MODUL 3

PEMROGRAMAN WEB

CONTROL STATEMENT

A. TUJUAN PERKULIAHAN

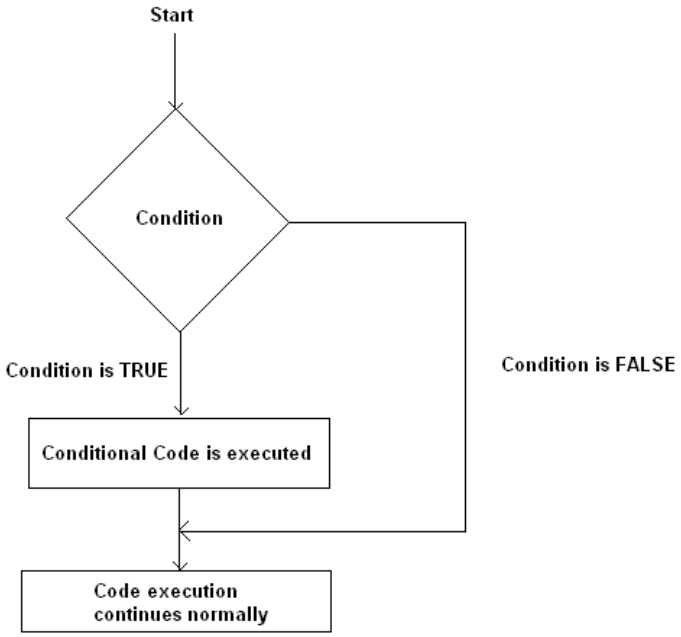
a. Mahasiswa mengenali konsep control statement pada bahasa PHP
b. Mahasiswa mampu mengimplementasikan kontrol perulangan
c. Mahasiswa mampu mengimplementasikan kontrol percabangan

B. ALOKASI WAKTU 3 x 50 menit

C. DASAR TEORI

1. CONTROL STATEMENT

Control statement atau statement kontrol merupakan komponen yang sangat penting dalam programming, karena dapat mengontrol aliran program/script. Program/script tidak hanya terdiri dari sebuah aliran saja (top - down), terkadang terdapat perintah yang harus diulang-ulang sampai beberapa kali. Selain mengulang perintah ada juga perintah yang hanya dijalankan pada kondisi tertentu atau berdasarkan syarat tertentu. Hal ini dapat diatur dengan menggunakan statement kontrol. Berikut adalah flowchart cara kerja control statement di PHP.

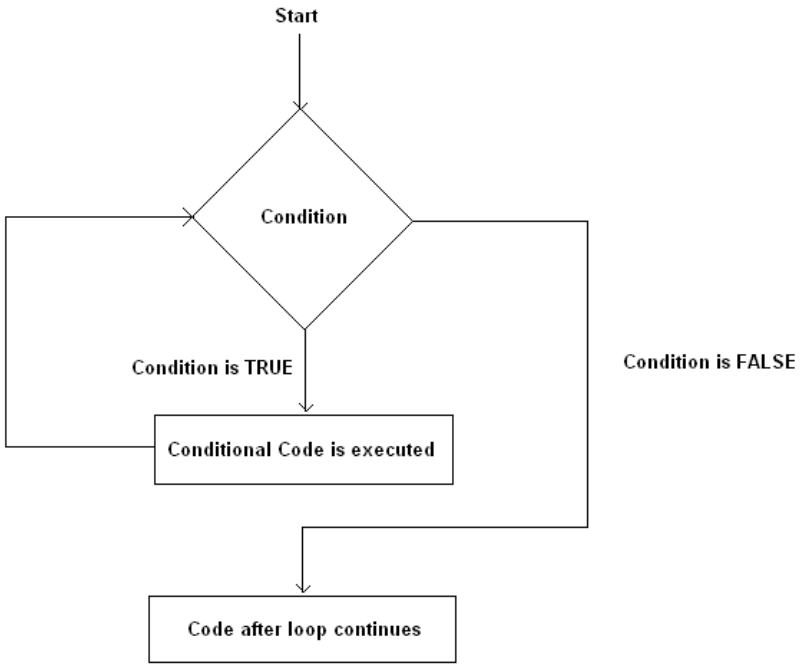


Dua jenis statement kontrol dalam dunia pemrograman terdiri dari statement kontrol kondisional (bersyarat) dan statement kontrol perulangan (looping). Statement kontrol kondisional merupakan statement kontrol yang digunakan untuk mengatur kapan suatu perintah akan dijalankan. Statement ini dapat mengatur kapan suatu perintah akan dijalankan, yaitu ketika telah dipenuhinya suatu syarat tertentu. Sedangkan statement kontrol perulangan digunakan untuk mengatur perintah yang dijalankan secara berulang-ulang. Dalam bahasa pemrograman PHP, terdapat dua buah statement kontrol kondisional, yaitu **IF** dan **SWITCH**. Sedangkan yang termasuk statement kontrol perulangan adalah: **FOR, WHILE, DO WHILE** dan **FOREACH**.

