Decision-making

Decision-making is about deciding the order of execution of statements based on certain conditions. PHP language handles decision-making by supporting the following statements.

When you write a computer program, regardless of the programming language, you often need to execute different set of statements depending on some satisfied condition. The process of determining the order in which statements execute in a program is called decision-making or flow of Control. The most common type of decision statement is the ***if*** statement which you study following Chapter.

- ***if*** statement
- ***if, else*** *statement*
- ***if, else if. else*** *statement*
- ***switch, case, default*** statement
- **conditional operator** statement

❖ Decision making with if statement

1. if statements

The ***if*** statement may be implemented in different forms depending on the evaluation of a conditional expression return a value of **True**. The syntax for a simple ***if*** statement is as follows:

Syntax: ***For one statement without block or curly braces.***

```
if (conditional expression)
      Statement;
```

Syntax: For block or more than one statement.

```
if (conditional expression)
{      Statement1;
       Statement2;
       :
}
```

Flowchart of if statement

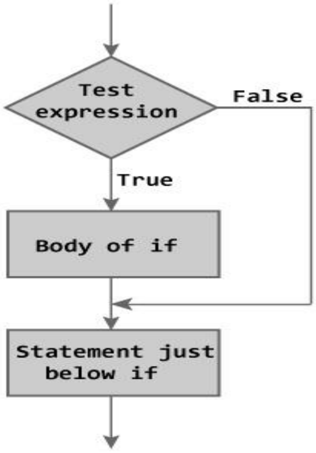


Figure: Flowchart of if Statement

If the expression is evaluated and found to be true, the single statement following the "if" is executed. If false, the following statement is skipped. Here a compound statement composed of several statements bounded by braces can replace the single statement.

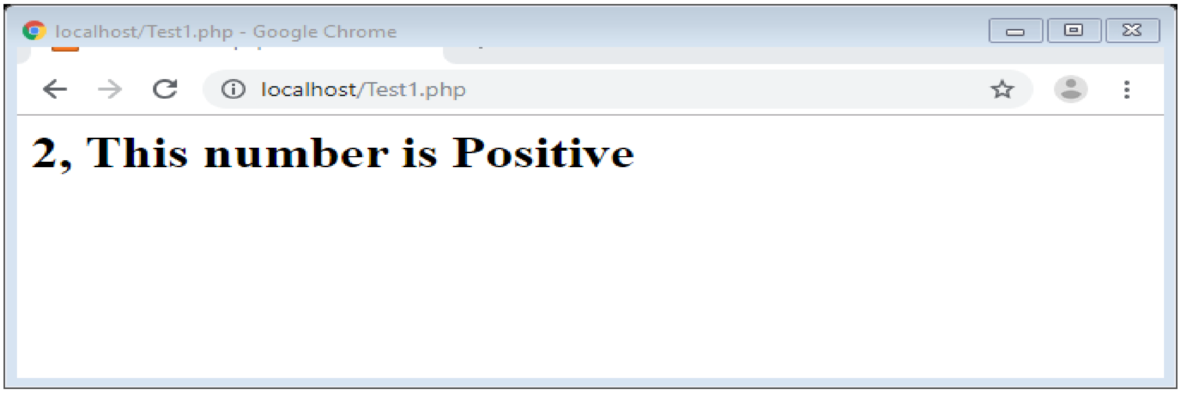
Example ***if*** condition is True than execute For one statements.

```
<?php
   $Number = 2;
   if ( $Number>0 )
      echo "<h1>  $Number, This number is Positive </h1>";
?>
```

//Number stored@ “number”

