UD2.- El lenguaje PHP.

Tipos de datos, constantes, operadores, operaciones abreviadas y comentarios.

Introducción a PHP

- print
- echo
- printf
- comentarios

Ejemplo, primer programa en PHP.

¿Qué es PHP?

PHP es un lenguaje de programación de propósito general, aunque su punto fuerte es el desarrollo web. Entre sus características más importantes están:

- El código se ejecuta en el servidor (en *Apache* mediante *mod_php*)
- El cliente recibe el resultado generado tras interpretar el código en el servidor.
- El código se almacena en archivos con extensión ***.php***.

Como vimos en la unidad anterior PHP es un lenguaje que permite programar pequeños scripts con gran rapidez.

Ejemplo, primer programa en PHP.

Ejemplo programa

En primer lugar, empezamos viendo un sencillo programa que muestra un texto por pantalla, y vamos a explicar cada una de las líneas de código de ese programa.

Nota**.- Recuerda guardar el archivo con la extensión **.php

Ejemplo de nuestro primer programa en PHP

```php
<?php
    echo "Bienvenid@ a la programación PHP";
?>
```

El programa muestra el mensaje:
"*Bienvenid@ a la programación PHP*"

Interpretación código (PHP)

PHP simplemente interpreta el texto del archivo de código con extensión **.php** hasta que encuentra uno de los caracteres especiales que delimitan el inicio de código **php**.
<?php

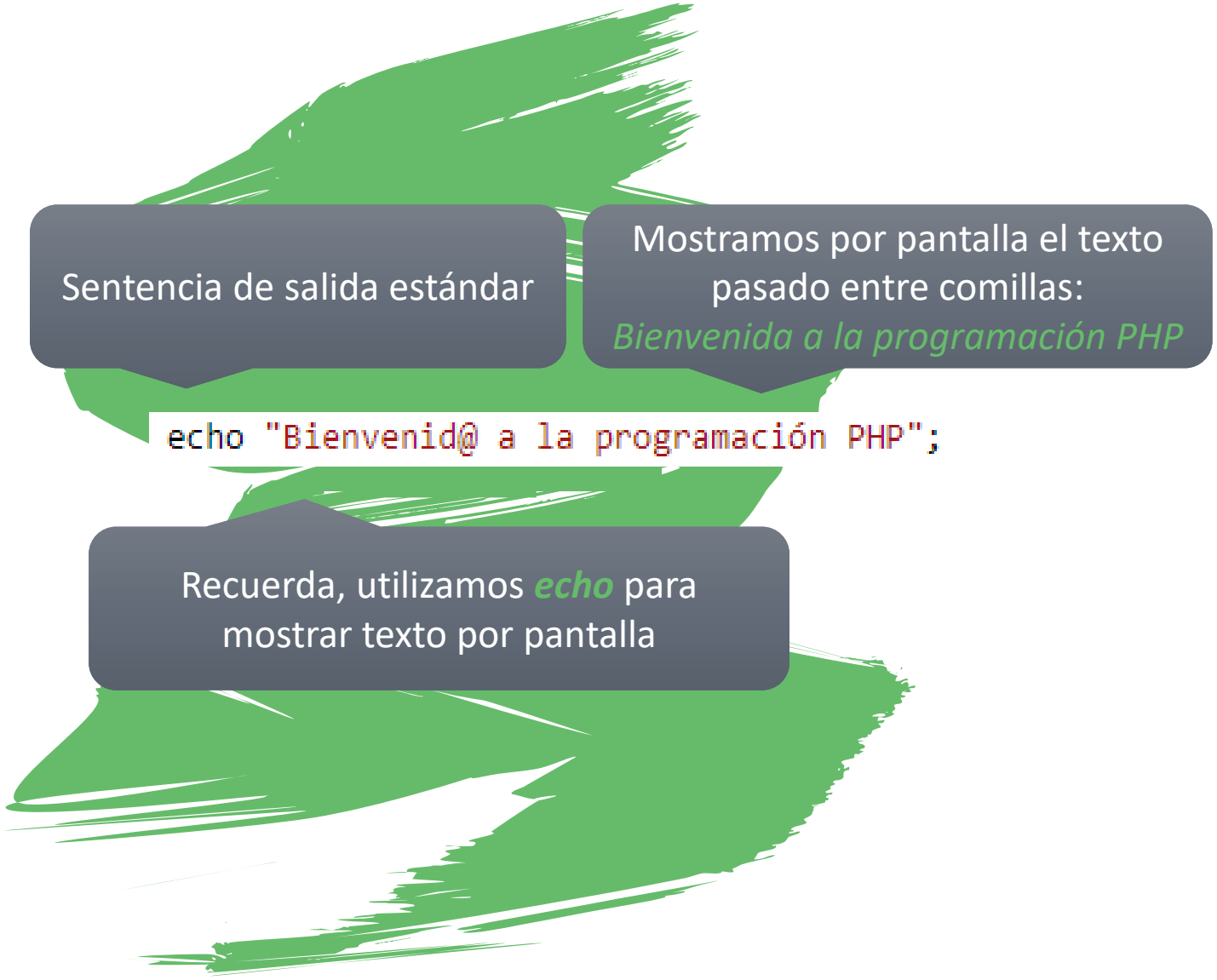
A esto se le conoce como **Código Embebido**

```
<!DOCTYPE html>
<html>
    <head>
        <title>Prueba PHP</title>
    </head>
    <body>
        <?php echo '<p>Hola mundo</p>'; ?>
    </body>
</html>
```

Los bloques de código **PHP**, se escriben entre **<?php** y **?>,** mientras que las sentencias se separan con **;**

Mostrar por Pantalla

echo se conoce como la sentencia de salida estándar. Permite al código en PHP mostrar información en pantalla.
La sentencia **echo** muestra (o imprime) una línea de texto en la pantalla.
La cadena dentro de los paréntesis es el argumento para la sentencia, "**Bienvenid@ a la programación PHP**".



