

LAPORAN PRAKTIKUM PEMROGRAMAN WEB

PERTEMUAN 7

ASSIGNMENT & OPERATOR ARITMATIKA



2411102441052

Angga Maulana Saputra

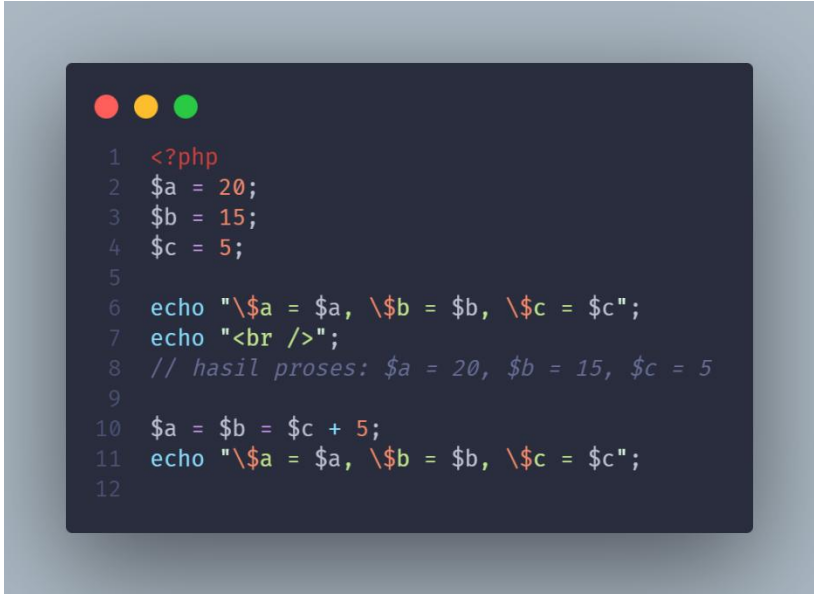
FAKULTAS SAINS DAN TEKNOLOGI

PROGRAM STUDI S1 TEKNIK INFORMATIKA

UNIVERSITA MUHAMMADIYAH KALIMANTAN TIMUR

Link Github: <https://github.com/nikamushi/tugashtmlcss/tree/main/pertemuan7>

Praktikum 7.1 (assignment by value)



```
1 <?php
2 $a = 20;
3 $b = 15;
4 $c = 5;
5
6 echo "\$a = $a, \$b = $b, \$c = $c";
7 echo "<br />";
8 // hasil proses: $a = 20, $b = 15, $c = 5
9
10 $a = $b = $c + 5;
11 echo "\$a = $a, \$b = $b, \$c = $c";
12
```

Kode di atas menunjukkan konsep assignment by value dalam PHP, yaitu penugasan nilai dari satu variabel ke variabel lain. Pada awalnya, variabel \$a, \$b, dan \$c masing-masing bernilai 20, 15, dan 5. Setelah baris `$a = $b = $c + 5;` dijalankan, operasi dilakukan dari kanan ke kiri, sehingga ekspresi `$c + 5` menghasilkan 10, kemudian nilai 10 tersebut diberikan terlebih dahulu ke \$b, lalu juga ke \$a. Akibatnya, nilai \$a dan \$b menjadi 10, sedangkan \$c tetap 5 karena tidak ikut diubah. Hal ini menunjukkan bahwa setiap variabel menyimpan salinan nilai tersendiri, sesuai dengan prinsip assignment by value.

Praktikum 7.2 (assignment by array)

2-byarray1.php

```
1  <?php
2  // Pembuatan array
3  $nama = array(
4      1 => "Andri",
5      2 => "Joko",
6      3 => "Sukma",
7      4 => "Rina",
8      5 => "Sari"
9  );
10
11 // Cara akses array
12 echo $nama[1]; // Andri
13 echo "<br />";
14 echo $nama[2]; // Joko
15 echo "<br />";
16 echo $nama[3]; // Sukma
```

Kode di atas menunjukkan pembuatan dan pengaksesan array dengan indeks yang ditentukan secara manual. Array \$nama dibuat menggunakan fungsi array() dengan indeks dimulai dari 1. Setiap elemen memiliki pasangan indeks dan nilai, misalnya 1 => "Andri", 2 => "Joko", dan seterusnya.

