



APRENDE **PHP** GUÍA PRÁCTICA

A C A D E M I A X

Contenido

1	Introducción	4
1.1	Bienvenida	4
1.1.1	Libro vivo	5
1.1.2	Alcance	5
1.2	Prerequisitos	5
1.3	¿Cómo evitar bloqueos?	6
2	Primeros pasos	7
2.1	Visión general	7
2.1.1	¿Qué es y por qué debes aprenderlo?	7
2.1.2	¿En dónde se utiliza?	7
2.1.3	¿Qué trabajos puedes conseguir?	8
2.1.4	¿Cuánto puedes ganar?	8
2.1.5	¿Cuales son las preguntas más comunes?	9
2.2	Historia, evolución, y versiones	9
2.3	Características y ventajas	10
2.4	Diferencias con otros lenguajes de programación	11
2.5	Configuración	12
2.5.1	IDE	12
2.5.2	Entorno	12
2.6	Hola Mundo	14
3	Gramática	15
3.1	Sintaxis	15
3.2	Comentarios	15
4	Tipos y variables	17
4.1	Variables	17
4.2	Listas	19

Contenido

4.3	Objetos	21
4.4	Constantes	23
5	Operadores	24
5.1	Operadores de aritméticos	24
5.2	Operadores comparativos	25
5.3	Operadores lógicos	26
6	Condicionales	27
6.1	Condición única	27
6.2	Múltiples condiciones	28
7	Bucles	30
7.1	Bucle for	30
7.2	Bucle while	31
8	Funciones	32
9	Programación orientada a objetos (POO)	36
9.1	Paradigma	36
9.2	Clases	36
10	Módulos	37
11	Siguientes pasos	39
11.1	Herramientas	39
11.2	Recursos	39
11.3	¿Que viene después?	40
11.4	Preguntas de entrevista	41

1 Introducción

1.1 Bienvenida

Bienvenid@ a esta guía de Academia X en donde aprenderás PHP práctico.

Hola, mi nombre es Xavier Reyes Ochoa y soy el autor de esta guía. He trabajado como consultor en proyectos para Nintendo, Google, entre otros proyectos top-tier, trabajé como líder de un equipo de desarrollo por 3 años, y soy Ingeniero Ex-Amazon. En mis redes me conocen como Programador X y comparto videos semanalmente en YouTube desde diversas locaciones del mundo con el objetivo de guiar y motivar a mis estudiantes mientras trabajo haciendo lo que más me gusta: la programación.

En esta guía vas a aprender estos temas:

- Gramática
- Tipos y variables
- Operadores
- Condicionales
- Bucles
- Funciones
- Programación orientada a objetos (POO)
- Módulos

La motivación de esta guía es darte todo el conocimiento técnico que necesitas para elevar la calidad de tus proyectos y al mismo tiempo puedas perseguir metas más grandes, ya sea utilizar esta tecnología para tus pasatiempos creativos, aumentar tus oportunidades laborales, o si tienes el espíritu emprendedor, incluso crear tu propio negocio en línea. Confío en que esta guía te dará los recursos que necesitas para que tengas éxito en este campo.

Empecemos!

1 INTRODUCCIÓN

1.1.1 Libro vivo

Esta publicación fue planeada, editada, y revisada manualmente por Xavier Reyes Ochoa. La fundación del contenido fue generada parcialmente por inteligencia artificial usando ChatGPT (Mar 14 Version) de OpenAI. Puedes ver más detalles en <https://openai.com/>

Esto es a lo que llamo un trabajo **VIVO**, esto quiere decir que será actualizado en el tiempo a medida que existan cambios en la tecnología. La versión actual es 1.0.0 editada el 23 de marzo de 2023. Si tienes correcciones importantes, puedes escribirnos a nuestra sección de contacto en <https://www.academia-x.com>

1.1.2 Alcance

El objetivo de esta guía es llenar el vacío que existe sobre esta tecnología en Español siguiendo el siguiente enfoque:

- Se revizan los temas con un enfoque práctico incluyendo ejemplos y retos.
- Se evita incluir material de relleno ya que no es eficiente.
- Se evita entrar en detalle en temas simples o avanzados no-prácticos.

Si deseas profundizar en algún tema, te dejo varios recursos populares y avanzados en la lección de recursos como el estándar de esta tecnología (que puede ser difícil de leer si recién estás empezando).

1.2 Prerequisitos

Antes de aprender PHP, necesitas lo siguiente:

1. Un computador: cualquier computador moderno tiene las capacidades de correr este lenguaje. Te recomiendo un monitor de escritorio o laptop ya que dispositivos móviles o ipads no son cómodos para programar.

1 INTRODUCCIÓN

2. Sistema operativo: conocimiento básico de cómo utilizar el sistema operativo en el que se ejecutará PHP (por ejemplo, Windows, MacOS, Linux). Te recomiendo tener el sistema operativo actualizado.
3. Conocimiento básico de la línea de comandos: se utiliza para ejecutar programas en PHP.
4. Un editor de texto: lo necesitas para escribir código de PHP. Los editores de texto más populares son Visual Studio Code, Sublime Text, Atom y Notepad ++.
5. Un navegador web y conexión al internet: es útil para investigar más sobre esta tecnología cuando tengas dudas. Los navegadores más populares son Google Chrome, Mozilla Firefox, Safari y Microsoft Edge. Se recomienda tener el navegador actualizado.

Si ya tienes estos requisitos, estarás en una buena posición para comenzar a aprender PHP y profundizar en sus características y aplicaciones.

Si todavía no tienes conocimiento sobre algunos de estos temas, te recomiendo buscar tutoriales básicos en blogs a través de Google, ver videos en YouTube, o hacer preguntas a ChatGPT. Alternativamente, puedes empezar ya e investigar en línea a medida que encuentres bloqueos enteniendo los conceptos en esta guía.

1.3 ¿Cómo evitar bloqueos?

Para sacarle el mayor provecho a esta guía:

1. No solo leas esta guía. La práctica es esencial en este campo. Practica todos los días y no pases de lección hasta que un concepto esté 100% claro.
2. No tienes que memorizarlo todo, solo tienes que saber donde están los temas para buscarlos rápidamente cuando tengas dudas.
3. Cuando tengas preguntas usa [Google](#), [StackOverflow](#), y [ChatGPT](#)
4. Acepta que en esta carrera, mucho de tu tiempo lo vas utilizar investigando e innovando, no solo escribiendo código.

5. No tienes que aprender inglés ahora pero considera aprenderlo en un futuro porque los recursos más actualizados están en inglés y también te dará mejores oportunidades laborales.
6. Si pierdas la motivación, recuerda tus objetivos. Ninguna carrera es fácil pero ya tienes los recursos para llegar muy lejos. Te deseo lo mejor en este campo!

2 Primeros pasos

2.1 Visión general

2.1.1 ¿Qué es y por qué debes aprenderlo?

PHP es un lenguaje de programación de código abierto muy popular, especialmente para el desarrollo web. Está diseñado para permitir a los desarrolladores crear sitios web dinámicos y aplicaciones web.

Debes aprender PHP porque es un lenguaje de programación de uso general, lo que significa que puedes usarlo para crear cualquier tipo de aplicación web. Además, es un lenguaje de programación muy popular, lo que significa que hay una gran cantidad de recursos disponibles para aprenderlo. Finalmente, es un lenguaje de programación fácil de aprender, lo que significa que incluso si eres un principiante, puedes aprenderlo rápidamente.

2.1.2 ¿En dónde se utiliza?

PHP es un lenguaje de programación de código abierto muy popular para el desarrollo web. Se utiliza principalmente para crear aplicaciones web dinámicas y sitios web interactivos. Se puede utilizar para crear contenido web, administrar bases de datos y construir aplicaciones web complejas. También se puede utilizar para crear aplicaciones de escritorio, aplicaciones móviles y aplicaciones de servidor.

2.1.3 ¿Qué trabajos puedes conseguir?

