Web Programming

Chapter 5: Php Part 2

Dr. Liew Siau Chuin
Faculty of Computer Systems
and Software Engineering
liewsc@ump.edu.my

PHP Data Types

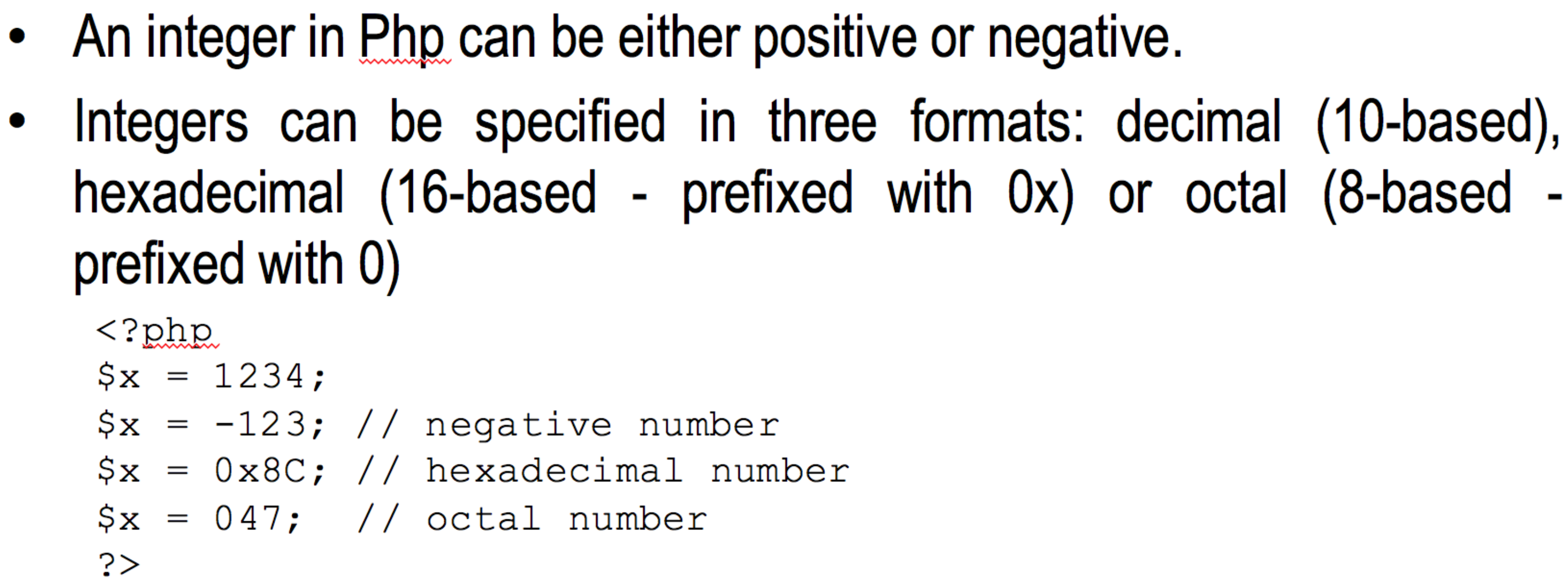
- PHP supports the following data types.
  - String
  - Integer
  - Floating point numbers
  - Boolean
  - Array
  - Object
  - NULL.

PHP Strings

- A string is a sequence of characters, like “I love PHP!”.
- A string can be any text inside a single or double quotes as shown below.

```
<?php  $x = “I love PHP!”;
echo $x;
echo "<br>";
$x = ‘I love PHP!';
echo $x;
?>
```

PHP Integer



- An integer in Php can be either positive or negative.
- Integers can be specified in three formats: decimal (10-based), hexadecimal (16-based - prefixed with 0x) or octal (8-based - prefixed with 0)

```
<?php
$x = 1234;
$x = -123; // negative number
$x = 0x8C; // hexadecimal number
$x = 047;  // octal number
?>
```

PHP Floating Point

- A floating point number is a number with a decimal point or a number in exponential form.

```
<?php
$x = 10.365;
$x = 2.4e3;
$x = 8E-5;
?>
```

PHP Boolean

- Booleans can be either TRUE or FALSE.

```
$x=true;
$y=false;
```

- Booleans are often used in conditional testing.

PHP Array

- An array stores multiple values in one single variable.

```
<?php
$fruit=array(”Apple",”Orange",”Pear");
echo "My fruit is a {$fruit[0]}";
?>
```

- Booleans are often used in conditional testing.

