Les Boucles

1) La boucle : tant que

Syntaxe en algorithme :

```
Tant que condition faire
   instruction
fin tant que
```

Syntaxe en C – PHP :

```
while (condition){
   instruction ;
}
```

La boucle tant que exécute une ou plusieurs instructions autant de fois qu'une condition reste vérifiée. Si la condition est au départ non vérifiée, il n'y aura aucune exécution du bloc d'instructions.
Dans le corps de la boucle, il faut prévoir une instruction dévolution de la condition, sinon on aura une boucle infinie.

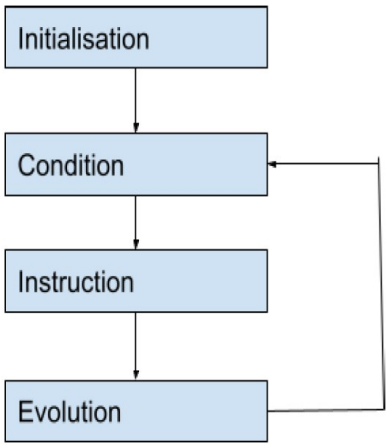
Exemple : Afficher tous les nombres pairs compris entre 2 et 20

```
Algo : nombres_pairs

Déclaration
   nb : entier

Début
   nb ← 2
   tant que nb <= 20 faire
      afficher(“Nombre pair : “, nb)
      nb ← nb+2
   fin tant que
fin nombres_pairs
```

A partir de cet algorithme, nous pouvons tirer le schéma général de la boucle tant que :



Traduction en C :

```
#include <stdio.h>

int main(){
   int nb ;
   nb=2 ;
   while (nb<=20){
      printf("Nombre pair : %d", nb) ;
      nb=nb+2 ;
   }
   return 0 ;
}
```

2) La boucle : faire tant que

Syntaxe en Algo :

```
faire
   instruction
tant que condition
```

Syntaxe en C – PHP :

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

Dans cette boucle, le test de la condition vient après l'exécution de l'instruction. Que si la condition est au départ non vérifiée, il y aura exécution une seule fois de l'instruction. Donc cette boucle exécute le bloc d'instructions au moins une fois. On dit que cette boucle est un schéma inverse de la première boucle tant que.

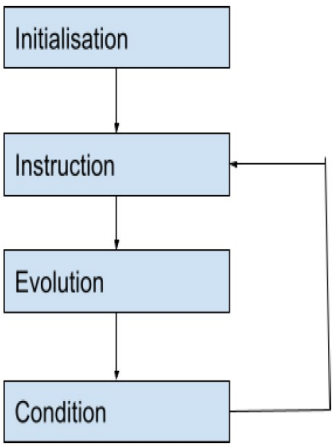
Exemple : Afficher tous les nombres pairs compris entre 2 et 20 :

```
Algo : nombres_pairs

Déclaration
   nb : entier

Début
   nb ← 2
   faire
      afficher("Nombre pair : ", nb)
      nb ← nb+2
   tant que nb <=20
fin nombres_pairs
```

A partir de cet algorithme, nous pouvons tirer le schéma général de la boucle tant que :



Traduction en C :

```
#include <stdio.h>

int main(){
   int nb ;
   nb=2 ;
   do{
      printf("Nombre pair : %d", nb) ;
      nb=nb+2 ;
   }while (nb <=20) ;
   return 0 ;
}
```

3) La boucle : pour

Syntaxe en Algo :

```
Pour indice allant de valeur_départ à valeur_fin pas de Pas faire
   instruction
fin pour
```

Syntaxe C – PHP :

```
for(initialisation ; condition ; évolution){
   instruction ;
}
```

La boucle pour exécute une ou plusieurs instructions un nombre de fois connu à l'avance, calculé à l'aide de la valeur_départ, la valeur-fin et le Pas
L'indice de parcours dans les langages évolués doit être de type énuméré : entier(int) ou bien caractère(char)
La boucle pour réalise automatiquement l'incrémentation ou la décrémentation de l'indice de la boucle

