ITM-711 Web Applications

PHP: Control Structures

Outline

- { } Statement
- if…else
- if…elseif
- switch
- break
- while
- do…while
- for

Making Decisions

- Decision making or flow control is the process of determining the order in which statements execute in a program
- The special types of PHP statements used for making decisions are called decision-making statements or decision-making structures

{ } Statement

- A **command block** is a group of statements contained within a set of braces
- Each command block must have an opening brace ({) and a closing brace (})

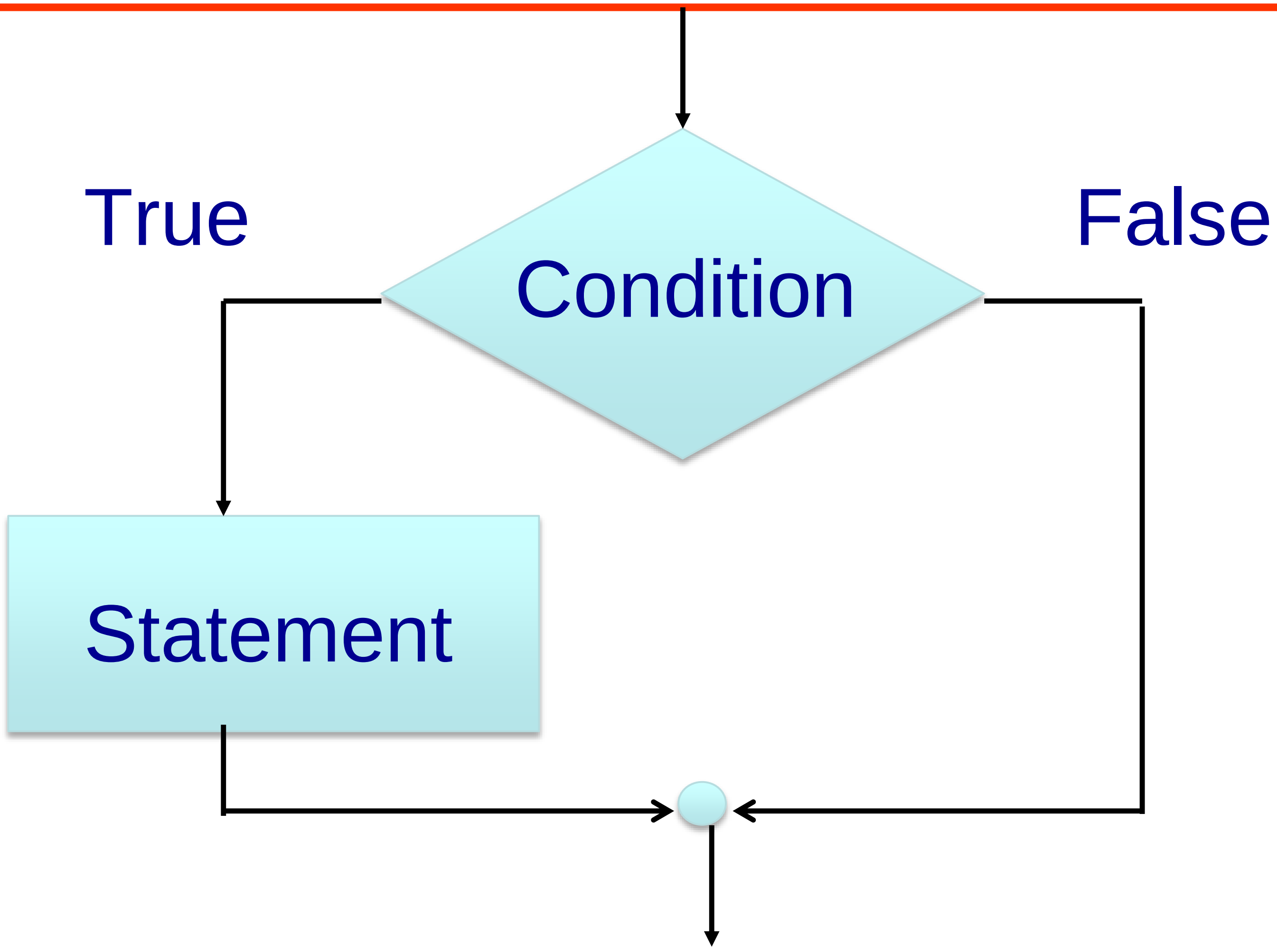
```
{
   statement1;
    statement2;
 }
```

