1. What is JavaScript? How can we generate output in it?

JavaScript (JS) is a programming language primarily used for adding interactivity and dynamic behavior to web pages. It runs in web browsers and enables features like animations, form validation, real-time updates, and more.

Some key features of JavaScript:

- **Client-Side Scripting**: Runs directly in the browser without needing a server.
- **Dynamic and Interactive**: Used to manipulate HTML and CSS to create interactive elements.
- **Event-Driven**: Responds to user actions like clicks, mouse movements, and key presses.
- **Supports APIs**: Works with APIs to fetch data, handle storage, and interact with external services.
- **Versatile**: Can be used for both frontend (React, Angular, Vue.js) and backend (Node.js) development.

```
console.log("Hello, World!");
document.write("Hello, World!");
alert("Hello, World!");
```

2. Create a simple arithmetic calculator using JavaScript.

```
function calculate() {
   let a = parseFloat(prompt("Enter first number:"));
   let operator = prompt("Enter operator (+, -, *, /):");
   let b = parseFloat(prompt("Enter second number:"));

   switch (operator) {
      case '+': alert(`Result: ${a + b}`); break;
      case '-': alert(`Result: ${a - b}`); break;
      case '*': alert(`Result: ${a * b}`); break;
      case '/': alert(`Result: ${a / b}`); break;
      default: alert("Invalid operator");
   }
}
calculate();
```

3. Provide examples of JavaScript conditional statements.

```
let age = 18;
if (age >= 18) {
   console.log("You are an adult.");
} else {
   console.log("You are a minor.");
}


let result = (age >= 18) ? "Adult" : "Minor";
console.log(result);
```

4. Illustrate loops and nested loops with suitable examples.

```
for (let i = 1; i <= 5; i++) {
   console.log(i);
}


for (let i = 1; i <= 3; i++) {
   for (let j = 1; j <= 3; j++) {
      console.log(`i: ${i}, j: ${j}`);
   }
}
```

5. Show use of events with their proper actions [onclick, onmouseover, onload, etc].

```
<button onclick="alert('Button clicked!')">Click Me</button>
<p onmouseover="this.style.color='red'">Hover over me!</p>
<body onload="console.log('Page loaded!')"></body>
```

6. Show the use of setTimeout() and setInterval() functions.

```
// setTimeout
setTimeout(() => {
   console.log("This runs after 2 seconds.");
}, 2000);

// setInterval
let counter = 0;
setInterval(() => {
   counter++;
   console.log(`Counter: ${counter}`);
}, 1000);
```

7. Use date object to access date and time to create a simple digital clock.

```
function updateClock() {
   let now = new Date();
   let time = now.toLocaleTimeString();
   console.log(time);
}
setInterval(updateClock, 1000);
```

8. What is Regular Expression? Show its use while validating form data.

A **Regular Expression (RegEx)** is a sequence of characters that defines a search pattern. It is used for pattern matching and manipulation in strings, such as validation, searching, and replacing text.

Uses of Regular Expressions:

- **Validating input fields** (e.g., emails, phone numbers, passwords)
- **Searching within strings** (e.g., finding words in text)
- **Replacing substrings** (e.g., replacing spaces with underscores)
- **Extracting specific data** (e.g., extracting dates from text)

```
let email = "test@example.com";
let regex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
if (regex.test(email)) {
   console.log("Valid email.");
} else {
   console.log("Invalid email.");
}
```

9. Using the jQuery library, show basic effect like show/hide.

```
<div id="box" style="width:100px;height:100px;background:red;"></div>
<button id="hide">Hide</button>
<button id="show">Show</button>
<script>
   $("#hide").click(() => $("#box").hide());
   $("#show").click(() => $("#box").show());
</script>
```

10. Use jQuery to demonstrate append() and prepend() functions.

```
<ul id="list">
   <li>Item 1</li>
</ul>
<button id="append">Append</button>
<button id="prepend">Prepend</button>
<script>
   $("#append").click(() => $("#list").append("<li>Appended Item</li>"));
   $("#prepend").click(() => $("#list").prepend("<li>Prepended Item</li>"));
</script>
```