- Desarrollador web
- Programador de aplicaciones web
- Desarrollador de sitios web
- Desarrollador de aplicaciones web
- Desarrollador de bases de datos
- Desarrollador de contenido web
- Desarrollador de sistemas web
- Desarrollador de aplicaciones móviles
- Desarrollador de aplicaciones de escritorio
- Desarrollador de aplicaciones de comercio electrónico
- Desarrollador de aplicaciones de redes sociales
- Analista de sistemas web
- Diseñador web
- Administrador de sistemas web
- Administrador de bases de datos
- Diseñador de interfaces de usuario
- Desarrollador de aplicaciones de software

2.1.4 ¿Cuánto puedes ganar?

El salario que puedes ganar usando PHP depende de muchos factores, como tu experiencia, el lugar donde trabajes y el tipo de trabajo que estés realizando. Según Glassdoor, el salario promedio para un desarrollador de PHP en los Estados Unidos es de \$86,846 anuales.

Es importante tener en cuenta que estos son solo promedios y que el salario real que puedes ganar puede ser mayor o menor, dependiendo de los factores mencionados anteriormente. Además, siempre es una buena idea investigar y hacer preguntas sobre los salarios y las condiciones laborales antes de aceptar un trabajo.

2.1.5 ¿Cuales son las preguntas más comunes?

1. ¿Qué es PHP?
2. ¿Cómo se instala PHP?
3. ¿Cómo se usa PHP para crear una página web?
4. ¿Cómo se conecta PHP a una base de datos?
5. ¿Cómo se usa PHP para enviar un correo electrónico?
6. ¿Cómo se usa PHP para cargar archivos en un servidor?
7. ¿Cómo se usa PHP para crear una sesión?
8. ¿Cómo se usa PHP para validar un formulario?
9. ¿Cómo se usa PHP para proteger una página web?
10. ¿Cómo se usa PHP para crear una cookie?

Al finalizar este recurso, tendrás las habilidades necesarias para responder o encontrar las respuestas a estas preguntas.

2.2 Historia, evolución, y versiones

PHP es un lenguaje de programación de código abierto muy popular, especialmente para el desarrollo web. Fue creado originalmente por el desarrollador de software Rasmus Lerdorf en 1995. Desde entonces, ha pasado por varias versiones y ha evolucionado hasta convertirse en uno de los lenguajes de programación más populares del mundo.

Las primeras versiones de PHP eran principalmente un conjunto de herramientas para el desarrollo web, como una herramienta para la creación de formularios web. Con el tiempo, se han añadido nuevas características, como soporte para bases de datos, soporte para lenguajes de programación orientados a objetos, y soporte para el desarrollo de aplicaciones web.

Actualmente, la última versión estable de PHP es la versión 7.4. Esta versión incluye mejoras en el rendimiento, soporte para nuevas características, como la sintaxis de

tipo de datos, y mejoras en la seguridad.

Además de la versión estable, hay también versiones de desarrollo, como la versión 8.0, que se está desarrollando actualmente. Esta versión incluye mejoras en el rendimiento, soporte para nuevas características, como la sintaxis de tipo de datos, y mejoras en la seguridad. Esta versión también incluye mejoras en la interoperabilidad con otros lenguajes de programación.

2.3 Características y ventajas

PHP es un lenguaje de programación de código abierto muy popular para el desarrollo de sitios web dinámicos. Está diseñado para ser fácil de usar y es compatible con la mayoría de los servidores web. Esto significa que los desarrolladores pueden crear sitios web dinámicos sin tener que aprender un lenguaje de programación completamente nuevo.

Algunas de las principales características de PHP incluyen:

- Soporte para bases de datos: PHP es compatible con una amplia gama de bases de datos, incluyendo MySQL, Oracle, PostgreSQL y Microsoft SQL Server. Esto significa que los desarrolladores pueden conectar sus sitios web a una base de datos y almacenar y recuperar datos de ella.
- Facilidad de uso: PHP es un lenguaje de programación fácil de aprender y usar. Esto significa que los desarrolladores no necesitan tener una gran cantidad de conocimientos técnicos para crear sitios web dinámicos.
- Seguridad: PHP es un lenguaje de programación seguro. Esto significa que los desarrolladores pueden crear sitios web seguros que protegen la información de los usuarios.
- Rendimiento: PHP es un lenguaje de programación rápido. Esto significa que los sitios web creados con PHP se cargan y ejecutan más rápido que los sitios web creados con otros lenguajes de programación.

- **Flexibilidad:** PHP es un lenguaje de programación flexible. Esto significa que los desarrolladores pueden crear sitios web que se pueden personalizar fácilmente para satisfacer las necesidades de los usuarios.

2.4 Diferencias con otros lenguajes de programación

PHP es un lenguaje de programación de alto nivel, interpretado y de código abierto. Está diseñado para el desarrollo web y es uno de los lenguajes más populares para el desarrollo de aplicaciones web.

Algunas de las principales diferencias entre PHP y otros lenguajes de programación son:

- PHP es un lenguaje de programación de alto nivel, mientras que otros lenguajes de programación como C, C++ y Java son lenguajes de programación de bajo nivel.
- PHP es un lenguaje interpretado, mientras que otros lenguajes como C, C++ y Java son lenguajes compilados.
- PHP es un lenguaje de programación orientado a objetos, mientras que otros lenguajes como C, C++ y Java son lenguajes de programación estructurados.
- PHP es un lenguaje de programación de código abierto, mientras que otros lenguajes como C, C++ y Java son lenguajes de programación de código cerrado.
- PHP es un lenguaje de programación diseñado para el desarrollo web, mientras que otros lenguajes como C, C++ y Java son lenguajes de programación diseñados para el desarrollo de aplicaciones de escritorio.

2.5 Configuración

2.5.1 IDE

Los archivos de PHP son archivos de texto. Puedes editarlos con **editores de texto** como Notepad en Windows o Notes en MacOS pero es recomendado utilizar un **IDE** (Integrated Development Environment) que es una aplicación de edición de código más avanzado que le da colores a tu código para que sea más fácil de leer y tengas funciones de autocompletado, entre otras. Algunos IDEs populares son [Brackets](#), [Atom](#), [Sublime Text](#), [Vim](#), y [Visual Studio Code](#).

El editor recomendado para practicar el código que vamos a ver es Visual Studio Code (o VSCode) que puedes bajar desde <https://code.visualstudio.com/>

2.5.2 Entorno

Existen varias maneras de instalar PHP pero en general se requiere:

1. **Instalar un servidor web:** Primero, necesitas instalar un servidor web para ejecutar archivos PHP. Puedes usar Apache, Nginx o cualquier otro servidor web.
2. **Instalar PHP:** Después, necesitas instalar PHP.
3. **Configurar el servidor web:** Una vez que hayas instalado el servidor web y PHP, necesitas configurar el servidor web para que pueda ejecutar archivos PHP.
4. **Visualizar el el hipertexto preprocesado por PHP:** Necesitas un navegador para visualizar el hipertexto preprocesado por PHP.

Estos pasos los puedes seguir de forma individual manualmente pero puede ser complejo y varían mucho dependiendo del sistema operativo que estés usando. Aquí utilizaremos la forma más simple que es simplemente descargar XAMPP. XAMPP

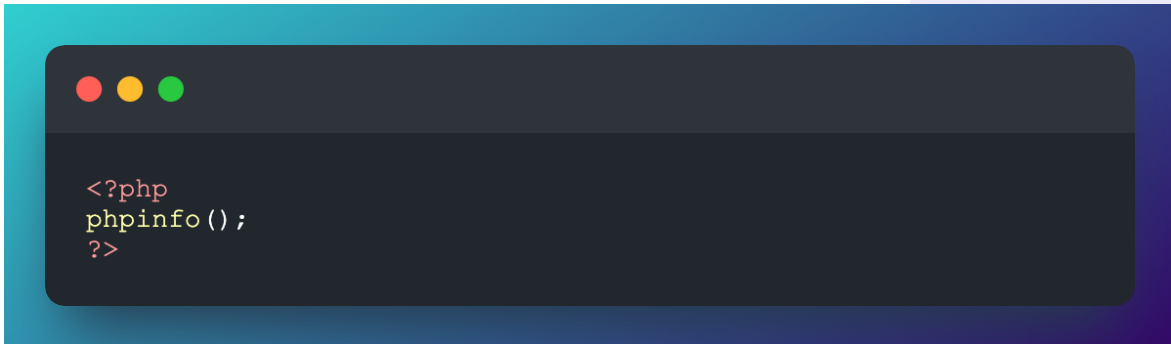
2 PRIMEROS PASOS

es un ambiente de desarrollo que viene con Apache, PHP, y otras herramientas adicionales para desarrollo web como la base de datos MariaDB, y el lenguaje Perl (que los puedes instalar pero están fuera del alcance de esta publicación).

