NAMA MATA KULIAH

UNIVERSITAS
INABA

MODUL – BAB XI

PENGULANGAN DAN FUNGSI

Dedy Aridarma, S.Kom, M.Kom

PERTEMUAN 11

PENGULANGAN DAN FUNGSI

Kemampuan yang diharapkan :

1. Mampu menerapkan konsep pengulangan dan fungsi dalam memecahkan suatu kasus

---

Pengulangan:

Perulangan dalam istilah lain disebut loop adalah fungsi dipakai dalam mengerjakan perintah program untuk mengerjakan perintah / pola yang sama secara berulang-ulang. Manfaat menggunakan perulangan di PHP adalah effisien dalam mengerjakan suatu perintah program. Nantinya program akan berjalan dan berhenti sesuai dengan kondisi yang telah diatur

- **Pernyataan While**

  While adalah perintah pada PHP dan bahasa pemrograman lainnya untuk membuat sebuah perulangan yang tidak di ketahui berapa kali perulangan tersebut di lakukan. Sering kali pada saat menuliskan kode program kita membutuhkan perintah perulangan while ini untuk membuat perulangan yang tidak perlu kita ketahui berapa kali perulangan harus di lakukan. misalnya seperti pada saat kita ingin menampilkan data dari database. Perulangan ini berguna untuk memproses suatu pernyataan atau beberapa pernyataan secara berulang-ulang hingga kondisi terpenuhi, Perulangan while memiliki 2 bagian yang harus anda tulis, antara lain :

  - Kondisi yang harus terpenuhi (Bernilai True) agar proses perulangan dilakukan
  - Baris perintah (Statement) yang akan diproses secara berulang, ketika kondisi bernilai true

Bentuk pernyataan while:

```
<?php
While (ekspresi)
{
Pernyatan_pernyataan
}
?>
```

- Kondisi adalah kondisi yang harus dipenuhi agar perulangan dapat dilakukan, selama kondisi bersifat TRUE maka perulangan akan tetap dijalankan, tetapi jika bersiftar FALSE perulangan akan berhenti, konsepnya sama dengan parameter kedua dari fungsi FOR
- Statement yang diulang adalah statement yang dijalankan ketika perulangan terjadi, didalam statement ini harus terdapat pemicu yang menyebabkan kondisi bernilai FALSE, agar tidak terjadi perulangan yang tidak pernah berhenti atau biasa dikenal dengan istilah (infinity loop).
- Tanda kurung kurawal pembuka dan penutup yang membatasi blok statement yang diulang diperlukan jika statement lebih dari 1 baris, tetapi jika hanya 1 baris anda bisa menghilangkan tanda kurung kurawal pembuka dan penutup

```php
<?php
//perulangan while
$i=1;
while ($i <= 10)
{
  echo "Bilangan $i";
  echo "<br />";
  $i=$i+1;

}


$x = 1;

while($x < 10) {
    echo "Angka $x <br>";
    $x++;
    }
?>
```

Perulangan didalam perulangan (Nested Looping) dengan While

menuliskan perulangan didalam perulangan dengan menggunakan while, misal akan menampilkan angka 1-10 sebanyak 10 baris

```
<?php
$a = 1;
while($a < 11)
{
$b = 1;
while($b < 11)
{
echo $b.", ";
$b++;
}
$a++;
echo "<br/>";
}
?>
```

Skip loop pada nilai tertentu

Dengan perulangan while dapat melakukan skip terhadap nilai tertentu ketika looping sedang berjalan, bisa menggunakan perintah continue; untuk melakukan skip pada proses perulangan.

```
<?php
$a = 1;
while($a < 10)
{
if ($a == 5) {
$a++;
continue;
}
echo $a;
$a++;
}?>
```

- **Pernyataan do – while**

  Perulangan Do While juga memiliki fungsi yang sama. Namun yang menjadi perbedaannya adalah pada cara kerjanya. Struktur perulangan While, pengecekan kondisi perulangan dilakukan di awal, sehingga jika kondisi tidak terpenuhi (bernilai False), maka perulangan tidak akan dijalankan. Struktur perulangan Do While berbeda, perulangan akan dijalankan terlebih dahulu, baru dilakukan pengecekan kondisi, sehingga perulangan akan tetap dilakukan minimal 1 kali, meskipun kondisi tidak terpenuhi (bernilai False).

  struktur dasar dari penulisan do-while di PHP

```
<?php
do {
statement yang diulang;
statement yang diulang;
}
 while (condition);
?>
```