PHP Null Value

- NULL value represents a variable that has no value.
- Variables can be emptied by setting the value to NULL:

```
<?php
$x=”Good morning!";
$x=null;
?>
```

PHP String Function

- Php provides a number of string functions. One of the most commonly used are such as strlen() and strpos().
- The strlen() function returns the length of a string, in characters.

```
<?php
echo
strlen(”Good morning!");
?>
```

PHP String Function

- A specified character or text within a string can be searched using this function.
- If a match is found, it will return the character position of the first match. If no match is found, it will return FALSE.

```
<?php
echo
strpos(”Good morning!","morning");
?>
```

PHP Constant

- A constant is an identifier that it's value cannot be changed during the script.
- It should start with a letter or underscore (no $ sign before the constant name).
- Constants are automatically global across the entire script.
- Use the define() function - it takes three parameters:
  - The first parameter defines the name of the constant,
  - The second parameter defines the value of the constant,
  - The optional third parameter specifies whether the constant name should be case-insensitive. Default is false.

PHP Constant

- Case-sensitive constant

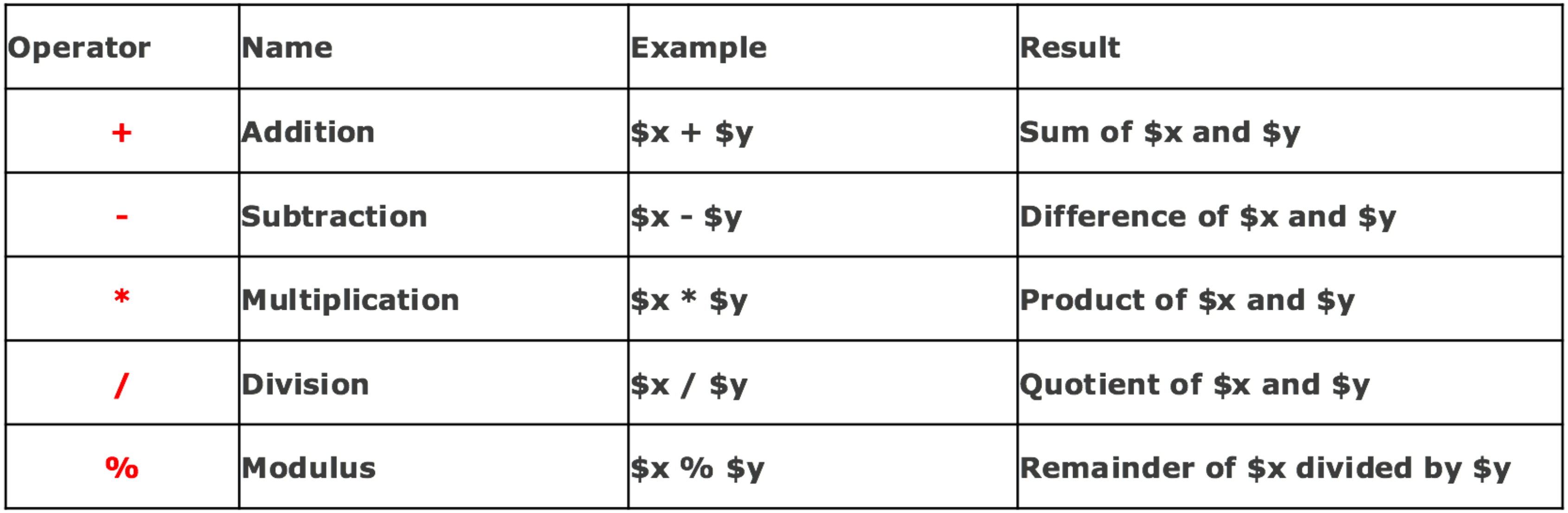
```
<?php
define("FRUIT", "Orange");
echo FRUIT;
?>
```

- Case-insensitive constant

```
<?php
define("FRUIT", "Orange",true);
echo fruit;
?>
```

