|  |  |  |
| --- | --- | --- |
| C:\Users\Admin\Pictures\JATA KPM.png  **BAHAGIAN PENDIDIKAN TEKNIK DAN VOKASIONAL**  **KEMENTERIAN PENDIDIKAN MALAYSIA**  **ARAS 5 & 6, BLOK E14, KOMPLEKS E,**  **PUSAT PENTADBIRAN KERAJAAN PERSEKUTUAN**  **KERTAS PENERANGAN**  ***(INFORMATION SHEET)*** | | |
| **KOD DAN NAMA NOSS** | IT-010-3: 2016 APPLICATION MODULE DEVELOPMENT | |
| **KOD DAN NAMA CU / WA** | C02 - APPLICATION MODULE DEVELOPMENT | |
| **NAMA PROGRAM** | TEKNOLOGI SISTEM PENGURUSAN PANGKALAN DATA DAN APLIKASI WEB | |
| **TAHAP DAN SEMESTER** | 3 (SEMESTER 2) | |
| **KOD DAN TAJUK KURSUS** | KPD 2023 WEB PROGRAMMING | |
| **NO.D5N TAJUK KOMPETENSI** | K4 WRITE MODULE CODE | |
| **NO. KOD KSKV** | KPD2023 / P(10/13) | Muka Surat : 1 Drp : 18 |
| **NO. KOD NOSS** | IT-010-3:2016-C02/ P(10/13) |

**TAJUK/TITLE: WRITE MODULE CODE**

**TUJUAN/PURPOSE:**

**Kertas penerangan ini adalah bertujuan menerangkan mengenai :**

* Function
* User-Defined Functions
* Function Parameters
* The Return Statement
* Predefined Variables
* PHP Form
* GET and POST
* MySQLi
* mysqli\_fetch\_assoc()
* mysqli\_fetch\_array()
* mysqli\_fetch\_row()

**PENERANGAN/***INFORMATION* **:**

**4. 8 FUNCTION**

**4.8.1 PHP User Defined Functions**

Besides the built-in PHP functions, we can create our own functions.

* A function is a block of statements that can be used repeatedly in a program.
* A function will not execute immediately when a page loads.
* A function will be executed by a call to the function.
* Create a User Defined Function in PHP

A user-defined function declaration starts with the word function:

Syntax

function *functionName*() {  
*code to be executed*;  
}

**Note:** A function name can start with a letter or underscore (not a number).

**Tip:** Give the function a name that reflects what the function does!

Function names are NOT case-sensitive.

In the example below, we create a function named "writeMsg()". The opening curly brace ( { ) indicates the beginning of the function code, and the closing curly brace ( } ) indicates the end of the function. The function outputs "Hello world!". To call the function, just write its name followed by brackets ():

Example

<?php  
function writeMsg() {  
    echo "Hello world!";  
}  
  
writeMsg(); // call the function  
?>

**4.8.2 PHP Function Parameter**

## Function Parameters

Function parameters are declared after the function name and inside parentheses. They are declared much like a typical variable would be −

<?php

// multiply a value by 10 and return it to the caller

function multiply ($value) {

$value = $value \* 10;

return $value;

}

$retval = multiply (10);

Print "Return value is $retval\n";

?>

This will produce the following result −

Return value is 100

## 4.8.3 PHP Functions returning value

A function can return a value using the **return** statement in conjunction with a value or object. return stops the execution of the function and sends the value back to the calling code.

You can return more than one value from a function using **return array(1,2,3,4)**.

Following example takes two integer parameters and add them together and then returns their sum to the calling program. Note that **return** keyword is used to return a value from a function.

<html>

<head>

<title>Writing PHP Function which returns value</title>

</head>

<body>

<?php

**function** addFunction($num1, $num2) {

$sum = $num1 + $num2;

**return** $sum;

}

$return\_value = addFunction(10, 20);

echo "Returned value from the function : $return\_value";

?>

</body>

</html>

This will display following result −

Returned value from the function : 30

**4.9 PREDEFINED VARIABLES**

# **4.9.1 PHP - Form Introduction**

## Dynamic Websites

The Websites provide the functionalities that can use to store, update, retrieve, and delete the data in a database.

## What is the Form?

A Document that containing black fields, that the user can fill the data or user can select the data.Casually the data will store in the data base

### Example

Below example shows the form with some specific actions by using post method.