//Checks whether number you entered is greater than 0

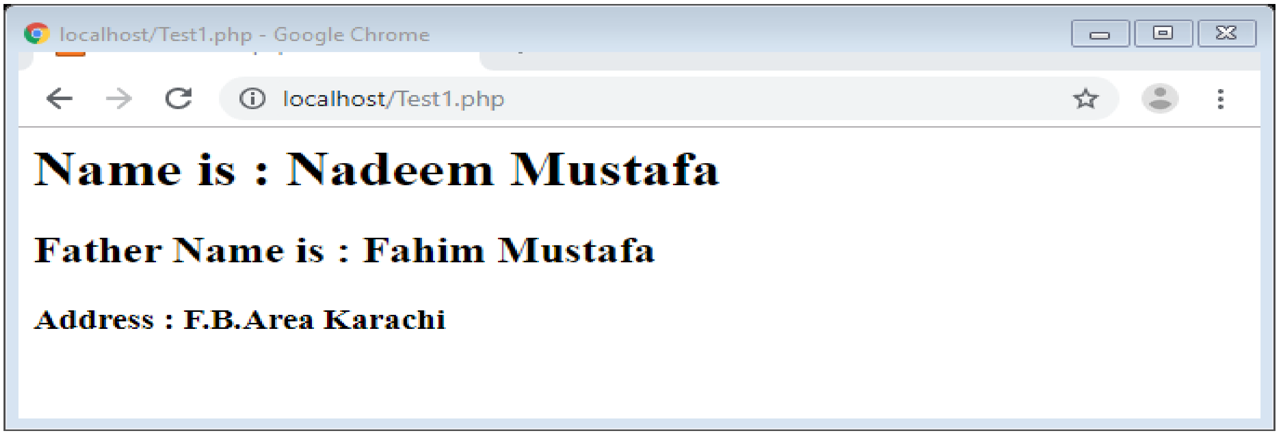
//if YES then Given number is positive will be print onto screen



2, This number is Positive

Example ***if*** condition is true than execute for block or more than one statements.

```
<?php
   $Roll=21;
  if ($Roll==21)
  {
     echo "<h1> Name is : Nadeem Mustafa </h1>";
     echo "<h2> Father Name is : Fahim Mustafa </h2>";
     echo "<h3> Address : F.B.Area Karachi </h3>";
  }
?>
```



Name is : Nadeem Mustafa

Father Name is : Fahim Mustafa

Address : F.B.Area Karachi

2. if, else statement:

This feature permits the programmer to write a single comparison, and then execute one of the two statements depending upon whether the test expression is true or false. The general form of the if-else statement is.

Flowchart of if...else statement

Syntax:

if(expression **)**
    statement1 **;**
else
    statement2 **;**

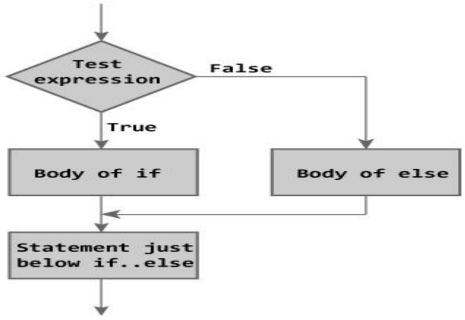
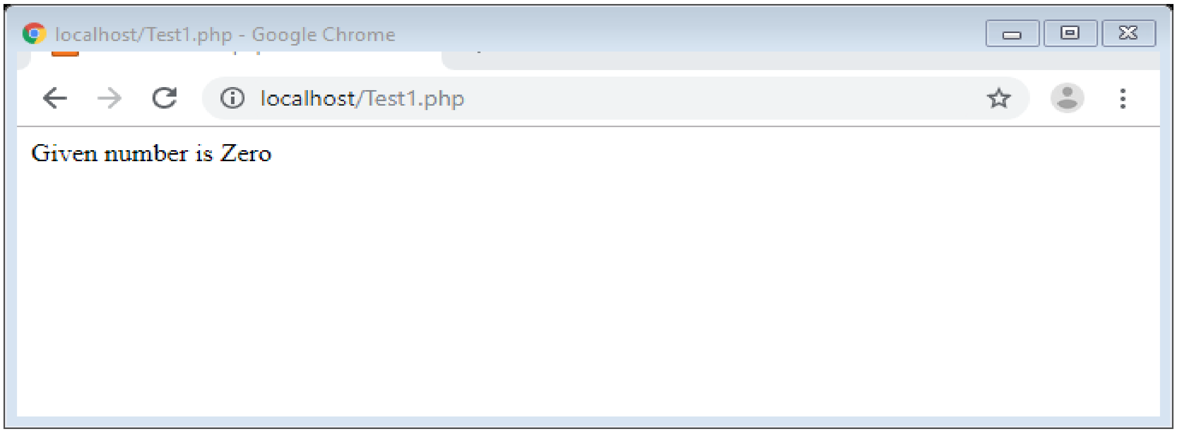


Figure: Flowchart of if...else Statement

Here also expression in parentheses must evaluate to (a Boolean) true or false. Typically you are testing something to see if it's true, and then running a code block (one or more statements) if it is true, and another block of code if it isn't. The statement1 or statement2 can be either simple or compound statement.