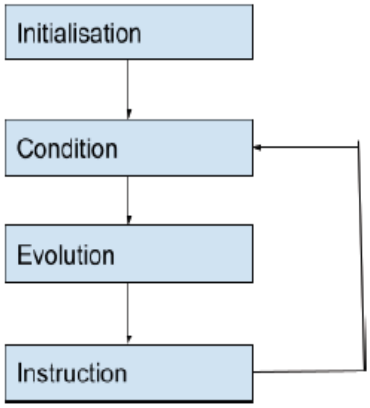
Exemple : Afficher tous les nombres pairs compris entre 2 et 20

```
Algo : nombres_pairs

Début
   nb : entier

Déclaration
   pour nb allant de 2 à 20 pas de 2 faire
      afficher(“Nombre pair : “ , nb)
   fin pour
fin nombres_pairs
```

Par défaut, le pas est de 1 ou -1, pas besoin de le spécifier dans la boucle



Traduction en C :

```
#include <stdio.h>

int main(){
   int nb ;
   for(nb=2; nb<=20; nb= nb+2){
      printf(“Nombre pair : %d“, nb) ;
   }
   return 0 ;
}
```

4) Série d'exercices

Exercice 1 : Écrire un Algo, C et PHP qui permet de saisir un entier et affiche tous les diviseurs

```
Algo : diviseurs

Déclaration
 nb, div : entier

Début
   afficher(“Donner un nombre entier : “)
```

```
   saisir(nb)
   div ← 1
   tant que div <= nb faire
      si nb%div=0
         alors afficher(“Diviseur : “, div)
      fin si
      div ← div+1
   fin tant que
fin diviseurs
```

Traduction en C :

```
#include <stdio.h>

int main ( ){
   int nb , div;
   printf ("Donner un nombre entier : ");
   scanf ("%d", &nb);
   div = 1 ;
   while (div <= nb){
      if (nb%div==0){
         printf ("\n Diviseur : %d ", div);
      }
      div = div + 1;
   }
   return 0;
}
```

Traduction en PHP :

```
<!DOCTYPE html>
<html>
   <head>
      <title>Diviseurs </title>
   </head>
   <body>
      <center>
         <h1> Recherche de diviseurs</h1>
         <form method="post" action="diviseurs.php">
            Nombre : <input type="text" name="nb">
            <input type="submit" name="Rechercher" value="Rechercher">
         </form>
         <?php
            if(isset($_POST['Rechercher'])) {
               $nb = $_POST['nb'];
               $div = 1 ;
               while ($div <= $nb){
                  if ($nb%$div==0){
                     printf ("<br/> Diviseur : %d ", $div);
                  }
```

```
                $div = $div + 1;
            }
        }
    ?>
  </center>
 </body>
</html>
```

Exercice 2 : Écrire un Algo, C, PHP qui permet de vérifier si un nombre entier saisi est un nombre parfait ou non
Un nombre entier est dit parfait s'il est égal à la somme de ses diviseurs sauf lui même

```
Algo : parfait

Déclaration
   nb, div, somme : entier

Début
   afficher ("Donner un nombre entier : ")
   saisir (nb)
   div ← 1
   somme ← 0
   tant que div < nb faire
      si nb % div = 0
         alors somme ← somme + div
      finsi
      div ← div + 1
   fin tant que
   si somme = nb
      alors afficher (nb , " est un nombre parfait")
      sinon afficher (nb , " n'est pas un nombre parfait")
   finsi
fin parfait
```

Traduction en C :

```
#include <stdio.h>

int main ( ){
   int nb , div, somme;
   printf ("Donner un nombre entier : ");
   scanf ("%d", &nb);
   div = 1 ;
   somme = 0 ;
   while (div < nb){
      if (nb%div==0){
         somme = somme + div ;
      }
      div = div + 1;
   }
   if (nb == somme){
```

```
        printf ("\n %d est un nombre parfait ", nb);
    }
    else{
        printf ("\n %d n'est pas un nombre parfait ", nb);
    }
    return 0;
}
```

