

2022/2023(1)  
IF184504 Web Programming

Lecture #3

**PHP Basic**

Misbakhul Munir **IRFAN SUBAKTI**

司馬伊凡

Мисбакхул Мунир **Ирфан Субакти**

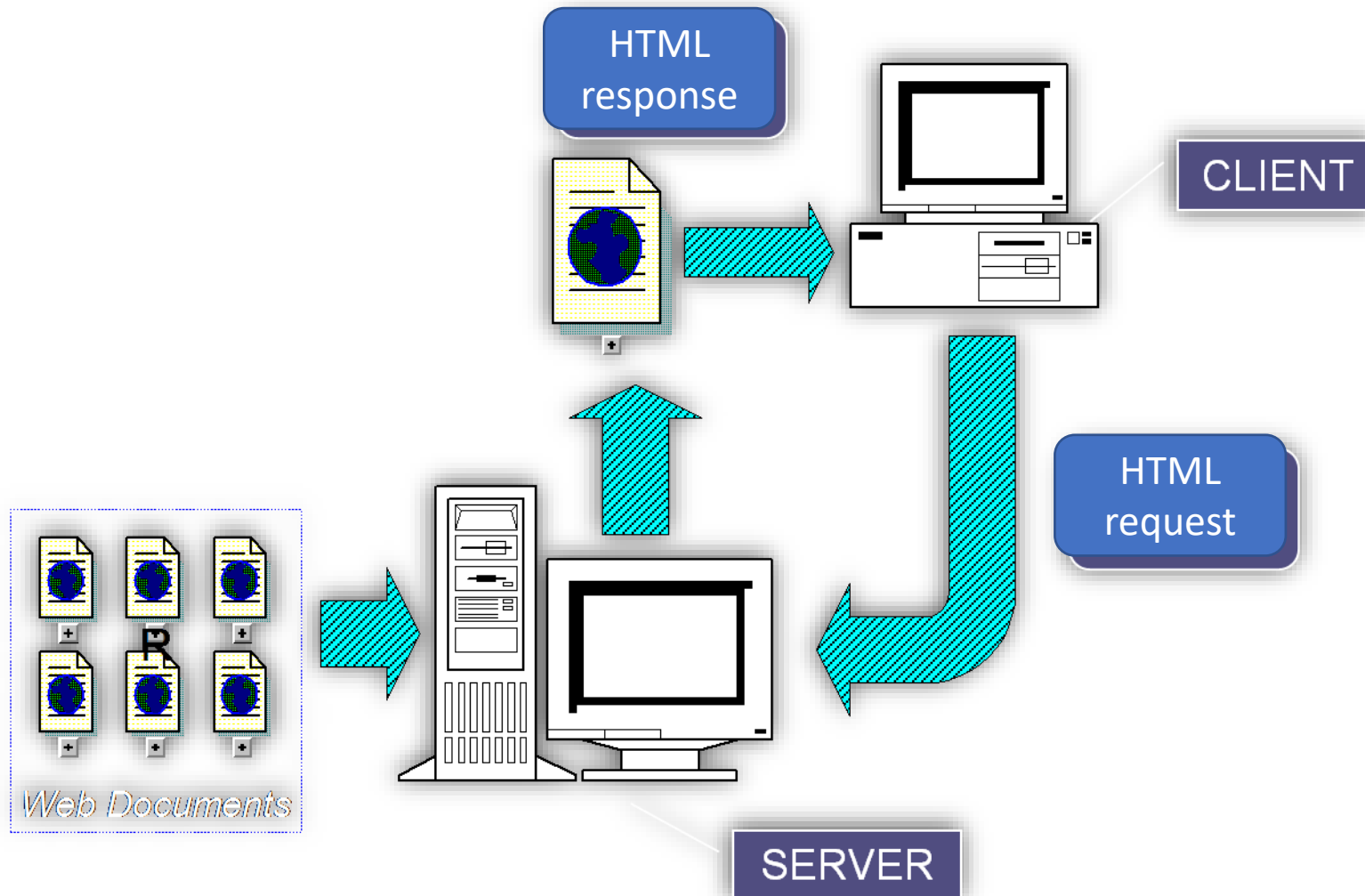
# PHP basic

- PHP → Personal Home Page
- PHP: Hypertext Preprocessor

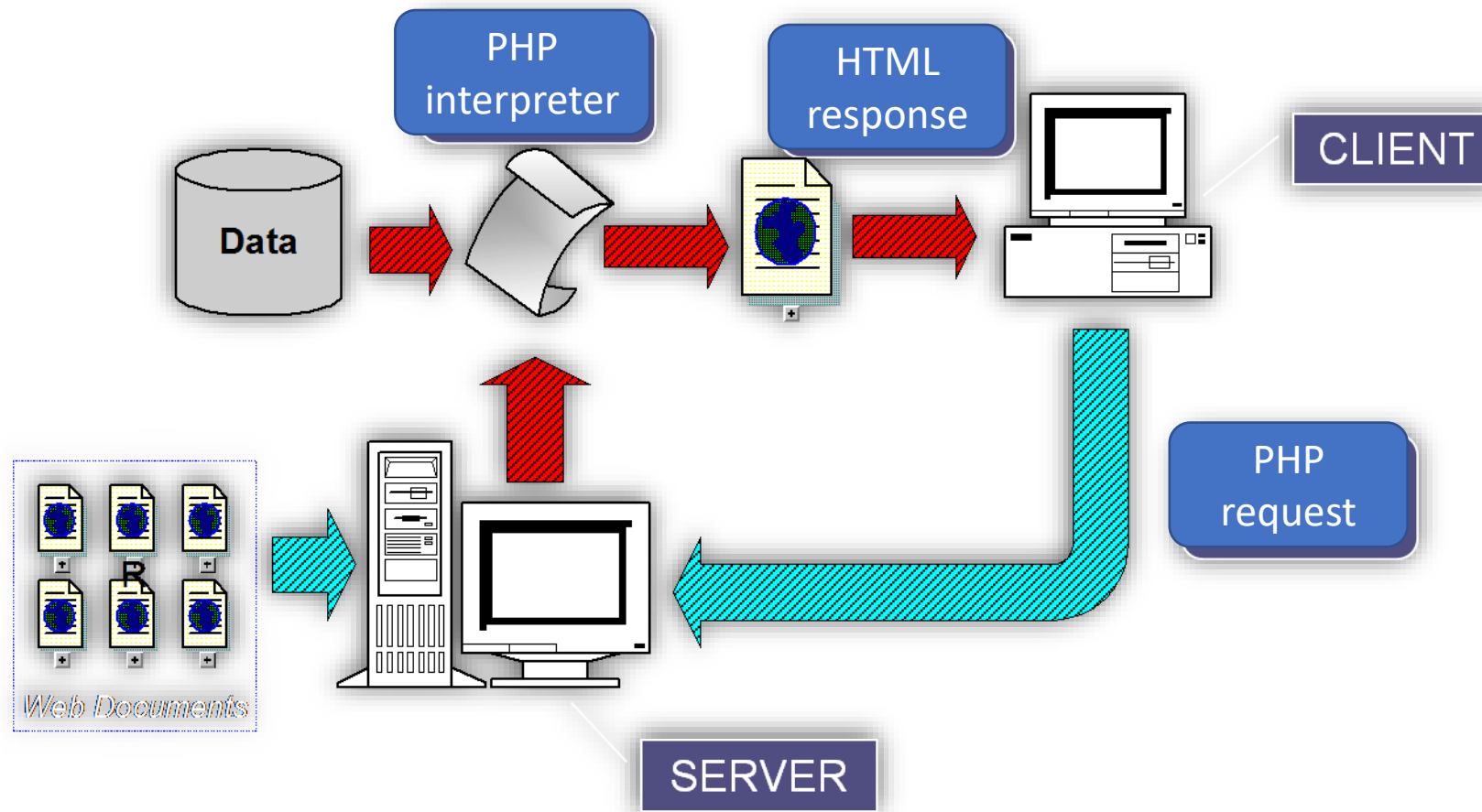
# Server Side Scripting (SSS)

- ASP (Active Server Page) and ASP.NET → [www.asp.net](http://www.asp.net) & [dotnet.microsoft.com/apps/aspnet](http://dotnet.microsoft.com/apps/aspnet)
- ColdFusion → [www.adobe.com/products/coldfusion-family.html](http://www.adobe.com/products/coldfusion-family.html)
- Java Server Pages → Jakarta Server Pages → [projects.eclipse.org/projects/ee4j.jsp](http://projects.eclipse.org/projects/ee4j.jsp)
- Perl → [perl.org](http://perl.org)
- Python → [python.org](http://python.org)
- PHP → [php.net](http://php.net)

# Static web architecture



# PHP (web) application architecture



# Web server & SSS installation

- Separated installation → PHP, Apache, MySQL
- Joined installation
  - XAMPP → [apachefriends.org](http://apachefriends.org) → will be used for now!
  - AppServ
  - WAMP
  - FoxServ
  - PHPTriad

# PHP

- What is that?
  - PHP → Personal Home Page
  - PHP: Hypertext Preprocessor

# PHP: the preparation

- Basic computer understanding
  - File, directory, creating file, etc.
- Ability for using text editor
  - Notepad, Notepad++, Sublime Text, DreamWeaver, Crimson Ed, vi, etc.
- Web Server for PHP has been installed
  - Apache Web Server, [XAMPP](#), etc.
- Ability for using web browser
  - Chrome, IE, Edge, Firefox, etc.