  Struktur Do While dimulai dengan perintah do, berikutnya dibuat tanda kurung kurawal pembuka dan penutup, diantara tanda kurung kurawal tersebut dapat menuliskan statement yang akan dijalankan ketika perulangan dilakukan, berikutnya anda dapat menuliskan baris program yang digunakan untuk menghentikan perulangan agar tidak terjadi infinity loop. Setelah itu baru dilakukan pengecekan kondisi, selama pengecekan kondisi nilainya TRUE maka perulangan akan dijalankan, tetapi jika nilainya FALSE maka perulangan akan berhenti, tetapi seperti yang dijelaskan diawal Struktur Do – While ini akan melakukan perulangan minimal 1 kali, meskipun nilai pengecekan kondisi bernilai FALSE.

```php
<?php
/* perulangan Do while */
$i = 0;
do {
  echo"pemrograman web <br>"; // blok akan di jalankan dulu 1x
  $i++;
} while($i < 5); //kondisi
echo "<hr />";
/* contoh 1 */

$i = 1;
do
{
    echo $i;
    echo "<br/>";
    $i++;
}
while($i < 10);
?>
```

- **Pernyataan for**

  Perulangan for adalah perulangan yang mengerjakan suatu pernyataan scara berulang-ulang. Jenis perulangan ini mempunyai 3 parameter penting yang pertama ada inisialisasi nilai awal. Kondisi untuk set perulangan akan berhenti ketika kondisi telah terpenuhi. Yang terakhir ada modifier untuk menambahkan nilai inisialisasi awal tersebut. Pada struktur perulangan FOR, terdapat 3 komponen yang perlu anda tulis antara lain :

  - Kondisi awal dari perulangan
  - perintah program yang akan dijalankan secara berulang
  - kondisi yang harus terpenuhi agar perulangan berhenti

  Struktur penulisan perulangan for PHP

```
<?php
for (kodisi awal; kondisi agar perulangan berhenti; increment)
{
statement yang dijalankan ketika perulangan berlangsung;
}
?>
```

- Kondisi awal adalah kondisi awal pada saat perulangan dilakukan, dapat membuat variabel yang digunakan untuk menyimpan nilai awal, yang akan dibandingkan nilainya pada parameter kedua for, yang membuat perulangan dilakukan atau berhenti
- Kondisi agar perulangan berhenti adalah kondisi yang harus dipenuhi agar perulangan dijalankan. selama kondisi yang dituliskan dibagian ini terpenuhi maka PHP akan tetap menjalankan statement yang dijalankan ketika perulangan berlangsung, kondisi ini biasanya membandingkan dengan variabel kondisi awal, semisal pada kondisi awal kita menuliskan $i = 1; lalu pada bagian kondisi agar perulangan berhenti kita tuliskan $i < 11; maka selama variabel $i nilainya kurang dari 11 maka perulangan akan tetap dijalankan.
- Increment adalah bagian yang digunakan untuk menjadikan variabel kondisi awal agar dapat memenuhi kondisi agar perulangan berhenti, anda bisa melakukan increment ataupun decrement pada variabel kondisi awal.
- Statement yang dijalankan ketika perulangan berlangsung, statement ini akan dijalankan ketika proses perulangan berlangsung. untuk statement ini ditulis diantara tanda kurung kurawal pembuka **({)** dan tanda kurung kurawal penutup **(})**


```php
<?php
/* perulangan For */
for($angka = 1; $angka <= 10; $angka++){
    echo "<h5>Angka ke-$angka</h5>";
}
echo "<hr />";
/* perulangan For menggunakan tanda titik : */
for( $i = 0; $i < 10; $i++ ):
echo "Pemrograman Web <br>";
endfor;
echo "<hr />";

/* Menuliskan Angka 1 - 10; */
for($i = 1;$i < 11;$i++)
{
    echo $i;
    echo "<br/>";
}
?>
```

