JAVA

1. **Create an array with the values (1, 2, 3, 4, 5, 6, 7) and shuffle it.**

ANS:

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

public class ShuffleArray {
   public static void main(String[] args) {
      Integer[] array = {1, 2, 3, 4, 5, 6, 7};

      List<Integer> list = new ArrayList<>(Arrays.asList(array));
      Collections.shuffle(list);

      Integer[] shuffledArray = list.toArray(new Integer[0]);

      System.out.println("Shuffled Array: " + Arrays.toString(shuffledArray));
   }
}
```

Output:

```
Shuffled Array: [3, 5, 4, 7, 6, 2, 1]
```

Explanation of the code:

These statements import necessary classes from the java.util package. These classes are used for working with lists and shuffling elements. An array of integers named array is created and initialized with the values from 1 to 7.converts the array to a list. This is done using Arrays. As List(array) to create a fixed-size list, and then it is wrapped in an ArrayList to make it mutable. The Collections. shuffle method is used to shuffle the elements in the list. This operation rearranges the elements in a random order.

The shuffled list is then converted back to an array named shuffledArray. The toArray method is used for this purpose. Finally, the shuffled array is printed to the console using System.out.println. The Arrays. toString method is used to convert the array into a string for display.

2. **Enter a Roman Number as input and convert it to an integer. (Example: IX = 9)**

ANS:

```
import java.util.Scanner;

public class RomanToInteger {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
```

```
        System.out.print("Enter a Roman numeral: ");
        String romanNumeral = scanner.nextLine();

        int result = romanToInt(romanNumeral);
        System.out.println("Integer equivalent: " + result);

        scanner.close();
    }

    public static int romanToInt(String s) {

        int result = 0;

        for (int i = 0; i < s.length(); i++) {
            char currentChar = s.charAt(i);
            int currentValue = charToValue(currentChar);

            if (i < s.length() - 1) {
                char nextChar = s.charAt(i + 1);
                int nextValue = charToValue(nextChar);

                if (currentValue < nextValue) {
                    result -= currentValue;
                } else {
                    result += currentValue;
                }
            } else {
                result += currentValue;
            }
        }

        return result;
    }

    public static int charToValue(char c) {

        switch (c) {
            case 'I': return 1;
            case 'V': return 5;
            case 'X': return 10;
            case 'L': return 50;
            case 'C': return 100;
            case 'D': return 500;
            case 'M': return 1000;
            default: return 0;
        }
    }
}
```

Output:

Enter a Roman numeral: IX

Integer equivalent: 9

Explanation of the code:

- Main Method: The entry point of the program, where a Roman numeral string is provided or can be entered by the user.
- romanToInt Function: This function takes a Roman numeral string as input and returns its integer equivalent. It iterates through the Roman numeral characters, compares their values, and performs the conversion.
- charToValue Function: A helper function that maps individual Roman numeral characters to their integer values.

3. **Check if the input is pangram or not. (A pangram is a sentence that contains all the alphabets from A to Z)**
   ANS:

```
public class PangramChecker {
   public static void main(String[] args) {
      String input = "The quick brown fox jumps over the lazy dog"; // Replace this with your
input string
      boolean isPangram = checkIfPangram(input);

      if (isPangram) {
         System.out.println("The input is a pangram.");
      } else {
         System.out.println("The input is not a pangram.");
      }
   }

   public static boolean checkIfPangram(String str) {
      str = str.toLowerCase();

      for (char ch = 'a'; ch <= 'z'; ch++) {
         if (str.indexOf(ch) == -1) {
            return false;
         }
      }

      return true;
   }
}
```

Output:
Input: The quick brown fox jumps over the lazy dog
Output: The input is a pangram.

Explanation of the code:

The main method takes an input string, and the checkIfPangram function checks if the input is a pangram.

The checkIfPangram function first converts the input string to lowercase to make the comparison case-insensitive.

It then iterates through the alphabet from 'a' to 'z'. For each letter, it checks if the letter is present in the string using the indexOf method. If any letter is not found, the function returns false. If all letters are found, the function returns true.

JavaScript

1. Take a sentence as an input and reverse every word in that sentence. Example - This is a sunny day > shiT si a ynnus yad