- **Instalar XAMPP:** Para instalar XAMPP puede visitar <https://www.apachefriends.org/> y descargar el instalador para tu sistema operativo. Simplemente sigue los pasos y asegúrate de escoger la instalación de Apache y PHP si son presentadas como opciones.
- **Iniciar servidor:** Una vez instalado, tendrás acceso a la aplicación de XAMPP en tu sistema operativo. Puedes abrir la aplicación de XAMPP, ir al tab de 'Manage Servers' (Administrar servidores), e iniciar el servidor de Apache seleccionándolo y presionando 'Start' (Iniciar).
- **Verificar servidor:** Después de iniciar el servidor, puedes verificar que el servidor esté funcionando yendo a la dirección <http://localhost> en tu navegador. Deberías ver la página de bienvenida de XAMPP.

Desde XAMPP puedes navegar a la carpeta donde están todos los archivos que necesitas. Debes ir al tab de 'Welcome' (Bienvenida) y presionar el botón 'Open Application Folder' (Abrir carpeta de aplicación). Una vez abierta esta carpeta, puedes navegar a la carpeta 'htdocs' en donde estarán todos los archivos de tu proyecto que se renderizan como página web en el navegador. Esta es la carpeta raíz de tu servidor web.

- **Verificar PHP:** Una vez que hayas configurado el servidor web, puedes verificar la instalación de PHP creando un archivo de prueba. Crea un archivo llamado 'test.php' en la carpeta raíz de tu servidor web y agrega el siguiente código:



Ahora, abre tu navegador y visita la dirección <http://localhost/test.php>. Si todo está bien, verás una página con información sobre la configuración de PHP.

- **Ejecutar archivos PHP:** Una vez que hayas verificado la instalación, puedes empezar a ejecutar archivos PHP. Simplemente crea un archivo PHP en la carpeta raíz de tu servidor web y abre la dirección http://localhost/nombre_de_carpeta/nombre_del_archivo.php en tu navegador para ejecutar el archivo.

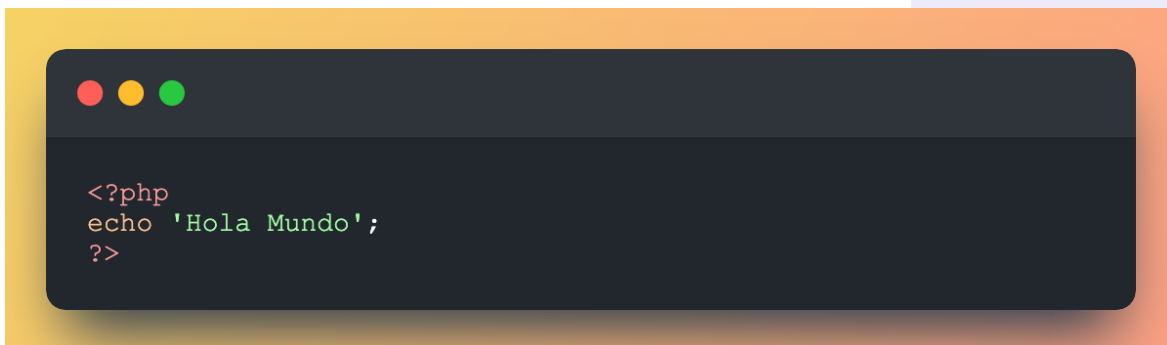
2.6 Hola Mundo

“Hola Mundo” es un ejemplo clásico que se utiliza para mostrar el funcionamiento básico de un lenguaje de programación.

Ejemplo:

En este ejemplo, se imprime el texto “Hola Mundo” en la consola.

3 GRAMÁTICA



También podrías modificar este código con tu nombre. Si es “Juan”, debería imprimir en la consola “Hola, Juan” .

Reto:

Modifica el ejemplo anterior para imprimir “Hola Universo” en la consola.

3 Gramática

3.1 Sintaxis

PHP es un lenguaje de programación de alto nivel con una sintaxis sencilla y fácil de aprender. Utiliza palabras clave específicas para definir estructuras de control, variables, funciones y otros elementos. La indentación es importante para mantener una buena estructura y legibilidad del código. El conjunto de caracteres por defecto es UTF-8, y la sensibilidad a mayúsculas y minúsculas es importante para la correcta interpretación de los nombres de variables, funciones y otros elementos.

3.2 Comentarios

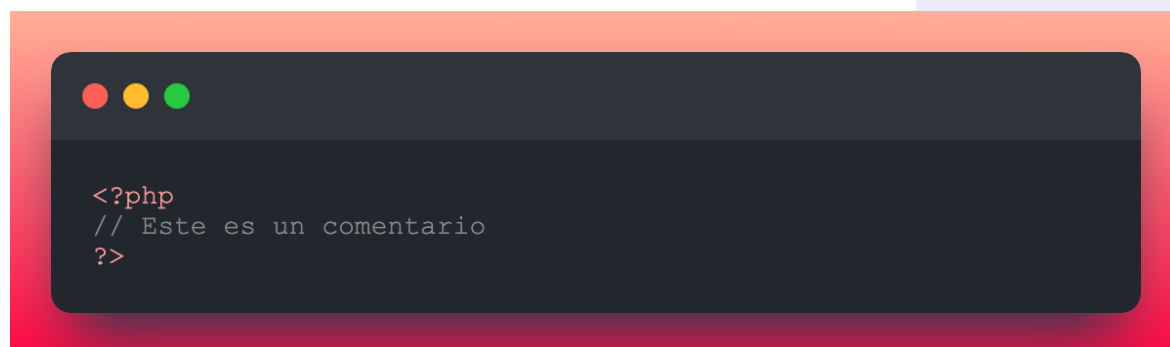
Los comentarios en PHP son líneas de texto que no son interpretadas como parte del código. Se usan para proporcionar información adicional sobre el código, como

3 GRAMÁTICA

explicaciones, notas, o instrucciones para el desarrollador. Los comentarios también se pueden usar para deshabilitar código temporalmente, sin tener que eliminarlo completamente.

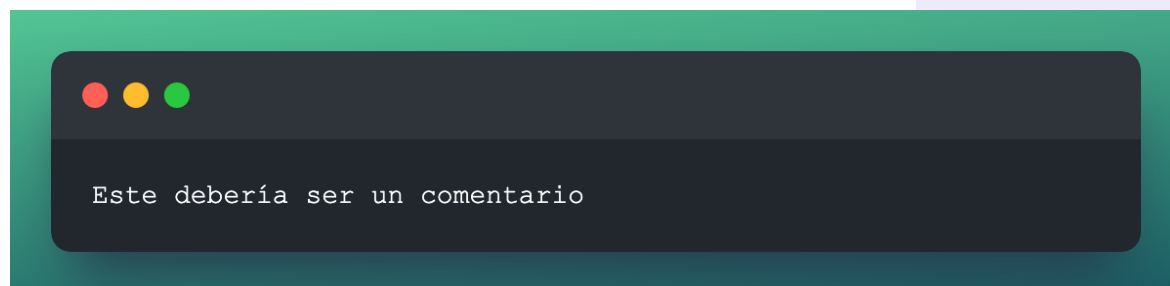
Ejemplo:

Este código es un comentario.