- Pernyataan break
  - ✓ Ketika proses perulangan berjalan, ingin segera keluar dari perulangan jika sebuah kondisi tertentu telah terpenuhi, sehingga sisa proses perulangan tidak perlu dijalankan.
  - ✓ Untuk keperluan inilah PHP menyediakan instruksi **break**. **Break** berfungsi sebagai perintah kepada web server untuk menghentikan perulangan secara prematur, yaitu menghentikan perulangan di luar dari yang direncanakan.
  - ✓ Perintah break dapat di letakkan di posisi manapun di dalam perulangan, namun biasanya kita akan membuat logika IF untuk menentukan kapan perintah break akan dijalankan.

```php
/* pernyataan break */
$i=0;
while ($i < 100)
{
  $i++;
  if ($i==13)
  {
  break;
  }
echo $i;
    }

    echo "<hr />";
    for ($i=0; $i <10; $i++)
    {
      for ($j=0; $j <10; $j++)
      {
        if ($i==4)
        {
          break;
        }
        echo $i;
      }
        echo "<br />";
    }
```

- Pernyataan Continue
  - ✓ Perintah continue juga digunakan untuk men-interupsi perulangan dalam PHP, namun jika perintah break digunakan untuk menghentikan perulangan, maka perintah continue hanya akan menghentikan perulangan untuk 1 iterasi saja, lalu proses perulangan akan dilanjutkan.
  - ✓ Arti dari continue ini adalah sebuah instruksi kepada PHP untuk melewati sisa perintah dalam perulangan, dan langsung lompat ke nilai counter berikutnya

```php
/* pernyataan continue */
for ($i=0; $i <10; $i++)
{
  if ($i==5)
  {
  continue;
  }
echo $i;
echo "<br />";
}
echo "<hr />";

for ($i=0; $i <10; $i++)
{
  for ($j=0; $j <10; $j++)
  {
      if ($i==4)
        {
        continue 2;
        }
      echo $i;
  }
echo "<br />";
}
?>
```

- Pernyataan Exit

  Exit di dalam PHP berfungsi untuk menghentikan atau membatalkan semua perintah dari sebuah fungsi atau script PHP yang kita buat, maksudnya yaitu jika kita memiliki sebuah proses dan jika kita ingin menghentikan proses tersebut kita bisa menggunakan yang nama exit sehingga proses tersebut tidak bisa berjalan, dan fungsi dari exit ini juga dapat kita gunakan untuk pengecekan dalam sebuah program atau sistem yang kita rancang. Exit tersebut merupakan sebuah fungsi jadi untuk menuliskan fungsi maka kita menggunakan sepasang tanda kurung di belakang yaitu exit(). Dan fungsi dari exit() juga hampir sama dengan die() yaitu sama-sama untuk menghentikan atau membatalkan sebuah perintah ataupun proses.

```php
    /* pernyataan exit */
for($i = 1; $i <= 25; $i++)
  {
     echo "$i <br>";
     if ($i == 9)
        exit;
  }
  echo "selesai <br>";
```

- Sintaks Alternatif dalam pengulangan

  alternatif sintaks untuk beberapa struktur kendali seperti perulangan dan percabangan (pengkondisian) yang bisa diawali dengan colon **(:)** dan diakhiri dengan **endfor; endif; endwhile; endswitch; endforeach;** tergantung dengan struktur kendali yang digunakan. Alternatif sintaks ini sangat berguna didalam perulangan atau percabangan bersarang (nested).

```php
<?php
    /*Standar sintaks*/
        echo "Ini Standar sintaks";
        echo "<hr>";
    ?>


    <?="Ini Alternatif sintaks";
    /*Alternatif sintaks*/
    echo "<hr>";
    ?>


    <?php
    /*Standar sintaks for*/
        for($i=1; $i<=5; $i++){
            echo "Nilai ke $i <br>";
        }
        echo "<hr>";
    ?>


    <?php
    /*Alternatif sintaks for*/
        for($i=1; $i<=5; $i++):
            echo "Nilai ke $i <br>";
        endfor;
        echo "<hr>";
    ?>
```


Fungsi

- Pengertian Fungsi
  - ✓ Fungsi adalah suatu kumpulan blok kode, yang menerima suatu inputan, melakukan satu tugas tertentu, dan secara opsional ia bisa mengembalikan suatu nilai.
  - ✓ Banyak fungsi build-in dari php yang sering digunakan, seperti **print()**, **print_r()**, **unset()**, dll. Selain fungsi-fungsi tersebut, dapatjuga membuat fungsi sendiri sesuai kebutuhan.
  - ✓ Fungsi pada PHP dapat dibuat dngan kata kunci function, lalu diikuti dengan nama fungsinya.