# PHP: the preparation (continued)

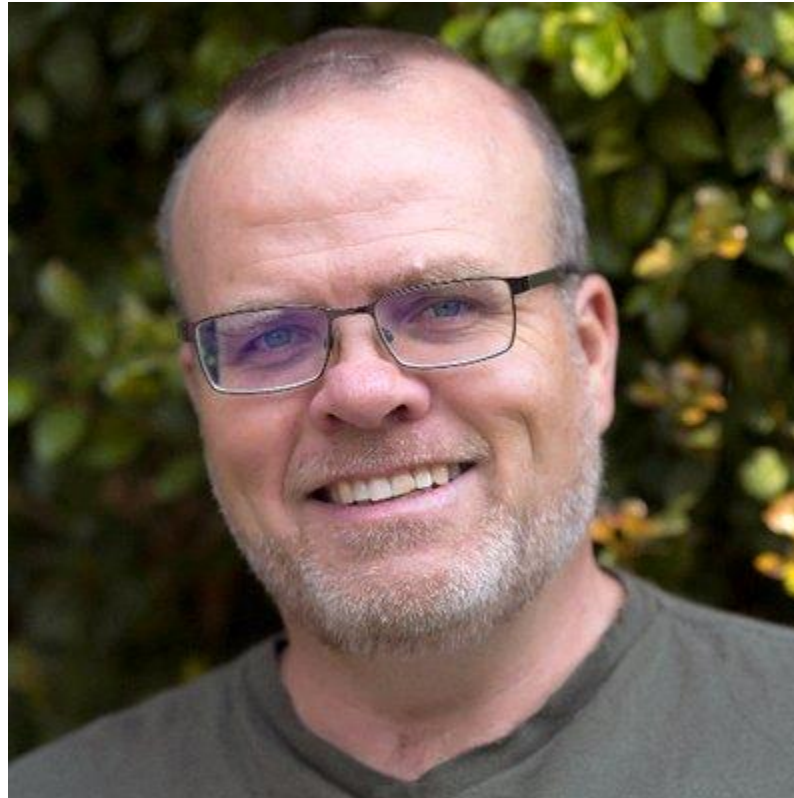
- Basic HTML (lecture #2)
- Googling ability

# PHP

- Recursive abbreviation of
  - PHP: Hypertext Preprocessor
- Rasmus Lerdorf: the co-author
  - Originally created in 1994
  - A group of developers including Jim Winstead (who later created blo.gs), Stig Bakken, Shane Caraveo, Andi Gutmans, and Zeev Suraski.
- Each statement ended by semicolon “;”
- Case sensitive for the user identifiers (identifier made by user), e.g., variable, constant, function, etc.
- Case **insensitive** for PHP built-in identifiers

# Rasmus Lerdorf: the PHP co-author

- [en.wikipedia.org/wiki/Rasmus\\_Lerdorf](https://en.wikipedia.org/wiki/Rasmus_Lerdorf)



