PRINT DATE AND TIME

```
<%@ page import="java.util.Date" %>
<%@ page contentType="text/html;charset=UTF-8" language="java" %>

<html>
<head>
   <title>Display Current Date & Time</title>
</head>
<body>
   <center>
      <h1>Display Current Date & Time</h1>
   </center>

   <%
      Date date = new Date();
      out.print("<h2 align=\"center\">" + date.toString() + "</h2>");
   %>
</body>
</html>
```

MERGE TWO ARRAYS

```
<?php

// Two example arrays
$array1 = [5, 9, 2, 1, 7];
$array2 = [4, 3, 8, 6, 10];

// Merge the arrays
$mergedArray = array_merge($array1, $array2);

// Sort the merged array in descending order
rsort($mergedArray);
```

```
// Display the sorted merged array
echo "Sorted merged array (descending order):<br>";
foreach ($mergedArray as $value) {
    echo $value . " ";
}

?>
```

PHP PROGRAMS

```
<?php

// Example string
$string = "Hello, world! This is a sample string for testing.";

// i) Finding the length of a string
$length = strlen($string);
echo "Length of the string: $length<br>";

// ii) Counting the number of words in a string
$wordCount = str_word_count($string);
echo "Number of words in the string: $wordCount<br>";

// iii) Reversing the string
$reversedString = strrev($string);
echo "Reversed string: $reversedString<br>";

// iv) Search for specific word in the string
$searchWord = "world";
if (strpos($string, $searchWord) !== false) {
    echo "The word '$searchWord' is found in the string.<br>";
} else {
    echo "The word '$searchWord' is not found in the string.<br>";
```

```
}


?>
```

xml program

```
<users>
   <user id="1">
      <name>John Doe</name>
      <email>john.doe@example.com</email>
      <phone>123-456-7890</phone>
   </user>
   <user id="2">
      <name>Jane Smith</name>
      <email>jane.smith@example.com</email>
      <phone>456-789-0123</phone>
   </user>
   <user id="3">
      <name>Michael Brown</name>
      <email>michael.brown@example.com</email>
      <phone>789-012-3456</phone>
   </user>
</users>



import org.w3c.dom.*;
import javax.xml.parsers.*;
import org.xml.sax.SAXException; // Import SAXException explicitly
import java.io.*;
import java.util.Scanner;

public class FetchUserDetails {
```

```
  public static void main(String[] args) {
    // Create a Scanner object to read user input
    Scanner scanner = new Scanner(System.in);

    // Prompt the user to enter a user id
    System.out.print("Enter User ID: ");
    String userId = scanner.nextLine(); // Read user input

    try {
      // Load and parse the XML file
      File xmlFile = new File("users.xml");
      DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
      DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
      Document doc = dBuilder.parse(xmlFile);

      // Optional, but recommended for parsing XML files
      doc.getDocumentElement().normalize();

      // Get all user elements
      NodeList userList = doc.getElementsByTagName("user");

      // Search for user with matching id
      for (int i = 0; i < userList.getLength(); i++) {
        Node userNode = userList.item(i);
        if (userNode.getNodeType() == Node.ELEMENT_NODE) {
          Element userElement = (Element) userNode;
          String id = userElement.getAttribute("id");

          if (id.equals(userId)) {
            String name = userElement.getElementsByTagName("name").item(0).getTextContent();
            String email = userElement.getElementsByTagName("email").item(0).getTextContent();
            String phone =
userElement.getElementsByTagName("phone").item(0).getTextContent();
```

```
                // Display user details
                System.out.println("User Details:");
                System.out.println("Name: " + name);
                System.out.println("Email: " + email);
                System.out.println("Phone: " + phone);
                return;
              }
            }
          }

          // If user with given id not found
          System.out.println("User with ID " + userId + " not found.");

      } catch (ParserConfigurationException | SAXException | IOException e) {
          e.printStackTrace();
      } finally {
          // Close the Scanner object
          scanner.close();
      }
   }
}
```

Payment Verification

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Payment Page</title>
  <style>
```

```
   .error { color: red; }
  </style>
</head>
<body>
  <h1>Payment Form</h1>
  <form id="paymentForm">
    <label for="cardNumber">Credit Card Number:</label>
    <input type="text" id="cardNumber" name="cardNumber"><br><br>
    <label for="expiryDate">Expiry Date (MM/YY):</label>
    <input type="text" id="expiryDate" name="expiryDate" placeholder="MM/YY"><br><br>
    <label for="cvv">CVV:</label>
    <input type="text" id="cvv" name="cvv"><br><br>
    <button type="submit">Submit Payment</button>
  </form>
  <p id="errorMessages" class="error"></p>

  <script>
    document.getElementById('paymentForm').addEventListener('submit', function(event) {
      event.preventDefault(); // Prevent form submission

      const cardNumber = document.getElementById('cardNumber').value.trim();
      const expiryDate = document.getElementById('expiryDate').value.trim();
      const cvv = document.getElementById('cvv').value.trim();
      const errorMessages = document.getElementById('errorMessages');

      let errors = [];

      // Credit card number validation
      const cardNumberPattern = /^[0-9]{16}$/;
      if (cardNumber === '') {
        errors.push('Credit Card Number is required.');
      } else if (!cardNumberPattern.test(cardNumber)) {
```

