Programmation en C/C++

Série 3b

Structures répétitives
Les boucles

 Important !

 Difficile !

Objectifs

Maîtriser les structures répétitives. Les boucles *For* et *While*. Imbrication de boucles.

Introduction

Les structures répétitives vont être très utiles en programmation : on ne va pas répéter 100 fois les lignes d'instructions demandant d'afficher ou de saisir 100 nombres entiers !
La boucle *For* sera utilisée pour répéter un certain nombre de fois, connu, une opération. La boucle *While* sera utile pour demander, par exemple, de saisir un nombre tant que celui-ci est différent d'une certaine valeur.

La structure répétitive "Pour" ou boucle "For"

L'instruction *for* permet, d'exécuter une instruction ou un bloc d'instructions un certain nombre de fois (généralement connu).

Syntaxe

for (condition d'initialisation ; condition de fin ; incrémentation)
instruction;

ou bien :

```
for ( condition d'initialisation ; condition de fin ; incrémentation )
{
instruction_1;
... ;
instruction_n;
}
```

Exemple 1

```
for (i=0; i<10; i++)
   printf("Bonjour ! ");
```

Cette boucle affichera 10 fois "Bonjour ! ", les uns à la suite des autres (pensez à l'espace !)

Exemple 2

```
for (i=0; i<10; i++)
 {
        printf("Bonjour ! \n");
        printf("Hello ! \n");
 }
```

Cette boucle affichera 10 fois la séquence :
Bonjour !
Hello !

Rq : Deux instructions sont répétées lors de l'exécution de cette boucle : les deux lignes d'instructions forment donc un bloc qui devra être matérialisé par des accolades.

Exemple 3

```
for (i=0; i<10; i++)
   printf("%d\n", i);
```

Voici ce qui se passe :

1. Juste avant d'entrer dans la boucle, l'initialisation i=0 est effectuée
2. Ensuite, la condition *i<10* est testée. Si elle est vrai, ce qui est le cas pour *i* ayant des valeurs de 1 à 9, l'instruction du corps de la boucle, ici *printf*("..."); est effectuée. Sinon, la boucle est terminée.
3. L'opération d'incrémentation *i*++ est effectuée à chaque tour de boucle.
4. Le programme reboucle à l'étape 2.


Attention ! Pas de ";" après la ligne du for

Remarque 1
for (i=1; i<=10; i++);
printf("Bonjour !");

Ce programme tourne à vide 10 fois puis affiche "Bonjour !" une seule fois !!!

Remarque 2
Il est possible de procéder à plusieurs initialisations et à plusieurs opérations en les séparant par des virgules. On peut aussi procéder à plusieurs tests en utilisant, par exemple, les connecteurs logiques && ou || :

```
for (i=0, j=1; i<10 && j<100; i++, j=j*2)
    printf("%d %d\n", i, j);
```

Ici, *i* et *j* sont respectivement initialisées à 0 et 1 ; on bouclera tant que *i* est inférieur à 10 ET *j* inférieur 100 ; à chaque tour de boucle *i* est incrémenté d'une unité et *j* multiplié par 2.

La structure répétitive "Tant que… faire …" ou *while … do…*

L'instruction *while* permet d'exécuter (et répéter) une instruction ou un bloc d'instructions tant qu'une condition est vraie (on teste la condition en entrée de boucle) :

Syntaxe

```
while (condition)
instruction;
```

ou bien :

```
while (condition)
{
instruction_1;
... ;
instruction_n;
}
```

Exemple

Par exemple, pour afficher les entiers de 0 à 9 :

```
i = 0;
while (i<10) {
    printf("%d\n", i);
    i++;
}
```

La structure répétitive " Faire … tant que…"

ou *do... while…*

L'instruction *do while* permet d'exécuter (et répéter) une instruction ou un bloc d'instructions tant qu'une condition est vraie (on teste, cette fois, la condition en sortie de boucle) :

Syntaxe

```
do
instruction;
while(condition);
```

ou bien :

```
do
{
instruction_1;
... ;
instruction_n;
}
while (condition);
```

Exemple

On souhaite afficher les entiers de 0 à 9 :

```
i = 0;
do {
    printf("%d\n", i);
    i++;
} while (i<10);
```

L'instruction *break*

L'instruction *break,* que l'on évitera d'utiliser, permet de sortir d'une boucle.

