Instituto Superior San Pablo Nº 9112

Desarrollo de Sistemas Web

Tec. Superior Análisis Funcional de Sistemas

Estructuras de control

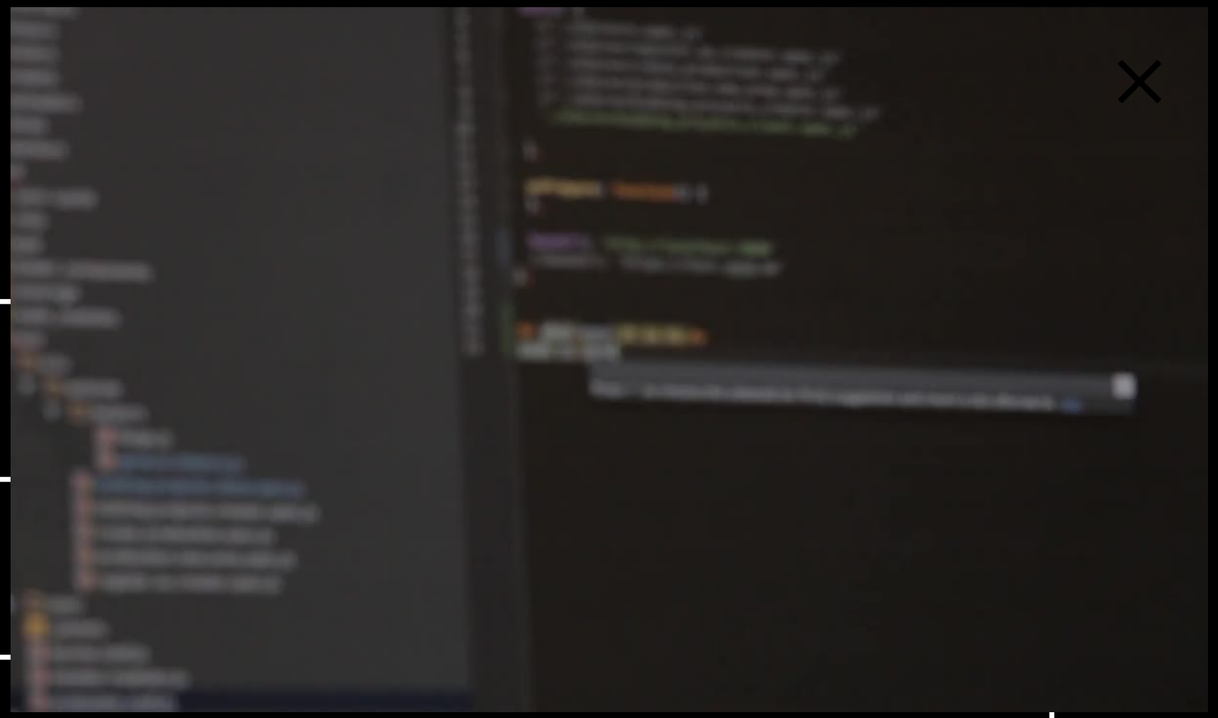
```php
for ($i = 1; $i <= 10; $i++) {
    echo "<br>";
    echo $i;
}
```

¿Para qué sirven?

Se utilizan para controlar el flujo de ejecución de un programa. En otras palabras, para tomar decisiones y realizar diferentes acciones en función de las condiciones que presentemos.

- Flujo condicional
- Flujo iterativo o bucles
- Flujo de excepciones

Flujo Iterativo



¿Para qué nos sirven?

Permiten que un bloque de código se ejecute repetidamente hasta que se cumpla una condición específica o la detengamos con alguna sentencia.

¿Cuándo las usamos?