2. PERULANGAN

Perulangan atau loop di PHP berguna ketika Anda ingin mengeksekusi potongan kode pada kondisi true secara berulang-ulang hingga suatu kondisi bernilai false. Jadi sebelum kode dieksekusi, akan dicek dulu nilai kebenaran kondisinya, dan segera setelah kondisi bernilai false, script akan melanjutkan eksekusi kode setelah loop.

Flowchart berikut menjelaskan cara kerja loop di PHP.



a. For

Secara umum, for loop digunakan untuk mengeksekusi potongan kode untuk jumlah waktu tertentu yang paling sering digunakan apabila persyaratan sudah didefinisikan. Dengan kata lain, jika Anda sudah tahu berapa kali Anda ingin mengeksekusi blok kode, itu adalah for loop yang merupakan pilihan terbaik. Berikut pseudocode dari loop.

```
for (expr1; expr2; expr3)
{
  // code to execute
}
```

Ekspresi `expr1` digunakan untuk menginisialisasi variabel, dan itu selalu dijalankan. Ekspresi `expr2` juga dieksekusi di awal loop, dan jika bernilai true, kode loop dieksekusi. Setelah eksekusi kode loop `expr3` dieksekusi. Secara umum, `expr3` digunakan untuk mengubah nilai suatu variabel yang digunakan di ekspresi `expr2`.

Berikut contoh sederhana penggunaan loop

```
<?php
for ($i=1; $i<=10; ++$i)
{
  echo sprintf("The square of %d is %d.</br>", $i, $i*$i);
}
?>
```

Program di atas menampilkan kuadrat dari sepuluh angka pertama. Ini menginisialisasi $i ke 1, berulang selama $i kurang dari atau sama dengan 10, dan menambahkan 1 ke $i pada setiap iterasi.

b. While

Sama seperti for, while loop digunakan ketika Anda ingin mengeksekusi potongan kode berulang kali sampai kondisi while bernilai false. Bedanya adalah parameter / ekspresi yang dibutuhkan hanya satu, yakni kondisi terminasi suatu variabel ekspresi. Posisi inisialisasi variabel ditulis di luar bracket while, sementara proses increment atau decrement ditulis pada body while. Berikut pseudocode dari while.

```
while (expression)
{
  // code to execute as long as expression evaluates to TRUE
}
```

Berikut adalah contoh dari penggunaan while di dalam PHP

```
<?php
$max = 0;
echo $i = 0;
echo ",";
echo $j = 1;
echo ",";
$result=0;

while ($max < 10 )
{
    $result = $i + $j;

    $i = $j;
    $j = $result;

    $max = $max + 1;
    echo $result;
    echo ",";
}
?>
```

Jika Anda terbiasa dengan seri Fibonacci, Anda mungkin mengenali apa yang dilakukan oleh program di atas — ia mengeluarkan seri Fibonacci untuk sepuluh angka pertama. While loop umumnya digunakan ketika Anda tidak tahu jumlah iterasi yang akan terjadi dalam satu loop. Hati-hati dalam penggunaan while, apabila Anda lupa menambahkan kondisi increment / decrement, maka Anda akan berada dalam **LOOPING FOREVER**.

c. Do ... while

Do...while mirip sekali dengan while. Ekspresi yang dibutuhkan hanya satu, yakni kondisi terminasi. Bedanya adalah do...while mengharuskan potongan kode untuk dijalankan terlebih dahulu, baru kemudian dicek kondisi pada akhir eksekusi kode. Apabila kondisi masih bernilai true, maka potongan kode di dalam do...while akan dijalankan lagi. Jika tidak, maka program akan lanjut mengeksekusi kode di bawahnya. Berikut pseudocode do...while.

```
 do
{
  // code to execute
} while (expression);
```

