfin, si aucun des tests précédents n'a marché, il exécute les instructions du « sinon ».

Notez que l'on peut mettre autant de else if que l'on veut.

On peut donc écrire :

SI la variable vaut ça,
ALORS fais ceci,
SINON SI la variable vaut ça ALORS fais ça,
SINON SI la variable vaut ça ALORS fais ça,
SINON SI la variable vaut ça ALORS fais ça,
SINON fais cela.

**Testez plusieurs conditions à la fois**

Il peut aussi être utile de faire plusieurs tests à la fois dans votre if.

Pour cela, il va falloir utiliser de nouveaux symboles :

| **Symbole** | **Signification** |
| --- | --- |
| && | ET |
| || | OU |
| ! | NON |

**Ajoutez une contrainte "ET" avec  &&**

On peut vouloir tester si l'âge est supérieur à 18 ET si l'âge est inférieur à 25.

Pour cela, il vous faudra écrire :

**if (age > 18 && age < 25)**

**Ajoutez une contrainte "OU" avec  ||**

Ce signe n'est pas facilement accessible sur nos claviers. Sur un clavier AZERTY français, il faudra faire Alt Gr + 6 ; sur un clavier belge, Alt Gr + &.

Imaginons un programme qui décide si une personne a le droit d'ouvrir un compte en banque. Pour ouvrir un compte en banque, il vaut mieux ne pas être trop jeune, ou bien avoir beaucoup d'argent. Notre test pourrait être :

**if (age > 30 || argent > 100000)**

**{**

 **printf("Bienvenue chez PicsouBanque !");**

**}**

**else**

**{**

 **printf("Hors de ma vue, miserable !");**

**}**

Ce test n'est valide que si la personne a plus de 30 ans ou si elle possède plus de 100 000 euros !

**Testez l'opposée d'une condition avec  !**

En informatique, le point d'exclamation signifie « non ».

Vous devez mettre ce signe avant votre condition pour dire « si cela n'est pas vrai » :

**if (!(age < 18))**

Cela pourrait se traduire par « si la personne n'est pas mineure ».

Si on avait enlevé le !  devant, cela aurait signifié l'inverse : « si la personne est mineure ».

Et puisqu'on y est, je voudrais vous lister quelques erreurs courantes du débutant.

**Évitez de faire ces erreurs**

**Oublier les deux signes  ==**

Pour tester si la personne a tout juste 18 ans, il faut écrire if (age == 18) avec  ==  .

Si vous ne mettez qu'un seul =, alors votre variable **prendra** la valeur 18. Ce qu'on veut faire ici, c'est tester la valeur de la variable, non pas la changer. Faites très attention à cela !

**Terminer une condition avec un  ;**

Un if est une condition. On met un  ;  à la fin d'une instruction, pas d'une condition.

**Faites des conditions de type vrai/faux avec les booléens**

Avant d'entrer dans les explications, voici un code source que je vous propose de tester :

**if (1)**

**{**

 **printf("C'est vrai");**

**}**

**else**

**{**

 **printf("C'est faux");**

**}**

Résultat :

C'est vrai

Mais… Il n'y a pas de condition dans le if  , juste un nombre. Qu'est-ce que ça veut dire ?

Si, ça en a, vous allez comprendre. Faites un autre test en remplaçant 1 par 0 :

**if (0)**

**{**

 **printf("C'est vrai");**

**}**

**else**

**{**

 **printf("C'est faux");**

**}**

Résultat :

C'est faux

Faites maintenant d'autres tests en remplaçant le 0 par n'importe quel autre nombre entier, comme 4, 15, 226, ou encore -10.

Qu'est-ce qu'on vous répond à chaque fois ? On vous répond : « C'est vrai ».

Si on met un 0, le test est considéré comme faux.

Si on met un 1 ou n'importe quel autre nombre, le test est vrai.

Ci-dessous, le test que vous faites est age >= 18.

if (age >= 18)

Par exemple, si age vaut 23,  alors le test est vrai, et l'ordinateur remplace, en quelque sorte, age >= 18 par 1. Ensuite, l'ordinateur obtient (dans sa tête) un if (1). Quand le nombre est 1, comme on l'a vu, l'ordinateur dit que la condition est vraie, donc il affiche « C'est vrai » ! Si la condition est fausse, il remplace age >= 18 par le nombre 0, et du coup la condition est fausse : l'ordinateur va lire les instructions du else.