Exemple

```
while (1) { /* la condition est donc toujours vraie !! */
    printf("%d\n", i);
    i++;
    if (i>10)
        break;
}
```

Conseils

Dans une boucle *while… do* : ne pas oublier d'initialiser préalablement la variable testée.

Attention à modifier la valeur de la variable testée au sein de la boucle *do…while* sinon on sera en présence d'une boucle sans fin.

Attention à effectuer un test adéquat car il y a risque de ne jamais entrer dans la boucle *while* (test toujours faux).

Imbrications de boucles

Il est possible d'imbriquer des boucles *For* les unes dans les autres.
Par exemple, lorsque vous devrez parcourir un tableau à 2 dimensions vous utiliserez 2 boucles *For* imbriquées ; une pour balayer les lignes et une autre pour balayer les colonnes.

Exemple

```
for (i=1  ;  i<nb ;   i++)     {
        …;
        …;
        for (j=1  ;  i<14 ;   j++)     {
                …;
                …;
        }     // fin de la boucle en j

}     // fin de la boucle en i
```

Cette notion est abordée dans les exercices.

La fonction sleep

La fonction *sleep* permet de marquer un temps d'arrêt entre 2 affichages ; ce qui peut être pratique lors de l'exécution d'une boucle, pour avoir le temps de visualiser les affichages répétitifs.

Sous Windows

En tête de programme taper : *#include<windows.h>*
Puis, taper : *Sleep (1000) pour marquer un arrêt de 1000 millisecondes*

Sous Linux

En tête de programme taper : *#include<unistd.h>*
Puis, taper : *sleep (1) pour marquer un arrêt de 1 seconde*

EXERCICES

Exercice 1

Énoncé

Écrire le programme qui affiche les nombres entiers de 30 à 50 compris les uns en dessous des autres.

Corrigé

```
#include<stdio.h>
#include<stdlib.h>
int main() {
int i;
for( i=30; i<=50; i++)
        {
        printf("%d\n",i);
        }
return 0;
}
```

Exercice 2

Énoncé

Écrire le programme qui affiche les nombres entiers de 50 à 30 compris les uns en dessous des autres.

Corrigé

```
#include<stdio.h>
#include<stdlib.h>
int main() {
int i;
for( i=50; i>=30; i--)
        {
        printf("%d\n",i);
        }
return 0;  }
```

Exercice 3

Énoncé

Écrire le programme qui affiche tous les multiples de 4 inférieurs ou égaux à 400.

Corrigé

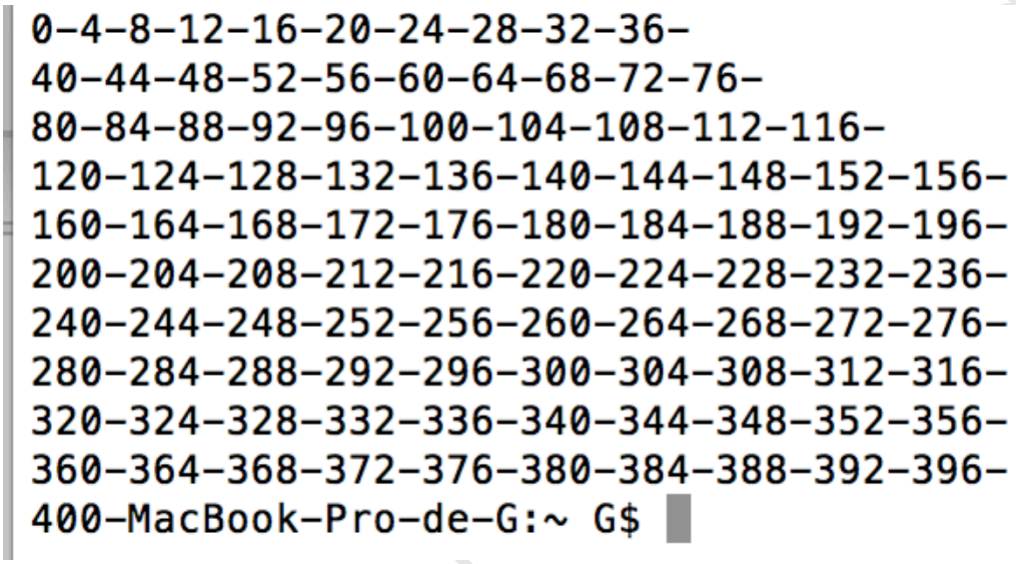
```
#include<stdio.h>
```

```
#include<stdlib.h>
int main() {
int i;
for( i=0; i<=100; i++)
        printf("%d-",i*4);
return 0;   }
```

