**Intro**

PHP (singkatan rekursif untuk PHP : Hypertext Preprocessor) adalah bahasa skrip “general-purpose” open source alias gratis yang banyak digunakan dan sangat cocok untuk pengembangan web dan dapat disematkan ke dalam HTML.

PHP merupakan bahasa pemrograman sisi server yang dirancang khusus untuk mengembangkan aplikasi web dinamis dan interaktif.

**Sejarah**

PHP (PHP: Hypertext Preprocessor) adalah bahasa pemrograman server-side yang digunakan untuk mengembangkan aplikasi web. PHP awalnya dikembangkan pada tahun 1994 oleh Rasmus Lerdorf sebagai sekumpulan script untuk mengelola website pribadinya. Pada awalnya, PHP disebut sebagai "Personal Home Page Tools".

Pada tahun 1995, Rasmus Lerdorf mengeluarkan versi pertama dari PHP yang kemudian dikenal sebagai PHP/FI (Forms Interpreter). PHP/FI merupakan versi PHP yang pertama kali dipublikasikan dan digunakan oleh banyak pengembang web.

Pada tahun 1997, Andi Gutmans dan Zeev Suraski, dua mahasiswa Universitas Technion di Israel, memperkenalkan Zend Engine, yaitu sebuah mesin rendering PHP yang menyediakan kinerja yang lebih baik dan fitur yang lebih lengkap. Zend Engine menjadi inti dari PHP hingga saat ini.

Pada tahun 1998, PHP 3 dirilis dengan lebih banyak fitur dan kemampuan dibandingkan dengan PHP/FI. PHP 3 memiliki dukungan untuk database MySQL dan PostgreSQL, dan juga mendukung pengembangan aplikasi web yang lebih canggih.

Pada tahun 2000, PHP 4 dirilis dengan banyak peningkatan, termasuk kinerja yang lebih baik dan dukungan untuk banyak database. PHP 4 juga memperkenalkan banyak fitur baru seperti penggunaan objek dan manajemen kesalahan yang lebih baik.

Pada tahun 2004, PHP 5 dirilis dengan peningkatan signifikan termasuk dukungan untuk pemrograman berorientasi objek yang lebih baik, kinerja yang lebih cepat, dan penanganan kesalahan yang lebih baik.

Pada tahun 2014, PHP 5.6 dirilis dengan banyak perbaikan keamanan dan fitur baru termasuk perbaikan keamanan dan penanganan error yang lebih baik, serta pengoptimalan kinerja yang lebih baik.

Pada tahun 2015, PHP 7 dirilis dengan peningkatan kinerja yang signifikan dan dukungan untuk pemrograman berorientasi objek yang lebih baik.

Saat ini, PHP masih menjadi salah satu bahasa pemrograman web yang paling populer dan banyak digunakan di seluruh dunia. Versi terbaru saat ini adalah PHP 8, yang dirilis pada tahun 2020 dengan peningkatan kinerja dan banyak fitur baru yang ditambahkan.

**How To Use?**

- Persiapan

Hal yang perlu dipersiapkan sebelum mencoba PHP adalah menginstall Web Server dan bahasa PHP tersebut. Bila tidak mau kerepotan menginstall Web Server (Apache, nginx, IIS, dan lain-lain) dan PHP secara terpisah, anda dapat menginstall software yang bernama XAMPP. Kelebihan XAMPP adalah dapat berjalan di semua platform (MAC, Linux, Windows). Apache, PHP, dan Database sudah tersedia dalam XAMPP.

**URL**

WPU : <https://www.youtube.com/watch?v=q3NVC5JxgVI&list=PLFIM0718LjIUqXfmEIBE3-uzERZPh3vp6&index=2>

PHP : <https://www.php.net/manual/en/intro-whatis.php>

Persiapan PHP : https://www.youtube.com/watch?v=o8oLQVYlpqw&list=PLFIM0718LjIUqXfmEIBE3-uzERZPh3vp6&index=3

**Sintaks**

**Tag PHP**

Penulisan kode PHP harus diawali dengan tag <?php pembuka lalu diakhiri dengan tag penutup ?>.