The following program demonstrates a legal if else statement:

```
<?php
  $number=0;                                    //Number stored@ "number"
 if( $number == 0 )                             //Checks whether number you entered is
                                                equal to 0 or Not.
    echo "Given number is Zero </br>";          //if YES then enters into if-body
 else                                           //If not then enters into else -body
     echo "Given number is not zero </br>";
?>
```



Given number is Zero

3. If .. else if.. else:

This brings up the other if-else construct; if, else, if, else. This construct is useful where two or more alternatives are available for selection.

Syntax:

```
if (condition)
        statement 1;
else if (condition)
        statement 2;
else if(condition)
        statement 3;
else
       statements 4 ;
```

The various conditions are evaluated one by one starting from top to bottom, on reaching a condition evaluating to true the statement group associated with it are executed and skip other statements. If none of expression is evaluate to true, then the statement or group of statement associated with the final else is executed.

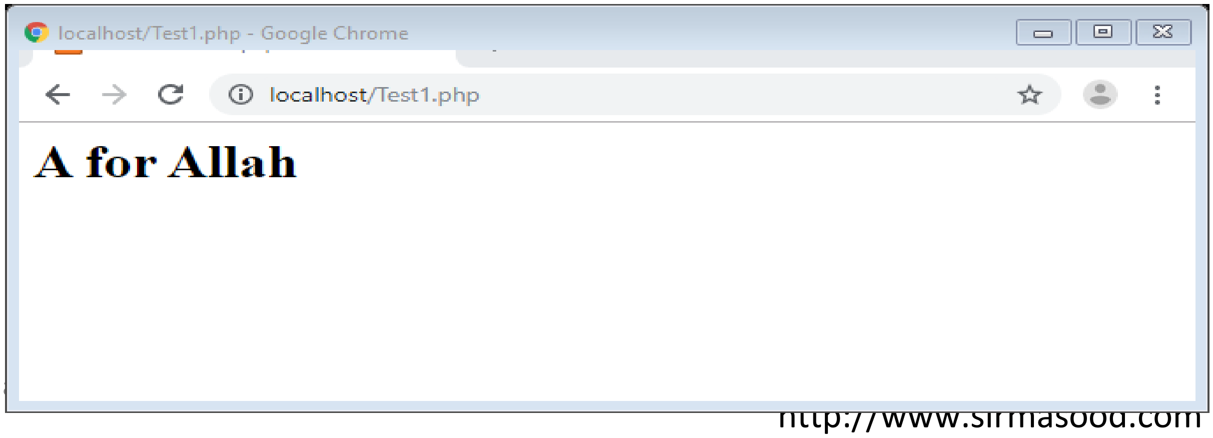
The following program demonstrates a legal if-else if-else statement:

```
 <?php
   $alphabet = "A";
   if($alphabet == "A" || $alphabet == "a" )
         echo "<h1>A for Allah</h1> ";
   else if($alphabet == "B" || $alphabet == "b" )
         echo "<h1> B for Bissmillah </h1>";
   else if($alphabet == "C" || $alphabet == "c" )
         echo "<h1> C for Captain </h1>";

  else
         echo"<h3><font color='red'>Unknown Character </font></h3>";
?>
```

//Ask or assign you to any Letter
//Letter stored@ "alphabet"
//Checks whether letter you entered is equal to "A".
//if YES then enters into if-body and display A for Allah control exit.
//If Not then enters into else-if Checks whether letter is equal to "B" then to display B for Bissmillah. And control exit.
// else-if Not then Check letter is equal "C" then display C for Captain and control exit.
//else-if Not then enters into else and display Unknown character and control exit

output will be depend on character value in $alphabet variable:-



A for Allah

- **Difference between if and else – if is that :**

If Statement	If Else if Statement
<?php	**<?php**
$per = 72;	$per = 67;
if($per >=80)	if($per >=80)
echo "Grade A+ ";	echo "Grade A+ ";
if ($per>=70)	else if ($per>=70)
echo "Grade A ";	echo "Grade A ";
if ($per>=60)	else if ($per>=60)
echo "Grade B ";	echo "Grade B ";
if ($per>=50)	else if ($per>=50)
echo "Grade C ";	echo "Grade C ";
?>	**?>**
Output:	**Output:**
Grade A Grade B Grade C	**Grade B**
// it will checks all conditions and execute all conditions that are true//	// It stop when the condition is true (i.e. not check further) and then exit from the if – structure//

Nested If Expression :

If there is another structure within ***if*** structure that is called nested if statement.