Exercice 4

Énoncé

Reprendre le programme précédent et l'améliorer afin, que tous les 10 affichages, il y ait un retour à la ligne.



Corrigé

```
#include<stdio.h>
#include<stdlib.h>
int main() {
int i, cpt =0;
for( i=0; i<=100; i++)
        {
        printf("%d-",i*4);
        cpt++;
        if (cpt==10)
                {
                printf("\n");
                cpt=0;
                }
        }
return 0;
}
```

Exercice 5

Énoncé

Écrire le programme qui affiche les nombres comme montrés ci-après. Attention ! Il n'y a pas de tiret après le nombre 100.

```
0-1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-
30-31-32-33-34-35-36-37-38-39-40-41-42-43-44-45-46-47-48-49-50-51-52-53-54-55-56
-57-58-59-60-61-62-63-64-65-66-67-68-69-70-71-72-73-74-75-76-77-78-79-80-81-82-8
3-84-85-86-87-88-89-90-91-92-93-94-95-96-97-98-99-100MBP-de-G:~ G$
```

Corrigé

```
#include<stdio.h>
#include<stdlib.h>
int main()
{int i;

for(i=0;i<100;i++)
        printf("%d-",i);

printf("100");
return 0;
}
```

Exercice 6

Énoncé

Écrire un programme qui affiche la table de multiplication de 4.

0 * 4 =0
1 * 4 = 4
…
10*4 = 40

Corrigé

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

int main() {
int i;

for(i=0;i<=10;i++)
        printf("%d*4=%d\n",i,i*4);

return 0;
}
```

Exercice 7

Énoncé

Modifier le programme précédent pour que l'utilisateur choisisse la table de multiplication qu'il veut afficher.

Corrigé

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

int main() {
int i;
int table;

printf("Quelle table voulez-vous afficher ?\t");
scanf("%d",&table);

for(i=0;i<=10;i++)
        printf("%d*%d=%d\n",i,table,i*table);

return 0;
}
```

Exercice 8

Énoncé

Écrire un programme qui demande de saisir un entier et qui repose la question indéfiniment si le nombre tapé n'est pas inférieur ou égal à 100. Utiliser une boucle *while*.

Corrigé

```
#include<stdio.h>
#include<stdlib.h>
int main() {
int a;

printf("Saisir un entier inferieur ou egal a 100:");
scanf("%d", &a);

while(a>100)  {
        printf("Merci de saisir un entier inferieur ou egal a 100:\n");
        scanf("%d",&a);
}

printf("Bravo! %d<=100",a);
return 0;
}
```

Exercice 9

Énoncé

Écrire, cette fois, un programme qui demande de saisir un entier supérieur ou égal à 100 et qui repose la question indéfiniment si la condition n'est pas respectée. Utiliser une boucle *do… while*.