<html>

<head>

<title>PHP Form Validation</title>

</head>

<body>

<?php

// define variables and set to empty values

$name = $email = $gender = $comment = $website = "";

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

$name = test\_input($\_POST["name"]);

$email = test\_input($\_POST["email"]);

$website = test\_input($\_POST["website"]);

$comment = test\_input($\_POST["comment"]);

$gender = test\_input($\_POST["gender"]);

}

function test\_input($data) {

$data = trim($data);

$data = stripslashes($data);

$data = htmlspecialchars($data);

return $data;

}

?>

<h2>Tutorials Point Absolute classes registration</h2>

<form method = "post" action = "/php/php\_form\_introduction.htm">

<table>

<tr>

<td>Name:</td>

<td><input type = "text" name = "name"></td>

</tr>

<tr>

<td>E-mail:</td>

<td><input type = "text" name = "email"></td>

</tr>

<tr>

<td>Specific Time:</td>

<td><input type = "text" name = "website"></td>

</tr>

<tr>

<td>Class details:</td>

<td><textarea name = "comment" rows = "5" cols = "40"></textarea></td>

</tr>

<tr>

<td>Gender:</td>

<td>

<input type = "radio" name = "gender" value = "female">Female

<input type = "radio" name = "gender" value = "male">Male

</td>

</tr>

<tr>

<td>

<input type = "submit" name = "submit" value = "Submit">

</td>

</tr>\

</table>

</form>

<?php

echo "<h2>Your Given details are as :</h2>";

echo $name;

echo "<br>";

echo $email;

echo "<br>";

echo $website;

echo "<br>";

echo $comment;

echo "<br>";

echo $gender;

?>

</body>

</html>

**4.9.2 GET and POST**

There are two ways the browser client can send information to the web server.

* The GET Method
* The POST Method

Before the browser sends the information, it encodes it using a scheme called URL encoding. In this scheme, name/value pairs are joined with equal signs and different pairs are separated by the ampersand.

name1=value1&name2=value2&name3=value3

Spaces are removed and replaced with the *+* character and any other nonalphanumeric characters are replaced with a hexadecimal values. After the information is encoded it is sent to the server.

**The GET Method**

The GET method sends the encoded user information appended to the page request. The page and the encoded information are separated by the **?**character.

http://www.test.com/index.htm?name1=value1&name2=value2

* The GET method produces a long string that appears in your server logs, in the browser's Location: box.
* The GET method is restricted to send upto 1024 characters only.
* Never use GET method if you have password or other sensitive information to be sent to the server.
* GET can't be used to send binary data, like images or word documents, to the server.
* The data sent by GET method can be accessed using QUERY\_STRING environment variable.
* The PHP provides **$\_GET** associative array to access all the sent information using GET method.

Try out following example by putting the source code in test.php script.

<?php

if( $\_GET["name"] || $\_GET["age"] ) {

echo "Welcome ". $\_GET['name']. "<br />";

echo "You are ". $\_GET['age']. " years old.";

exit();

}

?>

<html>

<body>

<form action = "<?php **$\_PHP\_SELF** ?>" method = "GET">

Name: <input type = "text" name = "name" />

Age: <input type = "text" name = "age" />

<input type = "submit" />

</form>

</body>

</html>

**The POST Method**

The POST method transfers information via HTTP headers. The information is encoded as described in case of GET method and put into a header called QUERY\_STRING.

* The POST method does not have any restriction on data size to be sent.
* The POST method can be used to send ASCII as well as binary data.
* The data sent by POST method goes through HTTP header so security depends on HTTP protocol. By using Secure HTTP you can make sure that your information is secure.
* The PHP provides **$\_POST** associative array to access all the sent information using POST method.

Try out following example by putting the source code in test.php script.

<?php

if( $\_POST["name"] || $\_POST["age"] ) {

if (preg\_match("/[^A-Za-z'-]/",$\_POST['name'] )) {

die ("invalid name and name should be alpha");

}

echo "Welcome ". $\_POST['name']. "<br />";

echo "You are ". $\_POST['age']. " years old.";

exit();

}

?>

<html>

<body>

<form action = "<?php **$\_PHP\_SELF** ?>" method = "POST">

Name: <input type = "text" name = "name" />

Age: <input type = "text" name = "age" />

<input type = "submit" />

</form>

</body>

</html>

**The $\_REQUEST variable**

The PHP $\_REQUEST variable contains the contents of both $\_GET, $\_POST, and $\_COOKIE. We will discuss $\_COOKIE variable when we will explain about cookies.

The PHP $\_REQUEST variable can be used to get the result from form data sent with both the GET and POST methods.

Try out following example by putting the source code in test.php script.