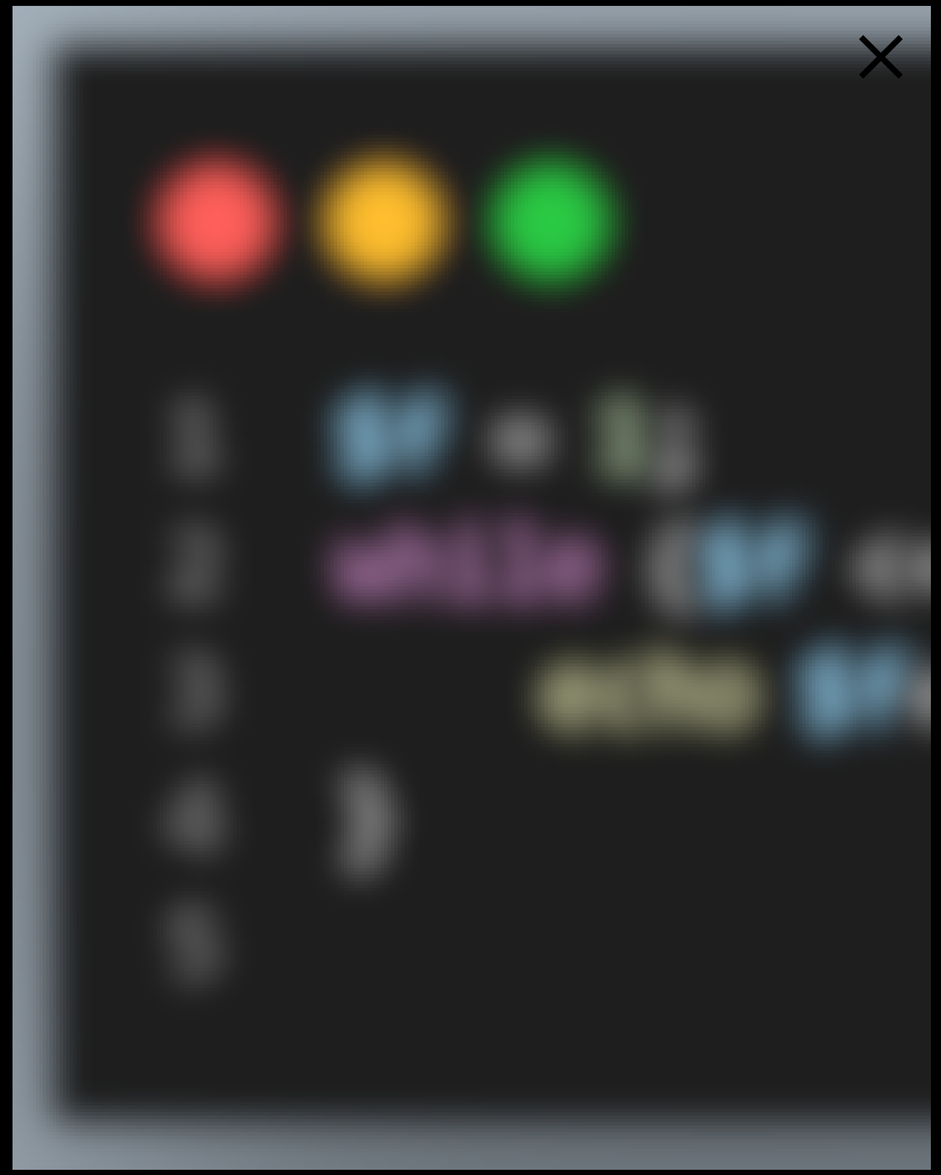
Se utilizan cuando necesitamos realizar una misma acción varias veces como mostrar datos de elementos o procesar datos de entrada.

¿Cuáles existen?

