1 Expressions and control statements in PHP

- Introduction:

- A script is a set of programming instructions that is interpreted at run time.
- PHP is a general purpose programming language originally designed for web application development.
- PHP is a recursive acronym for " PHP: Hypertext Preprocessor".
- PHP is an open-source, server side programming language.
- PHP files have extension ".php".
- It is used to manage dynamic content, databases, session tracking, even build entire e-commerce sites.

- Features of PHP:

- Web Application Features
- CLI (Command Line Interface)
- Built-in Module
- Object-Oriented Programming
- Pre-compilation
- Real Time Access Monitoring
- File I/O

- History:

- PHP was written in the C programming language by Rasmus Lerdorf in 1994.
- PHP originally stood for " Personal Home Page".
- The acronym was formally changed to ' PHP: Hypertext Preprocessor".

Scanned by TapScanner

- Advantages:
  - Speed
  - Easy to use/ Simplicity
  - Stable
  - Platform Independent/ Portability
  - Open Source
  - Built-in Database Connection Modules
  - Powerful Library support
  - Flexibility
  - Database Connectivity
  - Speed-up Custom web Application development.

- Disadvantages:
  - Security
  - Weak Type
  - Not suitable for Large Web Applications
  - Desktop Applications
  - Modification Problem

- Syntax of PHP:
  - A PHP programming script starts with <? php and ends with ?>.
  - Syntax:

```
<? php
    // PHP code
?>
```

  - The default file extension for PHP files is ".php".
  - A PHP script can be placed anywhere in the document.

Scanned by TapScanner

- Way to write PHP code or Syntax:
i) SGML style : <? php code.... ?>
ii) ASP style : <% php code..... %>
iii) Script style :
<script language="php"> PHP code.... </script>

- Variables :
- PHP variables are nothing but a name for storage locations in the memory.
- A variable can be used to store data values.
- Values can be of any type.
- Variables in PHP are identifiers prefixed with a dollar sign ($).
- For example ;

```
$ name = "Suraj";
$ num = 29;
```

- Variable Declaration :
- A variable is a name for piece of data stored in the memory during the program execution.
- A variable can start with a '$' sign, followed by the name of the variable.
- Syntax :

```
$ variable_name;
```

- for example,

```
$ name;
```

Rules for declaration of a variable :
i) A variable must be start with a '$' sign, followed by the name of the variable.

Scanned by TapScanner

ii) A variable name must be start with a letter or an underscore (_).

iii) A variable name cannot start with a number.

iv) A variable name can only contain alpha-numeric characters and underscores.

v) Variable names are case-sensitive.

vi) PHP variable can be of any length and does not contain white spaces.

- Assigning Values to a Variable:
  - A variable can store any type value.
  - A variable is automatically declared, when we assign a value to it.
  - A value can be assigned to a variable by using an assignment operator (=).
  - The variable is in the left-hand side of assignment operator & value is on the right hand side.
  - Syntax:

```
$variable_name = value;
```

  - for example,

```
$name = "suraj";
```

- Note: We can unset the variable by using unset() function but the reference is still set.

- Variable scope:
  - Scope of a variable is an area of program in which it is accessible/visible.
  - We can declare variables everywhere in the script.

Scanned by TapScanner

- There are four types of scope in PHP:

i) Local Scope:
- A variable which is declared inside the function is called as local variable.
- Local variable has access only within the function.
- Local variable cannot be accessed from outside the function.
- Example:

```
<? php
    $a = 4;
    function assign ()
    {
        $ a = 0; || Local variable
        echo $a;
    }
    assign();
?>
```

ii) Global Scope:
- Variables declared outside function are by default global variables and can accessed from any part of program.
- A global variable can be accessed in any part of a program except inside the function.
- Inside the function, these variables are accessed using the 'GLOBAL' keyword.
- Example:

```
< ? php
    $x = 15;
```

Scanned by TapScanner

```
$y = 20;
function addit()
{
    GLOBAL $x; // Global Variable
    $y = 10;
    $x++;
    $y++;
    printf "x = $x & y = $y";
}
addit();
?>
```

iii) Static Variable:

- In PHP, a static variable is a variable that keeps its value across multiple function calls.
- Static variables are used only within the functions.
- We can create a static variable simply by placing 'static' infront of the variable name.
- Example,

```
<? php
 //STATIC
 function keep-track()
 {
     STATIC $count = 0;
     $count ++;
     echo $count . "<br>";
 }
 keep-track();
 keep-track();
?>
```

Scanned by TapScanner

iv) Function Parameter:

- Function parameters are local, i.e. they are available only inside the function.
- Function parameters are declared after the function name and inside parentheses.
- Example,

```
<? php
    function multiply ($value)
    {
        $value = $value * 4;
        return $value;
    }
    $ retval = multiply (10);
    echo " Return value is $retval";
?>
```

- Datatypes:

- Datatype is defined as, a set of values and the allowable operations on those values.
- Datatype means the type of a value.
- PHP support eight primitive types:

- Scalar Types: Integer, Float, string, Boolean.
- Compound Types: Array and object.
- Special Types: Resource and NULL.

1) Integer:

- The integer datatype is used to store a numeric value without decimal point (.).
- PHP does not support unsigned integers.
- In PHP, is-int() or is-integer() are used to test whether a value is an integer.
- Example;

$a = 25 , $ b = 101;

Scanned by TapScanner

2) Floating Point Numbers:
- Floating point numbers are real numbers.
- It contains a dot (.).
- is_float() or is_real() are used to test whether a value is a point number or not.
- Example,

```
$a = 10.5 , $b = 59.6;
```

3) Strings:
- A string is a sequence of characters.
- String can be represented either in single or double quotes.
- Variables are expanded within double quotes not within single quotes.
- Use is_string() function to test whether the value is string or not.
- Example,

```
$name = "Suraj";
```

4) Boolean:
- Boolean value can be either TRUE or FALSE value
- Use is_bool() function to test whether value is boolean or not.
- Example,

```
$ isMarked = true;
```

5) Arrays:
- Array is used to store multiple values on a single variable.
- PHP array is a collection of the different types of values.

Scanned by TapScanner

- Arrays in PHP can be indexed numerically or associatively.
- Numerically indexed Array:

```
$ numbers = array (1,2,3,4,5);
```

- Associatively indexed Array:

```
$ person = array (
        "name" => "John",
        "age" => 30,
        "city" = "New York"
    );
```

- Array index starts from 0.
- We can also define array by using square brackets '[]'.
- array() function is used to create an array.

6) Objects:

- In PHP, an object is a datatype that allows you to encapsulate data and behavior into a single entity.
- Objects are instances of classes, which define the properties and methods of the object.
- We can create new instances of a class using the 'new' keyword.
- Each instance has its own set of properties, which can be accessed using the object operator (->).
- We can also methods using object operator.
- Use is_object to test whether a value is an object or not.

Scanned by TapScanner

7) Resources:

- In PHP, a resource is a special data type that represents an external resource such as file, a database connection, or a network socket.
- A resource is a reference to an external resource, rather than the resource itself.
- Resources are used in PHP to manage external resources that are not under the control of the PHP script.
- Resources are not normal variables, and they cannot be manipulated or copied like other data types.
- Use is_resource function to test whether a value is a resource or not.

8) Null:

- In PHP, null is a datatype that represents a variable with no value.
- It is used to indicate the absense of a value.
- Null is often used as a default value for variables or function arguments that may or may not be initialized or passed in.
- It can also be used to unset a variable or free up memory.
- Null is not same as the empty string (" "), an empty array ('[]') or the integer 0.
- These all are valid values in PHP, whereas null indicates that a variable has no value.
- Use is_null() to check whether a value is null or not.