**Short Open Tag PHP**

Short open tag (<=) merupakan pintasan tag pembuka output PHP. Untuk membuat program output kalimat sederhana yang pada umumnya seperti : <?php echo”Hello World!”; ?>, bila menggunakan short open tag maka menjadi <= “Hello World!”; ?>. hal tersebut akan menjadi sangat efisien dalam mengurangi waktu pengetikkan.

**Komentar**

Pembuatan komentar berfungsi untuk membantu programmer memberikan penjelas atau tanda pada kode program tertentu. Komentar tersebut tidak akan berpengaruh terhadap jalannya program tersebut, maka saat program PHP dijalankan tidak akan ditampilkan pada output. Pembuatan komentar 2 macam, yaitu untuk yang sebaris dan banyak baris.

**Escape From HTML**

Segala sesuatu di luar tag pembuka dan penutup diabaikan oleh parser PHP yang memungkinkan file PHP memiliki konten campuran. Ini memungkinkan PHP disematkan dalam dokumen HTML, misalnya untuk membuat template.

**URL**

PHP : <https://www.php.net/manual/en/language.basic-syntax.php>

WPU : https://www.youtube.com/watch?v=XTrU0GzMfCk&list=PLFIM0718LjIUqXfmEIBE3-uzERZPh3vp6&index=4

**Variabel**

Pengenalan Variabel: Variabel merupakan salah satu konsep dasar dalam pemrograman. Variabel digunakan untuk menyimpan nilai yang dapat diakses dan diubah selama proses eksekusi program. Dalam PHP, Anda dapat membuat variabel untuk menyimpan berbagai jenis data, termasuk angka, teks, array, dan objek.

Penulisan Variabel:

* Variabel dalam PHP diawali dengan tanda dolar ($), diikuti dengan nama variabel.
* Variabel harus diawali dengan huruf atau underscore (\_)
* Variabel tidak bisa diawali dengan angka
* Nama variabel bersifat case-sensitive, yang berarti $nama dan $Nama dianggap sebagai dua variabel yang berbeda.
* Contoh penulisan variabel yang valid: $nama, $umur, $jumlah\_barang.

**Variabel Scope**

Variabel scope merujuk pada wilayah atau bagian program di mana sebuah variabel dapat diakses dan digunakan. Dalam PHP, terdapat beberapa jenis scope yang mempengaruhi ketersediaan variabel, yaitu scope global, scope lokal, dan scope statis.

|  |
| --- |
| $nama = "John Doe"; // variabel global  function tampilkanNama() {  global $nama; // menggunakan variabel global di dalam fungsi  echo $nama;  }  tampilkanNama(); // Output: John Doe |

1. Scope Lokal: Variabel lokal adalah variabel yang hanya dapat diakses di dalam blok kode tertentu di mana variabel tersebut dideklarasikan. Variabel lokal dideklarasikan di dalam fungsi atau metode, dan tidak dapat diakses dari luar blok tersebut.

|  |
| --- |
| function tampilkanUmur() {  $umur = 25; // variabel lokal  echo $umur;  }  tampilkanUmur(); // Output: 25  // echo $umur; // akan menghasilkan error karena variabel tidak dapat diakses di luar fungsi |

1. Scope Statis: Variabel statis adalah variabel yang mempertahankan nilainya di antara pemanggilan fungsi. Variabel statis dideklarasikan di dalam fungsi dan nilainya akan diingat setelah fungsi selesai dieksekusi. Ketika fungsi dipanggil kembali, variabel statis akan menyimpan nilainya dari pemanggilan sebelumnya.

|  |
| --- |
| function hitungJumlah() {  static $jumlah = 0; // variabel statis  $jumlah++;  echo $jumlah;  }  hitungJumlah(); // Output: 1  hitungJumlah(); // Output: 2  hitungJumlah(); // Output: 3 |

**URL**

W3schools : https://www.w3schools.com/php/php\_variables.asp

Scope Variabel : https://www.w3schools.com/php/php\_variables\_scope.asp

PHP : https://www.php.net/manual/en/language.variables.php

**Echo/Print**

Dalam bahasa pemrograman PHP, terdapat dua perintah yang sering digunakan untuk menampilkan output ke browser atau konsol, yaitu **echo** dan **print**. Kedua perintah ini memiliki fungsi yang serupa, namun ada beberapa perbedaan kecil di antara keduanya.