```
      errors.push('Credit Card Number must be 16 digits.');
    }

    // Expiry date validation
    const expiryDatePattern = /^(0[1-9]|1[0-2])\/?([0-9]{2})$/;
    if (expiryDate === '') {
      errors.push('Expiry Date is required.');
    } else if (!expiryDatePattern.test(expiryDate)) {
      errors.push('Expiry Date must be in MM/YY format.');
    }

    // CVV validation
    const cvvPattern = /^[0-9]{3,4}$/;
    if (cvv === '') {
      errors.push('CVV is required.');
    } else if (!cvvPattern.test(cvv)) {
      errors.push('CVV must be 3 or 4 digits.');
    }

    // Display errors or submit the form
    if (errors.length > 0) {
      errorMessages.innerHTML = errors.join('<br>');
    } else {
      errorMessages.innerHTML = 'Payment successful!';
      // Here you can add code to process the payment, e.g., via AJAX
    }
  });
 </script>
</body>
</html>
```

REgisteration PAge

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Registration Page</title>
  <style>
    .error { color: red; }
  </style>
</head>
<body>
  <h1>User Registration</h1>
  <form id="registrationForm">
    <label for="username">Username:</label>
    <input type="text" id="username" name="username"><br><br>
    <label for="password">Password:</label>
    <input type="password" id="password" name="password"><br><br>
    <label for="confirmPassword">Confirm Password:</label>
    <input type="password" id="confirmPassword" name="confirmPassword"><br><br>
    <button type="submit">Register</button>
  </form>
  <p id="errorMessages" class="error"></p>

  <script>
    document.getElementById('registrationForm').addEventListener('submit', function(event) {
      event.preventDefault(); // Prevent form submission

      const username = document.getElementById('username').value.trim();
      const password = document.getElementById('password').value.trim();
      const confirmPassword = document.getElementById('confirmPassword').value.trim();
      const errorMessages = document.getElementById('errorMessages');
```

```
    let errors = [];

    // Username validation
    if (username === '') {
      errors.push('Username is required.');
    }

    // Password validation
    if (password === '') {
      errors.push('Password is required.');
    }

    // Confirm password validation
    if (confirmPassword === '') {
      errors.push('Confirm Password is required.');
    } else if (password !== confirmPassword) {
      errors.push('Passwords do not match.');
    }

    // Display errors or submit the form
    if (errors.length > 0) {
      errorMessages.innerHTML = errors.join('<br>');
    } else {
      errorMessages.innerHTML = 'Registration successful!';
      // Here you can add code to submit the form data, e.g., via AJAX
    }
  });
 </script>
</body>
</html>
```

Print prime numbers using JS

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Prime Numbers in Range</title>
</head>
<body>
  <h1>Prime Numbers in Range</h1>
  <p>Open the console to see the prime numbers.</p>
  <script>
    function isPrime(num) {
      if (num <= 1) return false;
      if (num <= 3) return true;

      if (num % 2 === 0 || num % 3 === 0) return false;

      for (let i = 5; i * i <= num; i += 6) {
        if (num % i === 0 || num % (i + 2) === 0) return false;
      }

      return true;
    }

    function printPrimesInRange(start, end) {
      for (let i = start; i <= end; i++) {
        if (isPrime(i)) {
          console.log(i);
        }
      }
    }
```

```
    // Example usage:
    const startRange = 1;
    const endRange = 100;
    printPrimesInRange(startRange, endRange);
  </script>
</body>
</html>
```

JDBC PROGRAM

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.Statement;
import java.sql.SQLException;

public class JdbcExample {

    public static void main(String[] args) {
        String url = "jdbc:mysql://localhost:3306/testdb"; // Replace with your database URL
        String user = "root"; // Replace with your database user
        String password = "password"; // Replace with your database password

        Connection connection = null;
        Statement statement = null;

        try {
            // 1. Establish a connection
            connection = DriverManager.getConnection(url, user, password);
            System.out.println("Connection established successfully.");
```

```
            // 2. Create a statement
            statement = connection.createStatement();

            // 3. Create table
            String createTableSQL = "CREATE TABLE IF NOT EXISTS Users ("
                    + "ID INT(11) NOT NULL AUTO_INCREMENT, "
                    + "NAME VARCHAR(20) NOT NULL, "
                    + "EMAIL VARCHAR(50) NOT NULL, "
                    + "PRIMARY KEY (ID))";
            statement.executeUpdate(createTableSQL);
            System.out.println("Table 'Users' created successfully.");

            // 4. Insert values into table
            String insertSQL1 = "INSERT INTO Users (NAME, EMAIL) VALUES ('John Doe',
'john@example.com')";
            String insertSQL2 = "INSERT INTO Users (NAME, EMAIL) VALUES ('Jane Doe',
'jane@example.com')";
            statement.executeUpdate(insertSQL1);
            statement.executeUpdate(insertSQL2);
            System.out.println("Values inserted successfully into 'Users' table.");

        } catch (SQLException e) {
            e.printStackTrace();
        } finally {
            // 5. Clean up the environment
            try {
                if (statement != null) statement.close();
                if (connection != null) connection.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
```

}