Read this first:

The control statement can be an assignment, a function call, a loop, a conditional statement or even a statement that does nothing or an empty statement.

Constant

A constant is an identifier (name) for a simple value. The value cannot be changed during the script.A valid constant name starts with a letter or underscore **without $ sign.**

Constants are automatically global across the entire script. There is no any keyword to define constant, define() is used to declare constant.

```
define(name, value, booleancase-insensitive)
```

name: *Specifies the name of the constant*

value: *Specifies the value of the constant*

case-insensitive: *Specifies whether the constant name should be case-insensitive. Default is false*

Operator

Operators are used to perform operations on variables and values.

PHP divides the operators in the following groups:

- Arithmetic operators
- Assignment operators
- Comparison operators
- Increment/Decrement operators
- Logical operators
- String operators
- Array operators

1. Arithmetic Operator used for arithmetic operations

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
**	Exponentiation	$x ** $y	Result of raising $x to the $y'th power

2. Assignment Operator

Operator	Meaning	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

3. Comparison Operator

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

4. Increment/Decrement Operator

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

5. String Operator

Operator	Name	Example	Result
.	Concatenation	$txt1 . $txt2	Concatenation of $txt1 and $txt2
.=	Concatenation assignment	$txt1 .= $txt2	Appends $txt2 to $txt1

6. Logical Operator

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

7. Ternary Operator

When we would like to assign one of two values to any variable based on any condition ternary operator is best option to shorten the code of if….else block.

```
$var = condition ? true : no;
```

Self Assessment	

<?php
$X = (2 + 3) * 2 + 3;
echo $X;
?>
	<?php
	$X = 12;
	function demo(){
	        $X = "Hello World";
	        global $X;
	        $Y = 13;
	        $Y++;
	        $Y++;
	        $Y++;
	        static $Z = 13;
	        $Z++;
	        echo "Value of X: ",$X,"
	";
	        echo "Value of Y: ",$Y,"
	";
	        echo "Value of Z: ",$Z,"
	";
	        }
	
	demo();
	demo();
	demo();
	?>

Output:	Output:

Conditional Statements

When you want to perform certain action on when certain conditions are true we used conditional statements. In PHP, if, if…else, if…elseif…elseif…else and switch are condition statements.

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

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

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

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

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

else {
    code to be executed if all conditions are false;
}
```

```
switch (n) {
    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;
}
```

Looping in PHP

When you need to execute a block of the code till the condition is true we use looping statements. In PHP- while, do….while, for and foreach loops are available.

- **while** - loops through a block of code as long as the specified condition is true

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

- **do...while** - loops through a block of code once, and then repeats the loop as long as the specified condition is true

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

- **for** - loops through a block of code a specified number of times

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

- **foreach** - loops through a block of code for each element in an array

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

Example of foreach:

```
<?php
$colors = array("red", "green", "blue", "yellow");

foreach ($colors as $value) {
    echo "$value <br>";
}
?>
```

Output:

Function

A function is a block or unit of statements that can be used to perform specific task. Generally, functions are used to avoid duplication of code and to improve code manageability.

Codes that are within the function are executed when function is called.

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

Functions can have three variant:

- Function with no parameter and not returning value
- Function with parameter and not returning value
- Function with parameter and returning value

Function with no parameter and not returning value

```php
<?php
    function fun1() {
    echo "Hello world!";
}

fun1(); // call the function
?>
```

Function with parameter and not returning value

```php
<?php
    function sum($no1, $no2) {
    echo $no1 + $no2;

}

sum(23,34); // call the function

sum(11,23); // call the function

sum(34,100.50); // call the function

?>
```

Function with parameter and returning value

```php
 <?php
     function sum($no1, $no2) {
     $ans = $no1 + $no2;

 }

 echo "Result: ".sum(23,34); // call the function

 echo "Result: ".sum(11,50); // call the function

 echo "Result: ".sum(99.5,1); // call the function

?>
```

Lab Assignment

1. Write a PHP script to create a function to display first 100 prime numbers.