Testez maintenant autre chose : envoyez le résultat de votre condition dans une variable, comme si c'était une opération (car pour l'ordinateur, **c'est** une opération !).

**int age = 20;**

**int majeur = 0;**

**majeur = age >= 18;**

**printf("Majeur vaut : %d\n", majeur);**

Comme vous le voyez, la condition age >= 18 a renvoyé le nombre 1 car elle est vraie. Du coup, notre variable majeur vaut 1 (on vérifie d'ailleurs cela grâce à un printf qui montre bien qu'elle a changé de valeur).

Faites le même test en mettant age == 10  , par exemple. Cette fois, majeur vaudra "0".

On dit qu'une variable à laquelle on fait prendre les valeurs 0 et 1 est **un booléen**. Et aussi que : 0 = faux ; 1 = vrai.

Pour être tout à fait exact, 0 = faux et tous les autres nombres valent vrai (on a eu l'occasion de le tester plus tôt). Ceci dit, pour simplifier les choses on va se contenter de n'utiliser que les nombres 0 et 1, pour dire si "quelque chose est faux ou vrai".

En langage C, il n'existe pas de type de variable "booléen". Donc, on est obligé d'utiliser un type entier comme int pour gérer les booléens.

Souvent, on fera un test if sur une variable booléenne :

**int majeur = 1;**

**if (majeur)**

**{**

 **printf("Vous etes majeur !");**

**}**

**else**

**{**

 **printf("Vous etes mineur");**

**}**

Comme  majeur  vaut "1", la condition est vraie, donc on affiche "Vous êtes majeur !".

Ce qui est très pratique, c'est que la condition peut être lue facilement par un être humain. On voit if (majeur)  , ce qui peut se traduire par "si vous êtes majeur". Les tests sur des booléens sont donc faciles à lire et à comprendre, pour peu que vous ayez donné des noms clairs à vos variables.

Tenez, voici un autre test imaginaire :

**if (majeur && vaccin)**

Ce test signifie "si vous êtes majeur ET que vous êtes vacciné".

vaccin est une autre variable booléenne qui vaut "1" si vous êtes vacciné, et "0" si vous ne l'êtes pas ! Bravo, vous avez tout compris !

Petite question : si on fait le test if (majeur == 1), ça marche aussi, non ?

Tout à fait. Mais le principe des booléens, c'est justement de raccourcir l'expression du if  , et de la rendre plus lisible. Avouez que if (majeur)ça se comprend très bien, non ?

**Retenez donc** :

* Si votre variable est censée contenir un nombre (comme un âge), faites un test sous la forme if (variable == 1).
Si au contraire votre variable est censée contenir un booléen (c'est-à-dire soit 1 soit 0 pour dire vrai ou faux), faites un test sous la forme if (variable).

**Optimisez votre code avec la condition  switch**

if... else est le type de condition le plus souvent utilisé car il permet de gérer tous les cas. En fait, il n'y a pas 36 façons de faire une condition en C.

Cela dit, if... else peut s'avérer quelque peu… répétitif. Prenons cet exemple :

if (age == 2)

{

 printf("Salut bebe !");

}

else if (age == 6)

{

 printf("Salut gamin !");

}

else if (age == 12)

{

 printf("Salut jeune !");

}

else if (age == 16)

{

 printf("Salut ado !");

}

else if (age == 18)

{

 printf("Salut adulte !");

}

else if (age == 68)

{

 printf("Salut papy !");

}

else

{

 printf("Je n'ai aucune reponse pour ton age");

}

**Construisez un switch**

Pour éviter d'avoir à faire des répétitions quand on teste la valeur d'une seule et même variable, il existe une autre structure que if... else. Elle s'appelle switch !

Voici un switch basé sur l'exemple qu'on vient de voir :

**switch (age)**

**{**

**case 2:**

 **printf("Salut bebe !");**

 **break;**

**case 6:**

 **printf("Salut gamin !");**

 **break;**

**case 12:**

 **printf("Salut jeune !");**

 **break;**

**case 16:**

 **printf("Salut ado !");**

 **break;**

**case 18:**

 **printf("Salut adulte !");**

 **break;**

**case 68:**

 **printf("Salut papy !");**

 **break;**

**default:**

 **printf("Je n'ai aucune reponse pour ton age");**

 **break;**

**}**

Le principe est le suivant :

1. Vous écrivez switch (maVariable) pour dire « je vais tester la valeur de la variable maVariable».
2. Vous ouvrez ensuite des accolades que vous refermez tout en bas.
3. À l'intérieur de ces accolades, vous gérez tous les cas :case 2  , case 4  , case 5  ,case 45  …

Vous devez mettre une instruction break; obligatoirement à la fin de chaque cas.