Mari kita lalui ke contoh kasus untuk memahami kemungkinan penggunaan kasus di mana Anda dapat menggunakan do-while loop.

```
<?php
$handle = fopen("file.txt", "r");
if ($handle)
{
    do
    {
        $line = fgets($handle);

        // process the line content

    } while($line !== false);
}
fclose($handle);
?>
```

Pada contoh di atas, program mencoba membaca file baris demi baris. Pertama, program telah membuka file untuk dibaca. Dalam kasus tersebut, kita tidak yakin apakah file tersebut mengandung konten. Jadi, kita perlu menjalankan fungsi fgets setidaknya sekali untuk memeriksa apakah suatu file tersebut berisi konten. Jadi kita bisa menggunakan do-while loop disini. do-while mengevaluasi kondisi setelah iterasi pertama loop.

d. Foreach

Foreach merupakan sintaks percabangan yang digunakan pada variabel array. Berikut pseudocode foreach.

```
foreach ($array as $element)
{
  // code to execute
}

foreach ($array as $key=>$value)
{
  // code to execute
}
```

Foreach membutuhkan parameter berupa variabel array yang kemudian diterjemahkan per elemen. Elemen dapat dipecah menjadi index dan isi dari array

tersebut, atau hanya isinya saja tanpa memperhatikan index. Mari kita lihat beberapa contoh.

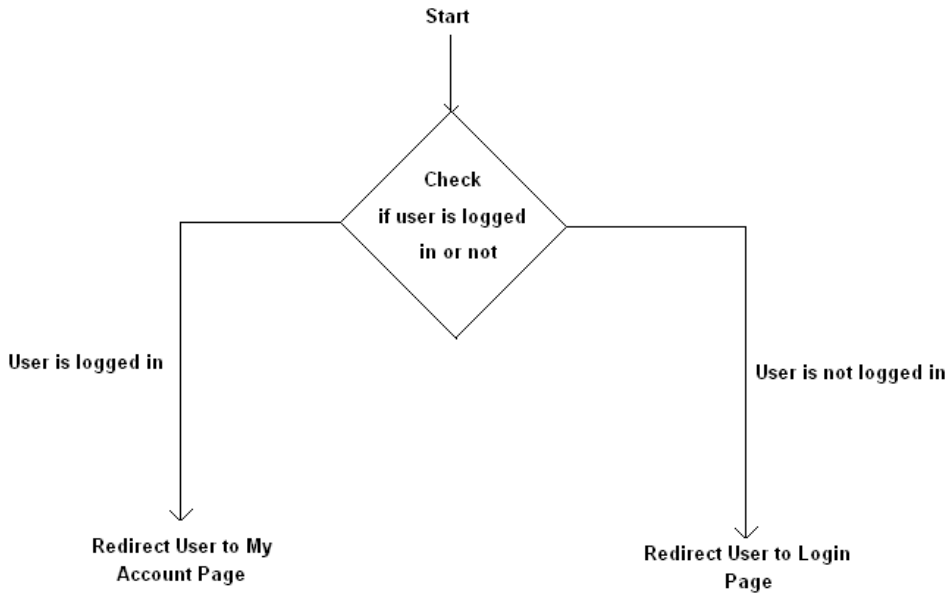
```
<?php
$fruits = array('apple', 'banana', 'orange', 'grapes');
foreach ($fruits as $fruit)
{
  echo $fruit;
  echo "<br/>";
}

$employee = array('name' => 'John Smith', 'age' => 30, 'profession'
=> 'Software Engineer');
foreach ($employee as $key => $value)
{
  echo sprintf("%s: %s</br>", $key, $value);
  echo "<br/>";
}
?>
```

Jika Anda ingin mengakses nilai array, Anda bisa menggunakan versi pertama foreach loop seperti yang ditunjukkan pada contoh di atas. Di sisi lain, jika Anda ingin mengakses kunci index dan nilai, Anda dapat melakukannya seperti yang ditunjukkan pada contoh $employee di atas.

3. PERCABANGAN

Struktur Kontrol Pecabangan merupakan suatu algoritma program memiliki suatu kondisi yang dimana kondisi tersebutlah yang akan menentukan perintah-perintah yang akan dijalankan oleh suatu program. Perintah dalam suatu kondisi akan dijalankan ketika kondisi tersebut bernilai benar, dan sebaliknya apabila kondisi bernilai salah maka perintah didalamnya tidak akan dijalankan. Berikut flowchart percabangan dengan kasus pengecekan user status.