`if` Statements

- Used to execute specific programming code if the evaluation of a conditional expression returns a value of `TRUE`
- The syntax for a simple `if` statement is:

```
if (expression)
    // do something
```

`if` Statements



Example

```
<?php
  if(isset($_GET["myNumber"])) {
     $a = $_GET["myNumber"];
     echo “ตัวเลขที่กรอก คือ ".$a;
     if(a>10)
      echo “ตัวเลขที่กรอกมีค่ามากกว่า 10”;
   }
?>
```

`if...else` Statements

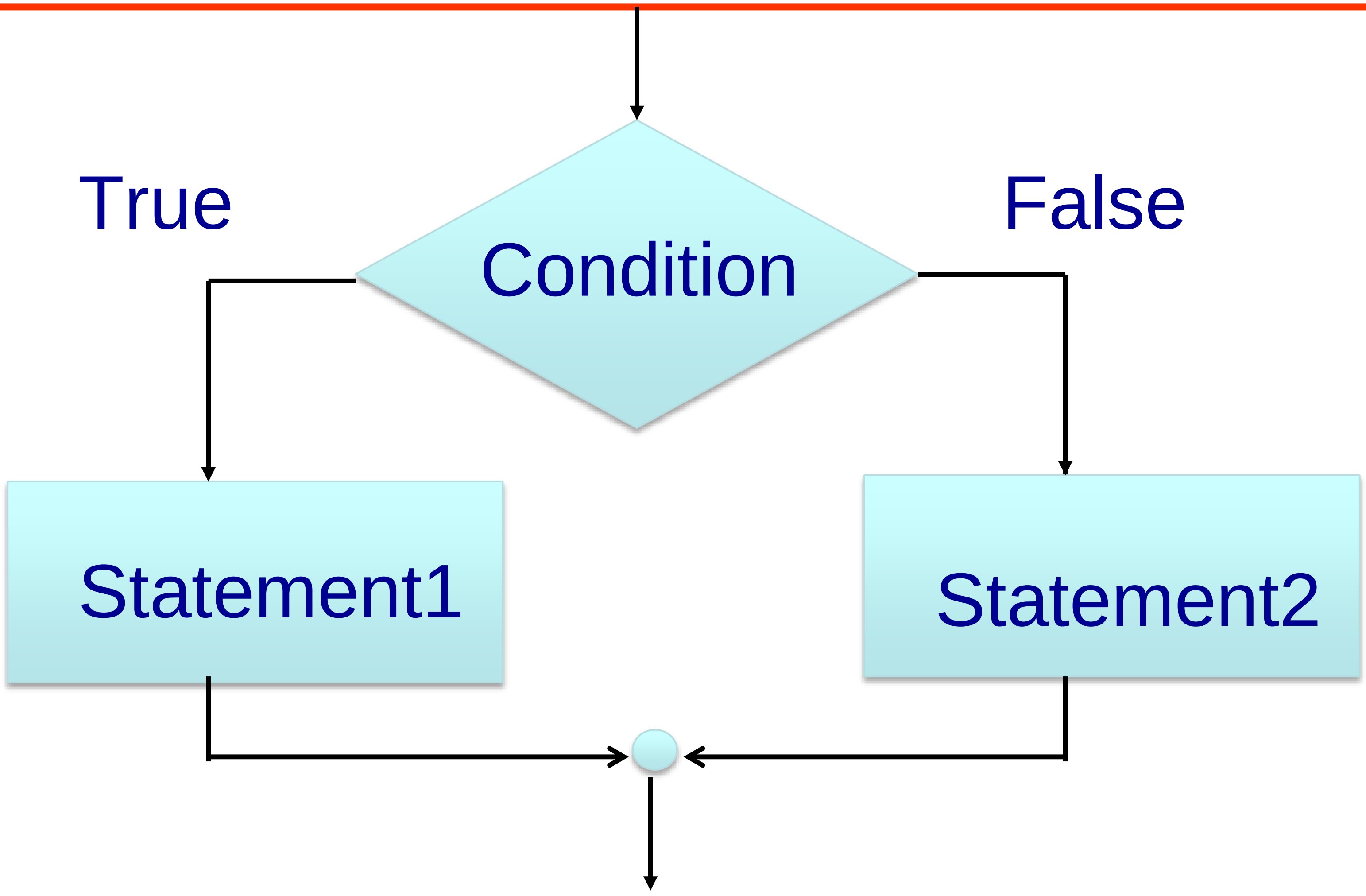
- An `if` statement that includes an `else` clause is called an **`if...else` statement**
- An `else` clause executes when the condition in an `if...else` statement evaluates to `FALSE`