## 1. Echo

Perintah **echo** digunakan untuk menampilkan satu atau lebih nilai ke output. Berikut adalah sintaks dari **echo**:

Beberapa hal yang perlu diperhatikan tentang **echo**:

* **echo** tidak mengembalikan nilai (void), sehingga tidak dapat digunakan dalam ekspresi aritmatika atau pernyataan **return**.
* Kita dapat menampilkan lebih dari satu nilai dengan memisahkan mereka menggunakan tanda koma (,).
* Kita juga dapat menampilkan teks dan variabel dalam satu perintah **echo**.

## 2. Print

Perintah **print** juga digunakan untuk menampilkan output ke browser atau konsol. Berikut adalah sintaks dari **print**:

Beberapa hal yang perlu diperhatikan tentang **print**:

* **print** mengembalikan nilai 1, sehingga dapat digunakan dalam ekspresi aritmatika dan pernyataan **return**.
* **print** hanya menerima satu nilai, sehingga tidak dapat menampilkan lebih dari satu nilai dalam satu perintah.
* Kita dapat menggabungkan teks dan nilai menggunakan operator penggabung string (.) seperti contoh di atas.

## Perbedaan antara Echo dan Print

Perbedaan utama antara **echo** dan **print** adalah:

* **echo** dapat menampilkan lebih dari satu nilai dengan memisahkan mereka menggunakan tanda koma (,), sedangkan **print** hanya menerima satu nilai.
* **echo** tidak mengembalikan nilai (void), sedangkan **print** mengembalikan nilai 1.
* Dalam hal performa, **echo** cenderung sedikit lebih cepat daripada **print**.

**URL**

W3schools : https://www.w3schools.com/php/php\_echo\_print.asp

**Tipe Data**

Dalam bahasa pemrograman PHP, terdapat beberapa tipe data yang dapat digunakan untuk menyimpan nilai atau informasi. Memahami tipe data ini sangat penting untuk mengelola dan memanipulasi data dengan benar. Berikut adalah beberapa tipe data utama dalam PHP:

Tipe Data String: Tipe data string digunakan untuk menyimpan teks atau karakter. String diapit oleh tanda kutip tunggal ('') atau tanda kutip ganda (""). Contoh penggunaan tipe data string adalah sebagai berikut:

Tipe Data Integer: Tipe data integer digunakan untuk menyimpan bilangan bulat. Contoh penggunaan tipe data integer adalah sebagai berikut:

Tipe Data Float: Tipe data float (atau sering disebut juga tipe data double) digunakan untuk menyimpan bilangan desimal. Contoh penggunaan tipe data float adalah sebagai berikut:

Tipe Data Boolean: Tipe data boolean digunakan untuk menyimpan nilai kebenaran (benar atau salah). Nilai benar direpresentasikan dengan **true** (tanpa tanda kutip), dan nilai salah direpresentasikan dengan **false** (tanpa tanda kutip). Contoh penggunaan tipe data boolean adalah sebagai berikut:

Tipe Data Array: Tipe data array digunakan untuk menyimpan kumpulan nilai dalam satu variabel. Nilai-nilai dalam array dapat memiliki tipe data yang berbeda. Contoh penggunaan tipe data array adalah sebagai berikut:

Tipe Data Objek: Tipe data objek digunakan untuk membuat objek yang merupakan instansi dari sebuah kelas. Objek menyimpan data (properti) dan perilaku (metode) yang terkait dengan kelas tersebut. Contoh penggunaan tipe data objek adalah sebagai berikut:

Untuk lebih banyak mengetahui tipe data lainnya bisa dilihat pada referensi

**URL**

W3schools : https://www.w3schools.com/php/php\_datatypes.asp

PHP : https://www.php.net/manual/en/language.types.php

**Operators**

Operator adalah simbol atau tanda khusus yang digunakan untuk melakukan operasi matematika, perbandingan, logika, atau manipulasi data dalam sebuah program. Operator dapat digunakan untuk menggabungkan, membandingkan, memodifikasi, atau mengubah nilai dari variabel atau ekspresi dalam bahasa pemrograman.

