Ejercicios a realizar para confirmar el concepto de HECHO de las diferentes historias de usuario

Ejercicio 1

Observa el Código que sigue

```
<?php
$navegador=getenv("HTTP_USER_AGENT");
¿>
<!DOCTYPE html>
<html>
  <head>
    <title>Código embebido para conocer el navegador cliente</title>
  </head>
  <body>
    <p>Estás utilizando un navegador <?php echo $navegador ?>.</p>
  </body>
</html>
```

a) Investiga qué hace el servidor web cuando recibe la petición de un recurso con extensión .php
b) Intenta describir, a grandes rasgos, qué es lo que hace el código anterior
c) Incluye el código anterior en un archivo llamado infoNavegador.php y observa el resultado en el navegador
d) Investiga cómo trata el servidor web el archivo infoNavegador.php antes de enviar el resultado de la petición al cliente. (¿realmente lo trata el servidor web?)
e) Consulta en la documentación para qué se utilizan los diferentes elementos del lenguaje PHP que aparecen el código. Vuelve a describir qué es lo que hace el código anterior, pero ahora de forma más exacta
f) Investiga las diferentes variables de entorno que se pueden manejar en el código PHP y de qué informan cada una de ellas. ¿Esas variables de entornos son propias de PHP? ¿Se podrían utilizar en otros lenguajes de lado servidor?
g) Investiga otras construcciones/funciones del lenguaje que se pueden utilizar para conocer información sobre el entorno.
h) Responde ¿Qué es lo envía el servidor web al cliente?
i) Investiga otros lenguajes de lado servidor que podrían haberse empleado en el código

Ejercicio 2

Crea un documento HTML e incluye en él código que muestre en el navegador tu nombre y estudios que estás realizando. Emplea diferentes formas PHP de llevar a la salida los datos que deseas mostrar

Ejercicio 3

Modifica el ejercicio 2 de forma que ahora no exista documento HTML alguno. Juega con las etiquetas de cierre de guion, observa los resultados e investiga las ventajas e inconvenientes de incluir o no esas etiquetas de cierre

Ejercicio 4

Investiga las diferentes construcciones del lenguaje y/o funciones con las que se pueden enviar datos a la salida. Modifica el ejercicio 2 empleando esas construcciones/funciones para mostrar en el navegador los datos solicitados

Ejercicio 5

Investiga en la documentación de PHP cuál es la sintaxis de PHP para especificar el ámbito de una sentencia. Observa el código de los ejercicios realizados hasta ahora y reflexiona sobre la sintaxis que has estado utilizando

Ejercicio 6

Ya sabes cómo se separan las sentencias en PHP. Investiga si en PHP es obligatorio incluir siempre el carácter de separación de sentencias y cuáles son sus ventajas e inconvenientes. Pon a prueba lo aprendido aplicándolo en los Ejercicios 2 y 3

Ejercicio 7

Modifica el ejercicio 2 e imitando el ejercicio 1 en el que se definen variables, experimenta a situar código en diferentes partes de la página HTML y observa el resultado. Si obtienes algún error o los resultados no son los esperados, intenta comprender qué es lo que lo causa, saca consecuencias y corrígelo

Ejercicio 8

Investiga qué son las directivas PHP y dónde se pueden definir y/o modificar.

Aprovechando los posibles errores que hayas obtenido en Ejercicio 7, modifica el ejercicio para incluir la directiva `error_reporting` asignándole diferentes valores cada vez, provoca que se produzcan errores y observa en la ventana del navegador los resultados obtenidos en función del valor asignado a la directiva `error_reporting`

Ejercicio 9

Explora los tipos que define el lenguaje PHP y su alcance. Investiga los 4 tipos escalares que define PHP y las conversiones implícitas que realiza el lenguaje y las funciones que se pueden utilizar para realizar conversiones explícitas. Investiga las diferentes funciones que integra PHP para conocer el tipo de datos de las variables.