Php Operators

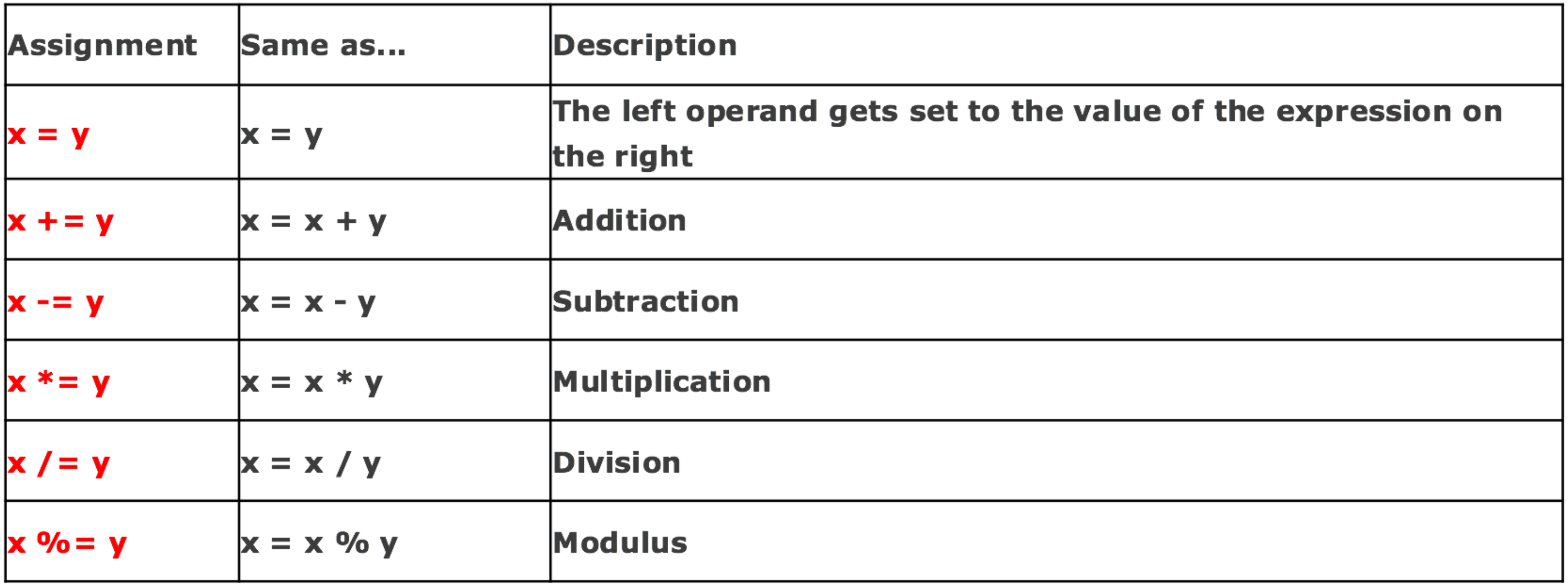
PHP Arithmetic Operators



Operator	Name	Example	Result
+	Addition	$x + $y	Sum of $x and $y
-	Subtraction	$x - $y	Difference of $x and $y
*	Multiplication	$x * $y	Product of $x and $y
/	Division	$x / $y	Quotient of $x and $y
%	Modulus	$x % $y	Remainder of $x divided by $y

