**MCQ:**

1. Which HTTP status code indicates a successful request?

a) 100

b) 200 ✅ (Correct Answer)

c) 300

d) 400

Explanation: HTTP status code 200 indicates a successful request, typically used for successful GET and POST operations.

2. How to collect data using the PHP superglobal when the form is submitted using the "GET" method?

a) $\_POST

b) $\_REQUEST

c) Both a & b

d) Both b & c

e) $\_GET ✅ (Correct Answer)

Explanation: Data submitted via the GET method is retrieved using $\_GET.

3. Find the correct answer to invoke a method in PHP?

a) $object->methodName(); ✅ (Correct Answer)

b) object->methodName();

c) $object.methodName();

d) object::methodName();

Explanation: In PHP, the arrow operator (->) is used to access methods or properties of an object.

4. How to define the following as an associative array in PHP?

a) $array = array("name" => "John", "age" => 25); ✅ (Correct Answer)

b) $array = myarray("name" => "John", "age" => 25);

c) $array = array("name" => "John", "25");

d) $array = array("John", "25");

Explanation: Associative arrays in PHP are defined using key-value pairs like "key" => "value".

5. What is the default port for HTTP traffic?

a) 443

b) 81

c) 80 ✅ (Correct Answer)

d) 8080

Explanation: Port 80 is the default port for HTTP. For HTTPS, it’s port 443.

6. Which operator is used for concatenating strings in PHP?

a) +

b) &

c) . ✅ (Correct Answer)

d) #

Explanation: The dot operator (.) is used for string concatenation in PHP.

7. What is the main difference between HTML and XHTML?

a) HTML is stricter than XHTML

b) XHTML is an extension of JavaScript

c) XHTML is not supported by web browsers

d) XHTML is a stricter version of HTML ✅ (Correct Answer)

Explanation: XHTML follows stricter syntax rules and is essentially HTML with XML compliance.

8. Which function is used to check if a string matches a specified pattern in PHP?

a) pattern\_match()

b) str\_match()

c) preg\_match() ✅ (Correct Answer)

d) match\_string()

Explanation: The preg\_match() function is used for pattern matching in PHP using regular expressions.

**Output Tracing:**

Question;

<?php

$value1 = 0.0;

$value2 = "0";

$value3 = false;

$value4 = "Hello";

if (empty($value1) && is\_string($value2) && ctype\_alpha($value2)) {

echo "Condition 1 is true <br>";

} else {

echo "Condition 1 is false <br>";

}

if (isset($value3) && empty($value1) && is\_numeric($value2)) {

echo "Condition 2 is true <br>";

} else {

echo "Condition 2 is false <br>";

}

if (isset($value4) && !empty($value4) && is\_string($value3)) {

echo "Condition 3 is true <br>";

} else {

echo "Condition 3 is false <br>";

}

?>

#### **Condition 1:**

php

Copy code

if (empty($value1) && is\_string($value2) && ctype\_alpha($value2))

* **empty($value1)**:
  + value1 = 0.0, which is considered "empty" in PHP. So, this returns true.
* **is\_string($value2)**:
  + value2 = "0", which is a string. So, this returns true.
* **ctype\_alpha($value2)**:
  + ctype\_alpha checks if the string contains only alphabetic characters. "0" is not alphabetic. So, this returns false.

Since the last condition (ctype\_alpha($value2)) is false, the entire condition is false.  
**Output**: "Condition 1 is false <br>"

#### **Condition 2:**

php

Copy code

if (isset($value3) && empty($value1) && is\_numeric($value2))

* **isset($value3)**:
  + value3 is defined (false), so isset returns true.
* **empty($value1)**:
  + As before, value1 = 0.0 is "empty". So, this returns true.
* **is\_numeric($value2)**:
  + value2 = "0", which is numeric as a string. So, this returns true.

All conditions are true.  
**Output**: "Condition 2 is true <br>"

#### **Condition 3:**

php

Copy code

if (isset($value4) && !empty($value4) && is\_string($value3))

* **isset($value4)**:
  + value4 = "Hello", which is defined. So, isset returns true.
* **!empty($value4)**:
  + value4 = "Hello", which is not empty. So, this returns true.
* **is\_string($value3)**:
  + value3 = false, which is not a string. So, this returns false.

Since the last condition (is\_string($value3)) is false, the entire condition is false.  
**Output**: "Condition 3 is false <br>"

Output:

Condition 1 is false

Condition 2 is true

Condition 3 is false

**Code writing:**

<!DOCTYPE html>

<html>

<body>

<form action="process.php" method="post">

Username: <input type="text" name="uname"><br>

Password: <input type="text" name="pass"><br>

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

</form>

</body>

</html>

**Requirements:**

* Validate the form fields (uname and pass) with the following rules:
  1. **Username (uname)** must not contain numbers.
  2. **Password (pass)** must be more than 6 characters.
  3. Both fields (uname and pass) are required.
* If the conditions are not met, display an error message:  
  "Please fill all the fields".
* If the inputs meet the criteria, submit the form.

**Correct PHP Code for Validation:**

<?php

// Check if form is submitted

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

$uname = $\_POST['uname'];

$pass = $\_POST['pass'];

// Check if both fields are filled

if (empty($uname) || empty($pass)) {

echo "Please fill all the fields";

}

else {

// Validate username: must not contain numbers

if (is\_numeric($uname)) {

echo "Invalid username: must not contain numbers.";

}

// Validate password: must be more than 6 characters

else if (strlen($pass) <= 6) {

echo "Invalid password: must be more than 6 characters.";

}

// If both validations pass

else {

echo "Form submitted successfully!";

}

}

}

?>

**How It Works:**

1. **Form Submission Check**:  
   The $\_SERVER["REQUEST\_METHOD"] ensures that the form is processed only when the POST method is used.
2. **Field Validation**:
   * empty($uname) and empty($pass) check if the fields are filled.
   * is\_numeric($uname) ensures the username doesn't contain numbers.
   * strlen($pass) validates the password length.
3. **Error Messages**:
   * Specific messages are displayed if the conditions are not met.
   * A success message is displayed if all criteria are satisfied.