Reto:

Comenta esta línea para que no cause errores y se lea como comentario:



4 Tipos y variables

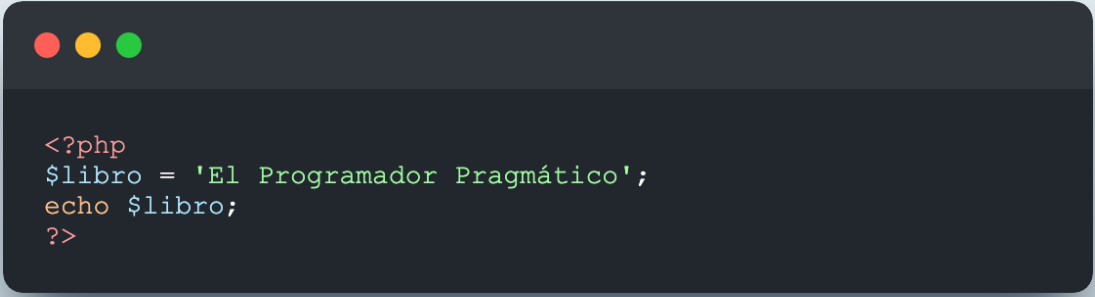
4.1 Variables

La manipulación de datos es una tarea fundamental de un lenguaje de programación. Para trabajar con datos necesitamos guardarlos en variables.

Una variable es un contenedor para almacenar datos que retiene su nombre y puede cambiar su valor a lo largo del tiempo. En los siguientes ejemplos vamos a ver varios tipos de datos que puedes guardar en variables.

Ejemplo:

Este código declara una variable llamada “libro” y le asigna el valor de una cadena de texto que contiene el título de un libro. Luego, imprime el valor de la variable “libro” en la consola.



```
<?php
$libro = 'El Programador Pragmático';
echo $libro;
?>
```

Texto es un tipo de dato útil para guardar información como números telefónicos y colores, entre otros. Este código asigna dos variables, una llamada telf que contiene un número de teléfono como una cadena de caracteres, y otra llamada color que contiene el color amarillo como una cadena de caracteres.

4 TIPOS Y VARIABLES

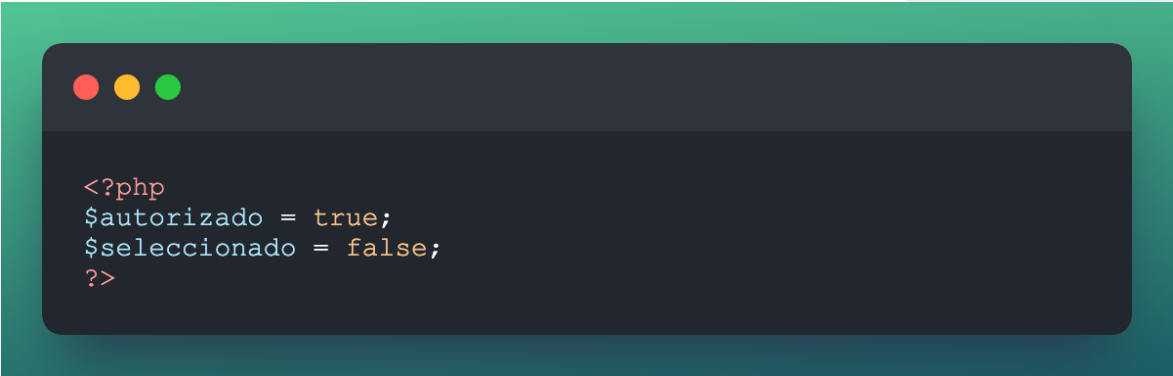
```
<?php
$teléfono = '406-234 2342';
$color = 'amarillo';
?>
```

También podemos guardar datos como números enteros y decimales. Estos datos se usan para realizar operaciones matemáticas y representar valores de peso, dinero, entre otros. Este código asigna dos variables, una con un valor entero (100) y otra con un valor decimal (1.967857).

```
<?php
$entero = 100;
$decimal = 1.967857;
?>
```

El tipo de dato booleano representa los valores de verdadero y falso. Este tipo de datos es útil, por ejemplo, para indicar si un usuario está autorizado a acceder a una app o no, entre varios usos. Este código crea una variable llamada “autorizado” que es verdadera y otra variable llamada “seleccionado” que es falsa.

4 TIPOS Y VARIABLES



```
<?php
$autorizado = true;
$seleccionado = false;
?>
```

Es importante saber que, en el mundo del código binario, el número 1 representa verdadero y 0 representa falso.

Reto:

Crea una variable “a” con 33 como texto e imprímela a la consola.

4.2 Listas

Las listas en PHP son una estructura de datos que nos permite almacenar una colección ordenada de elementos. Estos elementos pueden ser de cualquier tipo, desde números hasta sublistas. También los vas a encontrar con otros nombres como arreglos, matrices, arrays, etc.

Ejemplo:

Este código está imprimiendo el primer elemento de una matriz de números a la consola. En este caso, el primer elemento es 23.

4 TIPOS Y VARIABLES

```
<?php
$numeros = array(23, 45, 16, 37, 3, 99, 22);
echo $numeros[0]; // 23
?>
```

También existen listas de texto. Este código crea una variable llamada “animales” que contiene una lista de elementos, en este caso, tres animales: perro, gato y tigre.

```
<?php
$animales = array('perro', 'gato', 'tigre');
?>
```

Y esta es una lista de datos mixtos. Este código crea una variable llamada “datosMixtos” que contiene una lista de elementos de diferentes tipos, como texto, números, booleanos y otra lista.



```
<?php
$datosMixtos = array('texto', 69, true, array('lista dentro
de otra lista'));
?>
```

Reto:

Crea una lista en PHP que contenga los nombres de los meses del año. Accede al 3er índice e imprímelo en la consola.

4.3 Objetos

Los objetos en PHP son una forma de almacenar y organizar datos mapeándolos de uno a uno. Estos datos se almacenan en forma de pares clave-valor, donde la clave suele ser una cadena de texto y el valor puede ser cualquier tipo de dato. También los vas a encontrar con otros nombres como mapas, diccionarios, etc.

Ejemplo:

Este código crea un objeto llamado jugadores que contiene dos pares clave-valor. El primer par clave-valor es 10: 'Messi' y el segundo es 7: 'Cristiano Ronaldo' . Luego, se imprime el valor asociado con la clave 10, que es 'Messi' .

4 TIPOS Y VARIABLES

```
<?php
$jugadores = array(
    10 => 'Messi',
    7  => 'Cristiano Ronaldo'
);
echo $jugadores[10]; // 'Messi'
?>
```

También podemos mapear de texto a texto. Este código crea un objeto llamado “países” que contiene claves y valores. Las claves son códigos de países (EC, MX, AR) y los valores son los nombres de los países (Ecuador, México, Argentina).

```
<?php
$países = array(
    "EC" => "Ecuador",
    "MX" => "México",
    "AR" => "Argentina"
);
?>
```

E incluso podemos mapear de texto a listas de texto, entre muchas otras opciones. Este código establece un objeto llamado “emails” que contiene dos pares clave-valor. Cada clave es un nombre de persona y el valor es un arreglo de direcciones de correo electrónico asociadas con esa persona.



```
<?php
$emails = array(
    'Juan' => array('juan@gmail.com'),
    'Ricardo' => array('ricardo@gmail.com', 'rick@aol.com')
);
?>
```

Reto:

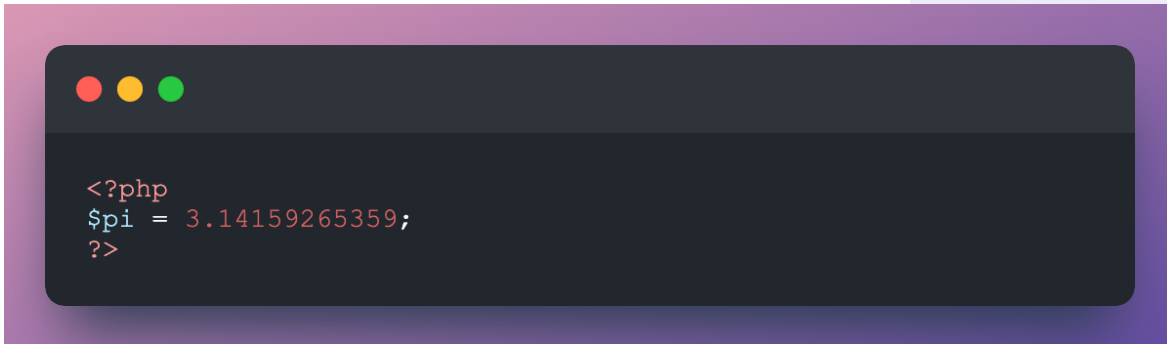
Crea un objeto en PHP que represente una computadora, con sus respectivas características (marca, modelo, procesador, memoria RAM, etc).

