Web Development with PHP
Practical - 2
PHP Variables & Loops

PHP *echo* and *print* functions

Echo and Print methods of PHP are not built-in functions of PHP language, but they are language constructs. A language construct is accepted/executed by the PHP parser as it is, in other words the PHP parser doesn't have to parse and modify a language construct to execute it, as it is ready for execution by default. Hence, language constructs are faster than any built-in functions.

echo() function is used to print or output one or more strings. It is specifically mentioned string here because, the syntax of the echo function is:

```
echo(string)
```

Although you can use echo() function to output anything, a PHP parser will automatically convert it into string type. echo doesn't need parenthesis, although you can use parenthesis if you want.

```
<?php
    echo "I am open";
    echo ("I am enclosed in parenthesis");

    echo 'This','is','a','broken','sentence';

    echo "This is a
     multiline sentence
     example";

    $str = "I am a string variable";
    echo $str;

    echo "Hello, this is a \"beautiful\" picture";

    $weird = "Stupid";
    echo "I am $weird";
    echo 'I am $weird';
?>
```

When using double quotes the value of the string variable gets printed, while using single quotes, the variable is printed as it is.

The PHP print is exactly the same as echo, with the same syntax and same usage. Replace echo with print in all the above examples.

PHP Constants

Constants are variables whose value cannot be changed. In other words, once you set a value for a constant, you cannot change it. In PHP, there are two ways to define a constant:

- Using the define() method.

  It defines a constant at runtime.

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

  - ➔ name: Name of the constant
  - ➔ value: Value of the constant
  - ➔ case-insensitive: Specifies whether the constant name is case sensitive or not. Its default value is false, which means, by default, the constant name is case sensitive.

    ```
    <?php
        define(OMG, "Oh! my God.");
        echo OMG;
    ?>
    <?php
        define(OMG, "Oh! my God.");
        echo omg;
    ?>
    <?php
        define(OMG, "Oh! my God.", true);
        echo omg;
    ?>
    ```

- Using the const keyword.

  We can also define constants in PHP using the const keyword. But we can only use the const keyword to define scalar constants, i.e. only integers, floats, booleans and strings, while define() can be used to define array and resource constants as well. When we define a constant using the const keyword, the constant name is always case sensitive.

  The const keyword defines constants at compile time. It is a language construct, not a function.

  ```
  <?php

      const OMG = "Oh! my God.";

      echo OMG;

  ?>
  ```

Another very important point to remember is that while naming a constant, we don't have to use $ symbol with the constant's name.

PHP Data Types

PHP data types are used to hold different types of data or values. PHP supports 8 primitive data types that can be categorized further in 3 types:

- ***Scalar Types***
    - It holds only a single value. There are 4 scalar data types in PHP.
        - ➔ **boolean** - A boolean data type can have two possible values, either True or False.
        - ➔ **integer** - An integer value can be negative or positive, but it cannot have a decimal.
        - ➔ **float** - Float data type is used to store any decimal numeric value. A float value can also be either negative or positive.
        - ➔ **string** - String data type in PHP and in general, is a sequence of characters(or anything, it can be numbers and special characters too) enclosed within quotes. You can use single or double quotes.
- ***Compound Types***
    - It can hold multiple values. There are 2 compound data types in PHP.
        - ➔ **array** - An array is used to store multiple values, generally of the same type, in a single variable.
        - ➔ **object** - An object is an instance of the class which holds the local variables with values assigned and using the object we can call the local methods defined in the class.
- ***Special Types***
    - There are 2 special data types in PHP.
        - ➔ **resource** - A resource is a special variable, holding a reference to an external resource. It typically holds special handlers to open files and database connections.
        - ➔ **NULL** - NULL data type is a special data type which means nothing. It can only have one value, and that is NULL.

PHP - Scope of the variable

- Local
  - The variables that are declared within a function are called local variables for that function. These local variables have their scope only in that particular function in which they are declared. This means that these variables cannot be accessed outside the function, as they have local scope. Hence a variable declaration outside the function with the same name is completely different from the variable declared inside the function.

```
<?php
    function mytest()
    {
        $lang = "PHP";
        echo "Local Scope in  " .$lang;
    }
    mytest();
     //using $lang (local Var) outside the function
     //will generate an error
    echo $lang;
?>
```

- Global
  - The global variables are the variables that are declared outside the function. These variables can be accessed anywhere in the program. To access the global variable within a function, use the GLOBAL keyword before the variable. However, these variables can be directly accessed or used outside the function without any keyword. Therefore there is no need to use any keyword to access a global variable outside the function.

```
<?php
    $name = "PHP - Global Scope";     //Global Variable
    function global_var()
    {
        global $name;
        echo "Variable inside the function: ". $name;
        echo "</br>";
    }
    global_var();
    echo "Variable outside the function: ". $name;
?>
```

  - Another way to use the global variable inside the function is the predefined $GLOBALS array.

```
<?php
    $num1 = 55;      //global variable
    $num2 = 3;     //global variable
    function global_var()
    {
            $sum = $GLOBALS['num1'] + $GLOBALS['num2'];
            echo "Sum of global variables is: " .$sum;
    }
    global_var();
?>
```

- Static
    - It is a feature of PHP to delete the variable, once it completes its execution and memory is freed. Sometimes we need to store a variable even after completion of function execution. Therefore, another important feature of variable scoping is static variables. We use the static keyword before the variable to define a variable, and this variable is called a static variable. Static variables exist only in a local function, but it does not free its memory after the program execution leaves the scope.