- While
- Do-While
- For
- Foreach

- Fácil de usar
- Condición verdadera
- Comprueba en cada ciclo

while



Continúa hasta que la condición deje de ser verdadera

WHILE 🔁

```php
$f = 1;
while ($f <= 10) {
    echo $f++;
}
```

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

Do-While

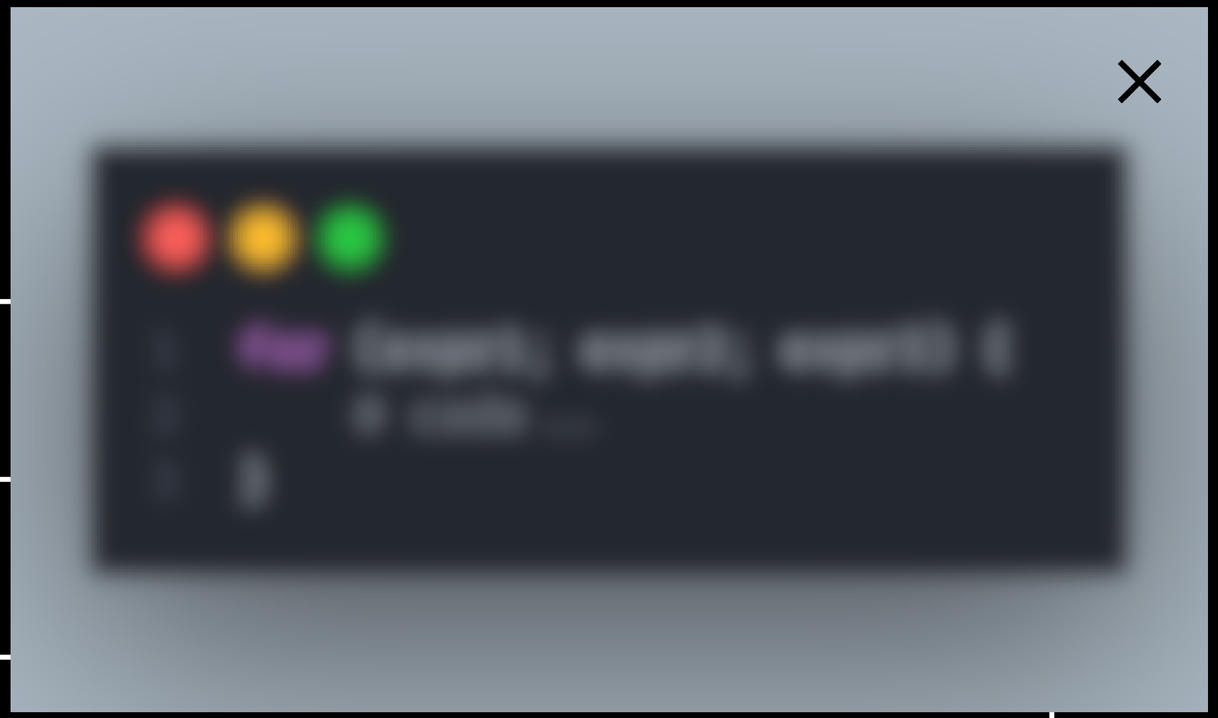
- Similar al WHILE
- Condición verdadera
- Comprueba al final

A diferencia del WHILE, este ciclo se ejecuta al menos una vez.

DO-WHILE

```php
$i=0;
do {
    echo $i++;
} while ($i < 10);
```

FOR



¿Para qué nos sirven?

Este tipo de iteración nos va a servir para repetir una acción un número determinado de veces

¿Cuándo las usamos?

Tiene muchos usos, sin embargo los más comunes son recorrer Arrays, mostrar elementos en un sitio web y realizar cálculos repetitivos.

¿Cómo se escribe?

Si bien su sintaxis es sumamente fácil, según la documentación de PHP es una de las iteraciones más complejas. 😱

FOR 🔁

FOR 🔁

```php
for ($i = 1; $i ≤ 10; $i++) {
    echo "<br>";
    echo $i;
}
```

FOREACH

```php
foreach ($variable as $key) {
    # code ...
}
```

- Iterar un array
- Contenido
- KEY & Contenido

FOREACH

Funcionamiento

```php
$alumnos = [
    [
        "nombre" => "Juan",
        "edad" => 20,
        "notas" => [
            "mate" => 10,
            "fisica" => 8,
            "ingles" => 7
        ]
    ],
    [
        "nombre" => "Pedro",
        "edad" => 20,
        "notas" => [
            "mate" => 10,
            "fisica" => 8,
            "ingles" => 7
        ]
    ],
];
```

FOREACH

Funcionamiento

```php
//Foreach Alumnos
foreach ($alumnos as $alumno) {
    echo "Nombre: ".$alumno['nombre']." <br>";
    echo "Edad: ".$alumno['edad']." <br>";
    echo "Notas: <br>";
    echo "Mate: ".$alumno['notas']['mate']." <br>";
    echo "Fisica: ".$alumno['notas']['fisica']." <br>";
    echo "Ingles: ".$alumno['notas']['ingles']." <br>";
    echo "<hr>";
}
```

```php
$alumnos = [
    [
        "nombre" => "Juan",
        "edad" => 20,
        "notas" => [
            "mate" => 10,
            "fisica" => 8,
            "ingles" => 7
        ]
    ],
    [
        "nombre" => "Pedro",
        "edad" => 20,
        "notas" => [
            "mate" => 10,
            "fisica" => 8,
            "ingles" => 7
        ]
    ],
];
```

Ejercitación

 20'

 Realiza un programa que muestre por pantalla la tabla de multiplicar del número 7 utilizando un bucle for.

 Realiza un programa que pida al usuario un número y muestre por pantalla todos los números pares desde 0 hasta ese número utilizando un bucle while.

