# UD2.- El Lenguaje PHP

## (Arrays, Matrices y  Manipulación de cadenas).

Arrays, **Matrices** y Manipulación de **cadenas**.

# Arrays

arrays

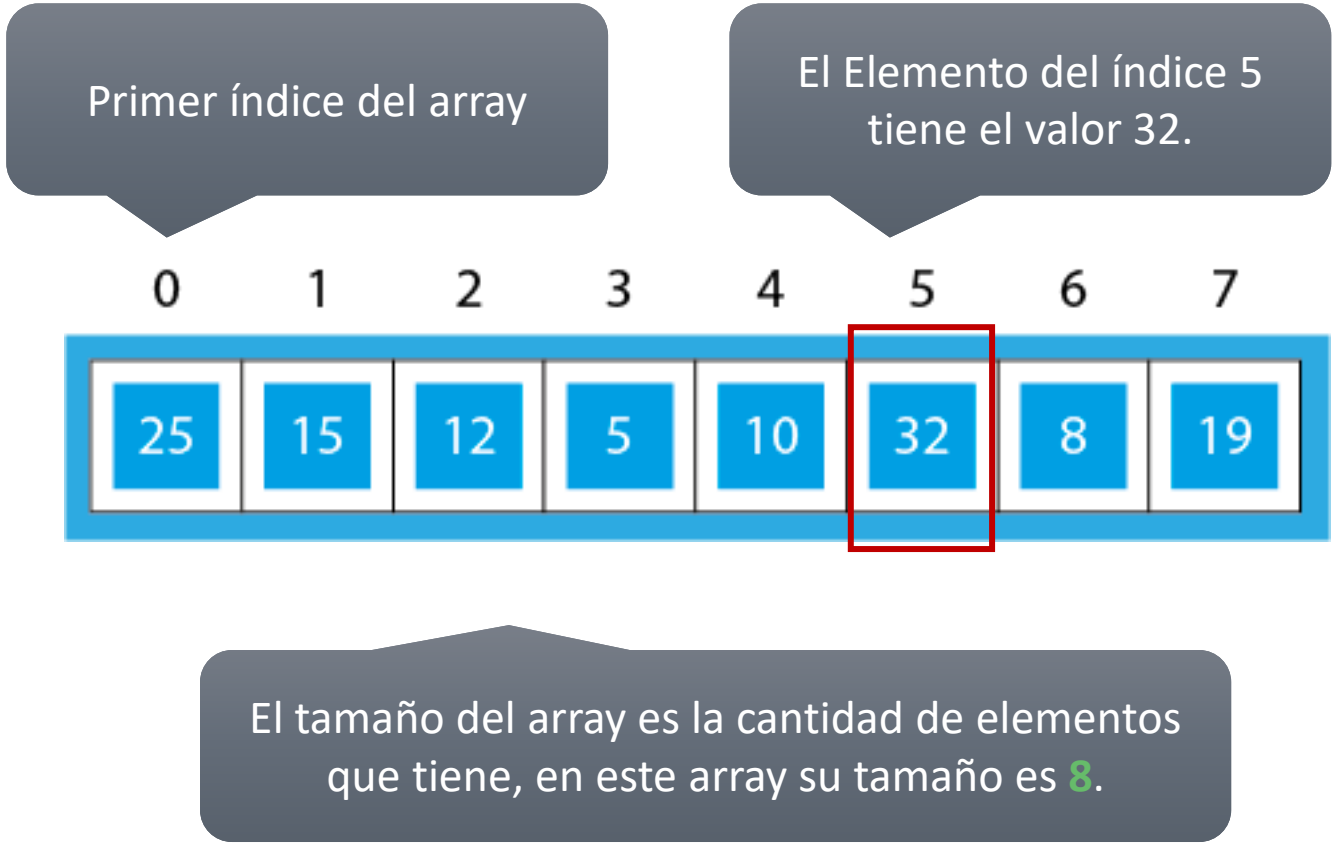
matrices

strings

# Arrays

Un *array* es conjunto de datos. Los arrays nos permiten almacenar varios datos en una sola variable, de forma que podemos acceder a esos datos utilizando distintos tipos de índices. En PHP existen tres tipos de arrays:

- **Numéricos**: la posición que ocupa cada elemento en el array la indica un número, y podemos acceder a esa posición indicando ese número entre corchetes. Las posiciones empiezan a numerarse desde la 0.
- **Asociativos**: la posición que ocupa cada elemento en la lista viene dada por un nombre, y podemos localizar cada elemento a través del nombre que le hemos dado, llamado *clave*.
- **Mixtos**: son arrays de varias dimensiones, donde en algunas se utilizan índices numéricos, y en otras, índices asociativos.