Syntax:

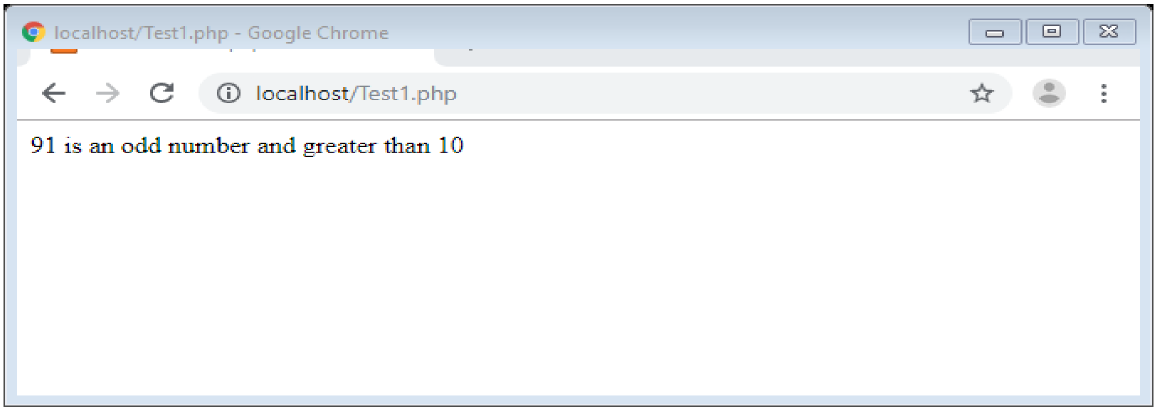
```
if( expression ){                    ←  Outer if
     if( expression1 ){          Inner If ←
            statement block1;
      }
    else{
          statement block2;
    }
}                          ←   Outer if ends here
else{
  statement block3;
}
```

if *'expression' is false* the **'statement-block3'** will be *executed*, otherwise it continues to perform the test for 'expression 1' . ***If*** *the 'expression 1' is* ***true*** the **'statement-block1'** is *executed* otherwise 'statement-block2' is executed.

Example

The if-else statement can also use to test for Nested conditions. The following example uses two conditions so that if the first test fails, we want to perform a second test before deciding what to do:

```
<?php
   $n=91;
    if ($n%2 != 0)
           if ($n>10)
                    echo "$n is an odd number and greater than 10 </br>";
           else
                     echo "$n is an odd number and less than 10 </br>";
     else
               echo "$n is an even number</br>";
?>
```



```
91 is an odd number and greater than 10
```

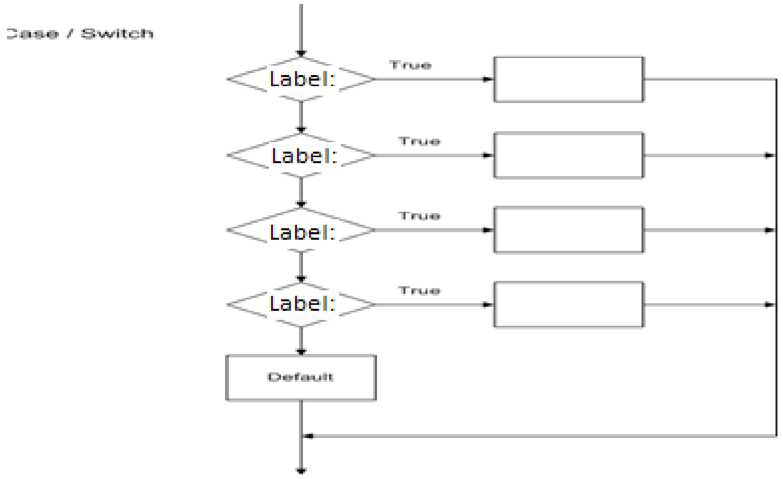
4. Switch statement

Switch statement is used to solve multiple option type. Another condition PHP statement that is used for controlling program flow is the switch statement. The switch statement controls program flow by executing a specific set of statement depending on the value of an expression. A ***switch*** statement consist of the following components: the keyword ***switch***, an expression, an opening brace, one or more ***case*** statements, a ***default*** label, and a closing brace. A case statement consists of a ***case*** label, the executable statements, and the keyword ***break*** use for exit the switch block. The syntax for the switch statement is as follows:

Syntax:

```
switch(expression)
{
    case label-1:
            statement-1;
            break;
    case label-2:
            statement-2;
            break;
    case label-3:
            statement-3;
            break;
     default:
           statement-otherwise-default;
 }
```

Flow Chart of Switch Case

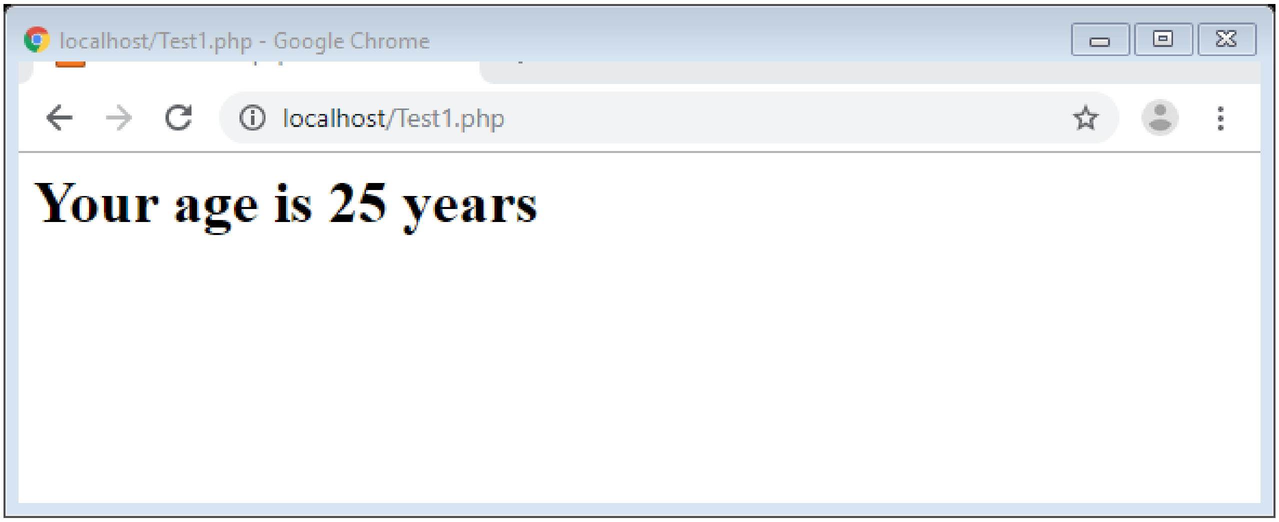


A case label consists of the keyword case, followed by a literal value or variable name It could be character, integer, string and floating point value, followed by a colon. PHP compares the value returned from the switch statement expression to the literal value or value of the variable named following the case keyword. If a match is found, the statement following the case label statement executes.

Example:

```
<?php
   $val=25;
   switch($val)
   {
     case "A":
         echo "<h1> Your grade is A </h1>";
         break;
     case "Nadeem":
         echo "<h1> Hi! Nadeem </h1>";
         break;
     case 25:
         echo "<h1> Your age is 25 years </h1>";
         break;
     case 5.6:
         echo "<h1> Your Height is 5.6 tall </h1>";
```

```
            break;
        default:
            echo "<h1> <font color='red'> Unknown Value</font> </h1>";
    }
?>
```