Mostrar por Pantalla

Opcionalmente, en el caso de que sólo tengamos instrucciones echo, podemos sustituir la estructura ***<?php ... ?>*** por la estructura **<?= ... ?>** y ahorrarnos la instrucción echo. Así, el párrafo que se muestra en el ejemplo anterior lo podríamos poner así:

Sentencia de salida estándar

Mostramos por pantalla el texto pasado entre comillas:
Bienvenida a la programación PHP

```
<?= "Bienvenid@ a la programación PHP"; ?>
```

Mostrar por Pantalla

Además de la sentencia **echo** podemos utilizar las sentencias **print** y **printf** para mostrar información por pantalla.
Veamos como utilizarlos:

Las líneas de código que vemos a continuación son equivalentes.

```
echo "Bienvenid@ a la programación PHP";
```

La sentencia ***print*** muestra los mensajes en la misma línea, a no ser que introduzcamos una secuencia de escape '***\n***'

```
print "Bienvenid@ a la programación PHP";
```

La sentencia ***printf*** se utiliza para mostrar los mensajes con formato, igual que sucede en el lenguaje **C**.

```
printf("Bienvenid@ a la programación PHP");
```

Todas muestran el mismo mensaje de texto en una línea.

Mostrar por Pantalla

Se pueden concatenar varios valores usando el operador (.):

Aunque se verán las variables y los operadores en profundidad un poco más adelante en esta misma unidad, podemos observar en este caso como se utiliza el operador '.' para concatenar el texto con el valor de la variable

```
$resultado = 4;
echo "El resultado es: " . $resultado;
```

Al ejecutar el programa se mostrará el siguiente mensaje:
El resultado es: 4

También se puede escribir así

```
$resultado = 4;
echo "El resultado es $resultado";
```

Mostrar por Pantalla

printf Se comporta de forma similar al *printf* original de C:

- Número variable de parámetros. El primero debe ser la cadena a mostrar por pantalla.
- Se pueden incluir una serie de caracteres especiales que determinarán el tipo de dato por el que se sustituirán en la salida por pantalla.
  - **%d** para tipos enteros (long , int).
  - **%f** para números reales (float y double).
  - **%s** para representar una cadena.

Ejemplos de *printf()*

```
$nombre = "Pedro";
$edad = 30;
printf("Mi nombre es %s y tengo %d años.", $nombre, $edad);
```

```
Mi nombre es Pedro y tengo 30 años.
```

```
$numero_decimal = 123.456;
printf("El valor es: %f\n", $numero_decimal);
```

```
El valor es: 123.456000
```

```
$precio = 19.99;
printf("El precio es: %.2f€\n", $precio); // Mostrará 2 decimales
```

```
El precio es: 19.99€
```

Entrada de datos

En **PHP**, la entrada de datos "por teclado" generalmente se refiere a la recepción de datos a través de un formulario HTML, que luego se procesan en el script PHP usando las variables superglobales ***$_GET***, ***$_POST*** o ***$_REQUEST*** para capturar la información enviada desde el navegador del usuario. Para la entrada de datos en la consola cuando se ejecuta **PHP** en el servidor, se utiliza la función **readline()**.

Para leer datos de consola desde teclado cuando se ejecuta PHP en el servidor

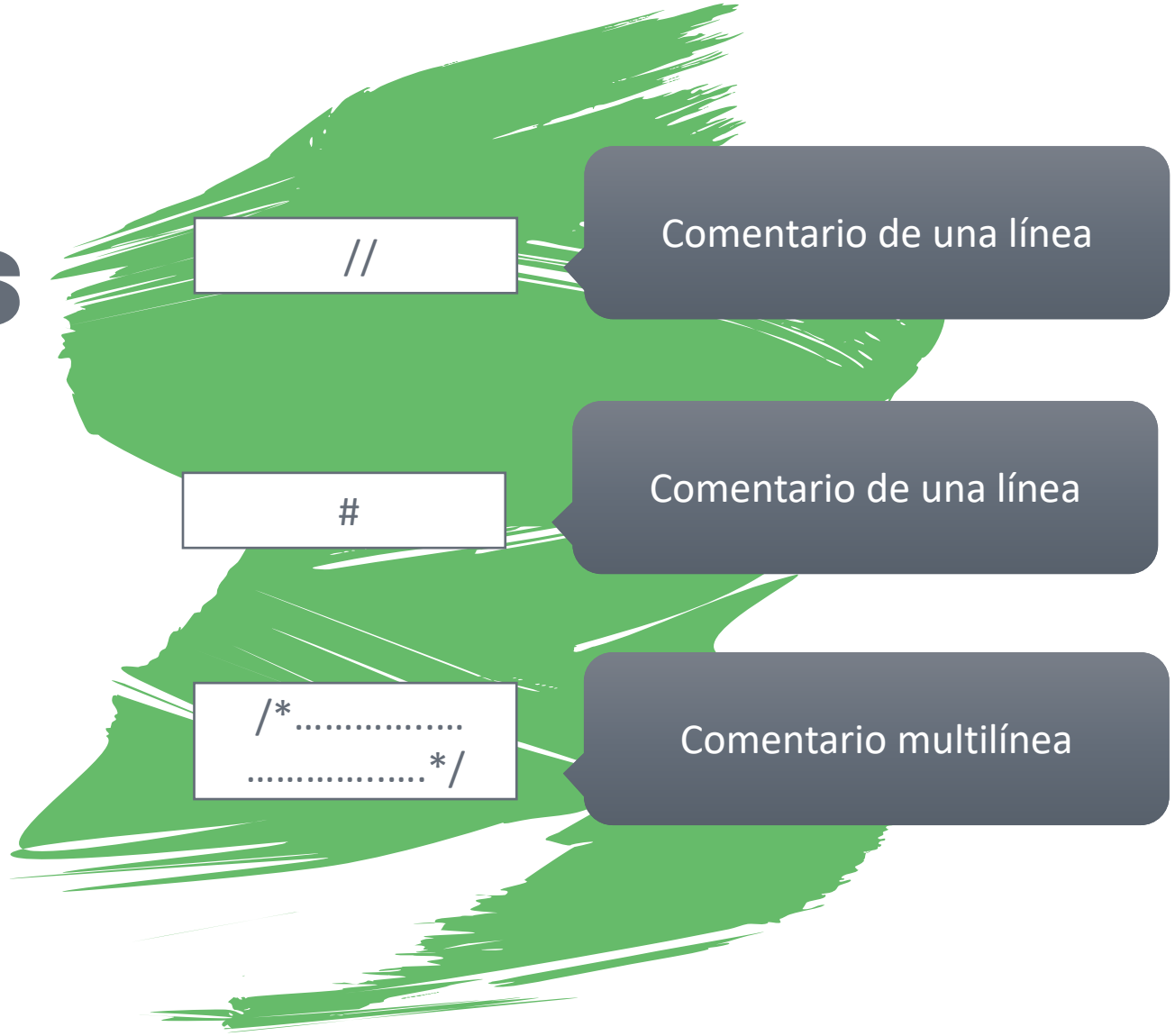
```php
<?php
// En un script de consola
$entrada = readline('Por favor, ingresa tu nombre: ');
echo "Tu nombre es: " . $entrada . "\n";
?>
```

Comentarios

//, #, /*....*/

Los comentarios son bloques de código que no interpretan los compiladores y que se utilizar para documentar nuestro código de programación.
Podemos utilizar // o # para realizar comentarios en una única línea.