4.4 Constantes

Las constantes son variables que no pueden ser reasignadas. Esto significa que una vez que se asigna un valor a una constante, este no puede ser cambiado.

Ejemplo:

Esta línea de código establece una constante llamada “pi” con un valor de 3.14159265359. Esta constante se puede usar para almacenar un valor numérico que no cambiará a lo largo del programa.



Reto:

Crea una constante llamada 'SALUDO' y asígnale el valor 'Hola Planeta' . Luego, imprime el valor de la constante en la consola.

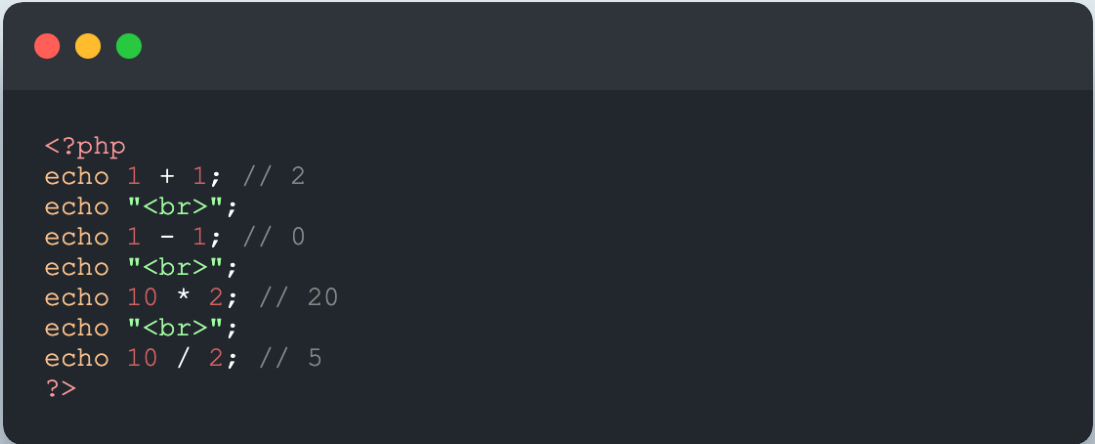
5 Operadores

5.1 Operadores de aritméticos

Los operadores aritméticos en PHP son usados para realizar operaciones matemáticas básicas. Estos operadores incluyen sumar, restar, multiplicar, dividir, entre otros.

Ejemplo:

Este código realiza operaciones matemáticas básicas, imprimiendo los resultados en la consola. Estas operaciones incluyen suma, resta, multiplicación y división.



```
<?php
echo 1 + 1; // 2
echo "<br>";
echo 1 - 1; // 0
echo "<br>";
echo 10 * 2; // 20
echo "<br>";
echo 10 / 2; // 5
?>
```

Reto:

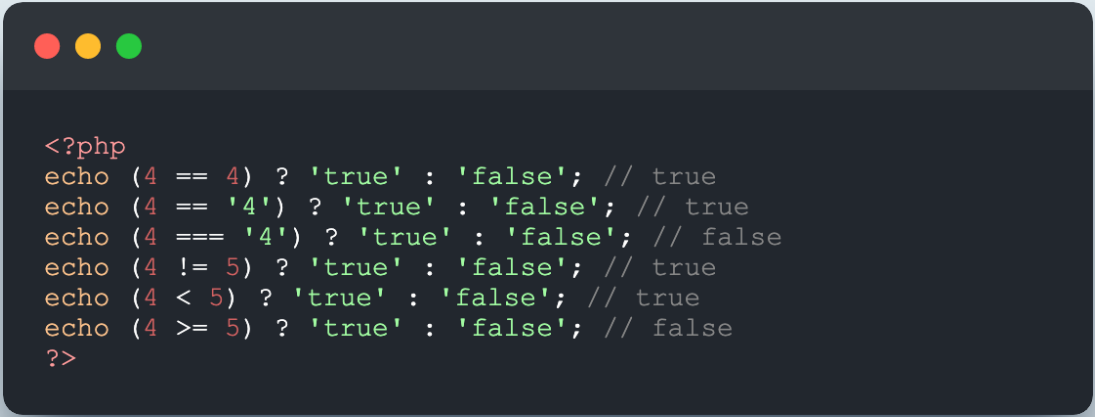
Usa los operadores aritméticos para calcular el área de un círculo con radio 5.

5.2 Operadores comparativos

Los operadores comparativos en PHP son usados para comparar dos valores y determinar si son iguales o diferentes. Estos operadores son útiles para determinar si dos valores son iguales, si dos valores no son iguales, si un valor es mayor o menor que otro, entre otros.

Ejemplo:

Este código muestra cómo se usan los operadores para comparar valores.



```
<?php
echo (4 == 4) ? 'true' : 'false'; // true
echo (4 == '4') ? 'true' : 'false'; // true
echo (4 === '4') ? 'true' : 'false'; // false
echo (4 != 5) ? 'true' : 'false'; // true
echo (4 < 5) ? 'true' : 'false'; // true
echo (4 >= 5) ? 'true' : 'false'; // false
?>
```

Reto:

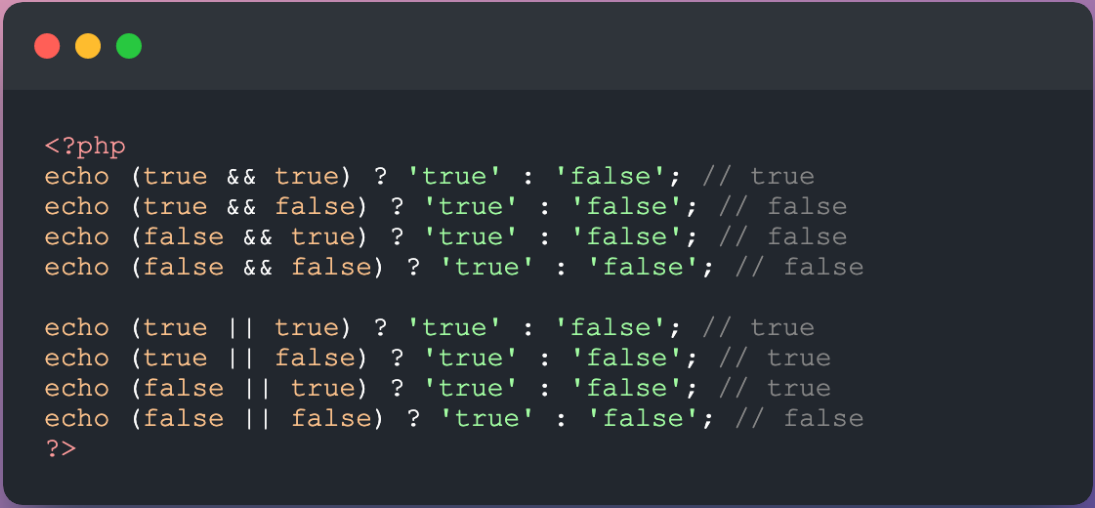
Escribe un programa que compare dos números 'a' y 'b' y determina si 'a' con el valor de 4 es mayor, menor o igual a 'b' con un valor de 2.

5.3 Operadores lógicos

Los operadores lógicos en PHP se usan para realizar comparaciones entre valores y devolver un valor booleano (verdadero o falso). Estos operadores pueden ser útiles, por ejemplo, si deseamos saber si un 'animal' es un gato y es un mamífero (al mismo tiempo).

Ejemplo:

Este código muestra el uso de los operadores lógicos 'y' y 'o' para evaluar expresiones booleanas. El operador 'y' devuelve verdadero si ambas expresiones son verdaderas, de lo contrario devuelve falso. El operador 'o' devuelve verdadero si al menos una de las expresiones es verdadera, de lo contrario devuelve falso.



```
<?php
echo (true && true) ? 'true' : 'false'; // true
echo (true && false) ? 'true' : 'false'; // false
echo (false && true) ? 'true' : 'false'; // false
echo (false && false) ? 'true' : 'false'; // false

echo (true || true) ? 'true' : 'false'; // true
echo (true || false) ? 'true' : 'false'; // true
echo (false || true) ? 'true' : 'false'; // true
echo (false || false) ? 'true' : 'false'; // false
?>
```

Reto:

Escribe una línea de código que devuelva verdadero si 'x' es mayor que 0 y 'y' es menor que 0.