Php Operators

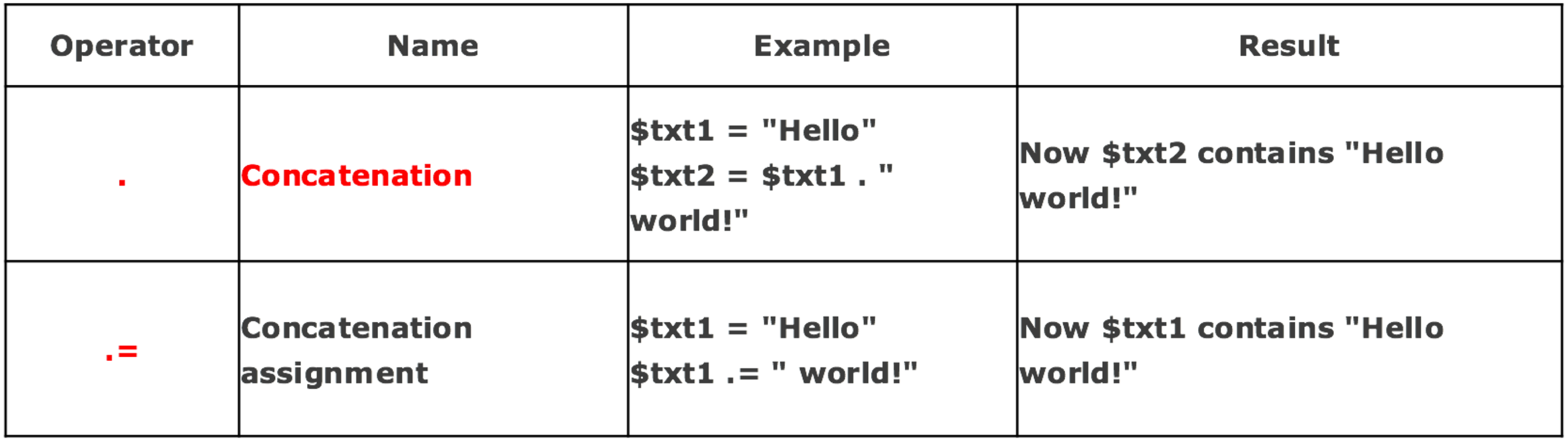
PHP Assignment Operators



Assignment	Same as...	Description
x = y	x = y	The left operand gets set to the value of the expression on the right
x += y	x = x + y	Addition
x -= y	x = x - y	Subtraction
x *= y	x = x * y	Multiplication
x /= y	x = x / y	Division
x %= y	x = x % y	Modulus

Php Operators

PHP String Operators



Operator	Name	Example	Result
.	Concatenation	$txt1 = "Hello" $txt2 = $txt1 . " world!"	Now $txt2 contains "Hello world!"
.=	Concatenation assignment	$txt1 = "Hello" $txt1 .= " world!"	Now $txt1 contains "Hello world!"

Php Operators

PHP Increment / Decrement Operators

Operator	Name	Description
++$x	Pre-increment	Increments $x by one, then returns $x
$x++	Post-increment	Returns $x, then increments $x by one
--$x	Pre-decrement	Decrements $x by one, then returns $x
$x--	Post-decrement	Returns $x, then decrements $x by one