Si queremos realizar comentarios que ocupen varias líneas podemos realizar la apertura del comentario /* en una línea y posteriormente cerrar el comentario con */. Todo el texto ubicado entre estos símbolos será considerado como comentario.



Ejemplo, de comentarios en PHP.

Ejemplos Comentarios

En PHP podemos utilizar comentarios de una línea o de bloque.

Nota**.- Recuerda guardar el archivo con la extensión **.php

Comentario de una línea // o también podemos utilizar #

```php
<?php

// Este es un comentario de una sola línea
# Este también es un comentario de una única línea
/*
  Este es
  un comentario
  que ocupa
  varias líneas
*/

?>
```

Comentario de varias líneas empiezan con /* y finalizan con */

Variables

Declaración

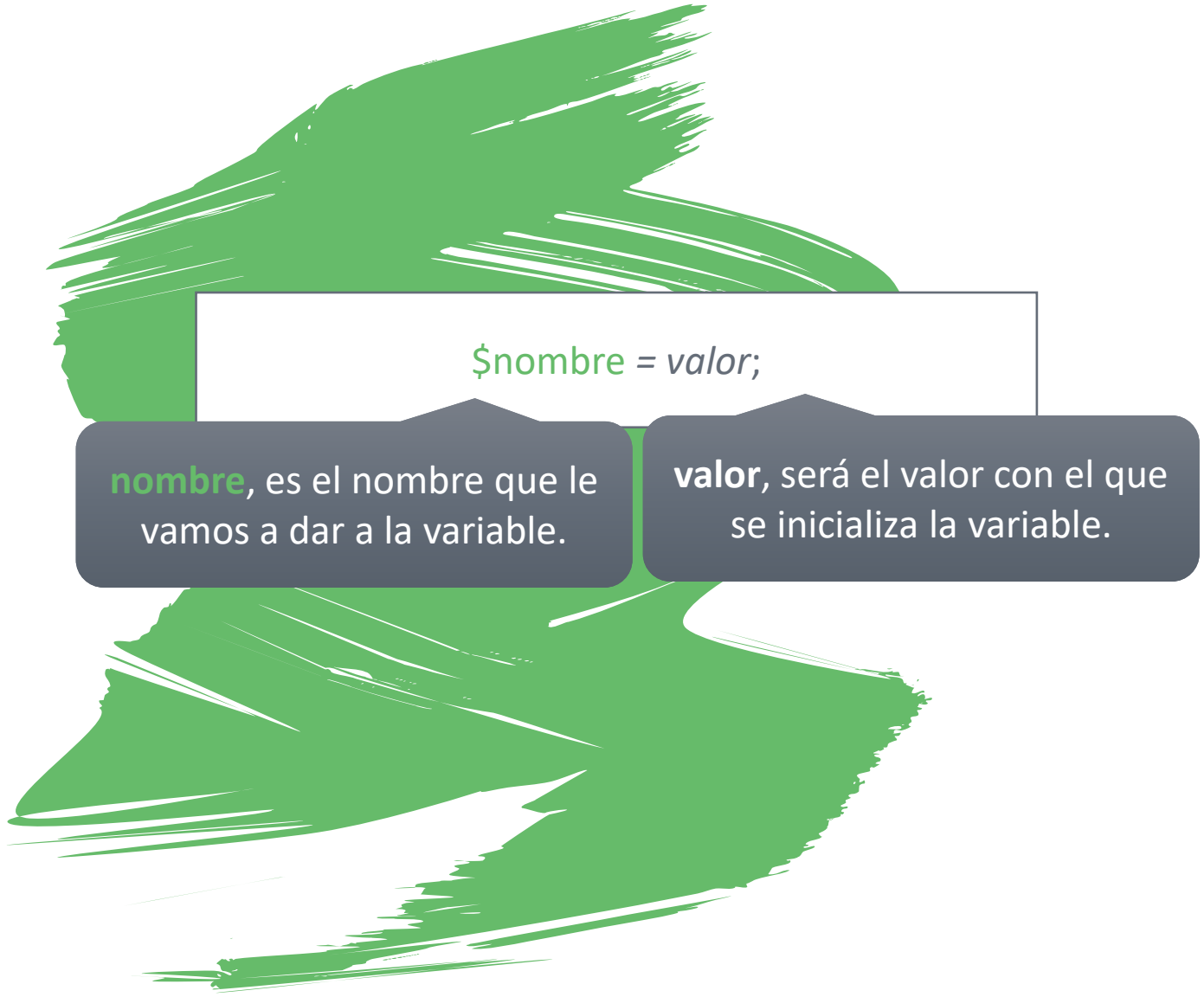
Las variables no se declaran previamente como en otros lenguajes (p.e. java). Se crean la primera vez que se les asigna un valor.

TIPO

El tipo de dato dependerá del valor con el que se inicialice la variable.

DECLARACIÓN

Van siempre precedidos del carácter '**$**'. El nombre de la variable debe empezar por una letra o '_', y puede estar formado por números, letras y guiones bajos.



Variables

Tipos de datos

PHP soporta los siguientes tipos básicos de datos (variables):

- **Booleano**: variables que sólo pueden almacenar dos valores, verdadero o falso, en PHP **TRUE** o **FALSE**, en mayúsculas.
- **Enteros**: números sin decimales. Se pueden representar en formato decimal (p.e. 15), formato octal (poniendo un 0 delante, p.e. 0156) o hexadecimal (poniendo 0x delante, p.e. 0x2B4).
- **Reales (decimales)**: números con decimales. La parte decimal queda separada de la entera con un punto. Por ejemplo, 3.14. También podemos usar notación científica, como por ejemplo 2.5e3, que equivale a $2.5 \cdot 10^3$.
- **Textos (cadenas)**: se pueden representar entre comillas simples o dobles. Si los representamos con comillas dobles, podemos intercalar variables dentro.