# PHP: the script writing

- Need to be surrounded by

`<? and ?>` OR

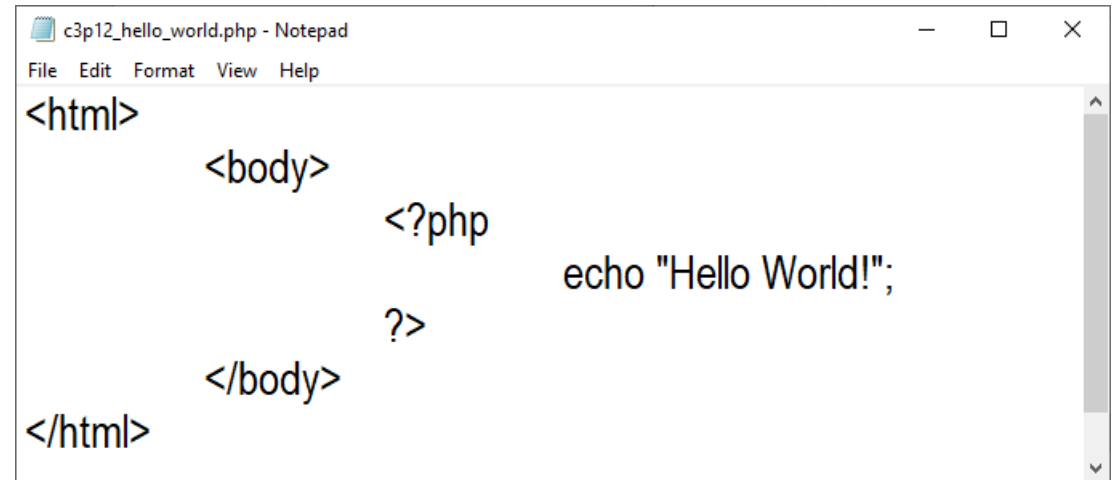
`<?php and ?>` OR

`<script language = "php"> and </script>` OR

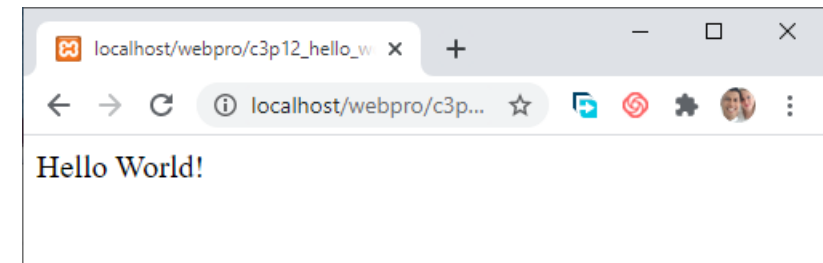
`<% and %>`

# Hello World: the program

```
<html>
  <body>
    <?php
      echo "Hello World!";
    ?>
  </body>
</html>
```

A screenshot of a Notepad window titled 'c3p12\_hello\_world.php - Notepad'. The window contains the following PHP code:

```
<html>
  <body>
    <?php
      echo "Hello World!";
    ?>
  </body>
</html>
```



# Variable

- Saving a value, data or information
- Variable name
  - Started by \$
  - Then the next character can be letter, number and any allowed character (ASCII 127-255)
- Unlimited length
- Case sensitive
- No need to be declared first
- Space is not allowed

# PHP data type

- Scalar types
  - Boolean
  - Integer
  - Float
  - String
- Compound types
  - Array
  - Object
- Special types
  - Resource
  - NULL

# Operators

- Arithmetic → +, -, \*, /, %, \*\* (exponentiation)
- Assignment → =
- Comparison → ==, ===, !=, <>, !==, >, <, >=, <=, <=>
  - === (identical), true if they are equal and of the same type)
  - !== (not identical)
  - <=> (spaceship), \$x <=> \$y, returns an integer less than, equal to, or greater than zero, depending on if \$x is less than, equal to, or greater than \$y; introduced in PHP 7
- Increment/decrement → ++, -- (pre & post)
- Logical → and, &&, or, ||, xor, !



# Operators (continued)

- String → . (concatenation), .= (concatenation assignment)
  - \$txt1 . \$txt2 → Concatenation of \$txt1 and \$txt2
  - \$txt1 .= \$txt2 → Appends \$txt2 to \$txt1
- Array → + (union), ==, === (identity), !=, <> (inequality), !== (non-identity)
- Conditional assignment → ?: (ternary), ?? (null coalescing)
  - \$x = expr1 ?? expr2
  - Returns the value of \$x.
  - The value of \$x is expr1 if expr1 exists, and is not NULL. If expr1 does not exist, or is NULL, the value of \$x is expr2. Introduced in PHP 7

# Comment

`/* ... */`

`//`

`#`

# Selection → branching

- Conditional statement

# if

```
if (condition) {  
    statement;  
}
```

```
if ($age >= 17) {  
    echo "Eligible to apply a driving license!";  
}
```

# if ... else

```
if (condition) {  
    statement-for-the-true-condition;  
} else {  
    statement-for-the-false-condition;  
}
```

```
if ($age >= 17) {  
    echo "Eligible to apply a driving license!";  
} else {  
    echo "Sit in the back seat, please!";  
}
```

?: → ternary

```
$var = (condition) ? true : false;
```

```
$status = ($age >= 17) ? "Eligible to apply a  
driving license!" : "Sit in the back seat,  
please!";
```

# if ... endif

```
if (condition) :  
    statement-for-the-true-condition;  
endif;
```

```
if ($age >= 17) :  
    echo "Eligible to apply a driving license!";  
endif;
```

# switch ... case

```
switch ($var) {  
    case 1: statement-1; break;  
    case 2: statement-2; break;  
    ...  
}
```

```
switch ($ron) {  
    case 88: echo "Premium"; break;  
    case 90: echo "Pertalite"; break;  
    case 92: echo "Pertamax"; break;  
    case 98: echo "Pertamax Turbo"; break;  
}
```



# Repetition → looping

- Looping statement

# for

```
for (init counter; test counter; increment) {  
    code to be executed for each iteration;  
}
```

```
for ($i = 1; $i <= 7; $i++) {  
    echo $i;  
}
```

# while

```
initialisation;  
while (condition) {  
    statement-to-be-repeated;  
    increment;  
}
```

```
$i = 0;  
while ($i <= 7) {  
    echo $i;  
    $i = $i + 1;  
}
```

# do ... while

```
initialisation;  
do {  
    statement-to-be-repeated;  
    increment;  
} while (condition)
```

```
$i = 0;  
do {  
    echo $i;  
    $i = $i + 1;  
} while ($i <= 7);
```

# foreach

```
foreach (array as $var) {  
    statement-to-be-repeated;  
}
```

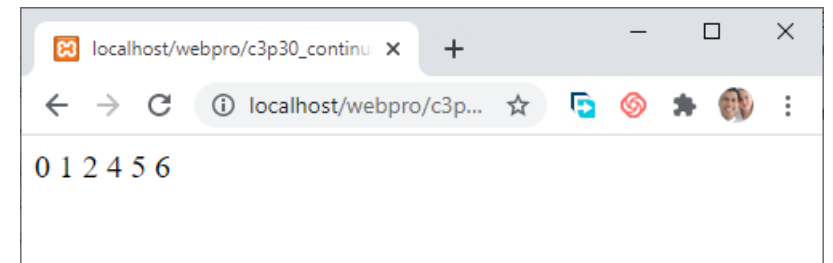
```
foreach ($_POST as $myInput) {  
    echo $myInput;  
}
```