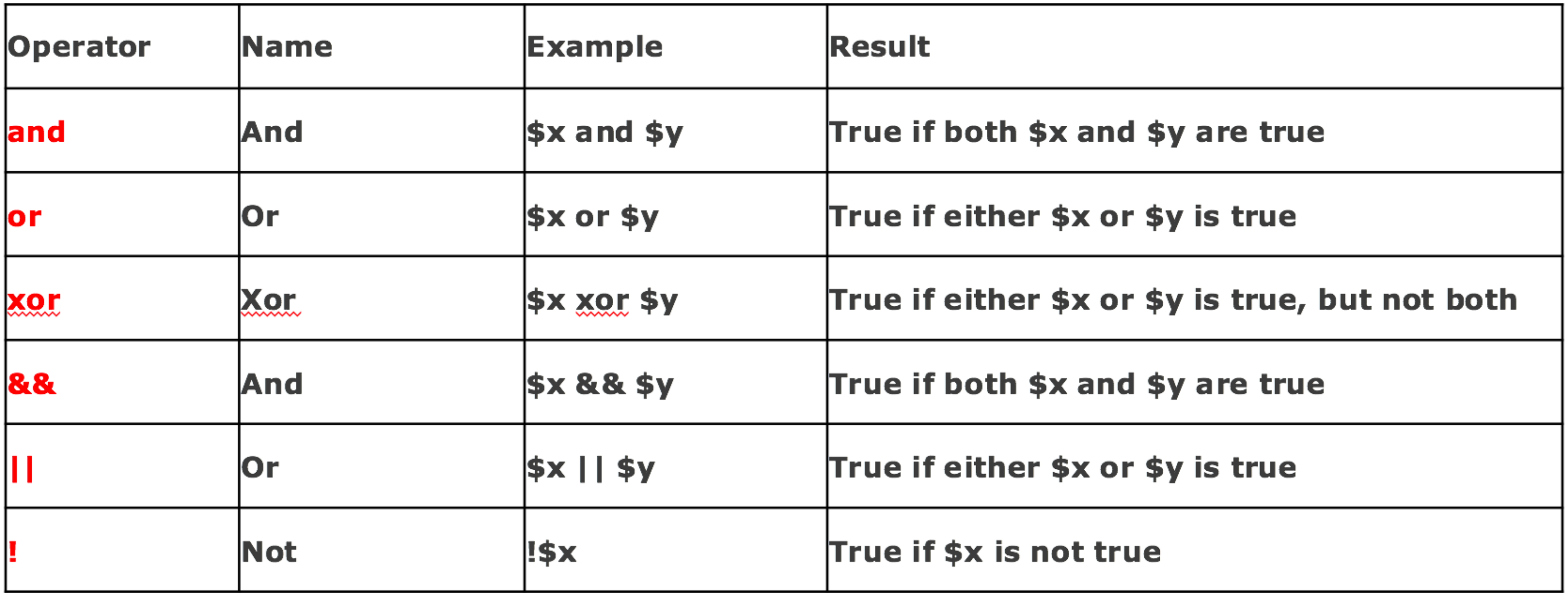
Php Operators

PHP Comparison Operators

Operator	Name	Example	Result
==	Equal	$x == $y	True if $x is equal to $y
===	Identical	$x === $y	True if $x is equal to $y, and they are of the same type
!=	Not equal	$x != $y	True if $x is not equal to $y
<>	Not equal	$x <> $y	True if $x is not equal to $y
!==	Not identical	$x !== $y	True if $x is not equal to $y, or they are not of the same type
>	Greater than	$x > $y	True if $x is greater than $y
<	Less than	$x < $y	True if $x is less than $y
>=	Greater than or equal to	$x >= $y	True if $x is greater than or equal to $y
<=	Less than or equal to	$x <= $y	True if $x is less than or equal to $y

Php Operators

PHP Logical Operators



Operator	Name	Example	Result
and	And	$x and $y	True if both $x and $y are true
or	Or	$x or $y	True if either $x or $y is true
xor	Xor	$x xor $y	True if either $x or $y is true, but not both
&&	And	$x && $y	True if both $x and $y are true
||	Or	$x || $y	True if either $x or $y is true
!	Not	!$x	True if $x is not true

Php Operators

PHP Array Operators

Operator	Name	Example	Result
+	Union	$x + $y	Union of $x and $y (but duplicate keys are not overwritten)
==	Equality	$x == $y	True if $x and $y have the same key/value pairs
===	Identity	$x === $y	True if $x and $y have the same key/value pairs in the same order and of the same types
!=	Inequality	$x != $y	True if $x is not equal to $y
<>	Inequality	$x <> $y	True if $x is not equal to $y
!==	Non-identity	$x !== $y	True if $x is not identical to $y

PHP Conditional Statements

- Conditional statements are used to perform different actions based on different conditions.
- In PHP we have the following conditional statements:
  - if statement
  - if...else statement
  - if...elseif....else statement
  - switch statement

PHP Conditional Statements

The if Statement

- The if statement is used to execute a portion of code only if a specified condition is true.
- Syntax

```
if (condition)
{
   code to be executed if condition is true;
}
```

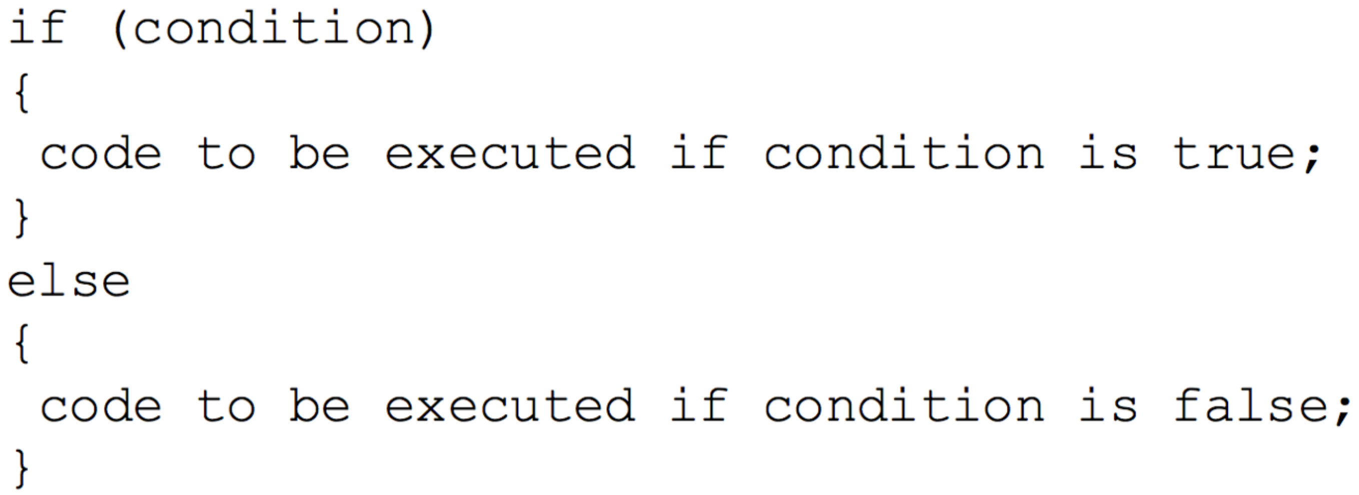
- Example

```
<?php
if ($kg<1)
{  echo ”It is an apple!";   }
?>
```