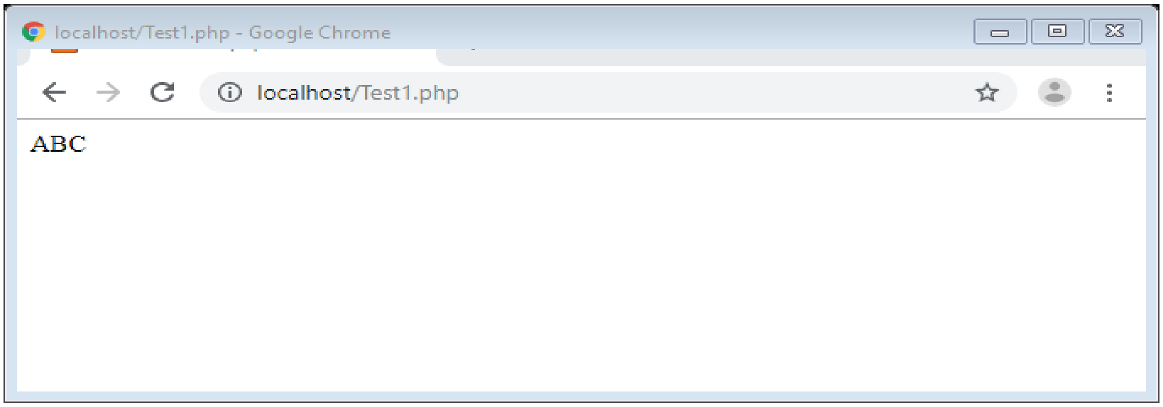
Your age is 25 years

Points to Remember

- ✓ It isn't necessary to use **break** after each block, but if you do not use it, all the consecutive block of codes will get executed after the matching block.
- ✓ Switch case is valid only for “equals to” condition, it doesn’t work for any other relational operators.

Example:

```
<?php
  $i = 1;
    switch($i)
    {
        case 1:          // compiler will check either i= 1?
          echo "A";      // if i=1, it will print “A”
                          //No break
        case 2:
          echo "B";      // No break
        case 3:
          echo  "C";
          break;
  }
?>
```



```
ABC
```

Explanation:

1. The output was supposed to be only **A** because only the first case matches, but as there is no break statement after the block, the next blocks are executed, until the cursor encounters a break.
2. **default** case can be placed anywhere in the switch case. Even if we don't include the default case switch statement works.

5. The Conditional (? :) Operator

We have covered **conditional operator ? :** in the previous condition expression which can be used to replace **if...else** statements. It has the following general form:

Syntax:

(Exp1) ?Exp2 : Exp3;

Or

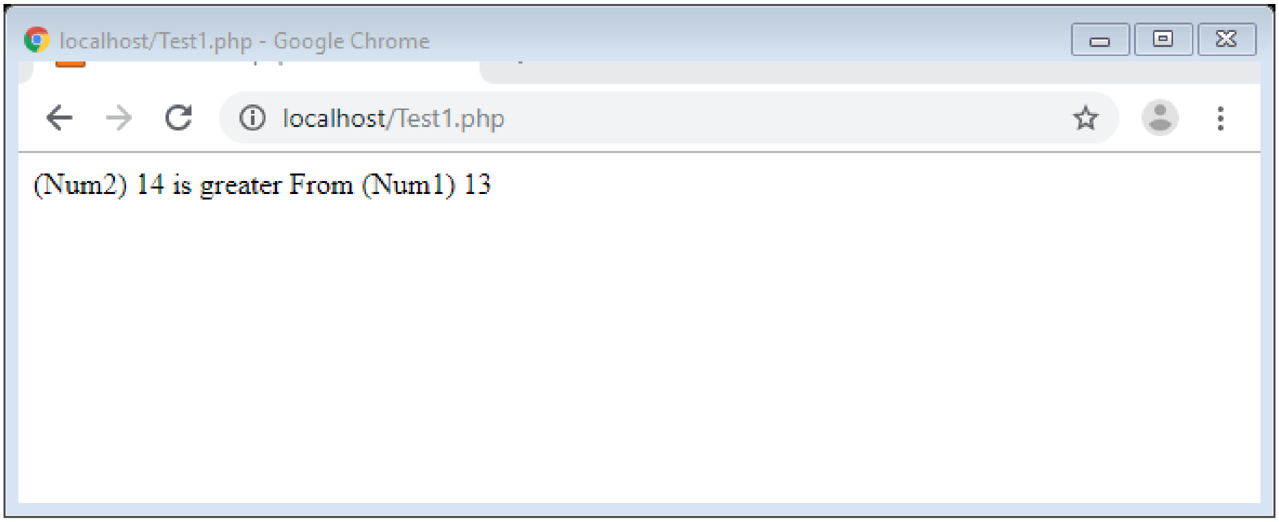
(Condition)? True : False;

Where Exp1, Exp2, and Exp3 are expressions. Notice the use and placement of the colon. The value of a **?**expression is determined like:

- Exp1 is evaluated. If it is true, then Exp2 is evaluated and becomes the value of the entire? Expression.
- If Exp1 is false, then Exp3 is evaluated and its value becomes the value of the expression.

Example:

```
<?php
    $Num1 =13;
    $Num2 = 14;
     $Result = ($Num1>$Num2)? "(Num1) $Num1 is greater From (Num2) $Num2": "(Num2)
$Num2 is greater From (Num1) $Num1";
     echo $Result;
 ?>
```



```
(Num2) 14 is greater From (Num1) 13
```

It will check *if ($Num1>$Num2)* then it will evaluate *Expression 1* i.e. , but since $Num1 is not greater than from $Num2 here so it will evaluate *expression 2* i.e.