Traduction en PHP :

```
<!DOCTYPE html>
<html>
    <head>
        <title>Parfait </title>
    </head>
    <body>
        <center>
            <h1> Test Nombre Parfait</h1>
            <form method="post" action="parfait.php">
                Nombre : <input type="text" name="nb">
                <input type="submit" name="Rechercher" value="Rechercher">
            </form>
            <?php
                if(isset($_POST['Rechercher'])) {
                    $nb = $_POST['nb'];
                    $div = 1 ;
                    $somme = 0;
                    while ($div < $nb){
                        if ($nb%$div==0){
                            printf ("<br/> Diviseur : %d ", $div);
                            $somme = $somme +$div;
                        }
                        $div = $div + 1;
                    }
                    if ($somme == $nb){
                        printf ("<br/> %d est un nb parfait", $nb);
                    }
                    else {
                        printf ("<br/> %d n'est pas un nb parfait", $nb);
                    }
                }
            ?>
        </center>
    </body>
</html>
```

Exercice 3 : Écrire un Algo, C et PHP qui permet de saisir un entier et une limite et affiche la table de multiplication du nombre jusqu'à la limite

```
Algo : table
```

```
Déclaration
   nb, mult, resultat, limite : entier

Début
   afficher ("Donner un nombre entier : ")
   saisir (nb)
   afficher ("Donner la limite de la table : ")
   saisir (limite)
   pour mult allant de 1 à limite faire
      resultat ← nb * mult
      afficher(mult, " * ", nb , " = ", resultat)
   fin pour
fin table
```

Traduction en C :

```
#include <stdio.h>

int main(){
   int nb, mult, resultat, limite;
   printf("Saisir un nombre entier : ");
   scanf("%d", &nb);
   printf("Saisir la limite : ");
   scanf("%d", &limite);
   for(mult=1; mult<=limite; mult= mult+1){
      resultat=nb*mult;
      printf("\n%d * %d = %d", mult, nb, resultat);
   }
   return 0;
}
```

Traduction en PHP :

```
<!DOCTYPE html>
<html>
   <head>
      <title>table </title>
   </head>
   <body>
      <center>
         <h1> Table de multiplication</h1>
         <form method="post" action="table.php">
            Nombre : <input type="text" name="nb">
            Limite : <input type="text" name="limite">
            <input type="submit" name="Afficher" value="Afficher">
         </form>
         <?php
            if(isset($_POST['Afficher'])) {
               $nb = $_POST['nb'];
               $limite = $_POST['limite'];
               for ($mult=1; $mult <= $limite; $mult++){
```

```
                $resultat = $mult * $nb;
                printf("<br/>%d*%d=%d", $mult, $nb, $resultat);
            }
        }
    ?>
  </center>
 </body>
</html>
```

Exercice 4 : Écrire un Algo, C et PHP qui permet de calculer le factoriel d'un nombre entier

```
Algo:factoriel

  Declaration
    nb,compt,fact : entier

Debut
  fact ← 1
  compt ← 1
  afficher(“Donner un entier”)
  saisir(nb)
  tant que compt <=nb faire
    fact ← fact*compt
    compt ← compt+1
  fin tant que
  afficher(“Le factoriel est:”,fact)
fin factoriel
```

Traduction C :

```
#include <stdio.h>

int main(){
  int nb, compt, fact;
  printf(“Donner un entier”);
  scanf(“%d”, &nb);
  fact=1;
  compt=1;
  while(compt <= nb){
    fact=fact*compt;
    compt+=1;
  }
  printf(“%d! = %d”,nb, fact);
  return 0;
}
```

Traduction PHP :

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
```

```
      <title>Exercice 4</title>
   </head>
   <body>
      <center>
         <h1> Exercice donne le factoriel</h1>
         <p> Donne le factoriel du chiffre donné</p>
         <form method="post" action="Ex_4.php">
            <table border="0">
               <tr>
                  <td>Donner un entier: </td>
                  <td> <input type="text" name="nb"> </td>
               </tr>
               <tr>
                  <td> </td>
                  <td> <input type="reset" name="Annuler" value="Annuler">  <input
type="submit" name="Valider" value="Valider"> </td>
               </tr>
            </table>
         </form>
         <?php
            if(isset($_POST['Valider'])){
               $nb = $_POST['nb'];
               $fact=1;
               $compt=1;
               while ($compt <= $nb){
                  $fact=$fact*$compt;
                  $compt+=1;
               }
               printf("<br>\n %d! = %d",$nb,$fact);
            }
         ?>
      </center>
   </body>
</html>
```