Scanned by TapScanner

- Expressions:
  - In PHP, an expression is a combination of variables, operators and function calls that evaluates to a value.
  - Expressions are used extensively in PHP code to perform calculations, comparisons and other operations.
  - Expressions can include logical & comparison operators.
  - These operators are used to evaluate conditions and make decisions in PHP code.
  - Example;

```
$x = 5 + 3;
$y = ($x * 2) / 3;
```

- Operators:
  - In PHP, operators are special symbols and keywords used to perform operations on variables, constants and other data types.
  - PHP support wide range of operators:

1) Arithmetic Operators:

+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus
**	Exponentiation

Scanned by TapScanner

2) Comparison Operators:

==	Equal
===	Identical
!==	Not Identical
!=	Not equal
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to

3) Logical Operators:

&& , and	AND
|| , or	OR
!	NOT
xor , ^	XOR

4) Assignment Operators:

=	assignment
+=	Addition assignment
-=	Subtraction assignment
*=	Multiplication assignment
/=	Division assignment
%=	Modulus assignment

5) Bitwise operator:

&	Bitwise AND
|	Bitwise OR
^	Bitwise XOR
~	Bitwise NOT
<<	Left shift

Scanned by TapScanner

>> Right shift

6) Conditional / Ternary Operator :

```
exp1 ? exp2 : exp3;
```

- Constants :

- In PHP, a constant is a value that cannot be changed during execution of a script.
- To define a constant in PHP, you can use the 'define()' function.
- Syntax:

```
define( name, value);
```

- Example;

```
$ a = 10; // variable
define( " PI", 3.14159);
```

- Decision making control statements :

i) if :

- In PHP, the 'if' statement is used for conditional execution of code.
- 'If' statement tests a condition and executes a block of code if the condition is true.
- Syntax:

```
if ( condition)
{
    // code to be executed
}
```

Scanned by TapScanner

2) if-else :

- In PHP, the 'if-else' statement is used for conditional execution of code based on a condition.
- The 'if-else' statement tests a condition and executes one block of code if the condition is true and a different block of code if the condition is false.
- Syntax:

```
if (condition)
{
    // code to be executed
}
else
{
    // code to be executed
}
```

3) nested if :

- In PHP, a nested 'if' statement is a conditional statement that is contained within another 'if' statement.
- A nested 'if' statement can be used to test additional conditions if the original condition is true.
- Syntax:

```
if (condition1)
{
    // code to be executed.
    if (condition2)
    {
        // code to be executed
    }
```

Scanned by TapScanner

4) Switch :

- In PHP, the switch statement is a control structure used to perform different actions based on different conditions.
- It provides an alternative to using multiple 'if' statements for testing the same variable.
- The switch statement tests a variable against a series of values and executes a block of code based on which value the variable matches.
- Syntax:

```
switch (variable)
{
    case value1:
        // code to be executed
        break;
    case value2:
        // code to be executed
        break;
    case value3:
        // code to be executed
        break;
    // more cases here
    default:
        // code to be executed
}
```

5) break ;

- In PHP, the break statement is used to immediately exit a loop or a switch statement.
- When executed inside a loop, the break

Scanned by TapScanner

statement causes the loop to terminate immediately and execution of the program continues with the first statement after the loop.
- When used inside a 'switch' statement, the 'break' statement causes execution to exit the switch block and continue with the first statement after the switch block.

6) Continue statement:
- In PHP, the continue statement is used to immediately skip to the next iteration of a loop.
- When executed inside a loop ~~to end and~~ the continue statement causes the current iteration of the loop to end and execution to continue with the next iteration of the loop.

• Loop control structure:

1) While loop:
- In PHP, a while loop is a control structure that allows you to execute a block of code repeatedly as long as a certain condition is true.
- Syntax:

```
while (condition)
{
    // code to be executed
}
```

Scanned by TapScanner

2) do-while loop:

- In php, the do-while loop is a type of loop that executes a block of code repeatedly as long as certain condition is true.
- There is a difference between while and do-while loop.
- In do-while loop, the code block is executed at least once before condition is checked.
- Syntax:

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

3) for loop:

- In PHP, a for loop is a control structure that is used to execute a block of code repeatedly a fixed number of times.
- The for loop consists of three parts:
  i) initialization
  ii) condition
  iii) increment
- Syntax:

```
for( initialization; condition; increment)
{
    // code to be executed
}
```

Scanned by TapScanner

4) foreach loop:

- In PHP, a foreach loop is a control structure that is used to iterate over an array or an object.
- It allows you to loop through each element of an array or each property of an object and perform an action on each one.
- The for each loop is particularly useful when you don't know the number of elements in the array or object in advance.
- Syntax:

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

Scanned by TapScanner