# Creación de Array Numérico

Se pueden crear de tres formas distintas:

1. Indicando entre paréntesis sus elementos, y anteponiendo la palabra **array**
2. Indicando a mano el índice que queremos rellenar, y el valor que va a tener (los índices intermedios que queden sin valor se quedarán como huecos vacíos).
3. Indicando el nombre del array con corchetes vacíos cada vez que queramos añadir un elemento. Así, se añade al final de los que ya existen.

1. Se define el array **$dias** con los valores “Lunes”, “Martes”, “Miércoles”

```php
$dias = array("Lunes","Martes","Miércoles");
```

El primer elemento (“Lunes”), está en la posición **0** del array.

2. Indicamos a mano los índices que queremos rellenar (0,1,2).

```php
$dias[0] = "Lunes";
$dias[1] = "Martes";
$dias[2] = "Miércoles";
```

3. Con corchetes vacíos, los elementos se van añadiendo al final de los elementos que ya existen en el array.

```php
$dias[] = "Lunes";
$dias[] = "Martes";
$dias[] = "Miércoles";
```

Ejemplos de uso de arrays.

# Ejemplo Arrays

Hemos visto como declarar arrays. Para sacar por pantalla algún valor, o usarlo en alguna expresión, pondremos el nombre del array y, entre corchetes, la posición que queremos:

Ejemplo de como acceder a la **segunda** posición de un array (*recuerda el primer elemento del array está en la posición 0*).

```
echo $dias[1];
```

Ejemplos de uso de arrays.

# Ejemplo Arrays

En el ejemplo superior vemos como declarar un *array* de caracteres, y en el ejemplo inferior se ha utilizado un bucle para recorrer un *array* de enteros y mostrar sus elementos.

Ejemplo de como declarar e inicializar un array de caracteres.

```php
$vocales = array('a','e','i','o','u');
```

Recorrer un array para mostrar sus elementos

```php
$numeros = array(10, 8, 6, 20, 18, 32, 45);

for($i=0;$i<count($numeros);$i++){
    echo "Elemento $i: $numeros[$i] <br />";
}
```

Ejemplos de uso de arrays.

# Ejemplo recorrer Arrays

Existe otra forma de recorrer los arrays, utilizando *foreach*.
Su sintaxis es:

*foreach(array as elemento)*

Ejemplo de como recorrer un array con **foreach**.

```php
$dias = array("Lunes","Martes","Miércoles");
foreach($dias as $dia)
    echo "$dia <br />";
```

Ejemplos de uso de arrays.

# Funciones de los Arrays

Algunas de las funciones más importantes existentes para trabajar con los arrays son:

- **$item = array_pop($array):** elimina el último *$item*.
- **array_push($array, $item):** añade un *$item* al final del array
- **$booleano = in_array($item, $array):** averigua si *$item* está en el *$array*.

Ejemplo de como utilizar funciones para trabajar con **arrays**.

```php
$frutas = ["sandía", "melocotón", "manzana"];

array_push($frutas, "naranjas");

$ultFruta = array_pop($frutas);
if (in_array("naranjas", $frutas))
    echo "<p>Quedan naranjas</p>";
else
    echo "<p>No quedan naranjas</p>";
```

# Funciones de los Arrays

Para eliminar elementos de un array, podemos utilizar las funciones **array_shift()** y **unset()**.
A continuación, vamos a ver unos ejemplos con cada una de ellas:

- **array_shift():**
  - Extrae el primer valor del array.
  - Acorta el array en un elemento.
  - Desplaza todos los elementos, y reindexa el array para empezar en 0.
- **unset():**
  - Elimina el elemento especificado sin reindexar el array.
  - Esto puede dejar “huecos” en los índices del array.