```
function reverseWordsInSentence(sentence) {
   const words = sentence.split(" ");
   const reversedWords = words.map(word => {
      return word.split('').reverse().join('');
   });
   const reversedSentence = reversedWords.join(" ");
   return reversedSentence;
}

const userInput = prompt("Enter a sentence:"); // Prompt the user for input
const reversedSentence = reverseWordsInSentence(userInput);

if (userInput) {
   console.log("Original sentence: " + userInput);
   console.log("Reversed sentence: " + reversedSentence);
} else {
   console.log("No input provided.");
}

Output:
Enter a sentence: Hello world
Original sentence: Hello world
Reversed sentence: olleH dlrow
```

Explanation of the code:

- The code defines a function reverseWordsInSentence(sentence) that takes a sentence as its parameter.
- Inside the function, it splits the sentence into words using the split method with a space (" ") as the separator. This creates an array called words containing individual words from the input sentence.
- It then uses the map method to create a new array called reversedWords. In this step, it takes each word in the words array, splits it into individual characters, reverses the characters' order, and joins them back together to form a reversed word.
- After reversing all words, it joins them back together into a reversedSentence using the space character as a separator.
- The function returns the reversedSentence.
- The code prompts the user to enter a sentence using the prompt function, and stores the input in a variable called userInput.
- It checks if the user provided any input. If the input is not empty or undefined, it proceeds with the following steps. If there is no input, it prints "No input provided."
- It calls the reverseWordsInSentence function with the userInput as the argument to get the reversedSentence.
- It then displays both the original and reversed sentences in the console

2. Perform sorting of an array in descending order

ANS:

```
const input = prompt("Enter numbers separated by spaces:");
const inputArray = input.split(" ");

if (input) {
   const numbers = inputArray.map(Number); // Convert input strings to numbers
   numbers.sort(function(a, b) {
      return b - a; // Compare in descending order
   });

   console.log("Sorted in descending order: " + numbers.join(", "));
} else {
   console.log("No input provided.");
}
```

Output:

Enter numbers separated by spaces: 5 2 10 1

Sorted in descending order: 10, 5, 2, 1

Explanation of the code:

1. The code uses the prompt function to get user input, which is expected to be a series of numbers separated by spaces.

2. It splits the input string into an array of substrings using the split method with a space (" ") as the separator. This creates an array called inputArray containing the individual number strings.

3. It checks if there is any input. If the input is not empty or undefined, it proceeds with the sorting process. If the input is empty, it prints "No input provided."

4. The map method is used to convert the array of number strings (inputArray) into an array of actual numbers. This is done by applying the Number function to each element, effectively parsing them into numeric values.

5. The sort method is used to sort the numbers array in descending order. It takes a custom comparison function that compares two numbers, a and b. The function subtracts b from a, which sorts the numbers in descending order.

6. Finally, the sorted numbers are displayed in descending order with a message in the console.

HTML

1. **Create a basic calculator using HTML, CSS, and JavaScript with the functionality of add, subtract, multiply and divide. Use the following picture for reference.**

Code.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Calculator</title>
  <style>
    /* Add your CSS styling here */
    body {
      font-family: Arial, sans-serif;
    }

    .calculator {
      width: 250px;
      background: lightgray; /* Change the background color to light gray */
      padding: 20px;
      border-radius: 10px;
      margin: 0 auto;
      text-align: center; /* Center the content horizontally */
    }
```

```
.input-container {
  display: flex;
  align-items: center;
  justify-content: space-between;
  margin: 0; /* Remove margin */
}

.input-text {
  color: #000; /* Text color is black */
  font-size: 24px;
  text-align: right;
  background: white; /* Background color is white */
  border: none;
  width: 70%; /* Adjust the width as needed */
  padding: 0 10px; /* Adjust the vertical padding to match the "AC" button */
  outline: none;
  height: 60px; /* Match the height of the "+" button */
}

.button {
  background: #333;
  color: #fff;
  font-size: 18px;
  padding: 10px;
  border: none;
  border-radius: 0; /* Remove border radius to make the buttons square */
  cursor: pointer;
  height: 60px; /* Fixed height for square buttons */
  margin: 0; /* Remove margin */
}

.ac-button {
  background: green; /* Green background color for "AC" button */
  color: #fff;
  font-size: 18px;
  padding: 10px 20px; /* Adjust padding to match the height of the "+" button */
  border: none;
  border-radius: 0; /* Remove border radius to make the button square */
  cursor: pointer;
  margin: 0; /* Remove margin */
}

.gray-bg-button {
  background: gray; /* Gray background color for + - / * buttons */
}

.button-container {
  display: grid;
  grid-template-columns: repeat(4, 1fr);
  gap: 0; /* Remove the gap between buttons */
```

```
        }

        .equals-button {
            background: #333; /* Restore the background color of "=" button */
        }

        /* Add negative margin to the "7" button to remove the gap with the "8" button */
        .button:nth-child(3) {
            margin-right: -2px;
        }
    </style>
</head>
<body>
    <div class="calculator">
        <div class="input-container">
            <input class="input-text" id="result" readonly>
            <button class="button ac-button" onclick="clearResult()">AC</button>
        </div>
        <div class="button-container">
            <button class="button" onclick="appendToResult(9)">9</button>
            <button class="button" onclick="appendToResult(8)">8</button>
            <button class="button" onclick="appendToResult(7)">7</button>
            <button class="button gray-bg-button" onclick="appendToResult('+')">+</button>
            <button class="button" onclick="appendToResult(4)">4</button>
            <button class="button" onclick="appendToResult(5)">5</button>
            <button class="button" onclick="appendToResult(6)">6</button>
            <button class="button gray-bg-button" onclick="appendToResult('-')">-</button>
            <button class="button" onclick="appendToResult(1)">1</button>
            <button class="button" onclick="appendToResult(2)">2</button>
            <button class="button" onclick="appendToResult(3)">3</button>
            <button class="button gray-bg-button" onclick="appendToResult('/')">/</button>
            <button class="button" onclick="appendToResult('.')">.</button>
            <button class="button zero-button" onclick="appendToResult(0)">0</button>
            <button class="button equals-button" onclick="calculateResult()">=</button>
            <button class="button gray-bg-button" onclick="appendToResult('*')">*</button>
        </div>
    </div>

    <script>
        function clearResult() {
            document.getElementById("result").value = "";
        }

        function appendToResult(value) {
            document.getElementById("result").value += value;
        }

        function calculateResult() {
            try {
                const result = eval(document.getElementById("result").value);
                document.getElementById("result").value = result;
```

```
            } catch (error) {
                document.getElementById("result").value = "Error";
            }
        }
    </script>
</body>
</html>
```