Ejemplos de tipos de datos.

```php
<?php
    $nombre = "Paco Guirau";

    $nota = 6.8;

    $edad = 12;

    $esInfantil = TRUE;

    echo "El alumno $nombre tiene una nota de $nota";
?>
```

Ejemplos de uso de declaración de variables.

Ejemplo variables

Como podemos observar se pueden declarar las variables de diversas formas. Se pueden declarar una a una en diferentes líneas del programa.
Veamos algunos ejemplos de declaración de variables.

```php
$edad = 19;
```

La variable será de tipo *entero,* llamada *edad,* e inicialmente tiene el valor *19*

```php
$pi = 3.14; //Declaración de pi como float


$nombre = "Paco";   //Declaración de nombre como string
```

Ejemplos de uso de conversiones de tipos.

Conversiones entre tipos

Si tenemos un tipo de dato, y lo queremos convertir a otro, PHP tiene una serie de funciones que nos permiten hacer estas conversiones, ***aunque estas conversiones son automáticas con las últimas versiones de PHP:***

- **intval($var)** *sirve para convertir un valor (en formato cadena, o real) a entero.*
- **doubleval($var)** *sirve para convertir un valor a número real.*
- **strval($var)** *sirve para convertir un valor a una cadena de texto.*

Ejemplo de uso de *intval(),* para convertir un texto a entero.

```php
<?php
    $textoEdad="18";    //variable de tipo texto
    $edad = intval($textoEdad); //variable de tipo numérico

?>
```

Ejemplos de uso de declaración de variables.

Estado de las variables

Es posible que, en algún momento de la ejecución del programa, una variable no tenga un valor definido, o queramos eliminar el valor que tiene. Para ello tenemos algunas instrucciones útiles:

- **unset($var)** *elimina la variable (como si lo se hubiese creado)*
- **isset($var)** *comprueba si la variable existe*
- **empty($var)** *comprueba si una variable está vacía*

Ejemplo de uso de *unset(),* y *empty(),* para eliminar una variable, y para comprobar si está vacía.

```php
<?php
    $var1="Ejemplo de uso";
    unset($var1);       // La variable var1 deja de existir
    $var1 = "";
    echo empty($var1);  // Muestra que es cierto (true, 1),
?>
```

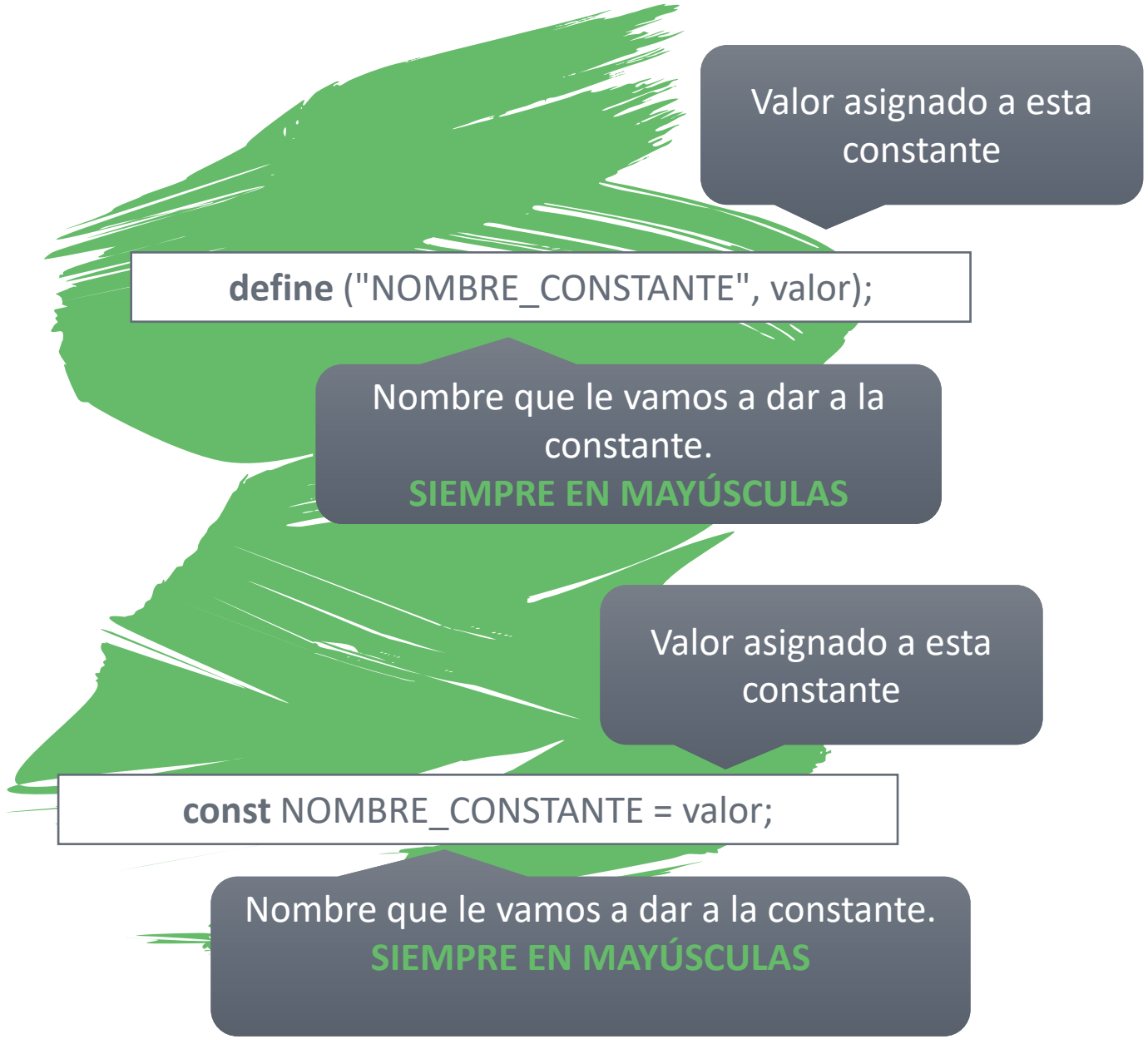
Constantes

Son un tipo especial de variables cuyo valor no cambia nunca, de ahí su nombre "*constantes*".
Se pueden declarar de dos formas:

- **define("NOMBRE", valor);**.
- **const NOMBRE=valor;**

USOS

Usa **`const`** si conoces la constante desde el inicio (dentro o fuera de clases).
Usa **`define()`** si necesitas declarar una constante **condicionalmente o en tiempo de ejecución**.