1. Operator Aritmatika:
   * Operator Penjumlahan (+): Digunakan untuk menjumlahkan dua nilai.
   * Operator Pengurangan (-): Digunakan untuk mengurangkan satu nilai dari nilai yang lain.
   * Operator Perkalian (\*): Digunakan untuk mengalikan dua nilai.
   * Operator Pembagian (/): Digunakan untuk membagi nilai satu dengan nilai yang lain.
   * Operator Modulus (%): Digunakan untuk mendapatkan sisa pembagian dua nilai.
   * Operator Increment (++) dan Decrement (--): Digunakan untuk menambah atau mengurangkan nilai variabel sebesar 1.
2. Operator Penugasan:
   * Operator Penugasan (=): Digunakan untuk memberikan nilai pada variabel.
   * Operator Penugasan Aritmatika (+=, -=, \*=, /=, %=): Digunakan untuk menggabungkan operasi aritmatika dengan operasi penugasan.
3. Operator Perbandingan:
   * Operator Sama dengan (==): Digunakan untuk memeriksa apakah dua nilai sama.
   * Operator Identik (===): Digunakan untuk memeriksa apakah dua nilai dan tipe datanya sama.
   * Operator Tidak Sama dengan (!=): Digunakan untuk memeriksa apakah dua nilai tidak sama.
   * Operator Tidak Identik (!==): Digunakan untuk memeriksa apakah dua nilai atau tipe datanya tidak sama.
   * Operator Lebih Besar (>): Digunakan untuk memeriksa apakah nilai pertama lebih besar dari nilai kedua.
   * Operator Lebih Kecil (<): Digunakan untuk memeriksa apakah nilai pertama lebih kecil dari nilai kedua.
   * Operator Lebih Besar Sama dengan (>=): Digunakan untuk memeriksa apakah nilai pertama lebih besar atau sama dengan nilai kedua.
   * Operator Lebih Kecil Sama dengan (<=): Digunakan untuk memeriksa apakah nilai pertama lebih kecil atau sama dengan nilai kedua.
4. Operator Logika:
   * Operator Logika AND (&&): Digunakan untuk memeriksa apakah kedua kondisi bernilai benar.
   * Operator Logika OR (||): Digunakan untuk memeriksa apakah salah satu dari kedua kondisi bernilai benar.
   * Operator Logika NOT (!): Digunakan untuk membalikkan nilai suatu kondisi.
5. Operator String:
   * Operator Penggabung String (.): Digunakan untuk menggabungkan dua string menjadi satu.
   * Operator Penggabung String dan Penugasan (.=): Digunakan untuk menggabungkan dua string dan menugaskan hasilnya ke variabel.
6. Operator Array:
   * Operator Union (+): Digunakan untuk menggabungkan dua array menjadi satu.

**URL**

W3schools : https://www.w3schools.com/php/php\_datatypes.asp

PHP https://www.php.net/manual/en/language.operators.php

**Percabangan**

Pengkondisian adalah salah satu konsep dasar dalam pemrograman yang memungkinkan kita untuk melakukan pemilihan atau pengecekan terhadap suatu kondisi tertentu. Dalam PHP, terdapat beberapa pernyataan pengkondisian yang dapat digunakan untuk mengontrol alur program. Dalam materi ini, kita akan membahas beberapa pernyataan pengkondisian utama dalam PHP beserta contoh penggunaannya.

1. Pernyataan If Pernyataan "if" digunakan untuk menjalankan blok kode tertentu jika suatu kondisi benar (true). Format dasar dari pernyataan "if" adalah sebagai berikut:
2. Pernyataan If-Else Pernyataan "if-else" digunakan untuk menjalankan blok kode tertentu jika kondisi benar (true), dan menjalankan blok kode lain jika kondisi salah (false). Format dasar dari pernyataan "if-else" adalah sebagai berikut:
3. Pernyataan If-ElseIf-Else Pernyataan "if-elseif-else" digunakan untuk melakukan pengecekan kondisi secara berurutan. Jika kondisi pertama benar, maka blok kode terkait akan dijalankan. Jika tidak, maka kondisi berikutnya akan diperiksa hingga kondisi yang cocok ditemukan atau mencapai bagian "else" sebagai kondisi default. Format dasar dari pernyataan "if-elseif-else" adalah sebagai berikut:
4. Pernyataan Switch Pernyataan "switch" digunakan untuk melakukan pengecekan kondisi dengan beberapa pilihan nilai yang mungkin. Setiap pilihan nilai memiliki blok kode terkait yang akan dijalankan jika nilai tersebut cocok. Format dasar dari pernyataan "switch" adalah sebagai berikut:

**URL**

W3schools (IF, ELSE IF) : <https://www.w3schools.com/php/php_if_else.asp>

W3schools (SWITCH) : https://www.w3schools.com/php/php\_switch.asp

PHP : https://www.php.net/manual/en/language.control-structures.php

**Loop**

Loop merupakan sebuah konstruksi dalam pemrograman yang digunakan untuk mengulang suatu blok kode berulang kali. Di dalam PHP, terdapat beberapa jenis loop yang dapat digunakan, yaitu:

Loop while: Loop while digunakan untuk mengulang blok kode selama suatu kondisi bernilai benar (true). Contoh penggunaannya sebagai berikut:

|  |
| --- |
| <?php  $counter = 1;  while ($counter <= 10) {  echo "Nilai counter: " . $counter . "<br>";  $counter++;  }  ?> |

Pada contoh di atas, blok kode akan diulang selama nilai **$counter** kurang dari atau sama dengan 10. Setiap kali diulang, nilai **$counter** akan ditampilkan dan kemudian nilai **$counter** akan ditambah 1.

Loop do-while: Loop do-while hampir sama dengan loop while, namun blok kode akan dijalankan setidaknya satu kali sebelum pengecekan kondisi dilakukan. Contoh penggunaannya sebagai berikut:

|  |
| --- |
| <?php  $counter = 1;  do {  echo "Nilai counter: " . $counter . "<br>";  $counter++;  } while ($counter <= 10);  ?> |

Pada contoh di atas, blok kode akan dijalankan minimal satu kali, kemudian dilakukan pengecekan kondisi. Jika kondisi masih bernilai benar, blok kode akan diulang kembali.

Loop for: Loop for merupakan loop yang memiliki struktur yang lebih kompleks dibandingkan loop while dan do-while. Loop for terdiri dari tiga bagian, yaitu inisialisasi, kondisi, dan perubahan nilai. Contoh penggunaannya sebagai berikut:

|  |
| --- |
| <?php  for ($counter = 1; $counter <= 10; $counter++) {  echo "Nilai counter: " . $counter . "<br>";  }  ?> |

Pada contoh di atas, variabel **$counter** diinisialisasi dengan nilai 1. Selama kondisi **$counter** kurang dari atau sama dengan 10 bernilai benar, blok kode akan diulang. Setiap selesai satu iterasi, nilai **$counter** akan ditambah 1.

Loop foreach: Loop foreach digunakan untuk mengulang elemen-elemen dalam sebuah array atau objek. Loop ini tidak mengharuskan penggunaan variabel penunjuk (iterator) seperti pada loop for. Contoh penggunaannya sebagai berikut:

|  |
| --- |
| <?php  $fruits = array("Apel", "Jeruk", "Mangga");  foreach ($fruits as $fruit) {  echo $fruit . "<br>";  }  ?> |

Pada contoh di atas, setiap elemen dalam array **$fruits** akan diambil satu per satu dan disimpan dalam variabel **$fruit**. Blok kode akan dijalankan untuk setiap elemen dalam array.

**URL**

W3shools : https://www.w3schools.com/php/php\_looping.asp

PHP : https://www.php.net/manual/en/language.control-structures.php

**Function**

Pengertian Function: Function adalah blok kode yang terorganisir secara logis dan dapat digunakan ulang dalam sebuah program. Function memungkinkan kita untuk memecah program menjadi bagian-bagian yang lebih kecil, yang masing-masing memiliki tugas tersendiri. Hal ini membantu dalam mengorganisir kode, meningkatkan keterbacaan, dan memudahkan pemeliharaan kode.