Si vous ne le faites pas, l'ordinateur lira les instructions en dessous (censées être réservées aux autres cas). break; commande en fait à l'ordinateur de "sortir" des accolades.

Enfin, le cas default correspond au else qu'on connaît bien maintenant. Si la variable ne vaut aucune des valeurs précédentes, l'ordinateur ira lire le default.

**Exemple:**

Je crois que le moment est venu de pratiquer un peu !

En console, pour faire un menu, on fait des printf qui affichent les différentes options possibles. Chaque option est numérotée, et l'utilisateur doit entrer le numéro du menu qui l'intéresse.

Voici par exemple ce que la console devra afficher :

=== Menu ===

1. Royal Cheese

2. Mc Deluxe

3. Mc Bacon

4. Big Mac

Votre choix ?

**Votre mission (si vous l'acceptez)** :

1. Reproduire ce menu à l'aide de printf.
2. Ajouter un scanf pour enregistrer le choix de l'utilisateur dans une variable choixMenu.
3. Faire un switch pour dire à l'utilisateur "tu as choisi le menu Royal Cheese", par exemple.

Alors, vous avez réussi ?

**Découvrez les conditions condensées : les ternaires**

Il existe une troisième façon de faire des conditions, plus rare : les **expressions ternaires**.

Les ternaires ne sont pas du tout indispensables ; personnellement je les utilise peu, car ils peuvent rendre la lecture d'un code source un peu difficile. Cela dit, il vaut mieux les connaître pour le jour où vous tomberez sur un code plein de ternaires !

Concrètement, c'est comme un if... else, sauf qu'on fait tout tenir sur une seule ligne.

Comme un exemple vaut mieux qu'un long discours, je vais vous donner deux fois la même condition :

1. La première avec un if... else.
2. Et la seconde, identique, mais sous forme d'une expression ternaire.

**1. Analysez cette condition en if... else**

Exemple : on a une variable booléenne majeur qui vaut vrai (1) si on est majeur, et faux (0) si on est mineur. On veut changer la valeur de la variable age en fonction du booléen, pour mettre "18" si on est majeur, "17" si on est mineur. C'est un exemple complètement stupide, je suis d'accord, mais ça me permet de vous montrer comment on peut se servir des expressions ternaires.

Voici comment faire cela avec un if... else  :

**if (majeur)**

 **age = 18;**

**else**

 **age = 17;**

Notez que j'ai enlevé les accolades, car elles sont facultatives s'il n'y a qu'une instruction.

**2. Analysez cette même condition en ternaire**

Voici un code qui fait exactement la même chose que le précédent, mais sous forme ternaire :

**age = (majeur) ? 18 : 17;**

Les ternaires permettent, sur une seule ligne, de changer la valeur d'une variable en fonction d'une condition.

Ici la condition est tout simplementmajeur  , mais ça pourrait être n'importe quelle condition plus longue, bien entendu.

Un autre exemple ?

**autorisation = (age >= 18) ? 1 : 0;**

Le point d'interrogation permet de dire « est-ce que tu es majeur ? ». Si oui, alors on met la valeur 18 dans age. Sinon (le : signifie else ici), on met la valeur 17.

**En résumé**

* Les **conditions** sont à la base de tous les programmes. C'est un moyen pour l'ordinateur de **prendre une décision** en fonction de la valeur d'une variable.
* Les mots-clés if  , else if  , else signifient respectivement « si », « sinon si », « sinon ». On peut écrire autant de else if que l'on veut.
* Un **booléen** est une variable qui peut avoir deux états : vrai (1) ou faux (0) (toute valeur différente de 0 est en fait considérée comme « vraie »). On utilise des int pour stocker des booléens, car ce ne sont en fait rien d'autre que des nombres.
* Le switch est une alternative au if quand il s'agit d'analyser la valeur d'une variable. Il permet de rendre un code source plus clair si vous vous apprêtez à tester de nombreux cas. Si vous utilisez de nombreux else if  , c'est en général le signe qu'un switch serait plus adapté pour rendre le code source plus lisible.
* Les **ternaires** sont des conditions très concises qui permettent d'affecter rapidement une valeur à une variable en fonction du résultat d'un test. On les utilise avec parcimonie, car le code source a tendance à devenir moins lisible avec elles.