LES BOUCLES OU ITERATIONS EN C

En programmation, une boucle est une séquence d'instructions qui est répétée jusqu'à ce qu'une certaine condition soit atteinte.

Il existe principalement deux types de boucles:

- **Boucles à entrée contrôlée**: dans ce type de boucles, la condition est testée avant d'entrer dans le corps de la boucle. boucle **for** et boucle **while** sont des boucles d'entrée contrôlées.
- **Boucles à sortie contrôlée**: dans ce type de boucles, la condition est testée ou évaluée à l'extrémité du corps de la boucle. Par conséquent, le corps de boucle s'exécutera au moins une fois, que la condition soit true ou false. **do-while** est une boucle de sortie contrôlé.

boucle for

Une boucle **for** est une structure de contrôle répétitive qui nous permet d'écrire une boucle exécutée un nombre de fois spécifique. La boucle nous permet de réaliser n nombre d'étapes sur une seule ligne.

Syntaxe :

```
for(initialisation ; condition ; mise à jour){
    // bloc d'instructions à répéter
}
```

Dans la boucle for, une variable de boucle est utilisée pour contrôler la boucle. Commencez par initialiser cette variable de boucle sur une valeur (**initialisation**), puis vérifiez si cette variable est inférieure ou supérieure à la valeur du compteur (**condition**). Si l'instruction est vraie, le corps de la boucle est exécuté et sa variable est mise à jour (**mise à jour**). Les étapes sont répétées jusqu'à ce que la condition de sortie arrive.

Exemple 1 :

```
#include < stdio.h>

int main(void){
    int i;

    for(i=0;i < 5;i++){
        printf("i = %d \n",i);
    }

    return 0;
}
```

```
i = 0
i = 1
i = 2
```

```
i = 3
i = 4
```

- **Initialisation**: Dans cette expression, nous devons initialiser le compteur de boucle à une valeur. par exemple: **i = 1;**
- **condition**: Dans cette expression, nous devons tester la condition. Si la condition est true, nous exécuterons le corps de la boucle et mettrons à jour l'expression, sinon nous sortirons de la boucle for. Par exemple: **i <=5;**
- **mise à jour** : après avoir exécuté le corps de la boucle, cette expression incrémente / décrémente la variable de la boucle d'une valeur. par exemple: **i++;**

boucle while

dans la boucle **for**, nous avons constaté que le nombre d'itérations était connu à l'avance, c'est-à-dire que nous savons combien de fois le corps de la boucle doit être exécuté. Les boucles **while** sont utilisées dans des situations où nous ne connaissons pas le nombre exact d'itérations de boucle auparavant. L'exécution de la boucle est terminée sur la base d'une condition.

Syntaxe :

```
initialisation
while(condition){

    // corps de la bucle

    mise à jour
}
```

Exemple 2 :

```

 #include < stdio.h>

 int main(void){
     int i;
     i=0; // initialisation
     while(i < 5){  // condition
         printf("i = %d \n",i);

         i++; // mise à jour
     }

     return 0;
 }

```

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

boucle do-while

Dans les boucles **do while**, l'exécution de la boucle est également terminée sur la base d'une condition de test. La différence principale entre la boucle **do while** et la boucle **while** est dans la boucle **do while**, la condition est testée à la fin du corps de la boucle;

Syntaxe :

```
initialisation
do{
    // corps de la boucle

    mise à jour
}while(condition);
```

Remarque : le point-virgule (;) à la fin de la boucle est obligatoire.

Exemple 3 :

```
#include < stdio.h>

int main(void){
    int i;
    i=0; // initialisation
    do{
        printf("i = %d \n";i);

        i++; // mise à jour
    }while(i < 5);// condition

    return 0;
}
```

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

Les points importants

- Utilisez la boucle **for** lorsque le nombre d'itérations est connu à l'avance, c'est-à-dire que le nombre de fois que le corps de la boucle doit être exécuté est connu.
- Utilisez les boucles **while** pour lesquelles le nombre exact d'itérations n'est pas connu mais la condition de fin de boucle est connue.
- Utilisez la boucle **do while** si le code doit être exécuté au moins une fois, comme dans les programmes pilotés par le menu.