

SEQUENCE 4 – STRUCTURES REPETITIVES

SEANCE 1 - DECOUVERTE - REPETER DES INSTRUCTIONS

OBJECTIFS

- Découvrir les structures : while, do ...while, for
- Différencier leur usage
- Vérifier des saisies

EXO : LIRE ET COMPRENDRE DES INSTRUCTIONS

Sans visual studio, lisez et exécutez-vous même ces lignes, indiquez ce qu'elles affichent et déduisez l'utilité de chacun s'ils en ont une !

<pre>int a =2, b = 4, cpt = 1, res; res = a; while (cpt < b) { res = res * a; cpt++; } Console.WriteLine(res);</pre> <p>Utilité : _____</p>	<pre>int res = 0, i =1 , n = 4 ; do { res = res + i; i++; } while (i <= n); Console.WriteLine(res);</pre> <p>Utilité : _____</p>
<pre>int nb = 10; for (int i = 1; i <= nb*2 ; i= i+2) { Console.WriteLine(i); }</pre> <p>Utilité : _____</p>	<pre>int i, j ; for (i = 1 ; i <= 5 ; i++) { for(j = 1 ; j <= i ; j++) Console.Write ("*"); Console.WriteLine () ; }</pre>

EXO 1 : REPETER DES ACTIONS AVEC WHILE, DOWHILE

1. Créez un répertoire « **Sequence_4_StructuresRepetitives** » puis un projet « **Exo1_RepeterBonjour** » dans une solution « **Seance1-DecouverteBoucles** ». Répétez à l'infini l'affichage de « Bonjour »

```
while(true)
{ Console.WriteLine("Hello !\n"); }
```

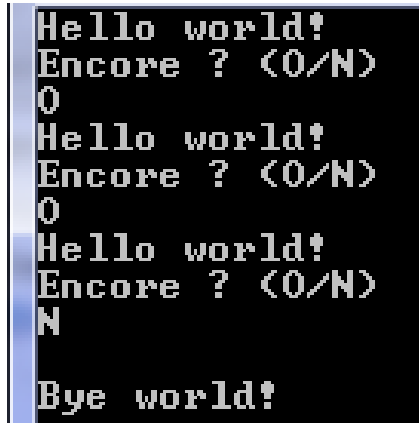
2. Stoppez la boucle avec la combinaison de touches : ctrl+C

3. Remplacez le while par un do while

```
do
    { Console.WriteLine("Hello !\n"); }
while(true) ;
```

Conclusion : pour une boucle infinie, while et do... while sont interchangeables.

4. Modifiez le code pour ne plus avoir de boucle infinie : c'est l'utilisateur qui décidera de relancer l'affichage oui ou non (O/N) . Rem : Votre programme **devra indiquer un message d'au revoir** en sortie de boucle.



```
Hello world!
Encore ? (O/N)
O
Hello world!
Encore ? (O/N)
O
Hello world!
Encore ? (O/N)
N
Bye world!
```

5. Essayez de remplacer le do... while par un while. Que faut-il faire ?

Conclusion : pour une boucle de « relance », le do... while est plus adapté.

EXO 2 : REPETER DES ACTIONS AVEC FOR

Ajoutez un projet « **Exo2_RepeterBonjourAvecFor** ».

Votre programme doit avec la structure **for**

- Afficher bonjour 100 fois
- Afficher « bonjour » un nombre de fois décidé par l'utilisateur
- Afficher les 10 premiers multiples de 3
- Afficher le nombre de multiple désiré par l'utilisateur du nombre désiré par l'utilisateur.

Conclusion : pour une boucle d'un nombre défini ou calculable, le for est plus adapté.

EXO 3 : JEU DE DEVINETTE

Ajoutez un projet « **Exo3_JeuDeDevinette** »

1. Votre programme doit faire le jeu suivant : « la devinette ». le programme choisit un nombre aléatoire entre 1 et 10, puis demande au joueur de deviner ce chiffre. Tant que le joueur n'a pas deviné, il doit réessayer. L'algorithme doit indiquer le nombre d'essai au bout duquel le joueur a gagné.
2. Que faut-il changer si on désire limiter le nombre d'essai à 5. Au bout de 5 essais, l'algorithme indique au joueur qu'il a perdu. Faites les modifications nécessaires.
3. Que faut-il ajouter pour que le joueur puisse rejouer à volonté ? Faites les modifications nécessaires.

EXO 4 : JEU DE DEVINETTE INVERSEE

Ajoutez un projet « **Exo4_JeuDevinetteInversee** » type console
Cette fois, l'utilisateur choisit un nombre au hasard.

Le programme demande à l'utilisateur un intervalle pour borner sa recherche
L'utilisateur indique un intervalle puis choisit un nombre compris dans l'intervalle qu'il garde en mémoire. Le programme fait une proposition de nombre. L'utilisateur lui indique s'il a le bon nombre, sinon si le nombre à trouver est plus grand ou plus petit. Et cela jusqu'à ce que le programme trouve le nombre choisit par l'utilisateur.

AIDE : pour trouver le chiffre en moins de coups possible, le programme doit à chaque fois donner le chiffre médian de l'intervalle.

ATTENTION :

- Vérifiez que cela fonctionne lorsque l'utilisateur choisit le nombre 0 et donne l'intervalle 0-20.
- Assurez-vous que l'utilisateur ne mente pas !