Define las siguientes variables:

```
$var1= 5230
$var2= 0
$var3= -45
$var4= true
$var5= false
$var6= 5230.23
$var7= -45.23
$var8= 0.0
$var9= "Lola la de las flores"
$var10= "825"
$var11= "0825"
$var12= "450 loros"
$var13= "Loros a 45.9"
$var14= "48E4"
$var15= "45.9"
$var16= 0.6
$var17= 0.3
$var18= 10
```

Para cada una de las variables anteriores, muestra en la ventana del navegador un mensaje con un texto parecido a:

`La variable $var` **`X`** `es de tipo` **`tipoDatosVariable`** `y su valor es` **`valor`**

Ejercicio 10

Investiga en profundidad (si no lo has hecho antes) el tipo de datos `string` de PHP. Responde a las preguntas:

¿Qué diferencia existe entre colocar el mensaje a mostrar entre comillas dobles ("") o simples (')?

¿Cuál de las dos formas consume más recursos?

¿Cuándo se debería emplear una u otra forma?

¿En cuál de las dos formas puedo escapar cada carácter especial?

¿Qué efecto tiene especificar el nombre de una variable entre comillas simples o dobles?

¿Qué ventajas e inconvenientes tienen los formatos Heredoc y Nowdoc sobre rodear las cadenas de mensajes entre comillas simples o dobles?

Para que las definiciones de cadenas Heredoc y Nowdoc se realicen correctamente ¿cuál es la regla que hay que observar?

Ejercicio 11

Investiga qué es una expresión el lenguaje PHP y cuál es su importancia

Ejercicio 12

Investiga los diferentes operadores aritméticos, de asignación, de incremento/decremento, lógicos y de cadena. Investiga si existen funciones PHP integradas que faciliten las operaciones y que mejoren la exactitud de los datos.

Partiendo de los datos definidos en el Ejercicio 9:

1. Realiza de cabeza (papel y lápiz) las siguientes operaciones, determinando cuál va a ser el resultado de la operación y cuál el tipo de datos resultante:

```
$var1 + $var3
$var3 + $var5
$var10 * $var1
$var1 / $var10
$var11 + $var3
$var9 - $var3
$var3 + $var12
$var13 + $var8
$var3 * $var14
$var18 + $var15
$var10++
--$var11
$var1 && $var9
$var8 || $var12
!$var11
!$var10
!$var13
$var13 && $var8
$var13 || $var8
$var4 . $var5
$var4 . $var1
$var10 . $var13
$var14 . $var16
$var1 . $var9
$var1 . $var3
```

2. Con todas y cada una de las operaciones anteriores, muestra en la ventana del navegador mensajes parecidos a:

   El resultado de **$op1 operando $op2** es **resultado** y el tipo de datos del resultado es **tipoDatos**

   Para mostrar el mensaje, emplea todas las combinaciones posibles que se ocurran aplicando las variantes que conoces como resultado de la investigación realizada en el ejercicio 10

Ejercicio 13

Investiga cómo se nombran las variables en PHP. Investiga las funciones matemáticas que define PHP.

¿Una variable definida como $var puede referenciarse como $Var?

Desde una página HTML, define tres variables significativas (que autodocumenten el código y que permitan saber claramente qué significa el dato dentro de la solución), una que mantenga el sueldo bruto del trabajador, otra que mantenga el tipo (tanto por ciento) que se le aplica de retención en razón del IRPF, una más que mantenga la cuota de la Seguridad Social a cargo del trabajador y, por último, una que recoja el sueldo neto que va a recibir el trabajador como resultado de restar al sueldo bruto la cuota de la seguridad social a cargo del trabajador y el importe de aplicar al sueldo bruto el tipo de retención a cuenta del IRPF.

Asigna a cada una de las variables los siguientes valores:

- Sueldo bruto, 2836,25€
- Tanto por ciento de la retención: 15,75%
- Cuota de la Seguridad Social a cargo del trabajador 385,60€

A través de una o varias expresiones, calcula el sueldo neto del trabajador. Por último, muestra un informe como el siguiente:

```
Sueldo bruto:                          X.XXX,XX
Retención a cuenta del I.R.P.F.:         XXX,XX
Cuota SS a cargo del trabajador:         XXX,XX
Suelo neto:                            X.XXX,XX
```

Utiliza las funciones matemáticas que necesites para obtener un resultado con solo dos decimales redondeados en función de si el valor decimal es mayor o igual a 0,50 o menor a 0,50.