2-byarray2.php

```
1  <?php
2  // Pembuatan array
3  $nama = ["Andri", "Joko", "Sukma", "Rina", "Sari"];
4
5  // Pengaksesan array
6  echo $nama[1]; //Joko
7  echo "<br />";
8  echo $nama[2]; //Sukma
9  echo "<br />";
10 echo $nama[3]; //Rina
```

Kode di atas menunjukkan cara membuat dan mengakses array menggunakan short array syntax ([]). Pada metode ini, PHP otomatis memberikan indeks mulai dari 0. Oleh karena itu, elemen pertama memiliki indeks 0, elemen kedua indeks 1, dan seterusnya.

Praktikum 7.3 (assignment by reference)

3-byreference1.php

```
1  <?php
2
3  $a = 20;
4  $b = $a;
5
6  echo "$a = $a, \$b = $b";
7  echo "<br />";
8  // Hasil proses: $a = 20, $b = 20
9
10 $a = $a + 5;
11 echo "$a = $a, \$b = $b";
12 echo "<br />";
13 // Hasil proses: $a = 25, $b = 20
14
15 $b = $b + 10;
16 echo "$a = $a, \$b = $b";
17 // Hasil proses: $a = 25, $b = 30
18
```

Kode ini menunjukkan konsep assignment by value, yaitu penugasan berdasarkan salinan nilai. Variabel \$a diberi nilai 20, lalu nilai tersebut disalin ke \$b dengan pernyataan \$b = \$a;. Karena yang disalin hanya nilainya, perubahan pada \$a tidak memengaruhi \$b, begitu pula sebaliknya. Setelah \$a ditambah 5, nilai \$a menjadi 25 sedangkan \$b tetap 20. Ketika \$b ditambah 10, hasil akhirnya adalah \$a = 25 dan \$b = 30. Hal ini menunjukkan bahwa kedua variabel menyimpan nilai yang terpisah.

3-byreference2.php

```
1 <?php
2
3 $a = 20;
4 $b = &$a;
5
6 echo "$a = $a, \b = $b";
7 echo "<br />";
8 // Hasil proses: $a = 20, $b = 20
9
10 $a = $a + 5;
11 echo "$a = $a, \b = $b";
12 echo "<br />";
13 // Hasil proses: $a = 25, $b = 25
14
15 $b = $b + 10;
16 echo "$a = $a, \b = $b";
17 // Hasil proses: $a = 35, $b = 35
18
```

Kode ini menunjukkan konsep assignment by reference, yaitu penugasan berdasarkan referensi. Variabel \$b dihubungkan langsung dengan \$a menggunakan operator & dalam pernyataan \$b = &\$a;. Dengan demikian, keduanya mengacu pada lokasi yang sama, sehingga perubahan pada salah satu variabel juga memengaruhi yang lain. Setelah \$a ditambah 5, keduanya menjadi 25, dan ketika \$b ditambah 10, keduanya menjadi 35. Ini membuktikan bahwa \$a dan \$b berbagi referensi yang sama, bukan sekadar salinan nilai.

Praktikum 7.4 (aritmatik)

```
1  <?php
2
3  $penjumlahan = 2 + 4;
4  $pengurangan = 6 - 2;
5  $perkalian = 5 * 3;
6  $pembagian = 15 / 3;
7  $modulus = 5 % 2;
8
9  echo "Hasil: 2 + 4 = " . $penjumlahan . "<br>";
10 // Hasil: 2 + 4 = 6
11 echo "Hasil: 6 - 2 = " . $pengurangan . "<br>";
12 // Hasil: 6 - 2 = 4
13 echo "Hasil: 5 * 3 = " . $perkalian . "<br>";
14 // Hasil: 5 * 3 = 15
15 echo "Hasil: 15 / 3 = " . $pembagian . "<br>";
16 // Hasil: 15 / 3 = 5
17 echo "Hasil: 5 % 2 = " . $modulus . "<br>";
18 // Hasil: 5 % 2 = 1
19
```

Kode di atas menunjukkan penggunaan operator aritmatika dalam PHP. Program melakukan lima operasi dasar matematika, yaitu penjumlahan (+), pengurangan (-), perkalian (*), pembagian (/), dan modulus (%). Setiap hasil operasi disimpan dalam variabel terpisah, kemudian ditampilkan menggunakan perintah echo.