<?php

if( $\_REQUEST["name"] || $\_REQUEST["age"] ) {

echo "Welcome ". $\_REQUEST['name']. "<br />";

echo "You are ". $\_REQUEST['age']. " years old.";

exit();

}

?>

<html>

<body>

<form action = "<?php **$\_PHP\_SELF** ?>" method = "POST">

Name: <input type = "text" name = "name" />

Age: <input type = "text" name = "age" />

<input type = "submit" />

</form>

</body>

</html>

Here $\_PHP\_SELF variable contains the name of self script in which it is being called.

**4.10 MYSQLI INTRODUCED**

# **4.10.1 PHP - Function MySQLi Fetch Assoc**

## Syntax

mysqli\_fetch\_assoc(result);

## Definition and Usage

It is used to fetches a result row as an associative array.

## Return Values

It returns an associative array of strings representing the fetched row.

## Parameters

|  |  |
| --- | --- |
| **Sr.No** | **Parameters & Description** |
| 1 | **result**  It specifies the result set identifier returned by mysqli\_query(), mysqli\_store\_result() or mysqli\_use\_result() |

## Example

Try out the following example

<?php

$connection\_mysql = mysqli\_connect("localhost","user","password","db");

if (mysqli\_connect\_errno($connection\_mysql)){

echo "Failed to connect to MySQL: " . mysqli\_connect\_error();

}

$sql = "SELECT name,salary FROM emp";

$result = mysqli\_query($connection\_mysql,$sql);

$row = mysqli\_fetch\_assoc($result);

print $row["name"];

print "\n";

print $row["salary"];

mysqli\_free\_result($result);

mysqli\_close($connection\_mysql);

?>

# **4.10.2 PHP - Function MySQLi Fetch Array**

## Syntax

mysqli\_fetch\_array(result,resulttype);

## Definition and Usage

It is used to fetchs a result row as an associative array

## Return Values

It returns an array of strings that corresponds to the fetched row.

## Parameters

|  |  |
| --- | --- |
| **Sr.No** | **Parameters & Description** |
| 1 | **result**  It specifies the result set identifier |
| 2 | **resulttype**  It specifies what type of array that should be produced |

## Example

Try out the following example

<?php

$connection\_mysql = mysqli\_connect("localhost","username","password","db");

if (mysqli\_connect\_errno($connection\_mysql)){

echo "Failed to connect to MySQL: " . mysqli\_connect\_error();

}

$sql = "SELECT name,age FROM emp";

$result = mysqli\_query($connection\_mysql,$sql);

$row = mysqli\_fetch\_array($result,MYSQLI\_NUM);

print $row[0];

print "\n";

print $row[1];

$row = mysqli\_fetch\_array($result,MYSQLI\_ASSOC);

print $row["name"];

print "\n";

print $row["age"];

mysqli\_free\_result($result);

mysqli\_close($connection\_mysql);

?>

# **4.10.3 PHP - Function MySQLi Fetch Row**

## Syntax

mysqli\_fetch\_row(result);

## Definition and Usage

The mysqli\_fetch\_row() function fetches one row from a result-set and returns it as an enumerated array.

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| result | Required. Specifies a result set identifier returned by mysqli\_query(), mysqli\_store\_result() or mysqli\_use\_result() |

## Technical Details

|  |  |
| --- | --- |
| **Return Value:** | Returns an array of strings that corresponds to the fetched row. NULL if there are no more rows in result set |
| **PHP Version:** | 5+ |

### Example

Fetch rows from a result-set:

<?php  
$con=mysqli\_connect("localhost","my\_user","my\_password","my\_db");  
// Check connection  
if (mysqli\_connect\_errno())  
  {  
  echo "Failed to connect to MySQL: " . mysqli\_connect\_error();  
  }  
  
$sql="SELECT Lastname,Age FROM Persons ORDER BY Lastname";  
  
if ($result=mysqli\_query($con,$sql))  
  {  
  // Fetch one and one row  
  while ($row=mysqli\_fetch\_row($result))  
    {  
    printf ("%s (%s)\n",$row[0],$row[1]);  
    }  
  // Free result set  
  mysqli\_free\_result($result);  
}  
  
mysqli\_close($con);  
?>

**SOALAN/***QUESTION***:**

1. Jelaskan definisi MySQLi berikut :
2. mysqli\_fetch\_assoc()
3. mysqli\_fetch\_array()
4. mysqli\_fetch\_row()

**RUJUKAN/***REFERENCE* **:**

1. <https://www.w3schools.com/php/func_mysqli_fetch_row.asp>
2. <https://www.w3schools.com/php/func_mysqli_fetch_assoc.asp>