6 Condicionales

6.1 Condición única

Los condicionales son estructura de control de flujo en PHP, es decir, controlan el flujo del código. Permiten ejecutar una sección de código si una condición es verdadera. También permiten ejecutar otra sección de código, si la condición es falsa.

Ejemplo:

Este código comprueba si una variable booleana (autorizado) es verdadera. Si es así, imprime un mensaje en la consola indicando que el usuario puede ingresar. Si no es así, imprime un mensaje indicando que el usuario no puede ingresar.



```
<?php
$autorizado = true;

if ($autorizado) {
    echo 'Puede ingresar'; // ✓
} else {
    echo 'No puede ingresar';
}
?>
```

Reto:


Escribe código condicional que revise si un número 'a' es par o impar. La variable 'a' tiene el valor de 17 y debe imprimir a la consola 'Es par' si es par o 'Es impar' si es impar.

6.2 Múltiples condiciones

Adicionalmente, los condicionales permiten ejecutar varias secciones de código de acuerdo a varias condiciones.

Ejemplo:


En este caso podemos ver que el código que se ejecuta depende de varios valores. Este código comprueba si una variable llamada entero es igual a 99 o 100. Si es igual a 99, imprime "Es 99" en la consola. Si es igual a 100, imprime "Es 100" en la consola. Si no es ni 99 ni 100, imprime "No 99 ni 100" en la consola.



```
<?php
$entero = 100;

if ($entero === 99) {
    echo 'Es 99';
} else if ($entero === 100) {
    echo 'Es 100'; // ✓
} else {
    echo 'No 99 ni 100';
}
?>
```

Este condicional es útil cuando hay una gran cantidad de posibles resultados para una expresión. Este código compara una variable (color) con diferentes valores y ejecuta una acción dependiendo del resultado de la comparación. En este caso, si la variable color es igual a 'amarillo', se imprimirá 'Advertencia' en la consola.



```
<?php
$color = 'amarillo';

switch ($color) {
    case 'verde':
        echo 'Éxito';
        break;
    case 'amarillo':
        echo 'Advertencia'; // ✓
        break;
    default:
        echo 'Error';
        break;
}
?>
```

Reto:

Crea un programa que evalúe una variable ‘numero’ y dependiendo del resultado, imprima un mensaje diferente. Si el número es mayor a 10, imprime “El número es mayor a 10” . Si el número es menor a 10, imprime “El número es menor a 10” . Si el número es igual a 10, imprime “El número es igual a 10” .

7 Bucles


7.1 Bucle for

Los bucles son otra estructura de control de flujo que se utilizan para iterar sobre una secuencia de elementos. También se los llama loops o ciclos.

Ejemplo:

7 BUCLES

Este código itera sobre una lista de animales e imprime cada elemento de la lista en la consola.



```
<?php
$animales = array('perro', 'gato', 'tigre');

foreach ($animales as $animal) {
    echo $animal . "\n";
}

// 'perro'
// 'gato'
// 'tigre'
?>
```

Reto:

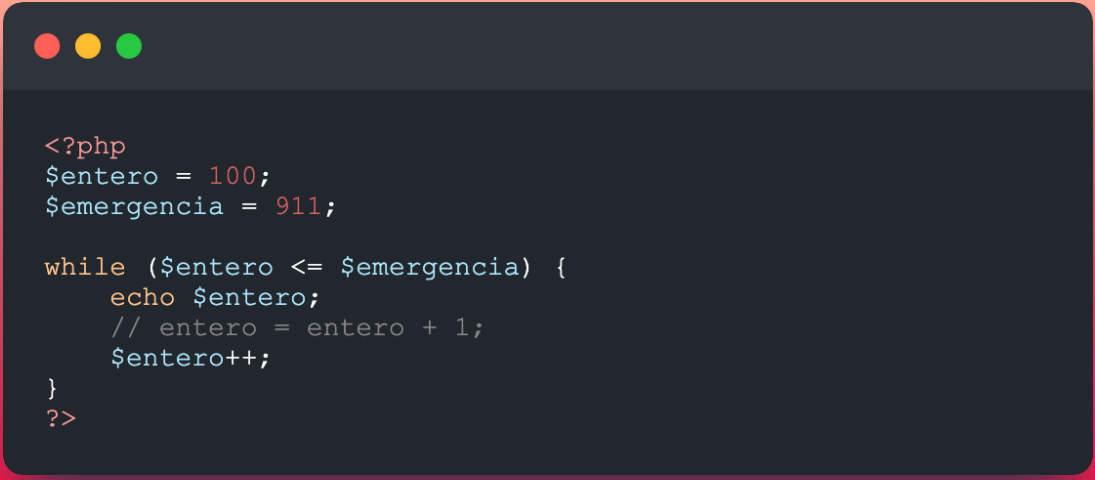
Escribe un bucle que imprima los números del 1 al 10 en orden ascendente.

7.2 Bucle while

while es un tipo de bucle en PHP que se usa para ejecutar una serie de instrucciones mientras se cumpla una condición. Esta condición se evalúa antes de cada iteración del bucle.

Ejemplo:

Este código establece un bucle while que imprime los números enteros desde 100 hasta 911 en la consola. El bucle while se ejecutará mientras la variable entero sea menor o igual a la variable emergencia. Cada vez que el bucle se ejecuta, la variable entero se incrementa en 1.



```
<?php
$entero = 100;
$emergencia = 911;

while ($entero <= $emergencia) {
    echo $entero;
    // entero = entero + 1;
    $entero++;
}
?>
```

Ten cuidado de no incluir una condición para parar el bucle. Esto podría seguir corriendo tu código indefinidamente hasta congelar tu computador.

Reto:

Crea un bucle while que imprima los números del 1 al 10 en la consola.

8 Funciones

En PHP, una función es un bloque de código diseñado para realizar una tarea específica y se puede reutilizar en varias partes el código. Las funciones se pueden definir para realizar cualquier tarea, desde realizar cálculos hasta mostrar mensajes en la pantalla.

Ejemplo:

Esta función toma dos argumentos (primero y segundo) de forma dinámica y los suma. Luego, imprime el resultado en la consola. Esta función se llama dos veces con diferentes argumentos para mostrar los resultados de la suma.

8 FUNCIONES

```
<?php
function sumar($primero, $segundo) {
    echo $primero + $segundo;
}

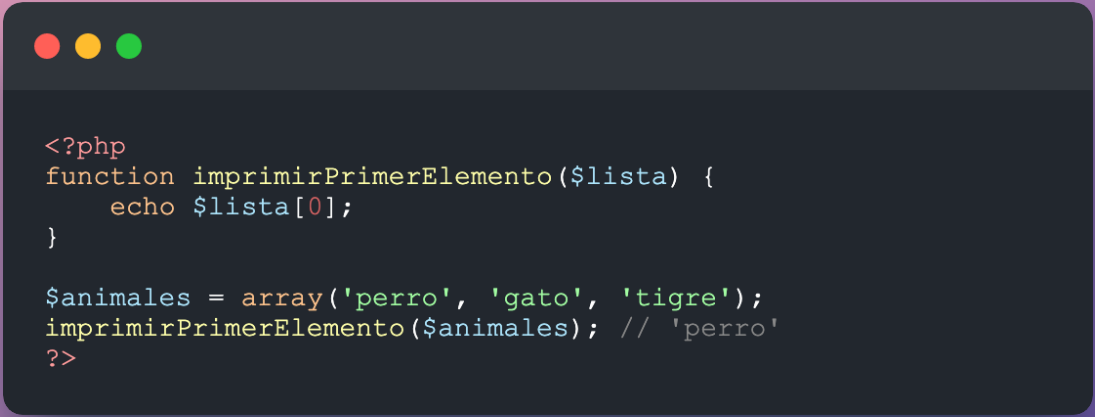
sumar(2, 2); // 4
sumar(3, 4); // 7
?>
```