Praktikum 7.5 (presedensi)

```
1  <?php
2
3  $a = 3 + 4 * 5 - 6;
4  echo $a;
5  // Hasil $a = 17
6  echo "<br />";
7  $a = (3 + 4) * 5 - 6;
8  echo $a;
9  // Hasil $a = 29
10
```

Kode di atas menunjukkan konsep presedensi operator dalam PHP. Pada ekspresi $\$a = 3 + 4 * 5 - 6$, operasi perkalian dilakukan lebih dulu sehingga hasilnya 17. Namun, saat ditulis $\$a = (3 + 4) * 5 - 6$, tanda kurung mengubah urutan operasi sehingga hasilnya menjadi 29.

Praktikum 7.6 (inc/decrement)

```
1  <?php
2
3  $x = 4;
4  $x++;
5  echo "Nilai x yang baru : " . $x;
6  echo "<br />";
7  // Hasil $x = 5
8  $x = 4;
9  $x--;
10 echo "Nilai x yang baru : " . $x;
11 // Hasil $x = 3
12
```

Kode di atas menunjukkan penggunaan operator increment dan decrement dalam PHP. Operator ++ digunakan untuk menambah nilai variabel sebesar 1, sedangkan -- digunakan untuk mengurangnya sebesar 1. Pada contoh pertama, nilai \$x yang awalnya 4 menjadi 5 setelah \$x++. Pada contoh kedua, nilai \$x kembali diatur ke 4 lalu dikurangi satu dengan \$x--, sehingga hasil akhirnya adalah 3.

Praktikum 7.7 (contoh studi kasus)

script5-1.php

```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Menghitung Komisi Salesman</title>
8 </head>
9
10 <body>
11   <h1>Menghitung Komisi Salesman</h1>
12   <?php
13
14     /*
15     Script ini akan menghitung komisi salesman berdasarkan nilai penjualan
16     yang dicapainya yang sebesar Rp. 1.500.000,-
17     Ketentuan komisinya adalah 5% dari nilai penjualan yang dicapai.
18     */
19     $nilaiJual = 1500000; // Nilai penjualan yang didapat salesmen
20     $komisi = 0.05; // Menghitung komisi yaitu 5% dari nilai penjualan
21     echo "<p>Nilai penjualan salesman : Rp. " . $nilaiJual . "</p>"; // Menampilkan nilai penjualan salesman
22     echo "<p>Komisi yang didapat salesman adalah Rp. " . $komisi . "</p>";
23     // Menampilkan hasil perhitungan komisi
24   ?>
25 </body>
26
27 </html>
```

Kode ini menghitung komisi salesman berdasarkan nilai penjualan sebesar Rp1.500.000 dengan persentase komisi 5%. Hasil perhitungan komisi ditampilkan di halaman web. Program ini menggunakan operasi aritmatika sederhana untuk mengalikan nilai penjualan dengan persentase komisi.

script5-2.php

```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5     <meta charset="UTF-8">
6     <meta name="viewport" content="width=device-width, initial-scale=1.0">
7     <title>Menghitung Gaji Bersih Karyawan</title>
8 </head>
9
10 <body>
11     <h1>Menghitung Gaji Bersih Karyawan</h1>
12     <?php
13
14         /*
15          Script ini akan menghitung gaji bersih karyawan yang dirumuskan dengan
16          Gaji Bersih = gaji Pokok + tunjangan - pajak;
17          Misalkan gaji pokoknya Rp. 1.000.000, tunjangan Rp. 500.000 dan
18          pajaknya 15% dari (gaji kotor = gaji pokok + tunjangan)
19          Berikut ini ada beberapa cara pembuatan script yang akan menghasilkan
20          output yang sama
21          */
22         // CARA KE - 1
23         $gajiPokok = 1000000; // Gaji pokok
24         $tunjangan = 500000; // Tunjangan
25         $gajiKotor = $gajiPokok + $tunjangan; // Hitung gaji kotor
26         $pajak = 0.15 * $gajiKotor; // Hitung gaji bersih
27         $gajiBersih = $gajiPokok + $tunjangan - $pajak; // Hitung gaji bersih
28         echo "<p>Gaji bersih karyawan adalah Rp. " . $gajiBersih . "</p>"; // Menampilkan gaji bersih
29         // CARA KE - 2
30         $gajiPokok = 1000000; // Gaji pokok
31         $tunjangan = 500000; // Tunjangan
32         $gajiKotor = $gajiPokok + $tunjangan; // Hitung gaji kotor
33         $gajiBersih = $gajiKotor - (0.15 * $gajiKotor); // Hitung gaji bersih
34         echo "<p>Gaji bersih karyawan adalah Rp. " . $gajiBersih . "</p>"; // Menampilkan gaji bersih
35         // CARA KE - 3
36         $gajiPokok = 1000000; // Gaji pokok
37         $tunjangan = 500000; // Tunjangan
38         $gajiBersih = $gajiPokok + $tunjangan - 0.15 * ($gajiPokok + $tunjangan); // Hitung gaji bersih
39         echo "<p>Gaji bersih karyawan adalah Rp. " . $gajiBersih . "</p>"; // Menampilkan gaji bersih
40     ?>
41 </body>
42
43 </html>
```

