La estructura while

Esta estructura sirve para iterar, es decir, realizar una o más acciones varias veces. Su sintaxis es:

```
while (condición) {
                acción a repetir;
                otra acción a repetir;
} //Terminó el bucle
```

Si queremos, por ejemplo, escribir los primeros 100 números naturales:

```
<?php
$i=1;
//El bucle siguiente se repetirá mientras $i sea menor que 101
while($i<101) {
      echo "$i<br>";
      $i++; //Incremeta $i. Equivale a $i=$i+1;
}
?>
```

Algunas notas sobre while:

- Al igual que en if, las llaves no son obligatorias si lo que se va a iterar es una sola sentencia.
- La variable en la condición debe inicializarse, es decir, debe tener valor definido la primera vez que es evaluada la condición del bucle. En nuestro ejemplo, la variable $i se inicializa en 1.
- Es importante verificar que la condición del bucle dé falso alguna vez, o quedaríamos “atrapados” en un bucle infinito.

La orden **break;** sirve para terminar el bucle inmediatamente. La orden **continue;** sirve para terminar la actual iteración bucle, y volver a evaluar la condición.

La estructura do-while

Es una estructura de bucle, similar a while. La principal diferencia es que la condición se evalúa **después** de que se ha ejecutado cada iteración. Por lo tanto, nos aseguramos de que el ciclo se ejecutará al menos una vez. Ejemplo:

```
<?php
$i = 0;
do {
      echo $i;
      $i--; //Decrementa $i en una unidad
} while ($i > 0);
/*La condición es falsa desde el principio ($i nunca fue mayor que cero).
Sin embargo, el ciclo se ejecuta una vez (imprime el número cero).*/
?>
```

La estructura "for"

“For” es el tipo de bucle más complejo. Lo explicaremos con un ejemplo: mostrar los primeros 100 números.

```
<?php
for($i=1; $i<101; $i++) {
     echo "$i ";
}
?>
```

La línea resaltada muestra las tres expresiones de un bucle “for”, separadas por punto y coma.
La primera es la inicialización, y se ejecuta incondicionalmente y por única vez, la primera vez que corre el ciclo.
La segunda es la condición. Es evaluada en cada iteración. Si es TRUE, se vuelve a iterar.
La tercera expresión es ejecutada al terminar cada iteración.
Cualquiera de las tres expresiones puede estar vacía.
Ejemplo: Mostrar por pantalla los 100 primeros números naturales, excepto el 73.

```
<?php
for($i=1; $i<101; $i++) {
      if($i==73) { continue;}
      echo "$i ";
}
?>
```

En el ejemplo anterior, se utliza *continue;* que sirve para “saltarse” este ciclo de la iteración, y pasar al siguiente.
Les dejo los enlaces a la documentación oficial:

- if http://www.php.net/manual/es/control-structures.if.php
- while http://www.php.net/manual/es/control-structures.while.php
- do-while http://www.php.net/manual/es/control-structures.do.while.php
- for http://www.php.net/manual/es/control-structures.for.php
- break http://www.php.net/manual/es/control-structures.break.php
- continue http://www.php.net/manual/es/control-structures.continue.php

1) Hacer tres programas en php que muestren la suma de los primeros 100 números. Usar "while" la primera vez, "do-while" la segunda, y "for" la tercera.
2) Pedir al usuario la base y la altura de un rectángulo. Dibujarlo con asteriscos. Ej 2 x 5

```
*****
*****
```

3) Pedir al usuario la altura de un triángulo. Dibujarlo con asteriscos. Ej: 4

```
*
**
***
****
```