`if...else` Statements

- The syntax for an `if...else` statement is:

```
if (expression)
    // do something
else
    // do another thing
```

`if..else` Statements



`if...else` Statements

- An `if` statement can be constructed without the `else` clause
- The `else` clause can only be used with an `if` statement

Example

```
<?php
     $a =1;
     $b = 2;
     if($a > $b)
      echo “a is greater than b”;
     else
      echo “a is less than or equal to b”;
?>
```

`if...elseif` Statements

- It executes another expression if the first fails
- The syntax for an `if..elseif` statement is:

```
if (expression1)
   // do something
elseif(expression2)
  // do another thing
elseif(expression3)
  // do another thing
         …
else // do another thing
```

Example

```
<?php
     $a =19;
     if($a == 1)
      echo “one”;
     elseif($a == 2)
      echo “two”;
     elseif($a == 3)
      echo “three”;
     elseif($a == 4)
      echo “four”;
     else
      echo “more than four”;
?>
```

`Switch` Statements

- The same variable is compared with many different values
- The `default` statement is used if none of the cases are true
- Statements are execute until it sees a `break` statement

Swith Statements

- The syntax for an `switch` statement is:

```
        switch ($variable_name) {
        case valueA:
          statements;
          break;  // optional
       case valueB:
          statements;
          break;  // optional
       default:
          statements;
   }
```

break

- `break` statement ends execution of the current for, while, do-while or switch structure.
- Break accepts an optional numeric argument which tells it how many nested enclosing structures are to be broken out of

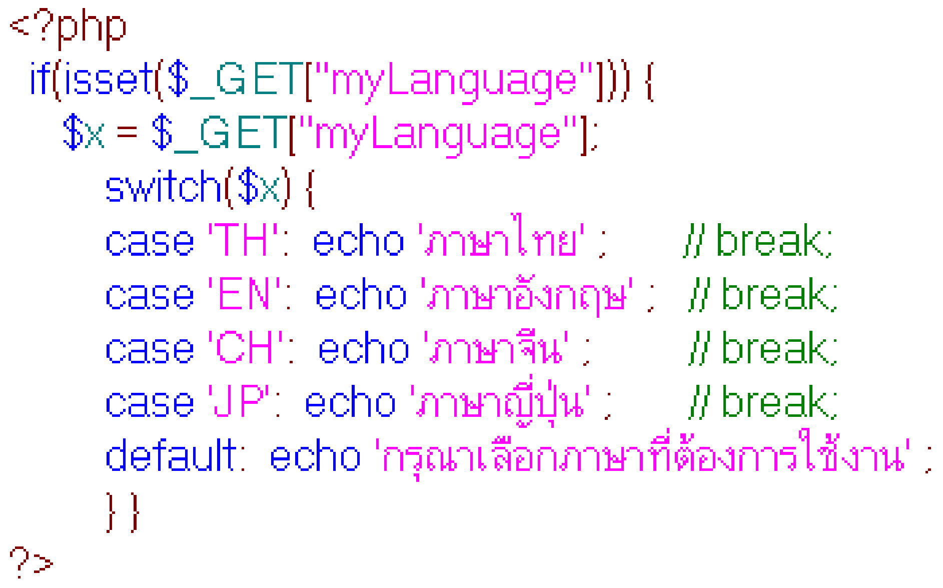
Example

```
<?php
     $a = 2;
     switch($a){
      case 0:
        echo “a equals to 0”;
        break;
      case 1:
        echo “a equals to 1”;
        break;
      default:
        echo “a is greater than 1”;
      }
?>
```

Example

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8"> </meta>
</head>
<body>
 <form action= "switch1.php" method= "GET">
   กรุณากรอกข้อมูลประวัติส่วนตัว:
   <select name = "myLanguage" >
          <option value= "TH"> ภาษาไทย</option>
          <option value= "EN"> ภาษาอังกฤษ</option>
          <option value= "CH"> ภาษาจีน</option>
          <option value= "JP"> ภาษาญี่ปุ่น</option>
      </select>
          <input type = "submit" value = "ตกลง">
         <hr>
</form>
<?php
  if(isset($_GET["myLanguage"])) {
     $x = $_GET["myLanguage"];
         switch($x) {
                 case 'TH':  echo 'ภาษาไทย';   break;
                 case 'EN':  echo 'ภาษาอังกฤษ'; break;
                 case 'CH':  echo 'ภาษาจีน';    break;
                 case 'JP':  echo 'ภาษาญี่ปุ่น';   break;
                 default:  echo 'กรุณาเลือกภาษาที่ต้องการใช้งาน';
                }}
?>
</body>
</html>
```

Example (without break)