Pada contoh di atas, program memeriksa apakah pengguna login atau tidak. Berdasarkan status login pengguna, mereka akan diarahkan ke halaman **Login** atau halaman **My Account**. Dalam kasus ini, struktur kontrol mengakhiri eksekusi kode dengan mengarahkan pengguna ke halaman lain.

a. If-else

Susunan dari If mengizinkan Anda untuk mengeksekusi sepotong kode jika ekspresi yang diberikan bersama dengan itu bernilai true. Di sisi lain, jika ekspresi bernilai false, itu tidak akan melakukan apa-apa. Lebih sering daripada tidak, Anda juga ingin menjalankan cuplikan kode yang berbeda jika ekspresi bernilai false. Di situlah pernyataan else muncul.

Anda akan selalu menggunakan pernyataan else dalam hubungan dengan pernyataan if . Pada dasarnya, Anda dapat mendefinisikannya seperti yang ditunjukkan pada pseudocode berikut.

```
if (expression)
{
  // code is executed if the expression evaluates to TRUE
}
else
{
  // code is executed if the expression evaluates to FALSE
}
```

Mari kita lihat contoh berikut untuk memahami cara kerjanya.

```
<?php
$age = 50;

if ($age < 30)
{
  echo "Your age is less than 30!";
}
else
{
  echo "Your age is greater than or equal 30!";
}
?>
```

Baris-baris kode di atas menunjukkan percabangan untuk menampilkan salah satu dari dua tulisan, `"Your age is less than 30!"` atau `"Your age is greater than or equal 30!"`. Untuk itu, sebuah kondisi dari variabel harus terpenuhi. Kita jumpai adanya variabel $age yang bernilai 50. Pada baris selanjutnya dicek, jika $age bernilai < 30, maka tulisan pertama akan muncul. Jika tidak tulisan kedua lah yang muncul dengan asumsi bahwa $age sudah pasti di atas 30.

b. If-else if-else

Kita bisa menganggap pernyataan elseif sebagai perpanjangan dari if-else construct. Jika Anda memiliki lebih dari dua pilihan untuk dipilih, Anda dapat menggunakan pernyataan elseif. Mari kita pelajari struktur dasar pernyataan elseif, seperti yang ditunjukkan pada pseudocode berikut.

```
if (expression1)
{
  // code is executed if the expression1 evaluates to TRUE
}
elseif (expression2)
{
  // code is executed if the expression2 evaluates to TRUE
}
elseif (expression3)
{
  // code is executed if the expression3 evaluates to TRUE
}
else
{
  // code is executed if the expression1, expression2 and
expression3 evaluates to FALSE, a default choice
}
```

Mari kita coba memahaminya menggunakan modifikasi contoh pada kasus if-else sebelumnya.

```
<?php
$age = 50;

if ($age < 30)
{
  echo "Your age is less than 30!";
}
elseif ($age > 30 && $age < 40)
{
  echo "Your age is between 30 and 40!";
}
elseif ($age > 40 && $age < 50)
{
  echo "Your age is between 40 and 50!";
}
else
{
  echo "Your age is greater than 50!";
}
?>
```

Seperti yang dapat Anda lihat dalam contoh di atas, kita memiliki beberapa kondisi, jadi kita telah menggunakan serangkaian pernyataan elseif. Jika dalam semua kondisi yang bernilai false, maka eksekusi pada kode yang disediakan di pernyataan else terakhir.

c. Switch

Pernyataan switch agak mirip dengan pernyataan elseif yang berarti ada lebih dari dua kondisi di dalam percabangan tersebut. Satu-satunya perbedaan adalah ekspresi yang sedang diperiksa.

Dalam hal pernyataan elseif Anda memiliki serangkaian kondisi yang berbeda, dan tindakan yang sesuai akan dijalankan berdasarkan suatu kondisi. Di sisi lain, jika Anda ingin membandingkan variabel dengan nilai yang berbeda, Anda dapat menggunakan pernyataan switch. Format penulisan switch dapat dilihat pada pseudocode berikut

```
switch ($param)
{
   case ‘case1’:
         // code is executed if the value equal case1
         break;
   case ‘case2’:
         // code is executed if the value equal case2
         break;
   default:
         //code is executed if nothing is equal the cases above
}
```

Mari kita coba pahami dengan menggunakan contoh

```
<?php
$favourite_site = 'Code';

switch ($favourite_site) {
  case 'Business':
    echo "My favourite site is business.tutsplus.com!";
    break;
  case 'Code':
    echo "My favourite site is code.tutsplus.com!";
    break;
  case 'Web Design':
    echo "My favourite site is webdesign.tutsplus.com!";
    break;
  case 'Music':
    echo "My favourite site is music.tutsplus.com!";
    break;
  case 'Photography':
    echo "My favourite site is photography.tutsplus.com!";
    break;
  default:
    echo "I like everything at tutsplus.com!";
}
?>
```

Seperti yang Anda lihat pada contoh di atas, kita ingin memeriksa nilai variabel $favourite_site dan berdasarkan pada nilai variabel $favourite_site kita ingin cetak pesan.