También puedes crear funciones que retornen un valor. Esto significa que una vez que se ejecuta una función, el valor devuelto se puede usar en otra parte del código. Esta función toma dos argumentos (primero y segundo) y los multiplica para devolver el resultado. En este caso, el resultado es 6.

```
<?php
function multiplicar($primero, $segundo) {
    return $primero * $segundo;
}

$resultado = multiplicar(3, 2);
echo $resultado; // 6
?>
```

Esta función imprime el primer elemento de una lista. En este caso, la lista es una lista de animales, y la función imprime el primer elemento de la lista, que es 'perro'. Funciones como esta son útiles para evitar la repetición de código y para ahorrar tiempo.



```
<?php
function imprimirPrimerElemento($lista) {
    echo $lista[0];
}

$animales = array('perro', 'gato', 'tigre');
imprimirPrimerElemento($animales); // 'perro'
?>
```

Finalmente, puedes ver otro ejemplo de una función bastante compleja. Esta función implementa el algoritmo de ordenamiento QuickSort para ordenar una lista de elementos. El algoritmo comienza tomando un elemento de la lista como pivote y luego coloca los elementos menores que el pivote a su izquierda y los elementos mayores a su derecha. Luego, se aplica el mismo algoritmo a las sublistas izquierda y derecha hasta que la lista esté ordenada. No tienes que entender todo lo que hace internamente pero te dará una idea de la complejidad que pueden tener programas como apps con miles de funciones.

```
<?php
function quicksort($lista) {
    if (count($lista) <= 1) {
        return $lista;
    }
    $pivote = $lista[0];
    $izquierda = [];
    $derecha = [];
    for ($i = 1; $i < count($lista); $i++) {
        $lista[$i] < $pivote ? array_push($izquierda,
        $lista[$i]) : array_push($derecha, $lista[$i]);
    }
    return array_merge(quicksort($izquierda), [$pivote],
    quicksort($derecha));
};

$numeros = [23, 45, 16, 37, 3, 99, 22];
$listaOrdenada = quicksort($numeros);
echo implode(' ', $listaOrdenada); // 3, 16, 22, 23, 37, 45,
99
?>
```

Reto:

Escribe una función que tome dos números como argumentos y devuelva el mayor de los dos.

9 Programación orientada a objetos (POO)

9.1 Paradigma

La Programación Orientada a Objetos (POO) es un paradigma de programación que se basa en la creación de objetos que contienen datos y funcionalidades. Estos objetos se relacionan entre sí para formar una estructura de datos compleja. Los lenguajes de programación orientados a objetos permiten que los programadores puedan crear objetos y usarlos para construir aplicaciones.

9.2 Clases

Las clases en PHP son una forma de definir objetos creando una plantilla. Se pueden usar para crear objetos con propiedades y métodos similares. Las propiedades son valores que pertenecen al objeto y los métodos son funciones que pertenecen al objeto.

Ejemplo:

En este ejemplo creamos una clase llamada 'Lenguaje' que inicializamos con las propiedades 'nombre' y 'año'. Adicionalmente creamos el método 'descripción' que imprime un texto usando las propiedades previas. Con esa clase podemos crear una instancia para el lenguaje 'HTML' y otra para 'CSS'. Ahora podemos crear, en teoría, un número infinito de lenguajes sin reescribir los objetos manualmente.

```
<?php
class Lenguaje {
    public $nombre;
    public $año;

    public function __construct($nombre, $año) {
        $this->nombre = $nombre;
        $this->año = $año;
    }

    public function descripcion() {
        echo $this->nombre . " fue creado en " . $this->año;
    }
}

$html = new Lenguaje('HTML', 1993);
$css = new Lenguaje('CSS', 1996);
$html->descripcion(); // 'HTML fue creado en 1993'
$css->descripcion(); // 'CSS fue creado en 1996'
?>
```

Reto:

Crea una clase llamada Auto que tenga propiedades como marca, modelo y año. Luego, crea un par de instancias de esta clase.

10 Módulos

Los módulos son una forma de incluir código externo de otro archivo. Esto permite a los desarrolladores reutilizar código y ahorrar tiempo al escribir código.

Ejemplo:

10 MÓDULOS

Este archivo define una función que toma dos argumentos (a y b) y luego imprime el resultado de la resta de a y b en la consola. El código exporta la función para que pueda ser usada en otros archivos. Exportar es una forma de compartir código entre archivos. Esto significa que un archivo puede exportar variables, funciones y objetos para que otros archivos los puedan usar.

```
<?php
function restar($a, $b) {
    echo $a - $b;
}
?>
```

Y en otro archivo podemos importar y utilizar esta función. Este código importa una función desde el archivo externo y luego la usa para restar 10 menos 2, resultando en 8.

```
<?php
include './restar.php';
echo restar(10, 2); // 8
?>
```

Reto:

Crea un archivo llamado que contenga una función para calcular el área de un rectángulo. Luego, importa la función en otro archivo y usala para calcular el área de un rectángulo con lados de 3 y 4.

11 Siguientes pasos

11.1 Herramientas

- **Apache:** Es un servidor web de código abierto, el cual es el más utilizado para alojar aplicaciones web desarrolladas con PHP.
- **MySQL:** Es un sistema de gestión de bases de datos relacionales, el cual es ampliamente usado para almacenar y recuperar información en aplicaciones web desarrolladas con PHP.
- **PHPMyAdmin:** Es una herramienta de administración de bases de datos MySQL, la cual permite a los usuarios realizar tareas como crear tablas, insertar datos, modificar tablas, etc.
- **XAMPP:** Es una distribución de Apache, MySQL, PHP y Perl, la cual es usada para desarrollar aplicaciones web localmente.
- **Composer:** Es una herramienta de gestión de dependencias para PHP, la cual permite a los desarrolladores instalar y actualizar librerías y frameworks de terceros.
- **Laravel:** Es un framework de código abierto para desarrollar aplicaciones web con PHP, el cual ofrece una gran cantidad de herramientas y funcionalidades para facilitar el desarrollo.

11.2 Recursos

1. **PHP Manual:** El Manual de PHP es una referencia completa y detallada de todas las funciones y características de PHP. Está escrito por los desarrolladores de PHP y contiene información sobre cada función, clase, método, variable y constante.

2. [PHP The Right Way](#): Esta guía es un recurso en línea para aprender sobre el desarrollo de PHP de la manera correcta. Está diseñado para ayudar a los desarrolladores a aprender las mejores prácticas y estándares de codificación para el lenguaje.
3. [PHP.net](#): Esta es la página principal de PHP.net, que contiene información sobre la última versión de PHP, así como enlaces a documentación, descargas, foros de discusión y recursos adicionales.
4. [Stack Overflow](#): Stack Overflow es una comunidad en línea para desarrolladores donde pueden preguntar y responder preguntas relacionadas con el desarrollo de software. Está especialmente útil para preguntas relacionadas con PHP.
5. [W3Schools](#): Esta es una guía en línea para aprender PHP. Contiene tutoriales paso a paso, ejemplos de código y referencias para ayudar a los desarrolladores a aprender el lenguaje.

11.3 ¿Que viene después?

1. Aprender un framework de PHP como Laravel, Symfony o CodeIgniter.
2. Aprender una base de datos relacional como MySQL, PostgreSQL o SQLite.
3. Aprender una base de datos no relacional como MongoDB.
4. Aprender HTML, CSS y JavaScript para crear interfaces de usuario.
5. Aprender sobre seguridad web para proteger tu aplicación.
6. Aprender sobre el alojamiento web para publicar tu aplicación.
7. Aprender sobre el desarrollo de APIs para conectar tu aplicación con otros servicios.
8. Aprender sobre el mantenimiento de aplicaciones para mantener tu aplicación actualizada.