# break & continue

- break → exits/ends a loop completely
- continue → shortcuts the current iteration & moves on to the next iteration

```
for ($i = 0; $i <= 10; $i++) {  
    if ($i == 3)  
        continue;  
    if ($i == 7)  
        break;  
    echo "$i ";  
}
```

```
1 <html>  
2 <body>  
3 <?php  
4     for ($i = 0; $i <= 10; $i++) {  
5         if ($i == 3) {  
6             continue;  
7         }  
8         if ($i == 7) {  
9             break;  
10        }  
11        echo "$i ";  
12    }  
13    ?>  
14 </body>  
15 </html>
```



# Form handling

- I/O

# Input form: the component

- `<FORM>` → tag
- `<ACTION>` → attribute
- `<METHOD>` → attribute
- **Submit** BUTTON



# Input form: the field

- Text
- Password
- Radio button
- Checkbox
- Combo box
- Text area

# <FORM> tag

- Define an input as a whole
- Multiple <FORM> are allowed in a page
- Many input fields are allowed in a <FORM> tag

# <ACTION> attribute

- Inside a <FORM> tag
- Define in which page an input form will be processed
- Can be filled by a particular page, e.g., my\_page.php, or an empty one ("")

# <METHOD> attribute

- Inside a <FORM> tag
- Define in how an input form will be processed
- It has 2 methods
  - POST
  - GET

# Submit BUTTON

- A trigger that a `<FORM>` will be processed
- Can be a button or any HTML component which is functioned as a button

# PHP form: the handling

- `$_POST` → for a form whose POST method
- `$_GET` → for a form whose GET method or for obtaining a variable from a URL
- `$_REQUEST` → for a form whose POST or GET methods

# Array

- An ordered map
- A special variable → hold more than one value a time

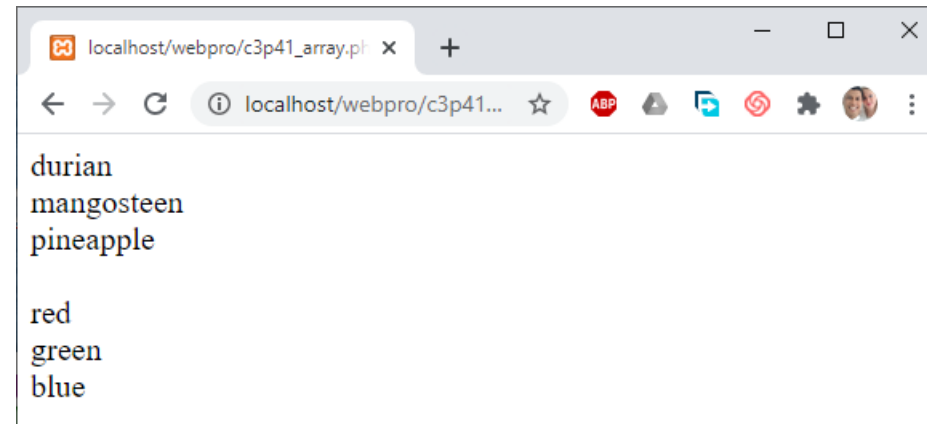
# Array: the explanation

- A structured data type → save the different types of data
- Element of array → building block of an array
- Array index → can be Integer or String



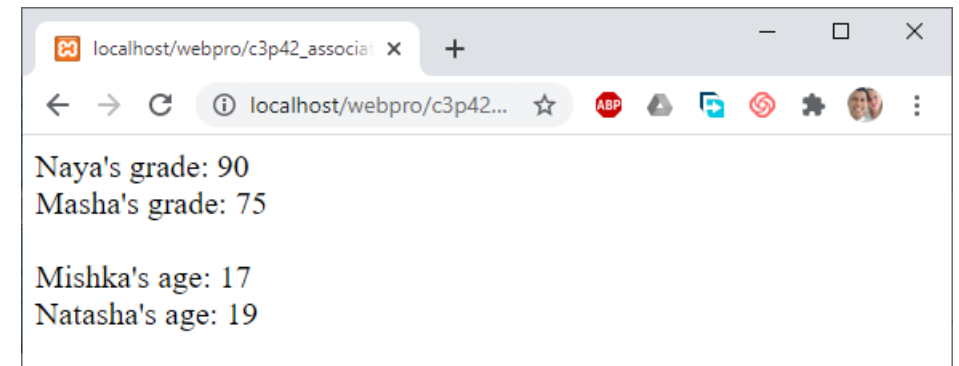
# Array: declaration & printing

```
1 <html>
2   <body>
3     <?php
4       $tropical_fruits = array("durian", "mango", "mangosteen", "pineapple");
5       echo $tropical_fruits[0]; echo "<br>"; // durian
6       echo $tropical_fruits[2]; echo "<br>"; // mangosteen
7       echo $tropical_fruits[3]; echo "<br>"; // pineapple
8       echo "<br>";
9       $myColour = array();
10      $myColour[] = "red"; // index 0
11      $myColour[] = "green"; // index 1
12      $myColour[] = "blue"; // index 2
13      echo $myColour[0]; echo "<br>"; // red
14      echo $myColour[1]; echo "<br>"; // green
15      echo $myColour[2]; echo "<br>"; // blue
16    ?>
17  </body>
18 </html>
```