Kode ini menghitung gaji bersih karyawan berdasarkan rumus:

Gaji Bersih = Gaji Pokok + Tunjangan - Pajak.

Pajak dihitung sebesar 15% dari gaji kotor. Program menunjukkan tiga cara penulisan berbeda yang menghasilkan hasil sama. Tujuannya adalah memahami perhitungan bertahap dan variasi penulisan ekspresi aritmatika di PHP.

script5-3.php

```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Konversi Waktu Tempuh Ke Detik</title>
8 </head>
9
10 <body>
11   <h1>Konversi Waktu Tempuh Ke Detik</h1>
12   <?php
13
14     /*
15     Script ini akan mengkonversi waktu yang dinyatakan dalam 10:16:42 (10 jam, 16 menit dan 42 detik) ke dalam satuan detik.
16     */
17
18     $jam = 10;
19     $menit = 16;
20     $detik = 42;
21     $jamKeDetik = $jam * 3600; // Konversi jam ke detik
22     $menitKeDetik = $menit * 60; // Konversi menit ke detik
23     $detikKeDetik = $detik; // Konversi ke detik
24     $totalDetik = $jamKeDetik + $menitKeDetik + $detikKeDetik; // Hitung total waktu dalam detik
25     echo "<p>Jika waktu " . $jam . ":" . $menit . ":" . $detik . " dinyatakan dalam satuan detik adalah : " . $totalDetik . "</p>"
26
27   ?>
28 </body>
29
30 </html>
```

Kode ini mengubah waktu yang dinyatakan dalam jam, menit, dan detik (contohnya 10:16:42) menjadi total detik. Prosesnya menggunakan rumus konversi:

total detik = (jam × 3600) + (menit × 60) + detik.

Hasil akhirnya menampilkan total waktu dalam satuan detik.

script5-4.php

```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Konversi jumlah detik ke satuan jam-menit-detik</title>
8 </head>
9
10 <body>
11   <h1>Konversi jumlah detik ke satuan jam-menit-detik</h1>
12   <?php
13
14     /*
15     Script ini merupakan kebalikan dari script5-3.php
16     Script ini akan mengkonversi waktu yang diketahui dalam satuan detik
17     ke dalam satuan jam-menit-detik;
18     Diketahui waktu dalam detik adalah 15789 detik, akan dikonversi ke
19     bentuk x jam, y menit dan z detik
20     */
21     $totalDetik = 15789; // Jumlah total detik mula-mula
22     // Mencari waktu dalam jam
23     $sisas = $totalDetik % 3600;
24     $dalamJam = ($totalDetik - $sisas) / 60;
25     // Sisa dari perhitungan jam digunakan untuk menghitung menitnya
26     $totalDetik = $sisas;
27     $sisas = $totalDetik % 60;
28     $dalamMenit = ($totalDetik - $sisas) / 60;
29     // Sisa dalam perhitungan menit digunakan untuk menghitung detiknya
30     $totalDetik = $sisas;
31     $sisas = $totalDetik % 60;
32     $dalamDetik = ($totalDetik - $sisas) / 60;
33     echo "<p>Hasil konversinya adalah : " . $dalamJam . " jam: " . $dalamMenit . " menit : " . $dalamDetik . " detik</p>";
34
35   ?>
36 </body>
37
38 </html>
```