PHP Conditional Statements

The if...else Statement

- It is used to execute some code if a condition is true and another code if the condition is false.
- Syntax



```
if (condition)
{
 code to be executed if condition is true;
}
else
{
 code to be executed if condition is false;
}
```

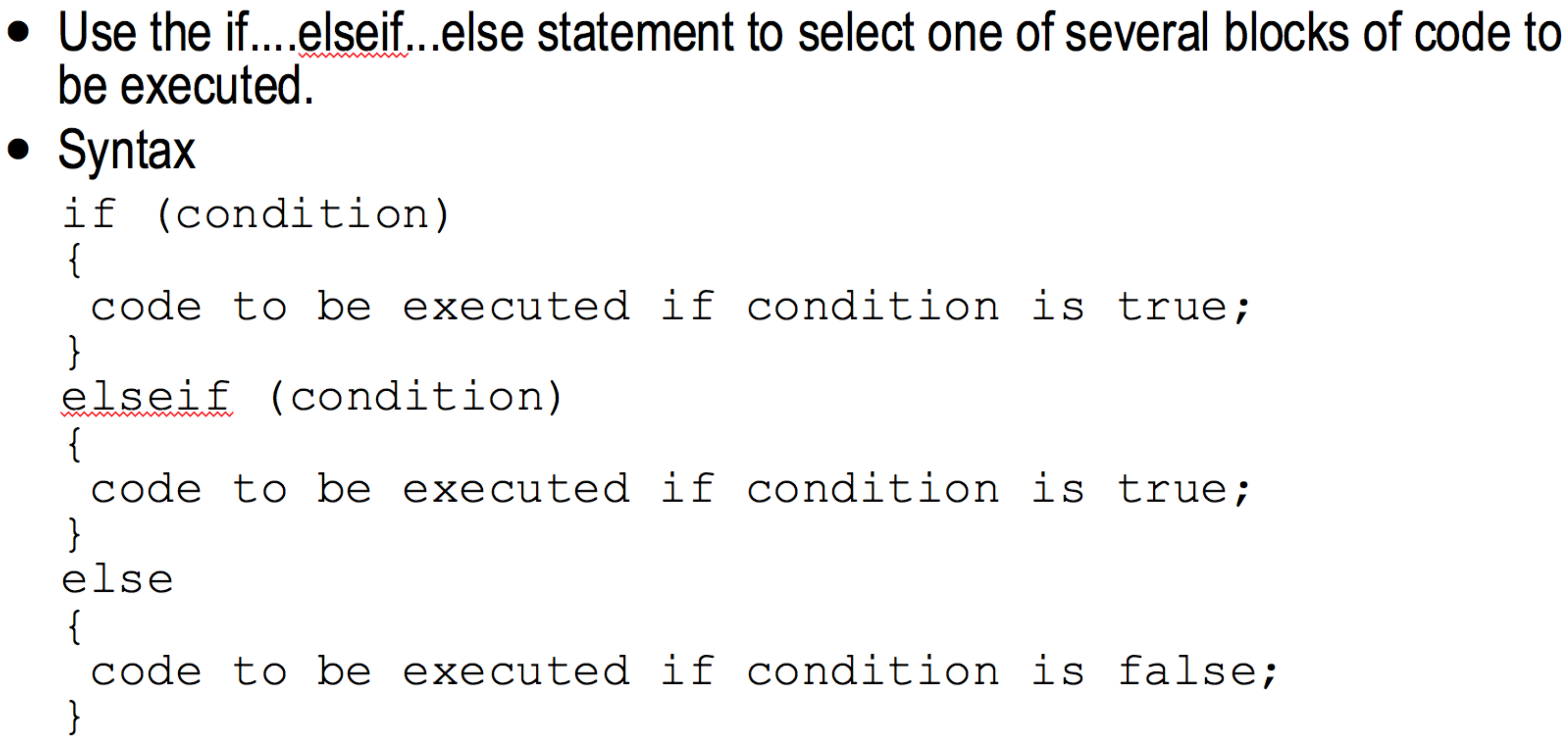
PHP Conditional Statements

```php
<?php

if ($kg<1)
  {
  echo ”It is an apple!";
  }
else
  {
  echo " It is a watermelon!";
  }
?>
```