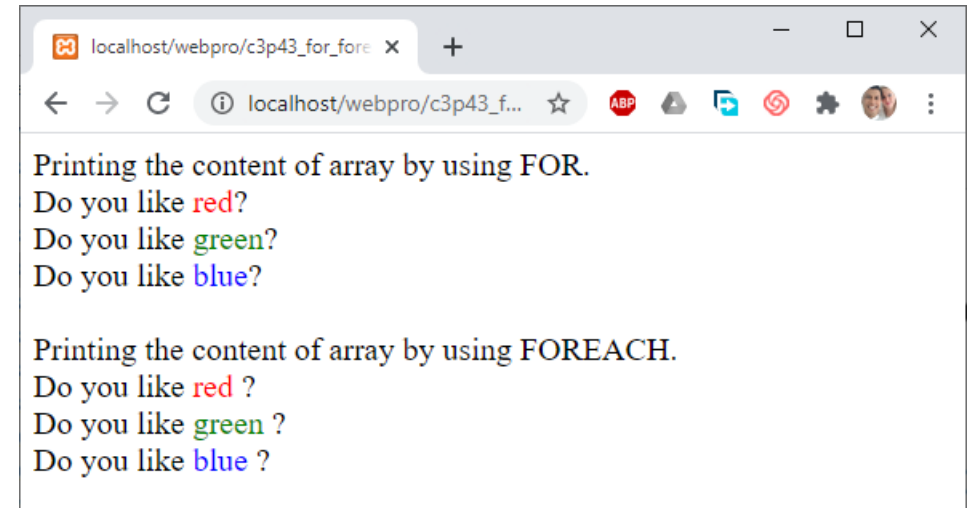
# Associative array: unordered index

```
1 <html>
2   <body>
3     <?php
4       // Associative array -> unordered index
5       $grade = array("Yarik" => 80,
6         "Naya" => 90,
7         "Masha" => 75,
8         "Ivan" => 85);
9       echo "Naya's grade: "; echo $grade['Naya']; echo "<br>"; // 90
10      echo "Masha's grade: "; echo $grade['Masha']; echo "<br>"; // 75
11      echo "<br>";
12      $age = array();
13      $age["Sergei"] = 18;
14      $age["Mishka"] = 17;
15      $age["Natasha"] = 19;
16      echo "Mishka's age: "; echo $age['Mishka']; echo "<br>"; // 17
17      echo "Natasha's age: "; echo $age['Natasha']; echo "<br>"; // 19
18    ?>
19  </body>
20 </html>
```



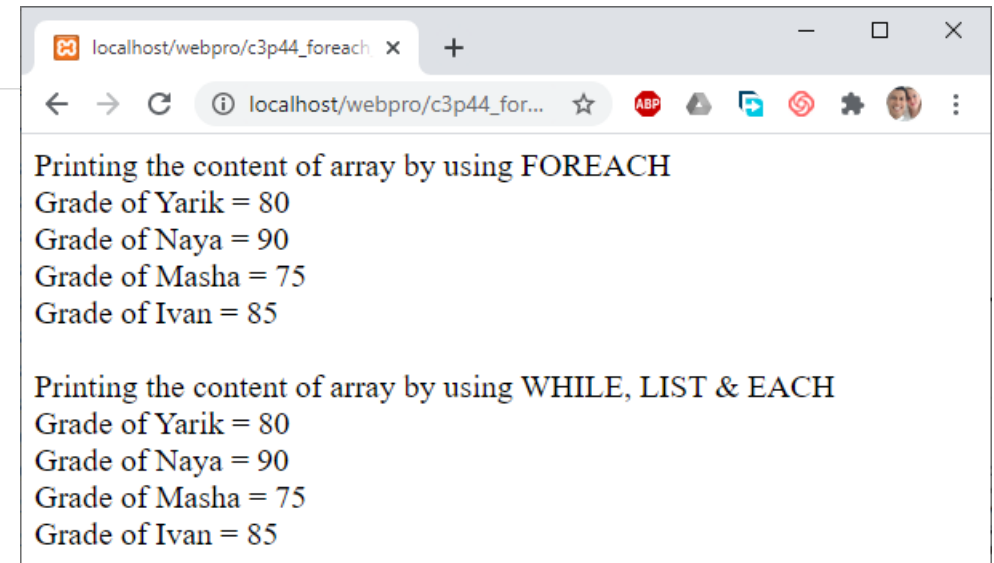
# Array printing: by using for & foreach

```
1 <html>
2   <body>
3     <?php
4       $colour = array();
5       $colour[] = "red";
6       $colour[] = "green";
7       $colour[] = "blue";
8       echo "Printing the content of array by using FOR.<br>";
9       FOR ($i = 0; $i < count($colour); $i++) {
10         echo "Do you like <font color = $colour[$i]>";
11         echo $colour[$i] . "</font>? <br>";
12       }
13       echo "<br>";
14       echo "Printing the content of array by using FOREACH.<br>";
15       FOREACH ($colour as $clr) {
16         echo "Do you like <font color = $clr>" . $clr . "</font> ?<br>";
17       }
18     ?>
19   </body>
20 </html>
```