Ejemplos de uso de declaración de constantes.

Ejemplo constantes

En los ejemplos de la derecha podemos ver como se realiza la declaración de algunas constantes:

Declaración de una constante de tipo *float*, llamada *PI*

```
define("PI", 3.14);
```

A esta constante le asignamos el valor 3.14

Declaración de una constante de tipo *float*, llamada *IVA*

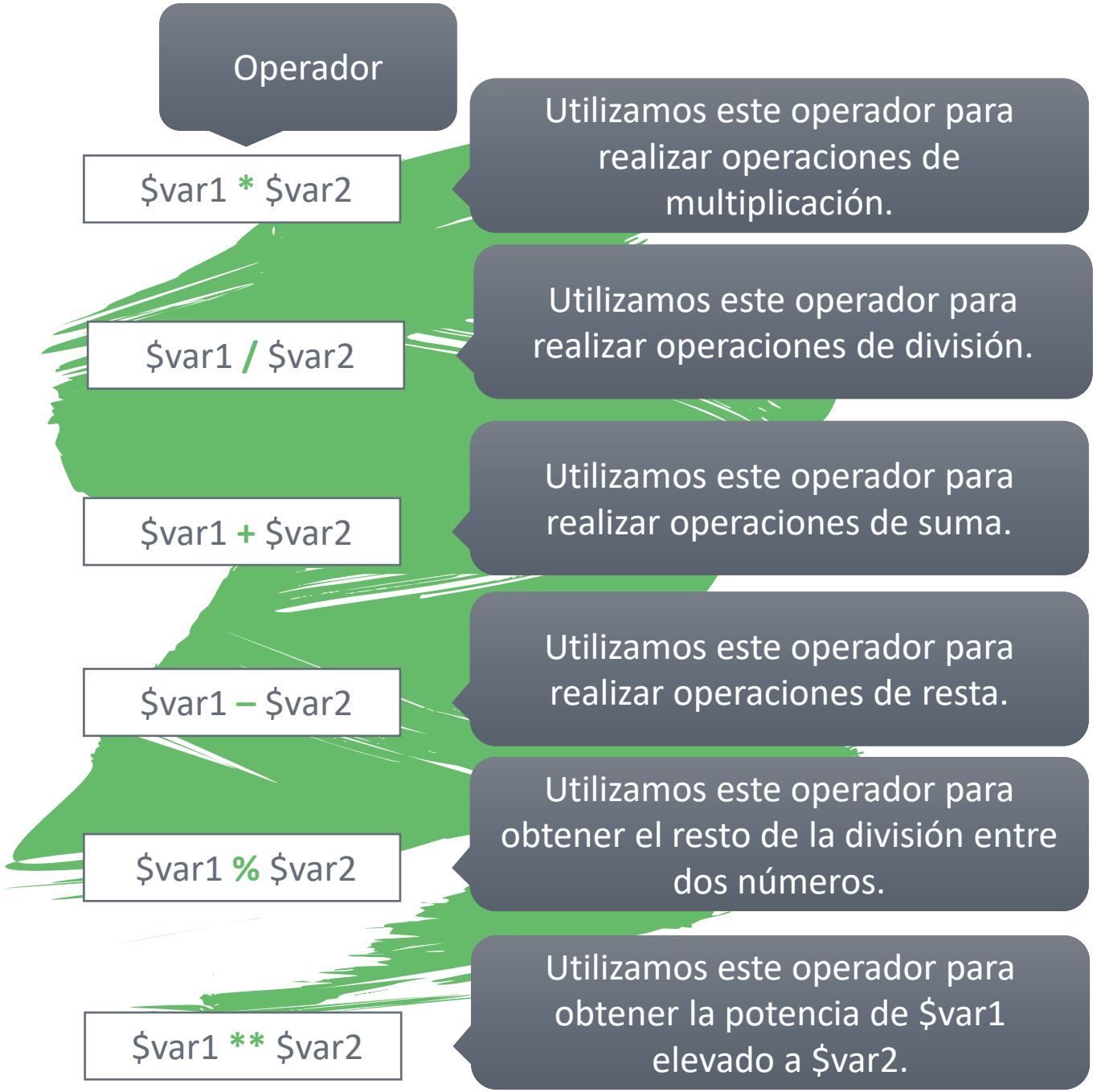
```
const IVA = 0.21;
```

A esta constante le asignamos el valor 0.21

```
echo PI, " ", IVA; // Cuidado, no se pone el símbolo dolar
```

Operadores Aritméticos

En PHP podemos realizar operaciones matemáticas, para ello el lenguaje nos proporciona los siguientes operadores: + (suma), - (resta), * (multiplicación), / (división), ** (potencia) y % (resto de la división).



Ejemplos de uso de operadores aritméticos.

Ejemplo

+ - * / %

Veamos algunos ejemplos de como utilizar los diferentes operadores aritméticos disponibles en java:

Ejemplo de utilización de los diferentes operadores

En c se almacena la suma de a + b.

```
$c = $a + $b;
$b = $c - 2;
$a = $a * 3;
$c = $c / 2;
$b = $b % 2;
$c = $c ** $a;
```