PHP Conditional Statements

The if...elseif....else Statement



- Use the if....elseif...else statement to select one of several blocks of code to be executed.
- Syntax

```
if (condition)
{
 code to be executed if condition is true;
}
elseif (condition)
{
 code to be executed if condition is true;
}
else
{
 code to be executed if condition is false;
}
```

PHP Conditional Statements

```
<?php

if ($kg<1)
  {
  echo ”It is an apple!";
  }
elseif($kg<1.5)
  {
  echo " It is a mango!";
  }
else
  {
  echo " It is a watermelon!";
  }
?>
```

PHP Conditional Statements

The switch Statement

- The switch statement is used to select one of many blocks of code to be executed.
- Syntax

```
switch (n)     //n, most often a variable, is evaluated once
{
 case label1:
 code to be executed if n=label1;
 break;
 case label2:
 code to be executed if n=label2;
 break;
 case label3:
 code to be executed if n=label3;
 break;
 ...
 default:
 code to be executed if n is different from all
 labels;
}
```

PHP Conditional Statements

```
<?php
$favfruit= "orange";
switch ($favfruit)
{
case "apple":
  echo "Your favorite fruit is apple!";
  break;
case "watermelon":
  echo "Your favorite fruit is watermelon!";
  break;
case "orange":
  echo "Your favorite fruit is orange!";
  break;
default:
  echo "Your favorite fruit is neither apple,watermelon or orange !";
}
?>
```

PHP Loop

- In PHP, we have the following looping statements:
  - while
  - do...while
  - for
  - foreach

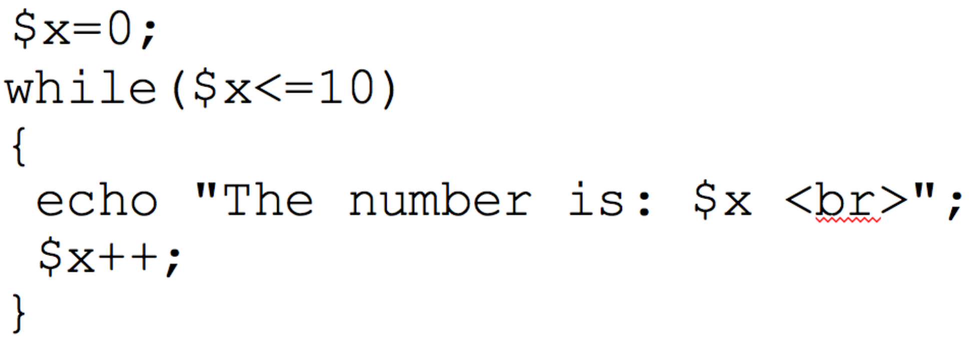
PHP Loop

PHP while Loop

- Executes a block of code as long as the specified condition is true.
- Syntax

```
while (condition is true)
{
 code to be executed;
}
```

- Example



```
 $x=0;
while($x<=10)
{
 echo "The number is: $x <br>";
 $x++;
}
```

PHP Loop

PHP do...while Loop

- The do...while loop will always execute the block of code once, it will then check the condition, and repeat the loop while the specified condition is true.
- Syntax

```
do
 {
  code to be executed;
 }
 while (condition is true);
```