```
<?php
 if(isset($_GET["myLanguage"])) {
   $x = $_GET["myLanguage"];
     switch($x) {
     case 'TH':  echo 'ภาษาไทย' ;       // break;
     case 'EN':  echo 'ภาษาอังกฤษ' ;  // break;
     case 'CH':  echo 'ภาษาจีน' ;        // break;
     case 'JP':  echo 'ภาษาญี่ปุ่น' ;       // break;
     default:  echo 'กรุณาเลือกภาษาที่ต้องการใช้งาน' ;
     } }
?>
```

`while`

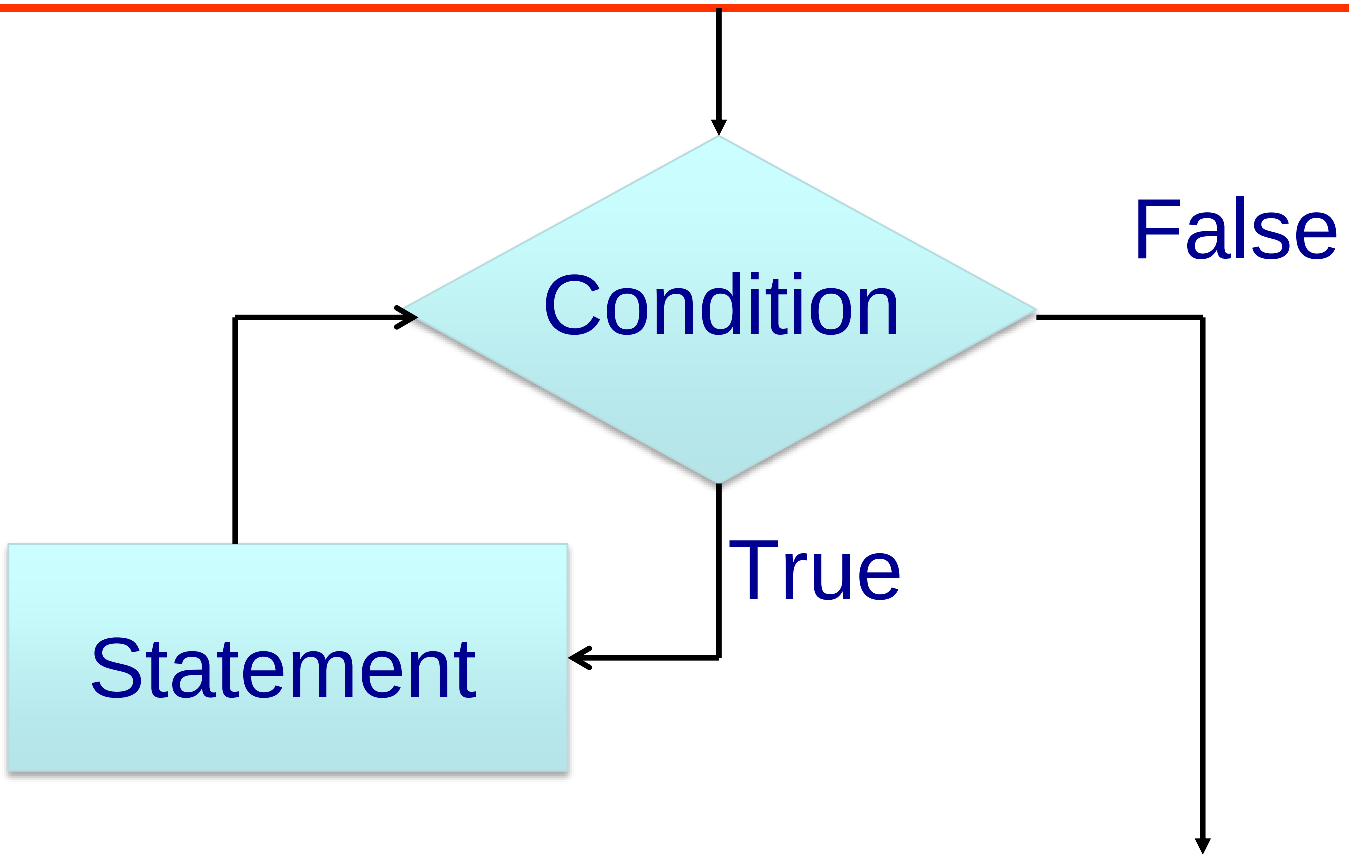
- It tells PHP to execute the nested statements repeatedly, as long as the while expression evaluates to `TRUE`
- If the first evaluation of the statement return `FALSE`, the `while` loop will not be executed at all

`while`

- The syntax for `while` statement is:

```
while (expression){
        statement;
        statement;
}
```

`while`



Example

```
<?php
   $x=1;

   while($x <= 5) {
      echo "The number is: $x <br>";
     $x++;
    }
?>
```

`do...while`

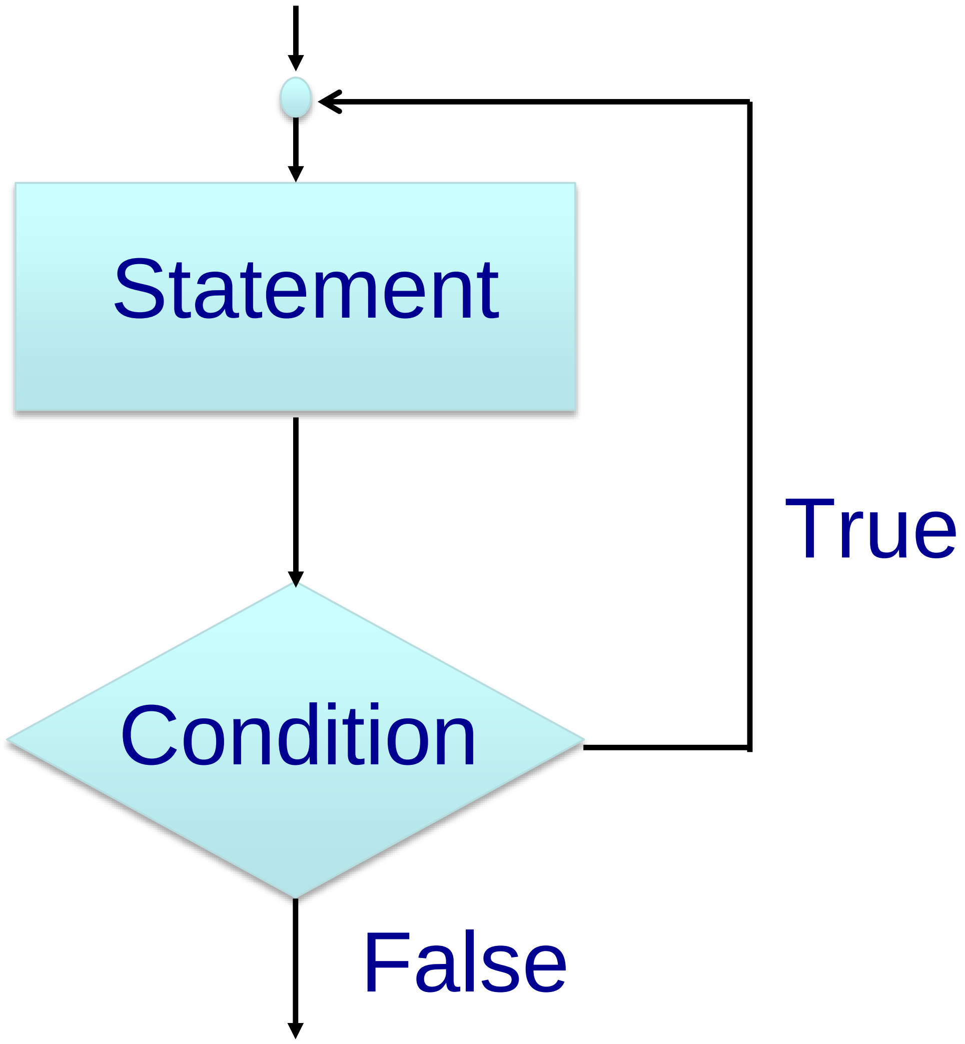
- The statement inside the loop will be executed at least once.
- The truth expression is checked at the end of each iteration instead of in the beginning.

`do...while`

- The syntax for an `do...while` statement is:

```
do{
    statement;
    statement;
 }while(expression);
```

do…while



Example

```
<?php
   $x=1;

  do {
        echo "The number is: $x <br>";
        $x++;
       } while ($x <= 5 );
?>
```

Example