Ejercicio 14

Investiga cuándo existe, para el lenguaje PHP, una variable. Investiga qué funciones integra PHP para informar al programador de si una variable ya existe, está ya inicializada o si su valor es NULL.

Define en una página HTML las siguientes variables y asígnales los valores que aparecen a su lado (si es que aparecen):

```
$var1 5230
$var2 0
$var3
$var4 true
$var5 false
$var6 ''
$var7 "Lola la de las flores"
$var8 null
```

Y, para cada una de las variables, muestra en una tabla HTML un informe como el siguiente:

| Expresión | Tipo de datos de la expresión | Tiene un valor | Tiene un valor NULL | Está definida |
|---|---|---|---|---|
| **Nombre variable y valor (si existe)** | **Tipo de datos del valor que contiene** | **¿Tiene o no un valor?** | **¿Su valor es null?** | **¿Está definida la variable?** |

Ejercicio 15

Investiga cómo se definen en PHP las constantes y cuáles son las normas de estilo que se suelen aplicar entre los programadores de PHP respecto a las constantes.

En una página HTML, define una constante para el valor de pi (3,14159265358979323846264338327950) y otra constante para el valor del número e (2,71828182845904523536028747135266). Define cada constante de una de las dos posibles formas de definir constantes.

Define una variable radio (de una circunferencia) con un valor de 102,75 centímetros y un logaritmo neperiano con un valor de 1,6094379124341.

Utilizando las constantes y las variables definidas, muestra en la página HTML, de una forma elegante (CSS), los siguientes informes:

```
El perímetro de la circunferencia con un radio de XXX,XX centímetros es de X,XXXX metros

El número que representa el logaritmo neperiano X,XXXXXXXXXXXXX es X
```

Vuelve a repetir el ejercicio, pero esta vez define las constantes por debajo de las sentencias que muestran el informe.

¿Qué ha pasado? ¿Cuál es la diferencia entre las dos formas de definir constantes? ¿Qué restricción hay que observar con la forma más moderna de definir valores constantes?

Ejercicio 16

Investiga qué ámbito tienen las variables dentro de los guiones PHP.

Vuelve a realizar el ejercicio 13 definiendo en guiones diferentes las variables que se emplean en el ejercicio. Obtén los resultados que se solicitan en el ejercicio y verifica lo aprendido en la investigación sobre el ámbito de las variables en PHP.

Ejercicio 17

Investiga las diferentes estructuras de decisión (y sus variantes) que define el lenguaje PHP. Investiga, también, los operadores relacionales que define PHP.

¿Qué diferencia hay entre especificar el operador == y el operador ===?

Define en una página HTML las siguientes variables y asígnales los valores que aparecen a su lado (si es que aparecen):

```
$var1 5230
$var2 0
$var3
```

Realiza las operaciones que se describen a continuación y muestra el informe de la operación y resultado correspondiente, parecido al del apartado 2 del Ejercicio 12

Divide $var1 entre $var2, pero si el valor del divisor es 0, debe mostrar el mensaje “División por 0, resultado infinito”.

Multiplica $va1 por $var3, pero si alguna de las variables no existe todavía, debe asignarle el valor 1.

Ejercicio 18

En una página HTML, define tres variables `$var1`, `$var2` y `$var3` y asígnalas cualquier valor numérico. Muestra en el navegador del cliente información sobre las variables y sobre qué variable mantiene el valor mayor y qué variable mantiene el valor menor (y cuáles son esos valores)

Ejercicio 19

En una página HTML, realiza un código PHP tal que, dadas tres variables numéricas con valores arbitrarios, ordene los valores de menor a mayor en las variables. Es decir, dadas las variables `$var1 = 10`, `$var2 = 5` y `$var3 = 6.2`, tras el proceso de ordenación las variables han de quedar de esta forma `$var1 = 5`, `$var2 = 6.2` y `$var3 = 10`. (Este proceso debe tener éxito para cualquier posible valor inicial que tengan las variables)

Ejercicio 20

Investiga las diferentes estructuras iterativas (y sus variantes) definidas por el lenguaje PHP. Investiga cuál es el comportamiento que define PHP para las sentencias break y continue.