11.4 Preguntas de entrevista

1. ¿Cuál es la diferencia entre PHP y HTML?
 - PHP es un lenguaje de programación, mientras que HTML es un lenguaje de marcado para crear páginas web.
2. ¿Qué es una variable en PHP?
 - Una variable en PHP es un contenedor para almacenar datos.
3. ¿Cómo puedo conectarme a una base de datos en PHP?
 - Puedes conectarte a una base de datos en PHP usando la función `mysql_connect()`.
4. ¿Cómo puedo enviar un correo electrónico desde PHP?
 - Puedes enviar un correo electrónico desde PHP usando la función `mail()`.
5. ¿Qué es una sesión en PHP?
 - Una sesión en PHP es un mecanismo para almacenar datos entre varias páginas web.
6. ¿Cómo puedo crear una cookie en PHP?
 - Puedes crear una cookie en PHP usando la función `setcookie()`.
7. ¿Cómo puedo leer una cookie en PHP?
 - Puedes leer una cookie en PHP usando la función `$_COOKIE`.
8. ¿Cómo puedo eliminar una cookie en PHP?
 - Puedes eliminar una cookie en PHP usando la función `setcookie()` con una fecha de expiración en el pasado.
9. ¿Cómo puedo cargar un archivo en PHP?

11 SIGUIENTES PASOS

- Puedes cargar un archivo en PHP usando la función `include()` o `require()`.
10. ¿Cómo puedo redireccionar una página web en PHP?
- Puedes redireccionar una página web en PHP usando la función `header()`.
11. ¿Qué es una función en PHP?
- Una función en PHP es un bloque de código que se puede reutilizar para realizar una tarea específica.
12. ¿Cómo puedo pasar parámetros a una función en PHP?
- Puedes pasar parámetros a una función en PHP usando los paréntesis después del nombre de la función.
13. ¿Cómo puedo devolver un valor de una función en PHP?
- Puedes devolver un valor de una función en PHP usando la palabra clave `return`.
14. ¿Qué es una clase en PHP?
- Una clase en PHP es una plantilla para crear objetos.
15. ¿Cómo puedo crear un objeto en PHP?
- Puedes crear un objeto en PHP usando la palabra clave `new`.
16. ¿Qué es una excepción en PHP?
- Una excepción en PHP es una situación anormal que se produce durante la ejecución de un programa.
17. ¿Cómo puedo manejar una excepción en PHP?
- Puedes manejar una excepción en PHP usando la palabra clave `try/catch`.
18. ¿Qué es una matriz en PHP?

11 SIGUIENTES PASOS

- Una matriz en PHP es una colección de datos indexada.
19. ¿Cómo puedo recorrer una matriz en PHP?
- Puedes recorrer una matriz en PHP usando la función `foreach()`.
20. ¿Qué es un bucle en PHP?
- Un bucle en PHP es una estructura de control que se repite hasta que se cumpla una condición.
21. ¿Cómo puedo crear un bucle en PHP?
- Puedes crear un bucle en PHP usando la palabra clave `while`.
22. ¿Qué es una cadena en PHP?
- Una cadena en PHP es una secuencia de caracteres.
23. ¿Cómo puedo concatenar cadenas en PHP?
- Puedes concatenar cadenas en PHP usando el operador de punto (`.`).
24. ¿Qué es una expresión regular en PHP?
- Una expresión regular en PHP es un patrón de búsqueda para encontrar cadenas de texto.
25. ¿Cómo puedo usar una expresión regular en PHP?
- Puedes usar una expresión regular en PHP usando la función `preg_match()`.
26. ¿Qué es una variable de ámbito en PHP?
- Una variable de ámbito en PHP es una variable que sólo está disponible dentro de un ámbito específico.
27. ¿Cómo puedo definir una variable de ámbito en PHP?
- Puedes definir una variable de ámbito en PHP usando la palabra clave `global`.

11 SIGUIENTES PASOS

28. ¿Qué es una constante en PHP?

- Una constante en PHP es una variable que no puede cambiar su valor.

29. ¿Cómo puedo definir una constante en PHP?

- Puedes definir una constante en PHP usando la función `define()`.

30. ¿Qué es una función anónima en PHP?

- Una función anónima en PHP es una función sin nombre.

31. ¿Cómo puedo crear una función anónima en PHP?

- Puedes crear una función anónima en PHP usando la palabra clave `function`.

32. ¿Qué es una clase abstracta en PHP?

- Una clase abstracta en PHP es una clase que no puede ser instanciada.

33. ¿Cómo puedo crear una clase abstracta en PHP?

- Puedes crear una clase abstracta en PHP usando la palabra clave `abstract`.

34. ¿Qué es una interfaz en PHP?

- Una interfaz en PHP es una plantilla para crear clases.

35. ¿Cómo puedo crear una interfaz en PHP?

- Puedes crear una interfaz en PHP usando la palabra clave `interface`.

36. ¿Qué es una herencia en PHP?

- Una herencia en PHP es un mecanismo para compartir código entre clases.

37. ¿Cómo puedo usar la herencia en PHP?

- Puedes usar la herencia en PHP usando la palabra clave `extends`.

38. ¿Qué es una clase padre en PHP?

11 SIGUIENTES PASOS

- Una clase padre en PHP es una clase de la que otras clases heredan.
39. ¿Qué es una clase hija en PHP?
- Una clase hija en PHP es una clase que hereda de otra clase.
40. ¿Qué es una clase estática en PHP?
- Una clase estática en PHP es una clase que no puede ser instanciada.
41. ¿Cómo puedo crear una clase estática en PHP?
- Puedes crear una clase estática en PHP usando la palabra clave static.
42. ¿Qué es una variable de clase en PHP?
- Una variable de clase en PHP es una variable que se comparte entre todas las instancias de una clase.
43. ¿Cómo puedo crear una variable de clase en PHP?
- Puedes crear una variable de clase en PHP usando la palabra clave static.
44. ¿Qué es un método estático en PHP?
- Un método estático en PHP es un método que no se puede llamar desde una instancia de una clase.
45. ¿Cómo puedo crear un método estático en PHP?
- Puedes crear un método estático en PHP usando la palabra clave static.
46. ¿Qué es una clase final en PHP?
- Una clase final en PHP es una clase que no puede ser heredada.
47. ¿Cómo puedo crear una clase final en PHP?
- Puedes crear una clase final en PHP usando la palabra clave final.

11 SIGUIENTES PASOS

48. ¿Qué es una función de clase en PHP?

- Una función de clase en PHP es una función que se puede llamar sin instanciar una clase.

49. ¿Cómo puedo crear una función de clase en PHP?

- Puedes crear una función de clase en PHP usando la palabra clave static.

50. ¿Qué es una variable de instancia en PHP?

- Una variable de instancia en PHP es una variable que sólo está disponible para una instancia de una clase.

51. ¿Cómo puedo crear una variable de instancia en PHP?

- Puedes crear una variable de instancia en PHP usando la palabra clave this.

52. ¿Qué es un método de instancia en PHP?

- Un método de instancia en PHP es un método que sólo se puede llamar desde una instancia de una clase.

53. ¿Cómo puedo crear un método de instancia en PHP?

- Puedes crear un método de instancia en PHP usando la palabra clave this.

54. ¿Qué es una clase anidada en PHP?

- Una clase anidada en PHP es una clase que se define dentro de otra clase.

55. ¿Cómo puedo crear una clase anidada en PHP?

- Puedes crear una clase anidada en PHP usando la palabra clave class.

56. ¿Qué es una función de flecha en PHP?

- Una función de flecha en PHP es una función anónima que se puede usar como una expresión.

11 SIGUIENTES PASOS

57. ¿Cómo puedo crear una función de flecha en PHP?

- Puedes crear una función de flecha en PHP usando la palabra clave arrow.

58. ¿Qué es una clase genérica en PHP?

- Una clase genérica en PHP es una clase que se puede usar con cualquier tipo de datos.

59. ¿Cómo puedo crear una clase genérica en PHP?

- Puedes crear una clase genérica en PHP usando la palabra clave generic.

60. ¿Qué es una excepción personalizada en PHP?

- Una excepción personalizada en PHP es una excepción que se puede crear para manejar situaciones anormales.

61. ¿Cómo puedo crear una excepción personalizada en PHP?

- Puedes crear una excepción personalizada en PHP usando la palabra clave throw.

62. ¿Qué es una función de devolución de llamada en PHP?

- Una función de devolución de llamada en PHP es una función que se puede pasar como parámetro a otra función.

63. ¿Cómo puedo crear una función de devolución de llamada en PHP?

- Puedes crear una función de devolución de llamada en PHP usando la palabra clave callback.