Output:

Explanation of the code:

- The calculator has a light gray background with rounded corners for a visually appealing design.
- The input field has a white background, black text color, and is right-aligned. It's a bit wider than before for improved readability.
- All buttons are square-shaped with a dark gray background for operators (+, -, /, *), green for the "AC" (clear) button, and dark gray for "=" (equal) button.
- The buttons are closely packed without any gap, ensuring a clean look.
- A negative margin is applied to the "7" button to eliminate the gap between "7" and "8" buttons.

- JavaScript functions are included for clearing the input, appending characters to the input field, and calculating the result.

2. Create a survey form with Fields; First Name, Last Name, Date of Birth, Country (dropdown), Gender (checkbox), Profession, email, and mobile number. All the input fields are necessary to submit the form. Create two buttons Submit and Reset. Reset will reset the form while clicking on submit, first, it will check all the fields and necessary validations and then a popup will appear displaying all the selected values with labels in front of it. On closing the popup, the form should reset all the values. Use the following image for reference

```
<!DOCTYPE html>
<html>
<head>
  <title>Survey Form</title>
  <style>
    /* Add your CSS styling here */
    body {
      font-family: Arial, sans-serif;
      background-color: #f0f0f0;
    }
    .form-container {
      background-color: #ffc0cb; /* Set a light pink background color */
      max-width: 400px;
      margin: 20px auto;
      padding: 20px;
      border: 1px solid #ccc;
      border-radius: 5px;
      box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    }
    .form-field {
      margin-bottom: 10px;
    }
    .popup {
      display: none;
```

```
            position: fixed;
            top: 0;
            left: 0;
            width: 100%;
            height: 100%;
            background: rgba(0, 0, 0, 0.7);
            align-items: center;
            justify-content: center;
            text-align: center;
        }
        .popup-content {
            background: #fff;
            border-radius: 5px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);
            padding: 20px;
        }
    </style>
</head>
<body>
    <div class="form-container">
        <h2>Survey Form</h2>
        <form id="surveyForm" onsubmit="return validateForm()">
            <div class="form-field">
                <label for="firstName">First Name:</label>
                <input type="text" id="firstName" required>
            </div>
            <div class="form-field">
                <label for="lastName">Last Name:</label>
                <input type="text" id="lastName" required>
            </div>
            <div class="form-field">
```

```
        <label for="dob">Date of Birth (YYYY-MM-DD):</label>
        <input type="text" id="dob" pattern="[0-9]{4}-[0-9]{2}-[0-9]{2}" required>
      </div>
      <div class="form-field">
        <label for="country">Country:</label>
        <select id="country" required>
          <option value="">Select a country</option>
          <option value="USA">USA</option>
          <option value="Canada">Canada</option>
          <option value="UK">UK</option>
          <option value="France">France</option>
          <option value="Germany">Germany</option>
          <option value="Japan">Japan</option>
          <option value="Australia">Australia</option>
          <option value="India">India</option>
          <option value="Brazil">Brazil</option>
          <option value="China">China</option>
          <option value="South Africa">South Africa</option>
          <option value="Mexico">Mexico</option>
        </select>
      </div>
      <div class="form-field">
        <label>Gender:</label>
        <input type="checkbox" id="male" name="gender" value="Male"> <label
for="male">Male</label>
        <input type="checkbox" id="female" name="gender" value="Female"> <label
for="female">Female</label>
      </div>
      <div class="form-field">
        <label for="profession">Profession:</label>
        <input type="text" id="profession" required>
      </div>
```