En una página HTML, realiza guion PHP que muestre en el navegador del cliente los cien primeros números naturales, uno por línea, informando, además, de si el número es par o impar, si es primo o no y de si el número pertenece a la sucesión de Fibonacci o no. Emplea un tipo de bucle diferente para cada una de las iteraciones que necesites realizar.

Ejercicio 21

En una página HTML, realiza un guion PHP que muestre en el navegador del cliente los números perfectos que hay entre 1 y 10000.

Un número perfecto es aquel que es igual a la suma de sus divisores. Ejemplo: 6=1+2+3.

Ejercicio 22

Investiga sobre los arrays de PHP. En concreto, como se definen y se inicializan (dos posibles formas) y los dos tipos de las claves de acceso a los elementos del array (índices numéricos clásicos y cadenas (arrays asociativos))

a) Define un array $meses a través de una de las dos posibles formas, con claves de índice numérico de base 0, cuyos elementos sean los nombres de los meses del año. Recorre el array con un bucle for y muestra los nombres en la ventana del navegador.
b) Define un array $meses2 empleando la segunda forma, diferente de la anterior, con claves de índice numérico de base 1, cuyos elementos sean los nombres de los meses del año. En la definición del array solo se debe especificar la clave numérica una única vez. Recorre el array con un bucle while ayudándote de la

función de array que informa de la cantidad de elementos que tiene el array. Muestra los nombres de los meses en la ventana del navegador

Ejercicio 23

Investiga la estructura de iteración alternativa al bucle `for` que es especialmente adecuada para recorrer arrays en PHP.

Recorre el array $meses2 empleando la estructura iterativa objeto de investigación. ¿Encuentras alguna diferencia en el recorrido? ¿Qué ventajas e inconvenientes ves entre las dos estructuras iterativas a la hora de recorrer arrays?

Define un array $diasMes cuyas claves sean los nombres de los meses del año (asociativo) y sus valores los días del mes de la clave (febrero 28 días).

¿Cómo se puede recorrer el array $diasMes?

Recorre el array $diasMes y muestra una tabla HTML en la ventana del navegador que muestre los datos que mantiene el array:

| Nombre del mes | Días | Mensaje |
|---|---|---|
| Enero | 31 | El mes de enero tiene 31 días |
| … | … | … |
| Diciembre | 31 | El mes de diciembre tiene 31 días |

Ejercicio 24

Continúa la investigación sobre los arrays. Los arrays pueden tener muchas dimensiones (claves o índices diferentes), entre diferentes dimensiones del array ¿tienen que ser todas las claves del array del mismo tipo? ¿Y dentro de la misma dimensión? ¿PHP realiza alguna conversión de tipo implícita según cómo se especifique el valor de la clave? ¿Qué sucede cuando en la definición del array, por la razón que sea, se especifica el mismo valor de clave más de una vez?

Define el array $diasSemana de la siguiente manera:

```
$diasSemana = ["lunes",
               "martes",
               "1"=>"miércoles",
               1.7=>"jueves",
               "01"=>"viernes",
               True=>"sábado",
               "domingo",];
```

Responde, ¿es sintácticamente correcta la definición anterior?

Si has respondido sí a la pregunta anterior, ahora responde, ¿qué contiene realmente el array? Si realizo un recorrido del mismo ¿qué se mostrará realmente?

En una página HTML, crea el código PHP necesario para definir y recorrer el array y mostrar su contenido (claves y elementos). Reflexiona sobre el resultado y obtén las conclusiones adecuadas. Si el resultado no coincide con lo las respuestas que has dado, vuelve a la documentación de PHP y repasa lo investigado sobre arrays y sus claves.