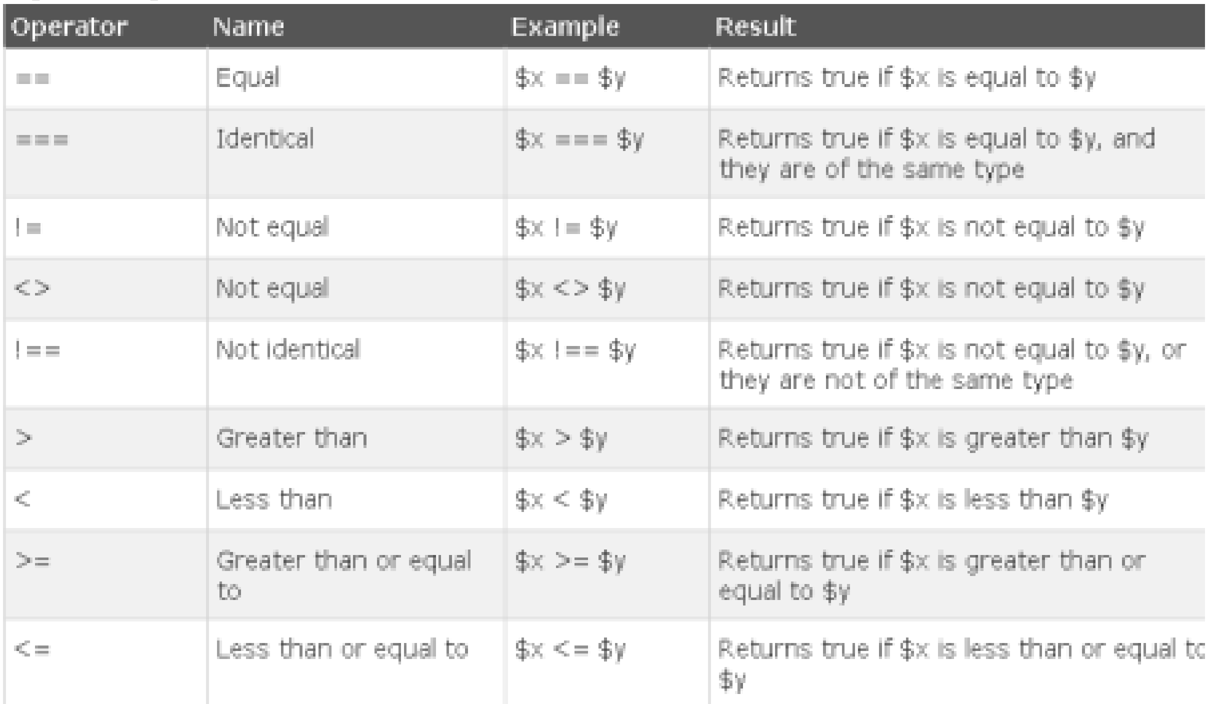
```
<?php
    function static_var()
    {
        static $num1 = 0;       //static variable
        $num2 = 100;          //Non-static variable
        //increment in non-static variable
        $num1++;
        //increment in static variable
        $num2++;
        echo "Static: " .$num1 ."</br>";
        echo "Non-static: " .$num2 ."</br>";
    }

     static_var();
     static_var();
     static_var();
     static_var();
     static_var();
?>
```

Operators

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

- Comparison operators



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

```
e.g
 <?php
 /*
 $x = 25;
 $y = 35;
 $z = "25";
 echo "<p> content from php file</p> <br>";
 echo "x==z is " . var_dump($x == $z)." <br>"; // Outputs: boolean true
?>
```

• var_dump($var) : gives the details about the datatype of variable • gettype($var) : return the datatype.

• settype($var) : change datatype of variable.

Conditional Loop

- **Simple if**

```
if(test-expr)
{
True part
}
statement
```

If...else statement

```
if(test-expr)
{
True part
}
else
{
False part
}
```

else if ladder :-

```
if(test-exp 1)
statement -1;
else if(test-exp 2)
 statement –
2;
 else if(test-exp 3)
 statement – 3;  .
 .
 .
 else
 false statement
```

switch statement :-

```
  switch(ch)
  {
        case value1:
           statement
  1;break;
        case value2:
           statement
  2;break;
        case value3:
           statement
  3; break;
   .
   .
   .
        default :
           statement
 false; break;
  }
```

- **While Loop**

```
while (expr) statement {
 statements; ….
}
```

- **Do while loop**

```
do{
 statements;
}while(expr);
```

for loop :-

```
for(intialization; condition; incr/decr)
{
        statement;
}
```

Exercise :

1. Create two variables of integer type and display the value of the variables.
2. Create a php script which declares all the scalar data types of variables and displays the value .
3. Create a php script which changes the data types using appropriate values. And display the new data types (integer to string , string to integer, Boolean to integer, integer to float, float to integer)
4. Create php script which declare two variable and perform all arithmetic operations on the variables.
5. Create a php script which shows all the comparison operators working.
6. Create a php script which performs string operations.
7. Create a php script which demonstrates the scope of local variables. {display the appropriate error also in output by making changes into script)
8. Create a php script which declares two global variables and demonstrates the scope of the global variable . {display the appropriate error also in output by making changes into script)
9. Create a php script which demonstrates the working and scope of static variables. (hint create function having static variable)
10. Create a php script which prints the first ten odd numbers.
11. Create a php script which checks whether the number is palindrome or not. E.d 121 is a palindrome number.
12. Create a php script which print the reverse of the number. E.g 123 => 321
13. Create a php script for given conditions.
    If age < 20 or age > 60 print message not valid age
    If age is between 20 to 35 , print message age is in range of 20 to 35
    If age is between 36 to 55 print message age is in range of 36 to 55

    Else print message age is more than 55
14. Create the php script which changes the background color of the div tag according to variable value. (hint `<div style="background-color:yellow;">`)