Mendefinisikan Function: Di PHP, sebuah function didefinisikan menggunakan kata kunci "function", diikuti dengan nama function, tanda kurung (), dan blok kode di dalamnya. Berikut adalah contoh pendefinisian function di PHP:

Function Scope: Variabel yang didefinisikan di dalam function hanya bisa diakses di dalam function tersebut, kecuali jika variabel tersebut dideklarasikan sebagai global. Sebaliknya, variabel yang didefinisikan di luar function dapat diakses di dalam function. Berikut adalah contoh penggunaan variabel scope di PHP:

|  |
| --- |
| $nama = "John";  function sapa() {  global $nama;  echo "Halo, " . $nama;  }  sapa(); // Output: Halo, John |

Parameter dan Return Value: Function dapat menerima parameter sebagai input untuk diproses di dalam function. Parameter didefinisikan di dalam tanda kurung () setelah nama function. Selain itu, function dapat mengembalikan nilai menggunakan pernyataan "return". Berikut adalah contoh penggunaan parameter dan return value di PHP:

|  |
| --- |
| function tambah($a, $b) {  $hasil = $a + $b;  return $hasil;  }  $hasil\_penjumlahan = tambah(3, 5);  echo $hasil\_penjumlahan; // Output: 8 |

**URL**

W3schools : https://www.w3schools.com/php/php\_functions.asp

PHP : https://www.php.net/manual/en/language.functions.php

**Arrays**

Array adalah struktur data yang digunakan untuk menyimpan kumpulan nilai-nilai dalam satu variabel. Dalam PHP, array dapat digunakan untuk menyimpan data dalam berbagai bentuk seperti angka, teks, objek, atau bahkan array lainnya. Array dalam PHP dapat memiliki indeks numerik atau indeks asosiatif.

**1. Membuat Array**

Anda dapat membuat array dalam PHP dengan menggunakan beberapa metode. Berikut ini adalah dua cara umum untuk membuat array:

a. Array Numerik:

|  |
| --- |
| $nama\_array = array(nilai1, nilai2, nilai3, ...); |

b. Array Asosiatif:

|  |
| --- |
| $nama\_array = array("kunci1" => nilai1, "kunci2" => nilai2, "kunci3" => nilai3, ...); |

**2. Mengakses Nilai Array**

Anda dapat mengakses nilai-nilai dalam array dengan menggunakan indeks numerik atau indeks asosiatif.

a. Array Numerik:

|  |
| --- |
| $nama\_array[indeks]; |

b. Array Asosiatif:

|  |
| --- |
| $nama\_array["kunci"]; |

**3. Mengubah Nilai Array**

Anda dapat mengubah nilai dalam array dengan menetapkan nilai baru ke indeks yang diinginkan.

|  |
| --- |
| $buah = array("apel", "jeruk", "mangga", "pisang");  $buah[1] = "anggur";  echo $buah[1]; // Output: anggur |

**4. Menambahkan Nilai ke Array**

Anda dapat menambahkan nilai baru ke array menggunakan fungsi **array\_push()** atau langsung menetapkan nilai baru ke indeks yang belum ada.

a. Menggunakan array\_push():

|  |
| --- |
| $nama\_array = array();  array\_push($nama\_array, nilai1, nilai2, ...);  $buah = array();  array\_push($buah, "apel", "jeruk");  echo $buah[0]; // Output: apel  echo $buah[1]; // Output: jeruk |

b. Menetapkan nilai baru ke indeks yang belum ada:

|  |
| --- |
| $nama\_array[indeks] = nilai;  $buah = array();  $buah[0] = "apel";  $buah[1] = "jeruk";  echo $buah[0]; // Output: apel  echo $buah[1]; // Output: jeruk |

**5. Perulangan Array**

Anda dapat menggunakan perulangan seperti **for**, **foreach**, atau **while** untuk mengakses dan memanipulasi nilai-nilai dalam array.

|  |
| --- |
| $buah = array("apel", "jeruk", "mangga", "pisang");  foreach ($buah as $nilai) {  echo $nilai . " ";  }  // Output: apel jeruk mangga pisang |

**URL**

W3schools : https://www.w3schools.com/php/php\_arrays.asp

PHP : https://www.php.net/manual/en/function.array