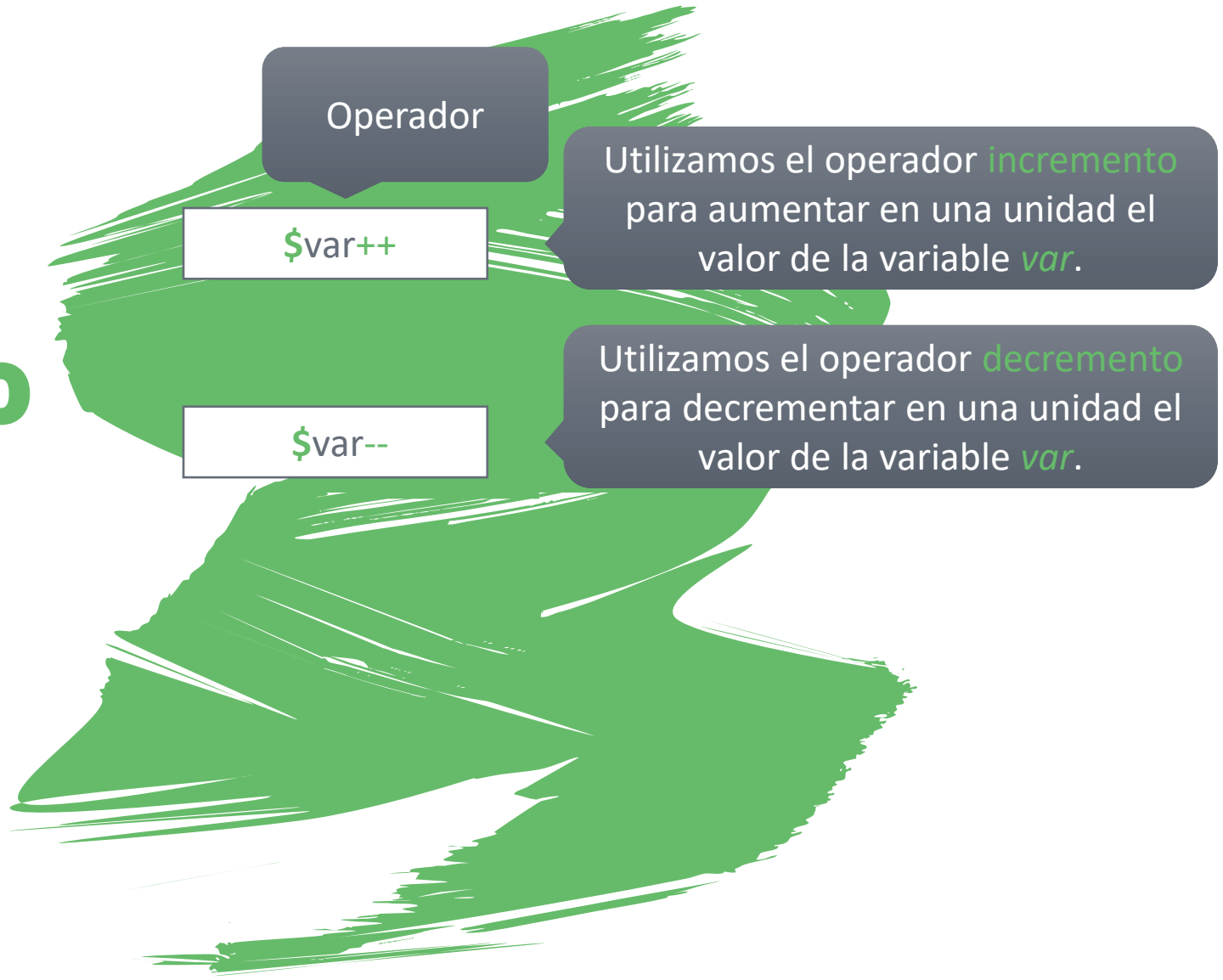
En b se almacena la resta de c - 2.

En c se almacena la división de c entre 2.

En b se almacena el resto de la división de b entre 2.

Operadores
In/Decremento

Además de los operadores aritméticos anteriores en PHP disponemos de dos operadores especiales:
++ Operador Incremento.
-- Operador Decremento.



Ejemplos de uso de operadores incremento y decremento.

Ejemplo

++ --

A continuación, vamos a ver ejemplos de utilización de los operadores de incremento y decremento.

Ejemplo Operador Incremento

```
$i=1;
$i++;
printf("El valor de i es %d", $i);
```

Después de ejecutar la sentencia i++ podemos ver como el valor de la variable i pasa a ser 2.

Ejemplo Operador Decremento

```
$j=5;
$j--;
printf("\nEl valor de j es %d", $j);
```

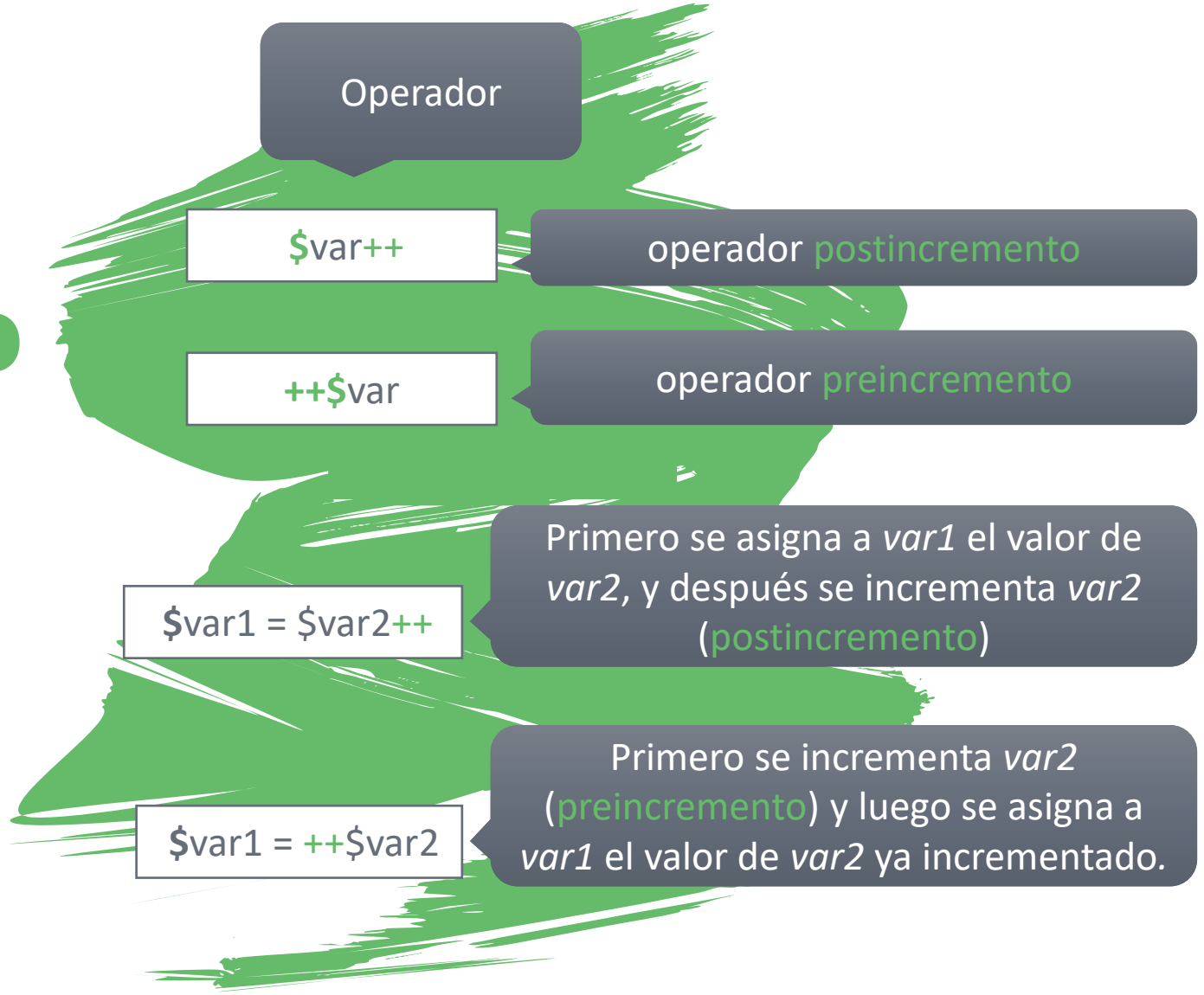
Después de ejecutar la sentencia j-- podemos ver como el valor de la variable j pasa a ser 4.