Points to Remember

- ✓ It is also called **"Ternary Operator"** as it works on three operands.

Exercise

Theory Questions

1. Decision-making structures cannot be nested. True or false with give any example.
2. How do you perform more than one statement when a condition is true?
3. Correct the syntax error line by line.

```
 <?php
1)      If   $x > 25
2)      {
3)              $Y = x;
4)      else
5)              %y= $z
6)       }
   ?>
```

4. What function of ***break*** keyword/statement and where we can use it?
5. What function of continue keyword/statement and where we can use it?

Practical Questions

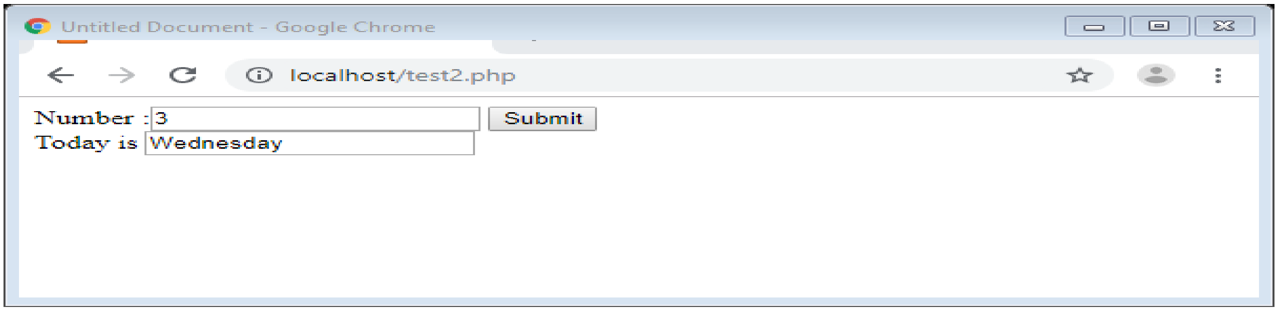
1. Write a simple program to construct a **Calculator** that can perform **Basic operation** Mathematical operation as well.
2. Write a simple program to check an alphabet entered by user is a ***Vowel*** or not **using if statement**, **if else if else statement** and **switch case statements**.

If Statement	If else if Statement	Switch Case

3. Write a code to check an integer number entered by user is **Even** or **Odd** using **Conditional operator statement**.
4. Write a program to input subject marks Math, English and Physics then to prepare marks sheet of a student that will show the Obtain marks, percentage and grade.
5. Write a program that input a number, then report whether the number is in the range from 1 to 100. Otherwise the number above to 100.
6. The following is supposed to cause an action or print message, whenever x is 5 and y is 9.

```
  <?php
      If ( $x >= $y )
          echo “x is greater than y”
      else
          echo “y is greater than x “;
 ?>
```

7. Write a switch statement that output messages indicating what day has been numerically input (for example 1-> Monday, 2-> Tuesday, 3-> Wednesday,...)



Number : 3

Today is Wednesday

Objective MCQ's

1. You can exit a switch statement using a(n) ___
   a) break
   b) end
   c) quit
   d) complete

2. When the value return by a switch statement expression does not match a case label, the statements within the___ label execute.
   a) Exception
   b) Else
   c) Error
   d) Default

3. In a simple if statement with no else, what happens if the condition following the if is false?
   a) The program search for the last else in the program.
   b) Nothing, or control falls through to the statements following the if.
   c) The body of the if statement is executed.
   d) The program as a whole is executed.

4. The conditional operator statement that

```
<?php
    $x=0 ;
    echo ( $x ==0 )? "x equal to zero": "x not equal to zero"  ;
 ?>
```

   a) Is incorrect syntax
   b) Is correct syntax, but x equal to zero will be print
   c) Cause a run time error
   d) Has no effect on the program

5. What value is assign in $income_tax to salary by the if statement when $salary is 55000?

```
<?php
   If ( $salary >70000 )
            $Income_tax= 1000;
   else if ( $salary > 50000 )
            $Income_tax = 500;
   else
             $Income_tax = 0;
?>
```

   a) 1000
   b) 500
   c) 0
   d) Nothing above all

6. Which is type of ternary operator?
   a) Logical operator
   b) Assignments operator
   c) Relational
   d) Conditional operator