Utilizamos **array_shift()** para eliminar el primer elemento del array, ‘manzana’, y almacenarlo en *$primerElemento*.

```php
$miArray = ['manzana', 'banana', 'naranja'];
$primerElemento = array_shift($miArray); //Elimina el 1er elemento del array

print_r($miArray); // Salida: Array ( [0] => banana [1] => naranja )
echo "Elemento eliminado: " . $primerElemento; // Salida: Elemento eliminado: manzana
```

Al mostrar el array, observamos como ‘manzana’ ha desaparecido, y los índices de los elementos se han cambiado.

En este ejemplo, utilizamos **unset($miArray[0])** para eliminar el elemento de índice 0.

```php
$miArray = [0 => 'manzana', 1 => 'banana', 2 => 'naranja'];
unset($miArray[0]);

print_r($miArray); // Salida: Array ( [1] => banana [2] => naranja )
```

Al mostrar el array, observamos como ‘manzana’ ha desaparecido, y los índices de los elementos se conservan.

# Creación de un Array Asociativo

Cada elemento es un par *clave-valor*. En vez de acceder por la posición, lo hacemos mediante una clave. Así pues, para cada clave se almacena un valor.
A la hora de recorrer este tipo de arrays, mediante **foreach** separamos cada elemento en una pareja ***clave*** => ***valor***

Se define el array asociativo **$capitales** con los pares ***clave-valor*** (País-capital)

```php
$capitales = [  "España" => "Madrid",
                "Portugal" => "Lisboa",
                "Italia" => "Roma",
                "Francia" => "Paris"];

$capitales["Alemania"] = "Berlín"; // añadimos un elemento

echo "La capital de Francia es {$capitales["Francia"]} <br />";

$capitales[] = "Budapest"; // se añade con la clave 0 !!! ¡¡¡No asignar valores sin clave!!!

foreach ($capitales as $pais => $capital) { // separamos cada elemento en clave => valor
    echo "$pais : $capital <br />";
}
```

Con corchetes vacíos [], los elementos se van añadiendo al final de los elementos que ya existen en el array.

Ejemplos de uso de arrays.

# Funciones

# Arrays Asociativos

Algunas de las funciones más importantes existentes para trabajar con los arrays asociativos son:

- **$claves = array_keys($array):** devuelve las claves del *$array asociativo*.
- **$tam = count($array):** devuelve el tamaño del *$array*
- **$sort($array):** ordenas los elementos del *$array*.
- **isset($array[item])**: indica si existe/tiene valor item dentro del array
- **unset($array[item]):** elimina el item del array (deja un hueco)

Ejemplo de como utilizar funciones para trabajar con **arrays**.