BREAK

10/15 Minutos

BREAK

```php
$i = 0;
while (++$i) {
    switch ($i) {
    case 5:
        echo "En 5, salgo del switch pero
            continúo ejecutando el while.<br />";
        break 1;
    case 10:
        echo "En 10; salgo del switch y del while.<br />";
        break 2;
    default:
        break;
    }
}
```

Es una estructura de control de flujo.
Se utiliza para interrumpir la ejecución de un bucle (como un bucle FOR, WHILE o DO-WHILE) o de un SWITCH-CASE, y saltar inmediatamente a la siguiente instrucción fuera del bucle o del switch-case.

Esta estructura es útil para hacer que la ejecución de un programa sea más eficiente y para manejar casos especiales que requieren una salida anticipada de alguna otra estructura.

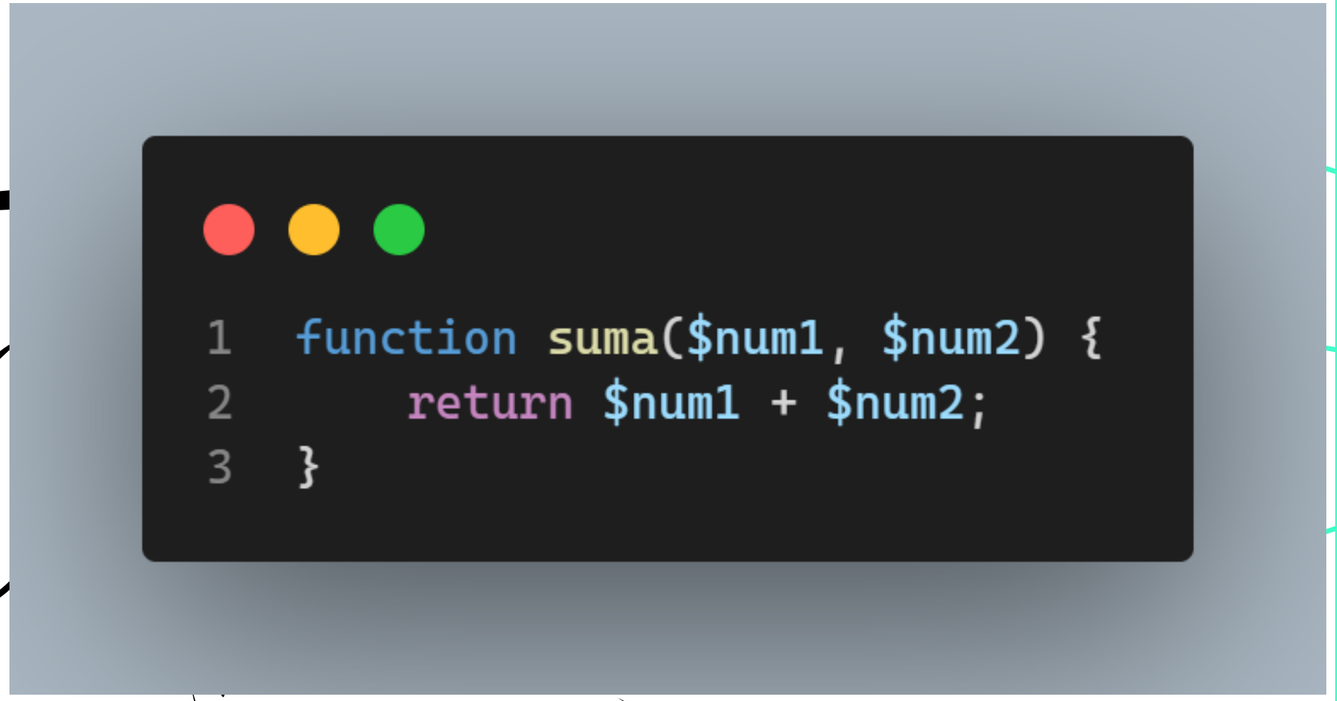
CONTINUE

```php
for ($i = 0; $i ≤ 10; ++$i) {
    if ($i % 2 ≠0) {
        continue;
    }
    echo "$i\n";
}
```

Es una estructura de control de flujo ITERATIVO.
Se utiliza para salar a la siguiente iteración de un bucle sin ejecutar el resto del código que sigue a la estructura **continue;** en esa iteración en particular.

Se utiliza comunmente en bucles FOR, WHILE y DO-WHILE.