# Array printing: while, list & each

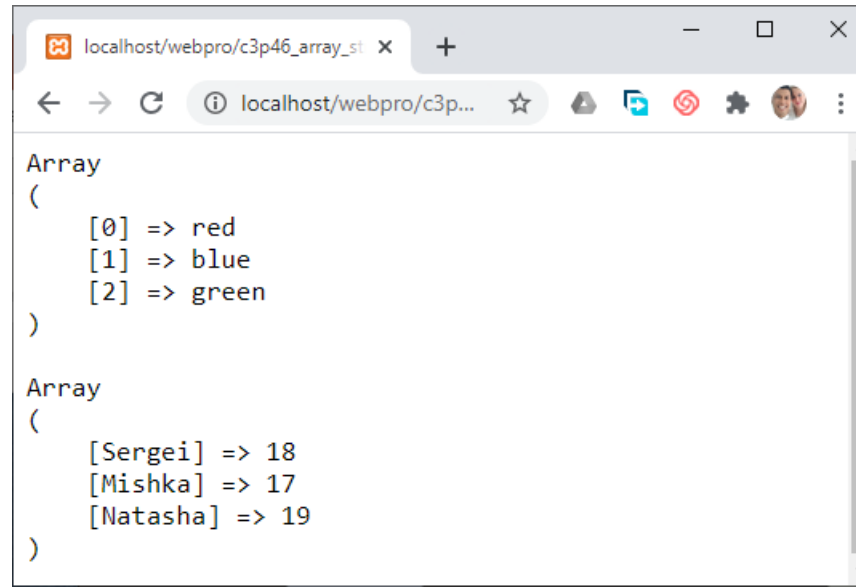
```
1 <html>
2   <body>
3     <?php
4       // Associative array -> unordered index
5       $grade = array("Yarik" => 80,
6         "Naya" => 90,
7         "Masha" => 75,
8         "Ivan" => 85);
9       echo "Printing the content of array by using FOREACH <br>";
10      FOREACH ($grade as $name => $grd) {
11        echo "Grade of $name = $grd<br>";
12      }
13      reset($grade);
14      echo "<br>";
15      echo "Printing the content of array by using WHILE, LIST & EACH<br>";
16      WHILE (LIST($name, $grd) = EACH($grade)) {
17        echo "Grade of $name = $grd<br>";
18      }
19    ?>
20  </body>
21 </html>
```