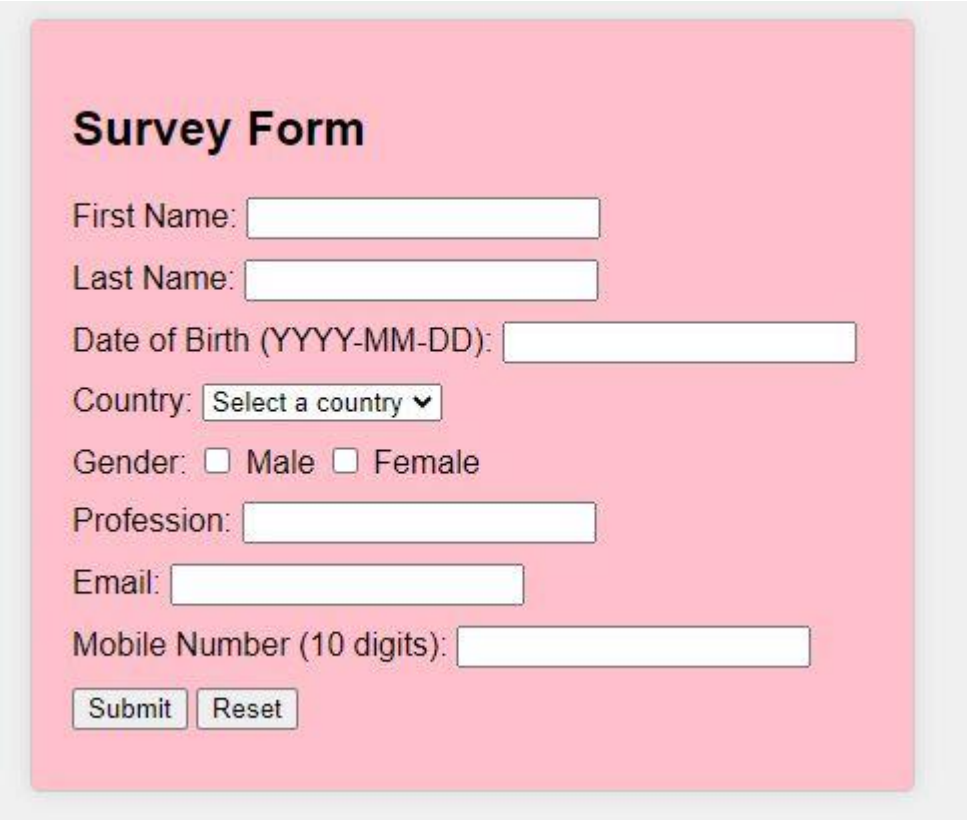
```
      <div class="form-field">
        <label for="email">Email:</label>
        <input type="email" id="email" required>
      </div>
      <div class="form-field">
        <label for="mobile">Mobile Number (10 digits):</label>
        <input type="tel" id="mobile" pattern="[0-9]{10}" required>
      </div>
      <div class="form-field">
        <button type="submit">Submit</button>
        <button type="button" onclick="resetForm()">Reset</button>
      </div>
    </form>
  </div>

  <div id="popup" class="popup">
    <div class="popup-content">
      <h2>Survey Submitted</h2>
      <p>First Name: <span id="popupFirstName"></span></p>
      <p>Last Name: <span id="popupLastName"></span></p>
      <p>Date of Birth: <span id="popupDob"></span></p>
      <p>Country: <span id="popupCountry"></span></p>
      <p>Gender: <span id="popupGender"></span></p>
      <p>Profession: <span id="popupProfession"></span></p>
      <p>Email: <span id="popupEmail"></span></p>
      <p>Mobile Number: <span id="popupMobile"></span></p>
      <button onclick="closePopup()">Close</button>
    </div>
  </div>

  <script>
```

```
function validateForm() {
    const firstName = document.getElementById("firstName").value;
    const lastName = document.getElementById("lastName").value;
    const dob = document.getElementById("dob").value;
    const country = document.getElementById("country").value;
    const gender = getSelectedGender();
    const profession = document.getElementById("profession").value;
    const email = document.getElementById("email").value;
    const mobile = document.getElementById("mobile").value;

    if (!firstName || !lastName || !dob || country === "" || gender === "" || !profession ||
!email || !mobile) {
        alert("Please fill out all required fields.");
        return false;
    }

    // Validate the email format
    const emailPattern = /^[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,4}$/;
    if (!email.match(emailPattern)) {
        alert("Please enter a valid email address.");
        return false;
    }

    // Validate the mobile number format (10 digits)
    const mobilePattern = /^\d{10}$/;
    if (!mobile.match(mobilePattern)) {
        alert("Please enter a valid 10-digit mobile number.");
        return false;
    }

        // Display the values in the popup
```

```
    document.getElementById("popupFirstName").textContent = firstName;
    document.getElementById("popupLastName").textContent = lastName;
    document.getElementById("popupDob").textContent = dob;
    document.getElementById("popupCountry").textContent = country;
    document.getElementById("popupGender").textContent = gender;
    document.getElementById("popupProfession").textContent = profession;
    document.getElementById("popupEmail").textContent = email;
    document.getElementById("popupMobile").textContent = mobile;

    // Show the popup
    document.getElementById("popup").style.display = "block";

    // Reset the form
    document.getElementById("surveyForm").reset();

    return false;
}

function getSelectedGender() {
    const checkboxes = document.querySelectorAll('input[name="gender"]:checked');
    const selectedGender = Array.from(checkboxes).map(checkbox => checkbox.value);
    return selectedGender.join(', ');
}

function resetForm() {
    document.getElementById("surveyForm").reset();
}

function closePopup() {
    document.getElementById("popup").style.display = "none";
}
```

```
    </script>
  </body>
  </html>
```