Operadores

Postincremento

Preincremento

En función de donde se coloquen los operadores podemos hablar de:
var++ Operador PostIncremento.
++var Operador PreIncremento.
En principio utilizados así tienen el mismo efecto. Sin embargo, si utilizamos expresiones como:
$var1 = $var2++;
El resultado es distinto de:
$var1 = ++$var2;



Ejemplos de uso de operadores postincremento y preincremento.

Ejemplo $a++ ++$a

A continuación, vamos a ver ejemplos de utilización de los operadores de postincremento y preincremento.

Ejemplo Operador postincremento

```
$a=2;
$b=3;
$a=$b++;
printf("El valor de a es %d", $a);
printf("El valor de b es %d", $b);
```

Primero se asigna b y luego se incrementa.

Al ejecutar este código a=3 y b=4.

Ejemplo Operador preincremento

```
$a=2;
$b=3;
$a=++$b;
printf("El valor de a es %d", $a);
printf("El valor de b es %d", $b);
```

Primero se incrementa b y luego se asigna.

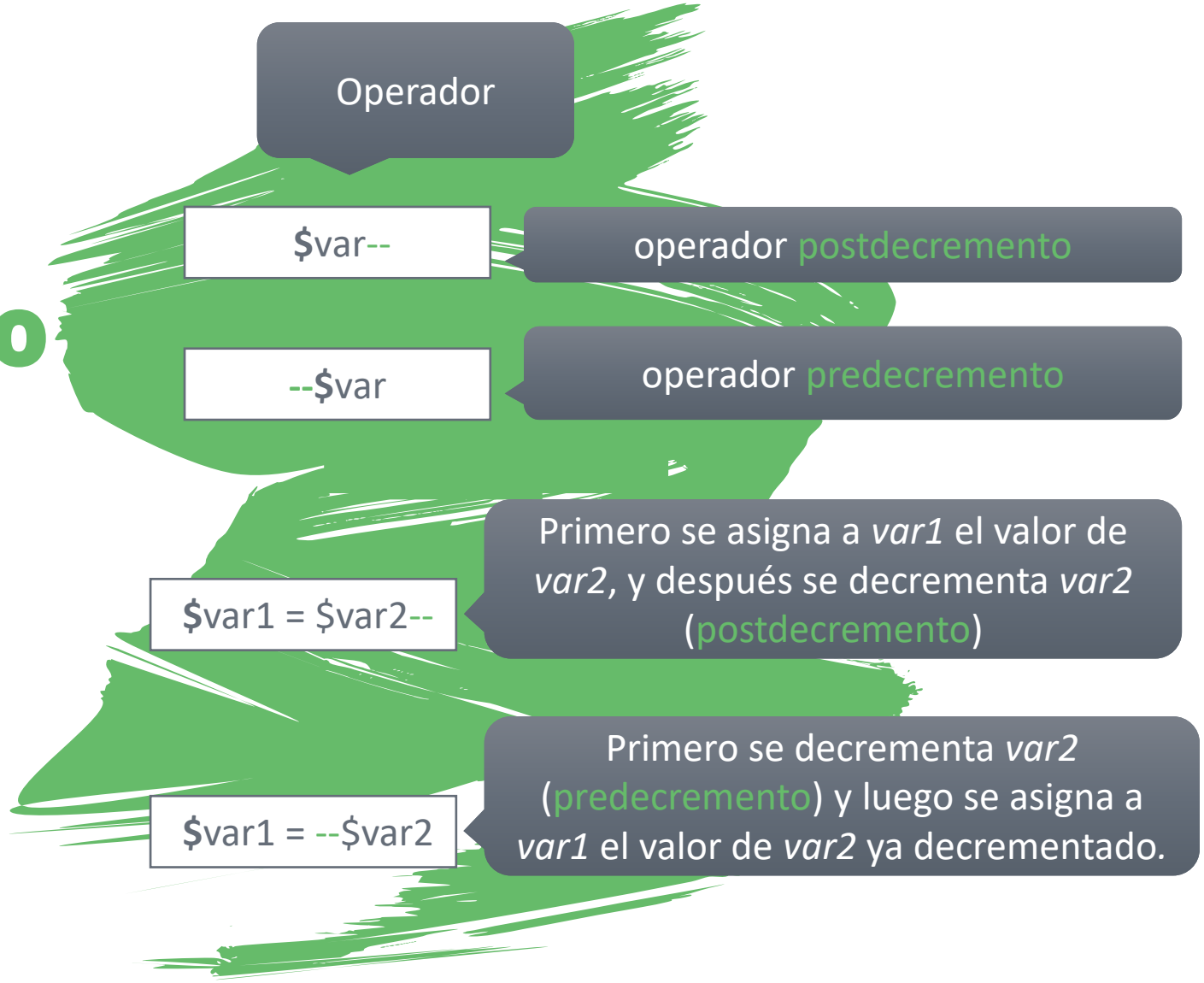
Al ejecutar este código a=4 y b=4.