Corrigé

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

```
#include<stdlib.h>
int main() {
int a;

do {
        printf("Saisir un entier superieur ou egal a 100 : ");
        scanf("%d",&a);
} while(a<100);

printf("Bravo!");
return 0;
}
```

Exercice 10

Énoncé

Imaginons un programme qui demande un code à 4 chiffres (entier). On suppose que le bon code est 4700. Le programme vérifie le code et s'il est bon il demande le montant à retirer et affiche la somme que l'on souhaite retirer, en euros. La question du retrait d'argent n'est posé que si le code est valable, autrement il faut retaper le code.

Corrigé

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

int main() {
int code,m;

do   {
        printf("Saisir votre code:");
        scanf("%d",&code);
}  while (code!=4700);

printf("Saisir votre montant:");
scanf("%d",&m);
printf("Vous avez demande : %d euros",m);
return 0;
}
```

Exercice 11

Énoncé

Améliorer le dispositif précédent pour simuler un vrai automate distributeur de billets : au bout de 3 essais infructueux il s'affiche "Carte avalée" ; si le code est bon on demande alors le montant à retirer et celui-ci ne doit pas dépasser 350 euros. Il s'affiche alors "Retirer vos billets".

Corrigé

```
#include<stdio.h>
#include<stdlib.h>
int main() {
int code;
int compteur=0;
int montant;

do   {
        compteur++;
        printf(" Essai %d\n Veuillez saisir votre code:",compteur);
        scanf("%d", &code);
}    while(code!=4700 && compteur <3);

if (code==4700)
        {
        printf("Veuillez saisir le montant de votre retrait:");
        scanf("%d",&montant);
        while(montant>350)
                {
                printf("Montant trop eleve\n Veuillez saisir un montant n'excedant pas
                350 euros:");
                scanf("%d",&montant);
                }

        printf("Retirez vos billets");
        }
else
        printf("carte avalee");

return 0;
}
```

Exercice 12

Énoncé

Écrire le programme qui demande de saisir un nombre entier. Si le nombre saisi est par exemple 3 il s'affichera :

3.1
3.2
3.3
2.1
2.2
2.3
1.1
1.2
1.3

Corrigé

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

int main()
{
   int n, i, j;
   printf("Saisir un entier: ");
   scanf("%d", &n);
   for (i = n; i > 0; i--)
      for (j = 1; j <= n; j ++)
         printf("%d.%d\n", i, j);

   return 0;
}
```

Exercice 13

Énoncé

Écrire le programme qui demande de saisir un nombre entier. Si le nombre saisi est par exemple 5 il s'affichera :

```
*
 *
  *
   *
    *
```

```
Entrer un entier: 5
*
 *
  *
   *
    *
```

Exemple d'affichage avec 5

Chaque étoile est décalée d'une espace (nom féminin en typographie) supplémentaire à chaque ligne. (0 espace avant l'étoile sur la première ligne, 1 espace avant l'étoile sur la ligne 2…)

Corrigé

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

int main()
{
   int nb, i, j;
```

```
    printf("Entrer un nombre entier: ");
    scanf("%d", &nb);
    printf("*\n");
    for (i = 1; i < nb; i++) {
        for (j = 1; j <= i; j++) {
            printf(" ");
        }
    printf("*\n");
    }
    return 0;
}
```

Exercice 14

Énoncé

Écrire le programme qui demande de saisir un nombre entier. Si le nombre saisi est par exemple 4 il s'affichera :

```
Saisir un entier:4
11
12
13
14
21
22
23
24
31
32
33
34
41
42
43
44
MacBook-Pro-de-G:~ G$
```

Si on a saisi 3 il s'affichera :

```
Saisir un entier:3
11
12
13
21
22
23
31
32
33
MacBook-Pro-de-G:~ G$
```

Corrigé

```
#include<stdio.h>
#include<stdlib.h>
int main() {
int i,j;
int a;
```

```
printf("Saisir un entier:");
scanf("%d",&a);

for(i=1;i<=a;i++)
{
		for(j=1;j<=a;j++)
		{
			printf( "%d%d\n",i, j);
		}
}
return 0;
}
```

Exercice 15

Énoncé

Le sapin de Noël…

Écrire le programme qui demande de saisir un nombre entier et affiche le sapin de Noël. Si le nombre saisi est par exemple 8 il s'affichera :