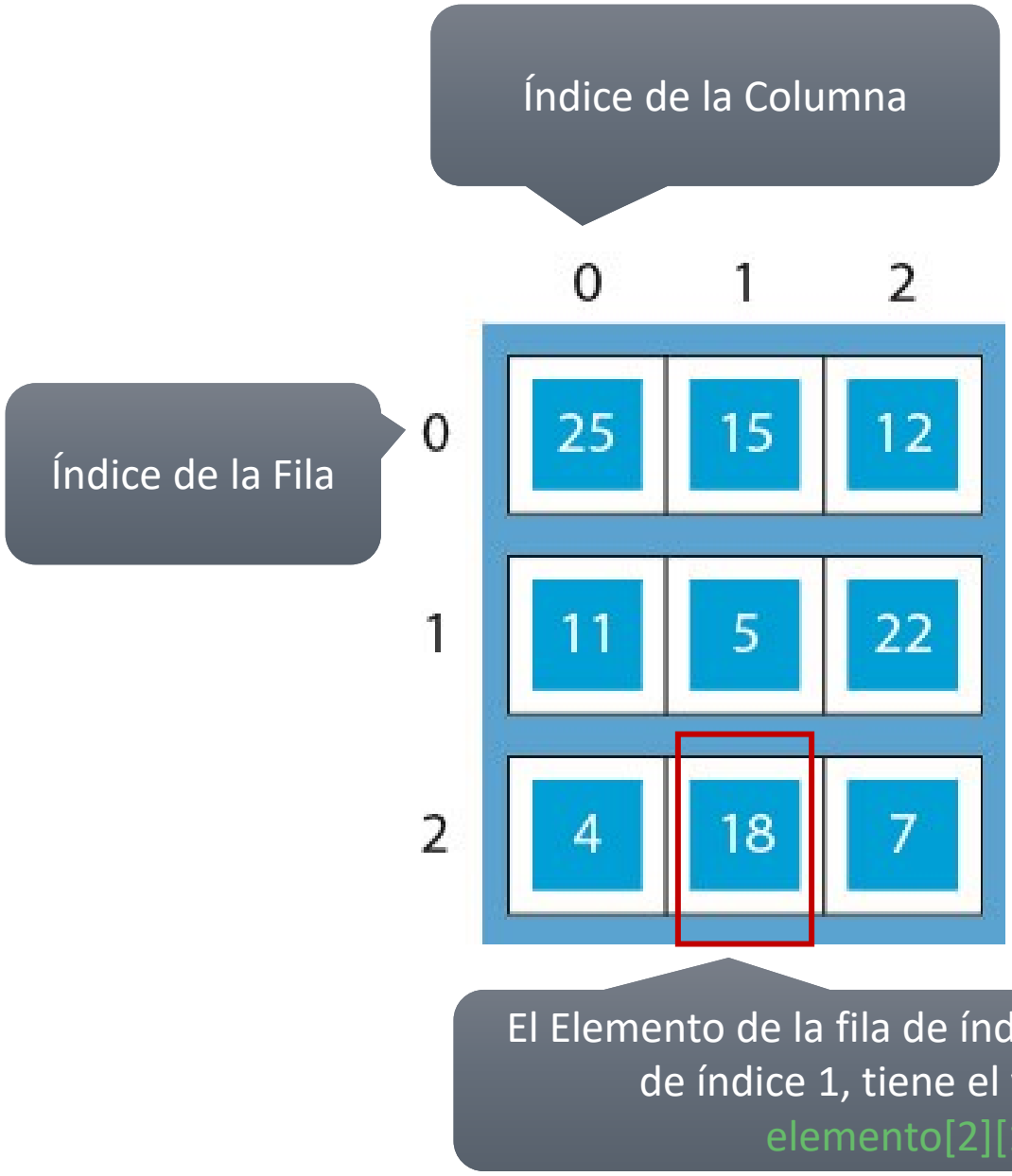
```php
$frutas = ["sandía", "melocotón", "manzana"];

array_push($frutas, "naranjas"); //agregamos naranjas al array

$ultFruta = array_pop($frutas);  //eliminamos el último elemento
/*Si existe el elemento naranjas en el array muestro "Quedan naranjas"
  En caso contrario muestro "No quedan naranjas"
*/
if (in_array("naranjas", $frutas))
    echo "<p>Quedan naranjas</p>";
else
    echo "<p>No quedan naranjas</p>";
```

# Matrices

Una matriz, es un *array* de varias dimensiones. Para acceder a cada elemento se hace mediante su índice, en este caso fila y columna en la *matriz.*

- La primera fila de una *matriz* es la ***0.***
- La primera columna de una *matriz* es la ***0.***

Para acceder a un elemento de la matriz indicamos primero el número de fila y a continuación el número de columna, ***$num[fila][columna]***



Ejemplos de uso de matrices.

# Ejemplo Matrices

Existen diversas formas de declarar y de instanciar las *matrices*. A continuación, se van a exponer dos ejemplos de como hacerlo.
También tenemos un par de ejemplos de como acceder de forma individual a un elemento de la matriz.

Ejemplo de como declarar e inicializar una matriz de enteros llamada edades.

```php
$edades = array(
    array(26,18,34),    // Fila 0
    array(8,7,15),      // Fila 1
    array(3,13,23)      // Fila 2
);
```

```php
$gente = array(
    array('nombre' => 'Luis', 'edad' => 14), // Fila 0
    array('nombre' => 'Silvia', 'edad' => 48)  // Fila 1
);
```

Ejemplo con claves asociativas, las claves como '**nombre**' y '**edad**' hacen el array más legible y fácil de usar.

Ejemplo de acceso a elementos de la matriz.

```php
echo $edades[0][1];     //Muestro 18
echo $edades[2][0];     //Muestro 3

echo $gente[0]['nombre']; // Imprime "Luis"
echo $gente[1]['edad'];   // Imprime 48
```

Ejemplos de uso de matrices.