Operadores

Postdecremento

Predecremento

En función de donde se coloquen los operadores podemos hablar de:
$var-- Operador PostDecremento.
--$var Operador PreDecremento.
En principio utilizados así tienen el mismo efecto. Sin embargo, si utilizamos expresiones como:
$var1 = $var2--;
El resultado es distinto de:
$var1 = --$var2;



Ejemplos de uso de operadores postdecremento y predecremento.

Ejemplo

$a-- --$a

A continuación, vamos a ver ejemplos de utilización de los operadores de postdecremento y predecremento.

Ejemplo Operador postdecremento

```php
$a=2;
$b=3;
$a=$b--;
printf("El valor de a es %d", $a);
printf("El valor de b es %d", $b);
```

Primero se asigna b y luego se decrementa.

Al ejecutar este código a=3 y b=2.

Ejemplo Operador predecremento

```php
$a=2;
$b=3;
$a=--$b;
printf("El valor de a es %d", $a);
printf("El valor de b es %d", $b);
```

Primero se decrementa b y luego se asigna.

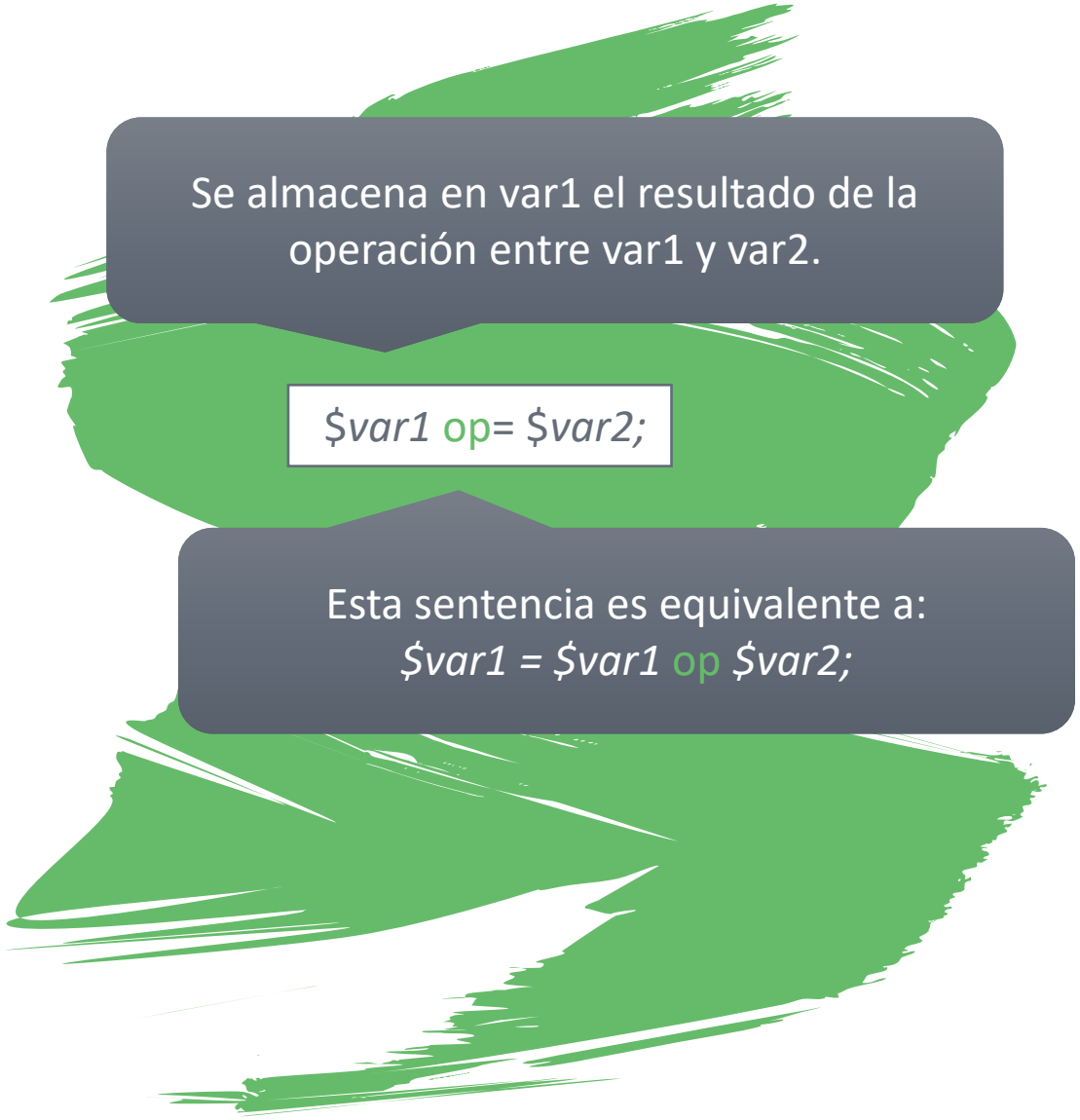
Al ejecutar este código a=2 y b=2.

Operaciones Abreviadas

Operaciones Abreviadas

En ejemplos anteriores hemos visto como se utilizaba la misma variable a un lado y al otro de un operador (p.e. $a=$a*3). Este tipo de operaciones se puede resolver también utilizando una notación abreviada: *$a*=3*.
De este modo ambas operaciones son totalmente equivalentes:

```
$a = $a*3;
$a*=3;
```



Ejemplos de uso de operaciones abreviadas.

Ejemplo Operaciones

A continuación, podemos ver unos ejemplos de la utilización de diferentes operaciones abreviadas:

Ejemplo de utilización de operaciones abreviadas

```php
$a=2;
$b=3;
$c=4;
$d=5;
$b+=$a; #Esta operación equivale a poner $b = $b + $a
$a*=2; #Esta operación equivale a poner $a = $a * 2
$c/=2; #Esta operación equivale a poner $c = $c / 2
$d%=3; #Esta operación equivale a poner $d = $d % 3
```

Intenta averiguar el valor de las variables después de cada operación.