```
<?php
   $x = 6;

   do {
      echo "The number is: $x <br>";
      $x++;
     } while ($x <= 5);
?>
```

`for`

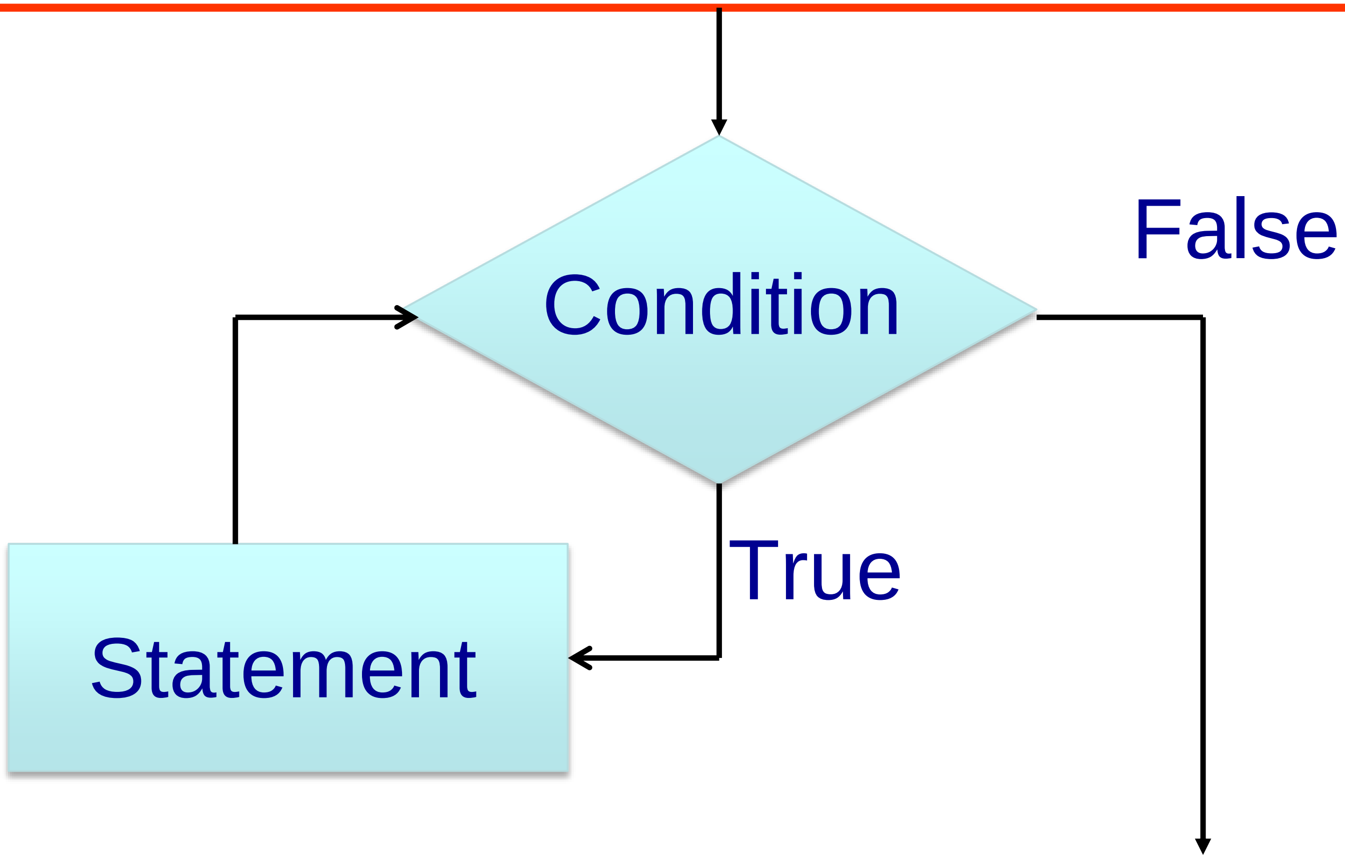
- `for` loop is used if you know how many times you want to execute the statements
- The syntax for `for` statement is:

```
for(initial;condition;inc/dec){
    statement;
    statement;
}
```

`for`

- The first expression `(initial)` is evaluated once unconditionally at the beginning of the loop
- In the beginning of each iteration, the second expression `(condition)` is evaluated. If the result is `True,` the loop continues and the nested statements are executed. If the result is `False,` the loop ends.
- At the end of each iteration, the third expression `(increment/decrement)` is evaluated.

`for`



Example

```
<?php

  for($i = 1; $i < 10; $i++) {
     echo "The number is $i ";
   }

?>
```