```
Veuillez saisir un nombre entier :
8
        *
       * *
      *   *
     *     *
    *       *
   *         *
  *           *
 *             *
```

Il faudra gérer espaces et étoiles !
Pensez à imbriquer des boucles…

Corrigé

```
#include <stdio.h>

int main() {
   int i, j, nb;

   printf("Veuillez saisir un nombre entier :\n");
   scanf("%d", &nb);

// Boucle externe correspondant au N lignes les unes sous les autres
   for (i = 1; i <= nb; i++) {

        // Boucle j des espaces avant l'étoile N°1
        for (j = i; j < nb; j++) {
```

```
            printf(" ");// on affiche une espace
        }
        // Affichage de la première étoile d'une ligne
        printf("*");

        // Boucle j des espaces après la première étoile d'une ligne
        for (j = 2; j < 2 * i - 1; j++) {
            printf(" ");// on affiche une espace
      }

       if (i == 1) {       //retour à la ligne car pas de deuxième étoile sur ligne 1
          printf("\n");
       }
      else {// affichage étoile N°2 et retour à la ligne
          printf("*\n");
      }
   }// fin de boucle en i

   return 0;
}
```

Exercice 16

Énoncé

Reprendre l'exercice précédent et, au lieu d'afficher un sapin, vous afficherez un "V" (sapin retourné).

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

int main(){
int i,j,nb;

printf("Veuillez saisir un entier :\n");
scanf("%d",&nb);

//Boucle qui gère les lignes du "V" sauf celle d'en bas qui ne comporte qu'une seule étoile
for (i=1;i<nb;i++)    {

        //Boucle qui gère les espaces avant l'étoile N°1
         for (j=1;j<i;j++)
            printf(" ");

        // affichage de l'étoile N°1
        printf("*");

        //Affichage des espaces entre les 2 étoiles
        for (j=1;j<=(nb-i-1)*2+1;j++)
```

```
            printf(" ");

        // Affichage de l'étoile N°2 et retour à la ligne
        printf("*\n");
}

//Cas des espaces avant la derniere etoile (en bas du "V)
for (j=1;j<nb;j++)
        printf(" ");

//affichage de la dernière étoile
printf("*\n");
return 0;
}
```

Exercice 17

Énoncé

Écrire un programme qui écrit la lettre T à l'aide d'étoiles. On demandera en premier de saisir un nombre entier *n* (avec *n* supérieur ou égal à 3). La branche verticale du "T" aura toujours *n* étoiles ; La branche horizontale du "T" aura *n* étoiles si *n* est impair et (*n-1*) étoiles si *n* est pair. Il faudra gérer les étoiles et les espaces.
Exemples :

```
Saisissez un nombre entier superieur ou egal a 3 :
5
*****
  *
  *
  *
  *
```

Exemple avec la valeur *n* =5

```
Saisissez un nombre entier superieur ou egal a 3 :
6
*****
  *
  *
  *
  *
  *
```

Exemple avec la valeur *n* =6

Corrigé

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

int main(){
   int nb, i, j;
   printf("Saisissez un nombre entier superieur ou egal a 3 :\n");
   scanf("%d",&nb);

//Cas des nombres impairs
   if (nb%2 != 0) // on teste si le reste de la division par 2 vaut 0
```

```
    {   //branche horizontale
        for (i=1; i<=nb; i++)
            printf("*");

        printf("\n");
        // branche verticale
        for (i=1;i<=nb-1;i++)
            {
                for(j=1;j<=((nb+1)/2)-1;j++)
                    printf(" ");

                printf("*\n");
            }
    }

  //Cas des nombres pairs
  else
    {   //branche horizontale
        for (i=1;i<nb;i++)
            printf("*");

        printf("\n");

        //branche verticale
        for (i=1;i<=nb-1;i++)
            {
                for(j=1;j<=(nb/2)-1;j++)
                    printf(" ");

                printf("*\n");
            }
    }
  return 0;
}
```