Untuk setiap nilai yang ingin Anda periksa dengan variabel $favourite_site, Anda harus mendefinisikan case block. Jika nilainya dicocokkan dengan sebuah case, kode yang terkait dengan case block tersebut akan dieksekusi. Setelah itu, Anda perlu menggunakan pernyataan break untuk mengakhiri eksekusi kode. Jika Anda tidak menggunakan pernyataan break, eksekusi skrip akan dilanjutkan hingga blok terakhir di dalam pernyataan switch.

Terakhir, jika Anda ingin mengeksekusi potongan kode yang jika nilai variabel tidak cocok dengan kasus apa pun, Anda dapat mendefinisikannya di bawah default block. Tentu saja, itu tidak wajib — itu hanya cara untuk menyediakan kasus default.

Jadi itulah kisah struktur kontrol bersyarat. Kita akan membahas loop di PHP di bagian selanjutnya.

D. PRAKTIKUM : Statement Control dengan PHP

Kegiatan 1

Cermati screenshot di bawah ini!

1. Buatlah script yang dapat menampilkan tampilan heading di atas dengan menggunakan **LOOPING**! **(simpan sebagai p030101.php).**
2. Modifikasilah script nomor 1 di atas, sedemikian hingga ketika headingnya genap (Heading 2, 4, 6) font colornya diberi warna merah seperti contoh screenshot berikut**!**

(simpan sebagai p030102.php)

Hint :

Untuk membuat heading dapat menggunakan tag sebagai berikut:

```
<h1>...<h1>
<h2>...</h2>
...
<h6>...</h6>
```

Kegiatan 2

Cermati screenshot di bawah ini

elemen 1 - 1	elemen 1 - 2	elemen 1 - 3	elemen 1 - 4	elemen 1 - 5
elemen 2 - 1	elemen 2 - 2	elemen 2 - 3	elemen 2 - 4	elemen 2 - 5
elemen 3 - 1	elemen 3 - 2	elemen 3 - 3	elemen 3 - 4	elemen 3 - 5
elemen 4 - 1	elemen 4 - 2	elemen 4 - 3	elemen 4 - 4	elemen 4 - 5

1. Buatlah script PHP untuk mengenerate tabel berukuran 5 x 6 (baris=5, kolom=6) menggunakan **LOOPING**!.
   (simpan sebagai p030201.php)
2. Modifikasilah script dari nomor 2 di atas, supaya cell dan tampilan angka dibedakan warnanya antara jumlah elemen baris dan kolom adalah angka genap dan ganjil. Hasil yang diharapkan seperti screenshot berikut

elemen 1 - 1	elemen 1 - 2	elemen 1 - 3	elemen 1 - 4	elemen 1 - 5
elemen 2 - 1	elemen 2 - 2	elemen 2 - 3	elemen 2 - 4	elemen 2 - 5
elemen 3 - 1	elemen 3 - 2	elemen 3 - 3	elemen 3 - 4	elemen 3 - 5
elemen 4 - 1	elemen 4 - 2	elemen 4 - 3	elemen 4 - 4	elemen 4 - 5

   ketentuan:
   - Jumlah ganjil: font color merah
   - Jumlah genap: cell color merah

   (simpan sebagai p030202.php)

Hint :

Untuk membuat tabel dengan jumlah baris 3 dan kolom 2 di HTML adalah menggunakan tag sebagai berikut:

```
<table border=”1”>
    <tr><td>Hello</td><td>Hello</td></tr>
    <tr><td>Hello</td><td>Hello</td></tr>
    <tr><td>Hello</td><td>Hello</td></tr>
</table>
```

E. REFERENSI

https://programmingpot.com/php-programming/php-control-structures/

https://code.tutsplus.com/id/tutorials/php-control-structures-and-loops--cms-31999

https://www.w3schools.com/php/