- The each() function is deprecated in PHP 7.2

# Array structure: printing

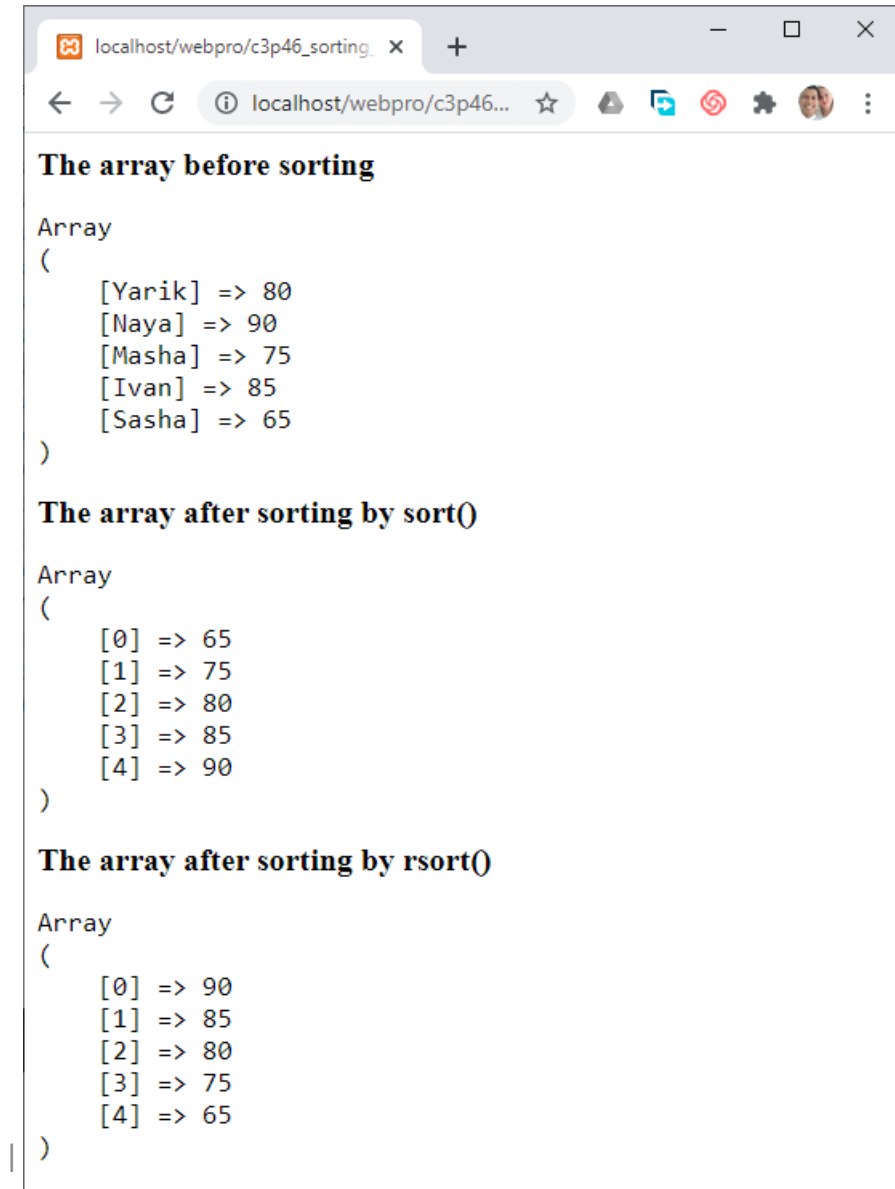
```
1 <html>
2   <body>
3     <?php
4       // Print out the array's structure
5       $colour = array("red", "blue", "green");
6       $age = array("Sergei" => 18,
7                   "Mishka" => 17,
8                   "Natasha" => 19);
9       echo "<pre>";
10      print_r($colour);
11      echo "<br>";
12      print_r($age);
13      echo "</pre>";
14    ?>
15  </body>
16 </html> |
```



# Array: sorting function

```
1 <html>
2   <body>
3     <?php
4       // Sorting array
5       $grade = array("Yarik" => 80,
6         "Naya" => 90,
7         "Masha" => 75,
8         "Ivan" => 85,
9         "Sasha" => 65);
10      echo "<b>The array before sorting</b>";
11      echo "<pre>";
12      print_r($grade);
13      echo "</pre>";
14      // Sort
15      sort($grade);
16      // Reset
17      reset($grade);
18      echo "<b>The array after sorting by sort()</b>";
19      echo "<pre>";
20      print_r($grade);
21      echo "</pre>";
22      // Reverse Sorting
23      rsort($grade);
24      // Reset
25      reset($grade);
26      echo "<b>The array after sorting by rsort()</b>";
27      echo "<pre>";
28      print_r($grade);
29      echo "</pre>";
30    ?>
31  </body>
32 </html>
```

15/09



```
The array before sorting
Array
(
    [Yarik] => 80
    [Naya] => 90
    [Masha] => 75
    [Ivan] => 85
    [Sasha] => 65
)

The array after sorting by sort()
Array
(
    [0] => 65
    [1] => 75
    [2] => 80
    [3] => 85
    [4] => 90
)

The array after sorting by rsort()
Array
(
    [0] => 90
    [1] => 85
    [2] => 80
    [3] => 75
    [4] => 65
)
```

ning |

# Function

- Procedure → function without return

# Function: without & with parameter

```
1 <html>
2   <body>
3     <?php
4       // Function: no parameter
5       function print_odd() {
6         for ($i = 0; $i < 100; $i++) {
7           if ($i % 2 == 1) {
8             echo "$i ";
9           }
10        }
11      }
12      // Call the function
13      echo "Call the function print_odd()<br>";
14      print_odd();
15      echo "<br>";
16      echo "<br>";
17      // Function: with the parameter
18      function print_odd2($begin, $end) {
19        for ($i = $begin; $i < $end; $i++) {
20          if ($i % 2 == 1) {
21            echo "$i ";
22          }
23        }
24      }
25
26      // Call the function
27      $a = 5;
28      $b = 70;
29      echo "Call the function print_odd() from $a to $b<br>";
30      print_odd2($a, $b);
31      echo "<br>";
32      echo "<br>";
33      // Function: with the return
34      function circleArea($radius) {
35        return 3.14 * $radius * $radius;
36      }
37      // Call the function
38      $r = 100;
39      echo "The area of a circle with the radius $r = ";
40      echo circleArea($r);
41    ?>
42  </body>
43 </html>
```