Output:



Explanation of the code:

HTML:

- The document defines a form with various input fields to collect survey information, including first name, last name, date of birth, country, gender, profession, email, and mobile number. It also provides buttons for submission and resetting the form.

- A div with the class "popup" is defined to create a hidden popup that will display the submitted survey information.

CSS:

- The CSS styles set the overall styling of the form container, input fields, buttons, and the popup. It provides a light pink background for the form and adjusts the layout and appearance of various elements to enhance the visual design.

- The "popup" class is initially set to "display: none" to hide the popup.

JavaScript:

- The JavaScript code defines several functions:

- validateForm(): Validates the form inputs, checks for required fields, valid email format, and a 10-digit mobile number. If any validation fails, it displays an alert message. If the form is valid, it displays the submitted information in the popup, shows the popup, and resets the form.
- getSelectedGender(): Collects the selected gender values from checkboxes and returns them as a comma-separated string.
- resetForm(): Resets the form to its initial state.
- closePopup(): Closes the popup by setting its display to "none."
- Event listeners are assigned to the form for submission and reset actions. When the form is submitted, the validateForm() function is called, and when the reset button is clicked, the resetForm() function is triggered.

- The survey form collects user input and validates it before displaying the submitted information in a popup. If the user's input is incomplete or contains formatting errors, the form provides feedback with alert messages. Once the form is successfully submitted, the popup displays the submitted information, and the form is reset for the next entry.