Ejercicio 25

Define el array $diasSemana de la siguiente manera:

```
$diasSemana = ["lunes", "martes", "miércoles", "jueves", "viernes", "sábado",
"domingo",];
```

Define el array $orden de la siguiente manera:

```
$orden = ["primer", "segundo", "tercer", "cuarto", "quinto", "sexto", "séptimo",
"octavo", "noveno", "décimo", "undécimo", "duodécimo", "decimotercero",];
```

En una página HTML, realiza un código PHP cuya ejecución cree, partiendo de los arrays anteriores, un array denominado `$ordenDias` con los siguientes valores:

`"primer"=>"lunes", "segundo"=>"martes", ..., "sexto"=>"sábado", "séptimo"=>"domingo"`

Y a continuación recorre `$ordenDias` y muestra un informe como el siguiente:

`El Lunes, es el primer día de la semana; el Martes, es el segundo día de la semana; …; el Domingo, es el séptimo día de la semana`

Ejercicio 26

Sigue investigando sobre arrays. Los arrays, ¿qué tipos de elementos pueden contener? ¿Tienen que ser los elementos del array todos del mismo tipo?

Investiga las funciones que define PHP que se pueden emplear con los arrays.

Partiendo de los valores de las variables del Ejercicio 9, define un array llamado `$valores` que contenga los valores de las variables.

Recorre el array con un bucle `for` y muestra en el navegador un informe parecido al del ejercicio 9:

`El elemento` **`valorClave`** `del array es de tipo` **`tipoDatosElemento`** `y su valor es` **`valor`**

Ejercicio 27

Define el siguiente array:

```
$peliculas=['enero' => [25=> ['La vida de Brian',
                              'El discreto encanto de la burguesía',],
                        28=> ['Los caballeros de la mesa cuadrada',
                              'El ángel exterminador',],
                       ],
            'marzo' => [3 => ['El sentido de la vida',
                              'El desprecio',],
                        4 => ['Al final de la escapada',
                              'Marnie, la ladrona'],
                        18=> ['Topaz',
                              'Bienvenido Mister Marshall',
                              'Una banda aparte',],
                        25=> ['El verdugo'],
                        26=> ['Calabuch',
                              'Vértigo',],
                       ],
            'abril' => [1 => ['Pierrot el loco',],
                       ],
           ];
```

El array mantiene las películas que un usuario ha visto en determinados días de determinados meses.

Recorre el array y muestra en una tabla HTML las películas que ha visto el usuario. La tabla PHP tendrá el formato

| **Mes** | **Día** | **Película** |
|---|---|---|
| Enero | 25 | La vida de Brian |
| Enero | 25 | El discreto encanto de la burguesía |
| … | … | … |
| Abril | 1 | Pierrot el loco |

Vuelve a recorrer el array, pero ahora el informe debe ser:

En enero, el día 25 vi las películas: La vida de Brian y El discreto encanto de la burguesía.

…

En abril, el día 1 vi la película Pierrot el loco.

Ejercicio 27

Investiga cómo se definen las funciones de usuario en PHP, cómo se declaran sus parámetros formales, los diferentes tipos de parámetros que se pueden definir (y sus tipos) y cómo se declaran los tipos de valores que puede devolver.

Si una función no declara tipos para sus parámetros formales, ¿qué tipos de valores puede recibir?

Si una función no declara el tipo que devuelve, ¿qué tipos de valores puede devolver?

Si una función declara tipos para sus parámetros formales, ¿puede un parámetro recibir valores de más de un tipo? ¿puede un parámetro recibir un valor null?

Si una función declara tipos para el valor que devuelve, ¿puede devolver unas veces un valor null y otras un valor false?

¿Es necesario definir una función antes de referenciarla?

¿Admite PHP la sobrecarga de funciones?

Repite el ejercicio 20, emplea arrays y define funciones para informar si un número es par, si es primo y si pertenece a la sucesión de Fibonacci. Define correctamente los parámetros (y sus tipos) que recibe y los tipos de los valores que devuelve.