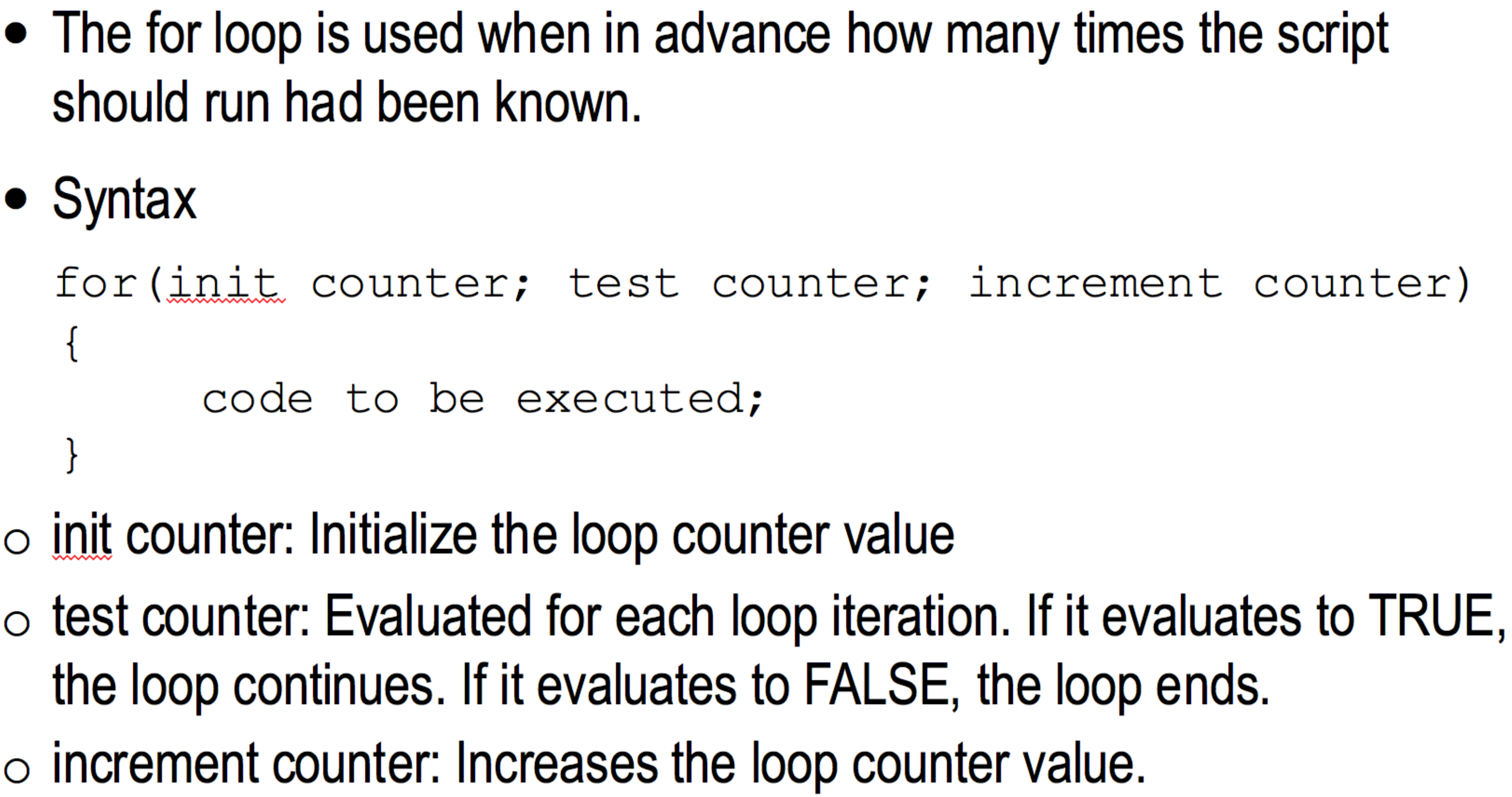
PHP Loop

- The do while loop would execute its statements at least once, even if the condition fails the first time.

-
  ```
  <?php
  $x=4;
  do
     {
     echo "The number is: $x <br>";
     $x++;
     }
  while ($x<=6)
  ?>
  ```

PHP Loop

PHP for Loop



- The for loop is used when in advance how many times the script should run had been known.
- Syntax

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

- init counter: Initialize the loop counter value
- test counter: Evaluated for each loop iteration. If it evaluates to TRUE, the loop continues. If it evaluates to FALSE, the loop ends.
- increment counter: Increases the loop counter value.

PHP Loop

- Example

```
<?php
for ($x=0; $x<=99; $x++)
{
  echo "The number is: $x <br>";
}
?>
```

PHP Loop

PHP foreach Loop

- The foreach loop works only on arrays, and is used to loop through each key/value pair in an array.
- Syntax

```
foreach ($array as $value)
{
   code to be executed;
}
```

- For every loop iteration, the value of the current array element is assigned to $value and the array pointer is moved by one, until it reaches the last array element.

PHP Loop

- Example

```
<?php
$fruits = array("apple", "orange",”watermelon");
foreach ($fruits as $value)
{
   echo "$value <br>";
}
?>
```

PHP Function

- A function is a block of statements that can be used repeatedly in a program.
- Include built-in functions and user defined functions.
- Php has more than 1000 built-in functions.

PHP Function

PHP User Defined Functions

- A user defined function declaration starts with the word "function":
- Syntax

```
function functionName()
{
    code to be executed;
}
```

- A function will not execute immediately when a page loads.
- A function will be executed by a call to the function.
- Example in the next slide.

PHP Function

```
<?php
   function writeMsg()
   {
     echo ”Good morning!";
   }

   writeMsg(); // call the function
?>
```

Activities

- Please refer to Tutorial 8 for more Php exercises.