RETURN

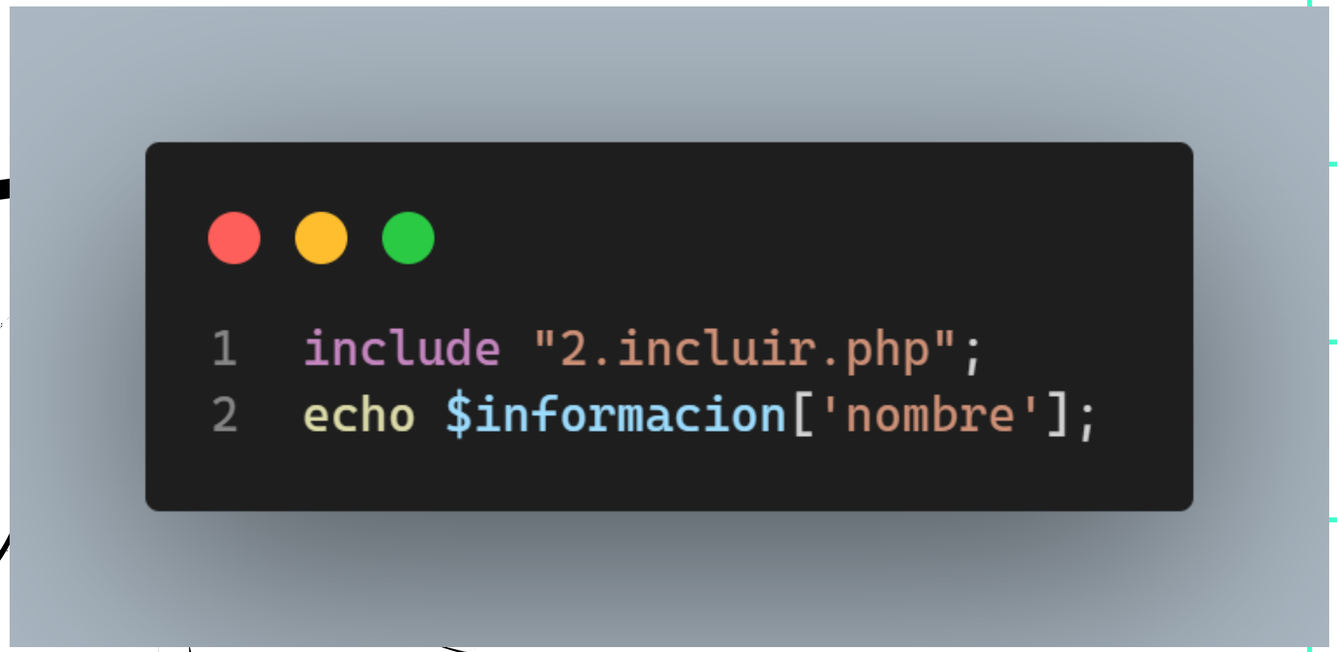


Es una estructura de control de flujo.
Se utiliza para salir de una función y devolver un valor opcional a quien la haya invocado.

Se utiliza comunmente en funciones. El valor devuelto puede ser de cualquier tipo de dato en PHP. Si no tiene valor de retorno, por defecto retorna un NULL

Esta estructura solo puede ser utilizada dentro de funciones

INCLUDE / INCLUDE_ONCE



Es una instrucción que se utiliza para incluir el contenido de un archivo en otro archivo PHP.

Es de suma utilidad cuando se desea dividir un programa en archivos más pequeños y reutilizables.

El contenido del archivo se inserta donde se encuentra la instrucción

REQUIRE / REQUIRE_ONCE

```php
require "2.incluir.php";
echo $informacion['nombre'];
```

Es una instrucción que se utiliza para incluir el contenido de un archivo en otro archivo PHP.

Es de suma utilidad cuando se desea dividir un programa en archivos más pequeños y reutilizables.

Si el archivo que se intenta incluir no existe o tiene fallas, REQUIRE dentendrá la ejecución.

Y ahora?

Momento
CODING

Ejercitación

 30'

1. Generar un array de 50 números enteros de forma aleatoria.
2. Utilizar estructuras de control a gusto para recorrer el array creado.
3. Determinar si el número 5, se encuentra incluído en los valores del arreglo.
4. Mostrar por pantalla el resultado de la operación (encontrado o no encontrado). En caso de que existiera el valor indicar su posición.

- *Indagar sobre el uso de la función rand() en la documentación oficial de PHP para el punto 1.*
- *Bonus: buscar en la documentación oficial array_push() y array_search(). Realizar el mismo ejercicio haciendo uso de estas funciones.*

Bibliografía

Pueden consultar más información en los siguientes enlaces.

Páginas web

- www.php.net