Kode ini mengambil Waktu awal dalam detik (misalnya 15789 detik) dikonversi menjadi jam, menit, dan detik. Prosesnya menggunakan operasi modulus (%) dan pembagian (/) untuk memisahkan satuan waktu dari total detik.

script5-5.php

```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Menghitung selisih dua buah waktu</title>
8 </head>
9
10 <body>
11   <h1>Menghitung selisih dua buah waktu</h1>
12   <?php
13
14     /*
15      Script ini akan mencari selisih antara waktu 10:34:45 dengan 12:25:31
16      Hasil selisih waktu dinyatakan dalam detik
17     */
18     $jam1 = 12;
19     $menit1 = 25;
20     $detik1 = 31;
21     $jam2 = 10;
22     $menit2 = 34;
23     $detik2 = 45;
24     $totalDetik1 = $jam1 * 3600 + $menit1 * 60 + $detik1; // Menghitung total detik untuk waktu pertama
25     $totalDetik2 = $jam2 * 3600 + $menit2 * 60 + $detik2; // Menghitung total detik waktu kedua
26     $selisih = $totalDetik1 - $totalDetik2; // Hitung selisih total detik dari kedua waktu
27     echo "<p>Selisih dari kedua waktu adalah " . $selisih . " detik</p>"
28
29   ?>
30 </body>
31
32 </html>
```

Kode ini menghitung selisih antara dua waktu (misalnya 12:25:31 dan 10:34:45). Kedua waktu dikonversi terlebih dahulu ke satuan detik, kemudian dicari selisihnya. Hasil akhir menunjukkan perbedaan waktu dalam satuan detik.

Soal1

```
1 <?php
2 $saldoAwal = 1000000;
3 $bunga = 0.0025;
4 $bulan = 11;
5 $saldoAkhir = $saldoAwal + ($saldoAwal * $bunga + $bulan);
6 echo "Saldo akhir setelah " . $bulan . " bulan adalah : Rp. " . $saldoAkhir . ",-";
7
```

Kode di atas untuk menghitung saldo akhir tabungan dengan bunga bulanan, namun terdapat kesalahan pada rumus perhitungannya. Penjumlahan dengan variabel \$bulan seharusnya diganti dengan perkalian agar bunga dihitung berdasarkan lama waktu tabungan. Rumus yang benar adalah $\$saldoAkhir = \$saldoAwal + (\$saldoAwal * \$bunga * \$bulan)$;, sehingga hasil perhitungan saldo akhir menjadi sesuai dengan jumlah bulan dan persentase bunga yang berlaku.

Soal2

```
1 <?php
2 $jumlahUang = 1575250;
3
4 $a = (int) ($jumlahUang / 100000);
5 $jumlahUang = $jumlahUang % 100000;
6
7 $b = (int) ($jumlahUang / 50000);
8 $jumlahUang = $jumlahUang % 50000;
9
10 $c = (int) ($jumlahUang / 20000);
11 $jumlahUang = $jumlahUang % 20000;
12
13 $d = (int) ($jumlahUang / 5000);
14 $jumlahUang = $jumlahUang % 5000;
15
16 $e = (int) ($jumlahUang / 100);
17 $jumlahUang = $jumlahUang % 100;
18
19 $f = (int) ($jumlahUang / 50);
20 $jumlahUang = $jumlahUang % 50;
21
22 echo "Jumlah Rp. 100.000 : " . $a . "<br />";
23 echo "Jumlah Rp. 50.000 : " . $b . "<br />";
24 echo "Jumlah Rp. 20.000 : " . $c . "<br />";
25 echo "Jumlah Rp. 5.000 : " . $d . "<br />";
26 echo "Jumlah Rp. 100 : " . $e . "<br />";
27 echo "Jumlah Rp. 50 : " . $f . "<br />";
28
```

Kode di atas untuk menghitung pecahan uang dari sejumlah nominal tertentu, dalam hal ini sebesar Rp1.575.250. Program bekerja dengan cara membagi jumlah uang berdasarkan nilai pecahan terbesar terlebih dahulu, yaitu Rp100.000, kemudian menghitung sisa uang menggunakan operator modulus (%) untuk dilanjutkan ke pecahan berikutnya (Rp50.000, Rp20.000, Rp5.000, Rp100, dan Rp50). Hasil akhirnya menampilkan jumlah masing-masing pecahan yang dibutuhkan untuk mencapai total uang tersebut.