

```
<?php
/* ejemplo 1 */

$i = 1;
while ($i <= 10) {
    echo $i++; /* el valor presentado seria
                $i antes del incremento
                (post-incremento) */
}

/* ejemplo 2 */

$i = 1;
while ($i <= 10):
    echo $i;
    $i++;
endwhile;
?>
```

Le dice a PHP que ejecute las sentencias anidadas, tanto como la expresión `while` se evalúe como true.

**while\_**

Al igual que con la sentencia `if`, se pueden agrupar varias instrucciones dentro del mismo bucle `while` rodeando un grupo de sentencias con corchetes, o utilizando la sintaxis alternativa

# SENTENCIAS DE CONTROL

```
<?php
$i = 0;
do {
    echo $i;
} while ($i > 0);
?>
```

**do-while**

Los bucles `do-while` son muy similares a los bucles `while`, excepto que la expresión verdadera es verificada al final de cada iteración en lugar que al principio.

La diferencia principal con los bucles `while` es que está garantizado que corra la primera iteración de un bucle `do-while` (la expresión verdadera sólo es verificada al final de la iteración)

```
<?php
if ($i == 0) {
    echo "i es igual a 0";
} elseif ($i == 1) {
    echo "i es igual a 1";
} elseif ($i == 2) {
    echo "i es igual a 2";
}

switch ($i) {
    case 0:
        echo "i es igual a 0";
        break;
    case 1:
        echo "i es igual a 1";
        break;
    case 2:
        echo "i es igual a 2";
        break;
}
?>
```

la sentencia continue se aplica a `switch` y actúa de manera similar a `break`. Si se tiene un `switch` dentro de un bucle y se desea continuar a la siguiente iteración de del ciclo exterior, se utiliza `continue 2`.

**switch\_**

es similar a una serie de sentencias `IF` en la misma expresión. En muchas ocasiones, es posible que se quiera comparar la misma variable (o expresión) con muchos valores diferentes, y ejecutar una parte de código distinta dependiendo de a que valor es igual. Para esto es exactamente la expresión `switch`.