11. Use jQuery to create plugins like accordion.

```
<div class="accordion">
   <h3>Section 1</h3>
   <div>Content 1</div>
   <h3>Section 2</h3>
   <div>Content 2</div>
</div>
<script>
   $(".accordion h3").click(function() {
      $(this).next().slideToggle();
   });
</script>
```

PHP Scripting

CRUD Based Simple Website

1. Database Schema

CREATE DATABASE crud_db (db.sql)

```
USE crud_db;

CREATE TABLE users (
   id INT AUTO_INCREMENT PRIMARY KEY,
   name VARCHAR(50) NOT NULL,
   email VARCHAR(100) UNIQUE NOT NULL,
   created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

2. Database Connection (config.php)

```
<?php
$servername = "localhost";
$username = "root";  // Default username for XAMPP
$password = "";      // Default password is empty
$dbname = "crud_db";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect_error) {
   die("Connection failed: " . $conn->connect_error);
}
?>
```

3. Homepage (Read Users) - index.php

```
<?php
include 'config.php';

$result = $conn->query("SELECT * FROM users");
?>

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CRUD App</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <h2>User List</h2>
  <a href="create.php">Add New User</a>
  <table border="1">
    <tr>
      <th>ID</th>
      <th>Name</th>
      <th>Email</th>
      <th>Actions</th>
    </tr>
    <?php while ($row = $result->fetch_assoc()): ?>
      <tr>
        <td><?= $row['id']; ?></td>
        <td><?= $row['name']; ?></td>
        <td><?= $row['email']; ?></td>
        <td>
          <a href="update.php?id=<?= $row['id']; ?>">Edit</a>
          <a href="delete.php?id=<?= $row['id']; ?>" onclick="return confirm('Are
you sure?');">Delete</a>
        </td>
      </tr>
    <?php endwhile; ?>
  </table>
</body>
</html>
```

4. Create User (create.php)

```
<?php
include 'config.php';

if ($_SERVER["REQUEST_METHOD"] == "POST") {
   $name = $_POST['name'];
   $email = $_POST['email'];

   $stmt = $conn->prepare("INSERT INTO users (name, email) VALUES (?, ?)");
   $stmt->bind_param("ss", $name, $email);

   if ($stmt->execute()) {
      header("Location: index.php");
      exit();
   } else {
      echo "Error: " . $stmt->error;
   }
}
?>

<!DOCTYPE html>
<html>
<head>
   <title>Add User</title>
</head>
<body>
   <h2>Add New User</h2>
   <form method="POST">
      <label>Name:</label>
      <input type="text" name="name" required><br>
      <label>Email:</label>
      <input type="email" name="email" required><br>
      <input type="submit" value="Add">
   </form>
</body>
</html>
```

5. Update User (update.php)

```
<?php
include 'config.php';

$id = $_GET['id'];
$result = $conn->query("SELECT * FROM users WHERE id = $id");
$user = $result->fetch_assoc();

if ($_SERVER["REQUEST_METHOD"] == "POST") {
   $name = $_POST['name'];
   $email = $_POST['email'];

   $stmt = $conn->prepare("UPDATE users SET name = ?, email = ? WHERE id = ?");
   $stmt->bind_param("ssi", $name, $email, $id);

   if ($stmt->execute()) {
      header("Location: index.php");
      exit();
   } else {
      echo "Error: " . $stmt->error;
   }
}
?>

<!DOCTYPE html>
<html>
<head>
   <title>Edit User</title>
</head>
<body>
   <h2>Edit User</h2>
   <form method="POST">
      <label>Name:</label>
      <input type="text" name="name" value="<?= $user['name']; ?>" required><br>
      <label>Email:</label>
      <input type="email" name="email" value="<?= $user['email']; ?>" required><br>
      <input type="submit" value="Update">
   </form>
</body>
</html>
```

6. Delete User (delete.php)

```
<?php
include 'config.php';

$id = $_GET['id'];
$conn->query("DELETE FROM users WHERE id = $id");

header("Location: index.php");
exit();
?>
```