# Ejemplo Matrices

En el siguiente ejemplo se han utilizado bucles anidados para recorrer una *matriz* y mostrar sus elementos.
Vemos las dos formas de recorrer matrices, con bucles ***foreach*** por un lado, y con bucles ***for*** por otro, ambos realizan la misma función en este caso.

## Recorrer una matriz para mostrar sus elementos con bucles foreach

```php
foreach ($edades as $fila) {    //Para recorrer la filas
    foreach ($fila as $valor) { //Para recorrer las columnas
        echo $valor . " "; // Imprime los valores de la matriz
    }
    echo "<br>"; // Para separar las filas en la salida
}
```

## Recorrer una matriz para mostrar sus elementos con bucles for

```php
for($fila=0;$fila<count($edades);$fila++){      //Para recorrer la filas
    for($col=0;$col<count($edades[0]);$col++){  //Para recorrer las columnas
        echo $edades[$fila][$col] . " "; // Imprime los valores de la matriz
    }
    echo "<br>"; // Para separar las filas en la salida
}
```

# Números Aleatorios

En PHP pueden generarse números aleatorios. También podemos generar números aleatorios criptográficamente seguros, es decir, números que son generados por una fuente de aleatoriedad confiable y difícil de predecir, como las fuentes del sistema operativo, a diferencia de los generadores de números pseudoaleatorios estándar, que pueden ser predecibles y no adecuados para tareas de seguridad.

Utilizamos la función **random_int()** para generar un número entero que sea seguro para usos criptográficos.

Las funciones **mt_rand()** y **rand()** son más rápidas y adecuadas para usos generales, como juegos o simulaciones.

Ejemplos de uso de las funciones para números aleatorios.

```php
// Genera un entero aleatorio entre 1 y 100
$numero_seguro = random_int(1, 100);
echo $numero_seguro;

// Genera un entero entre 1 y 100 con mt_rand()
$numero_mt = mt_rand(1, 100);
echo $numero_mt;

// Genera un entero entre 1 y 100 con rand()
$numero_rand = rand(1, 100);
echo $numero_rand;
```

# Manipulación de Cadenas

Una cadena es una sucesión de caracteres alfanuméricos.
PHP nos proporciona una serie de métodos o funciones, para *convertir a mayúsculas, o a minúsculas*, *obtener una subcadena*, *encontrar un texto...*
Para más información tenéis el enlace oficial a php: http://es.php.net/manual/es/ref.strings.php

*Para declarar un tipo **String**, debemos de indicar un nombre, y asignarle la cadena que queremos almacenar entre comillas “ ” o ‘ ’*

```
$nombreCad = “texto cadena”;
```

```
$nombreCad = ‘texto cadena’;
```

Ejemplos de declaración de strings.

# Ejemplos String

Veamos como declarar strings:

## Declaración de tipos **String**

```php
$saludo = "Hola";
$otro_saludo = 'Holi';
```

## Utilizamos el operador . para concatenar cadenas

```php
$saludo = "Hola";
$nombre = "Mundo";

$mensaje = $saludo . " " . $nombre . "!"; // Resultado: "Hola Mundo!"
```

# Métodos de String

- ***strlen($cadena),*** devuelve la longitud de un String.
- ***str_word_count($cadena)***: Cuenta el número de palabras en una cadena.
- ***strrev($cadena):*** Invierte una cadena.
- ***str_replace($buscar, $reemplazar, $cadena):*** Reemplaza todas las ocurrencias de una subcadena dentro de otra cadena.
- ***strtolower($cadena):*** Convierte toda la cadena a minúsculas.
- ***strtoupper($cadena)***: Convierte toda la cadena a mayúsculas.

Ejemplo de utilización de los métodos str_word_count(), str_replace(), strtoupper() y strrev() con Strings.

```php
$texto = "Este es un ejemplo de cadena.";
$numeroPalabras = str_word_count($texto); // $numeroPalabras será 6

$frase = "El gato que está triste y azul.";
$nuevaFrase = str_replace("azul", "blanco", $frase);
// $nuevaFrase será "El gato que está triste y blanco."

$s1 = "Una cadena de texto";
echo strtoupper($s1); //UNA CADENA DE TEXTO

$s2 = "Otra cadena de texto";
echo strrev($s2); //otxet ed anedac artO
```