Netaji Subhash Engineering College

Name: Raima Majumder

University Roll No.: 29242723028

Topic: PHP Fundamentals: Variables, Data Types, and Operators

Subject: PHP with MySQL

Paper Code: BCAC501

Stream: Bachelors of Computer Application

Year: 3rd

Semester: 5th

Session: 2025-2026

❑ *Introduction to PHP*

PHP, which stands for **Hypertext Preprocessor**, is a powerful and widely-used **server-side scripting language** designed specifically for web development. Unlike client-side languages such as JavaScript, which run in the browser, PHP runs on the **web server**, allowing it to generate dynamic page content, interact with databases, manage sessions, and perform a wide range of server-side functions. It is especially popular for creating websites that require frequent updates or user interaction, such as content management systems, forums, or e-commerce platforms.

One of the key features of PHP is that it can be **embedded directly into HTML**, making it easy to mix server-side logic with client-side content. When a user accesses a PHP page, the server processes the PHP code and then sends the resulting HTML back to the user's browser. This makes PHP an efficient way to create dynamic and interactive web pages without needing to reload full HTML files manually.

❑ *Basics of PHP syntax*

➢ ***PHP Tags***

- PHP code must be written **inside <?php ... ?> tags**
- These tags tell the web server where the PHP code begins and ends
- Anything **outside the tags** is treated as plain HTML

Example:

```
<?php
  echo "Hello, PHP!";
?>
```

➢ ***Statements and Semicolons***

- Every PHP statement **must end with a semicolon (;)**
- This separates one instruction from the next
- Missing semicolons can cause syntax errors

Example:

```
<?php
  $name = "Alice";
  echo $name;
?>
```

➢ ***Case Sensitivity***

- **Variables** are **case-sensitive**
- $Name and $name are different variables
- **Functions and keywords** are **not case-sensitive**
- echo, ECHO, and EcHo are all valid

➢ ***Comments in PHP***

Used to describe code or temporarily disable code execution.

Types of comments:

- // Single-line comment
- # Another way for single-line comment
- /* */ Multi-line comment

❑ *Variables in PHP*

➢ *What is a Variable?*

- A **variable** is a container for storing data such as strings, numbers, or arrays.
- In PHP, variables are **declared using the dollar sign ($)** followed by the variable name.

Example:

```
<?php
  $name = "Alice";
  $age = 25;
?>
```

➢ *Variable Naming Rules*

- Must begin with a **$** followed by a **letter** or **underscore (_)**
- Cannot start with a number (e.g., $1user is invalid)
- Can include letters, numbers, and underscores (e.g., $user_name)
- **Case-sensitive** – $Name and $name are **not** the same

➢ *Displaying Variable Values*

- Use echo or print to output variable values to the browser.

Example:

```
<?php
  $name = "John";
  echo "Hello, " . $name;  // Outputs: Hello, John
?>
```

- The . operator is used for **string concatenation**

❑ *PHP Data Types*

➢ What Are Data Types?

Data types define the **kind of value** a variable can hold. In PHP, variables can store different types of data such as text, numbers, or collections. PHP is a **loosely typed language**, meaning you don't need to declare a data type explicitly — it automatically detects the type based on the assigned value.

Data Type	Description	Example
String	Text enclosed in quotes	"Hello World"
Integer	Whole numbers	42, -7
Float	Decimal numbers	3.14, -2.5
Boolean	True or false values	true, false
Array	Collection of values	["red", "blue", "green"]
NULL	Represents a variable with no value	NULL

➢ Code Examples:

```
<?php
  $name = "Alice";           // String
  $age = 25;               // Integer
  $price = 9.99;            // Float
  $isOnline = true;          // Boolean
  $colors = ["red", "blue"]; // Array
  $nothing = NULL;           // NULL
?>
```

❑ *Operators in PHP*

➢ **What is an Operator?**

An **operator** in PHP is a **symbol** that tells the program to perform a specific action on one or more variables or values. Operators are used in tasks such as **math calculations**, **value comparisons**, **logic decisions**, **string manipulation**, and more.

PHP has several types of operators:

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

Arithmetic Operators

- **What Are Arithmetic Operators?**

Used to perform mathematical operations on numeric values.

- **List of Arithmetic Operators:**

Operator	Operator	Example	Result
+	Addition	5+2	7
-	Subtraction	5-2	3
*	Multiplication	5*2	10
/	Division	10/2	5
%	Modulus	10%3	1

Example:

```
<?php
 $a = 10;
 $b = 3;
 echo $a + $b;  // Output: 13
?>
```

Comparison Operators

- **What Are Comparison Operators?**
  Used to **compare two values**. Returns a **boolean result** (true or false).
- **List of Comparison Operators:**

Operator	Name	Example	Result
==	Equal	5 == “5”	True
===	Identical(type + value)	5 === “5”	False
!=	Not equal	5 != 3	True
<>	Not equal	5 <> 5	False
<	Greater than	5 > 3	True
>	Less than	2 < 3	True
>=	Greater or equal	4 >= 4	True
<=	Less or equal	3 <= 2	False

- **Example:**

```
<?php
  var_dump(5 == "5");   // true
  var_dump(5 === "5");  // false
?>
```

Logical Operators

- **What Are Logical Operators?**
  Used to **combine multiple conditions** and return boolean values.
- **List of Logical Operators:**

Operator	Name	Example	Result
&&	AND	True && false	False
||	OR	True || false	True
!	NOT	!true	false

- **Example:**

```
<?php
  $x = true;
  $y = false;
  var_dump($x && $y); // false
  var_dump($x || $y); // true
  var_dump(!$x);       // false
?>
```

Assignment Operators

- Used to **assign values to variables.**

Operator	Example	Meaning
=	$x = 2	Assign 2 to $x
+=	$x += 2	$x = $x + 2
-=	$x -= 2	$x = $x - 2
*=	$x *= 2	$x = $x * 2
/=	$x /= 2	$x = $x / 2

- **Example:**

```
<?php
  $x = 10;
  $x += 5;
  echo $x;  // Output: 15
?>
```

Increment/Decrement Operators

- Used to increase or decrease a value by 1.

Operator	Description	Example	Result
++$x	Pre-increment	++$x;	Increments before use
$x++	Post-increment	$x++;	Increments after use
--$x	Pre-decrement	--$x;	decrements before use
$x--	Post-decrement	$x--;	decrements after use

- **Example:**

```
<?php
 $count = 5;
 ++$count;
 echo $count;  // Output: 6
?>
```

String Operators

- Used to combine or append strings.

Operation	Description	Example	Result
.	Concatenation	"Hello" . " PHP"	"Hello PHP"
.=	Concatenation & assign	$msg .= " World"	Appends to $msg

- **Example:**

```
<?php
 $greeting = "Hello";
 $greeting .= " PHP!";
 echo $greeting;  // Output: Hello PHP!
?>
```

❑ *Conclusion*

In this presentation, we explored the fundamental building blocks of PHP — its syntax, variables, data types, and operators. We began by understanding how PHP code is written and how variables are used to store and manage data. We then looked at PHP's flexible data types like strings, integers, floats, booleans, arrays, and NULL, which allow developers to handle a wide range of information.

Operators were also discussed in detail, showing how they enable tasks like performing calculations, comparing values, combining logic, and manipulating strings. These concepts are essential for writing meaningful and dynamic PHP scripts. With a clear grasp of these basics, you're well-equipped to start developing interactive and data-driven web applications using PHP.

Thank You!