- ✓ Kode intruksi dapat di tulis di dalam kurung kurawal **({...}).**

- Deklarasi Fungsi
  - ✓ Fungsi pada PHP memiliki nama, dan juga bisa menerima parameter.
  - ✓ Peraturan pemberian nama fungsi, sama dengan nama variabel. Hanya saja tidak bisa menggunakan nama fungsi yang sama seperti fungsi bawaan php seperti misalnya **var_dump, empty, count**, dan lain-lain. Kita juga tidak bisa mendefinisikan satu nama fungsi yang sama sebanyak 2 kali
  - ✓ Parameter fungsi adalah suatu nilai yang kita masukan kedalam sebuah fungsi, nilai tersebut bisa berupa apa saja. Bisa berupa **string**, **boolean**, **integer**, bahkan ia juga bisa berupa fungsi yang lainnya
  - ✓ Pembuatan fungsi pada php

```php
<?php
/*Membuat Function / Fungsi*/
function salam(){
    echo "Asalamu'alaikum";
}

/*Memanggil Function / Fungsi*/
salam();

 ?>
```

- ✓ **Function PHP mengembalikan Nilai / Value**

  Fungsi dapat mengembalikan nilai dengan menggunakan statement return bersamaan dengan nilai atau objeknya. Return menghentikan eksekusi function dan mengirimkan nilai kembali ke kode panggilan. Pengembalian nilai dalam fungsi dapat menggunakan kata kunci **return**.

```php
<?php
// membuat fungsi
function hitungUmur($thn_lahir, $thn_sekarang){
  $umur = $thn_sekarang - $thn_lahir;
  return $umur;
}

echo "Umur saya adalah ". hitungUmur(1994, 2015) ." tahun";
?>
```

- Varabel Lokal
  - ✓ Variabel Scope (atau ruang lingkup variabel) adalah jangkauan kode program dimana perintah program masih bisa mengakses sebuah variabel.
  - ✓ Jika mendefenisikan sebuah variabel pada satu file PHP, maka variabel tersebut dapat diakses oleh seluruh kode program pada halaman yang sama. Namun jika variabel tersebut di defenisikan di dalam sebuah fungsi, variabel itu belum tentu bisa diakses dari luar fungsi tersebut.
  - ✓ Variabel yang didefenisikan di dalam sebuah fungsi, secara default tidak dapat diakses oleh kode program di luar fungsi tersebut. Dan begitu juga sebaliknya, variabel yang didefenisikan di luar fungsi, tidak bisa diakses dari dalam fungsi.

```php
 <?php
$nilai = 5; //variable Global

function tampil() {
    $nilai =5; //variable local
echo "$nilai";
echo "<br>";

}
tampil();
echo "$nilai";

?>
```

- Variabel Global
  - ✓ Sebuah variabel dideklarasikan diluar function dan mempunyai global scope. Maka variabel tersebut hanya bisa diakses dari luar function tersebut.
  - ✓ Perintah kata “global” digunakan untuk mengakses sebuah variable global didalam function, Jadi bisa mengakses variable yang telah dideklarasikan diluar function.

```php
<?php
$nilai = 5; //variable Global

function tampil() {
    $nilai =5; //variable local
echo "$nilai";
echo "<br>";

}
tampil();
echo "$nilai";

?>
```

- **Static Variabel**
  - ✓ Variabel statis adalah jenis variabel yang mempertahankan nilainya pada setiap pemanggilan fungsi. Untuk variabel normal, nilai dari variabel tersebut akan secara otomatis dihapus pada saat fungsi selesai dijalankan, dan akan dibuat ulang pada saat fungsi dipanggil.
  - ✓ Namun jika sebuah variabel dinyatakan sebagai static variabel, maka nilai variabel tersebut akan tetap dipertahankan walaupun fungsi telah selesai dijalankan. Biasanya fungsi ini dimanfaatkan jika kita ingin menghitung berapa kali sebuah fungsi dipanggil.

```php
<?php
function statis()
{
   static $a=0;
   $a=$a+1;
   return "Ini adalah pemanggilan ke-$a fungsi static() <br />";
}

echo statis();
echo statis();
echo statis();
echo statis();
?>
```

- **Variabel Super Global**
  - ✓ Variabel yang sudah di sediakan PHP, yang sudah otomatis ada tanpa perlu didefinisikan sendiri, dan ia bersifat global dalam artian bisa kita akses dari mana pun dan kapan pun.
  - ✓ Semua variable super global adalah Array Assosiative
    - $_SERVER
    - $_GET
    - $_POST
    - $_SESSION
    - $_COOKIE
    - $_REQUEST
    - $_ENV