Java 40 programs on arrays

1. Finding the Largest Element in an Array

```
public class LargestElement {
    public static void main(String[] args) {
        int[] arr = {5, 3, 9, 2, 8};
        int max = arr[0];
        for (int num : arr) {
            if (num > max) {
                max = num;
            }
        }
        System.out.println("Largest element: " + max);
    }
}
```

2. Finding the Smallest Element in an Array

```
public class SmallestElement {
    public static void main(String[] args) {
        int[] arr = {4, 1, 6, 3, 7};
        int min = arr[0];
        for (int num : arr) {
            if (num < min) {
                min = num;
            }
        }
        System.out.println("Smallest element: " + min);
    }
}
```

3. Reversing an Array

```
public class ReverseArray {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5};
        for (int i = 0; i < arr.length / 2; i++) {
            int temp = arr[i];
            arr[i] = arr[arr.length - 1 - i];
            arr[arr.length - 1 - i] = temp;
        }
        System.out.print("Reversed array: ");
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }
}
```

4. Sorting an Array Using Bubble Sort

```
public class BubbleSort {
    public static void main(String[] args) {
        int[] arr = {5, 1, 4, 2, 8};
        for (int i = 0; i < arr.length - 1; i++) {
            for (int j = 0; j < arr.length - 1 - i; j++) {
                if (arr[j] > arr[j + 1]) {
                    int temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;
                }
            }
        }
        System.out.print("Sorted array: ");
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }
}
```

5. Counting Even and Odd Elements in an Array

```
public class CountEvenOdd {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5, 6};
        int evenCount = 0, oddCount = 0;
        for (int num : arr) {
            if (num % 2 == 0) {
                evenCount++;
            } else {
                oddCount++;
            }
        }
        System.out.println("Even count: " + evenCount);
        System.out.println("Odd count: " + oddCount);
    }
}
```

6. Merging Two Arrays

```
public class MergeArrays {
    public static void main(String[] args) {
        int[] arr1 = {1, 3, 5};
        int[] arr2 = {2, 4, 6};
        int[] merged = new int[arr1.length + arr2.length];

        for (int i = 0; i < arr1.length; i++) {
            merged[i] = arr1[i];
        }
        for (int i = 0; i < arr2.length; i++) {
            merged[arr1.length + i] = arr2[i];
        }

        System.out.print("Merged array: ");
        for (int num : merged) {
            System.out.print(num + " ");
        }
    }
}
```

7. Finding the Sum of All Elements in an Array

```
public class ArraySum {
    public static void main(String[] args) {
        int[] arr = {5, 10, 15, 20, 25};
        int sum = 0;
        for (int num : arr) {
            sum += num;
        }
        System.out.println("Sum of all elements: " + sum);
    }
}
```

8. Copying an Array

```
public class CopyArray {
    public static void main(String[] args) {
        int[] original = {1, 2, 3, 4, 5};
        int[] copy = new int[original.length];

        for (int i = 0; i < original.length; i++) {
            copy[i] = original[i];
        }

        System.out.print("Copied array: ");
        for (int num : copy) {
            System.out.print(num + " ");
        }
    }
}
```

9. Removing Duplicates from an Array

```
import java.util.Arrays;

public class RemoveDuplicates {
    public static void main(String[] args) {
        int[] arr = {1, 2, 2, 3, 4, 4, 5};
        Arrays.sort(arr);
        int[] temp = new int[arr.length];
        int j = 0;

        for (int i = 0; i < arr.length - 1; i++) {
            if (arr[i] != arr[i + 1]) {
                temp[j++] = arr[i];
            }
        }
        temp[j++] = arr[arr.length - 1];

        int[] uniqueArr = Arrays.copyOf(temp, j);
        System.out.print("Array after removing duplicates: ");
        for (int num : uniqueArr) {
            System.out.print(num + " ");
        }
    }
}
```

10. Binary Search in a Sorted Array

```
import java.util.Arrays;

public class BinarySearch {
    public static void main(String[] args) {
        int[] arr = {3, 1, 4, 2, 5};
        Arrays.sort(arr);  // Array must be sorted for binary search
        int target = 4;
        int result = binarySearch(arr, target);

        if (result == -1) {
            System.out.println("Element not found.");
        } else {
            System.out.println("Element found at index: " + result);
        }
    }

    public static int binarySearch(int[] arr, int target) {
        int left = 0, right = arr.length - 1;
        while (left <= right) {
            int mid = left + (right - left) / 2;
            if (arr[mid] == target) {
                return mid;
            }
            if (arr[mid] < target) {
                left = mid + 1;
            } else {
                right = mid - 1;
            }
        }
        return -1;
    }
}
```

11. Finding the Largest Element in an Array

```
public class LargestElement {
    public static void main(String[] args) {
        int[] arr = {5, 3, 9, 2, 8};
        int max = arr[0];
        for (int num : arr) {
            if (num > max) {
                max = num;
            }
        }
        System.out.println("Largest element: " + max);
    }
}
```

12. Finding the Smallest Element in an Array

```
public class SmallestElement {
    public static void main(String[] args) {
        int[] arr = {4, 1, 6, 3, 7};
        int min = arr[0];
        for (int num : arr) {
            if (num < min) {
                min = num;
            }
        }
        System.out.println("Smallest element: " + min);
    }
}
```

13. Reversing an Array

```
public class ReverseArray {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5};
        for (int i = 0; i < arr.length / 2; i++) {
            int temp = arr[i];
            arr[i] = arr[arr.length - 1 - i];
            arr[arr.length - 1 - i] = temp;
        }
        System.out.print("Reversed array: ");
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }
}
```

14. Sorting an Array Using Bubble Sort

```
public class BubbleSort {
    public static void main(String[] args) {
        int[] arr = {5, 1, 4, 2, 8};
        for (int i = 0; i < arr.length - 1; i++) {
            for (int j = 0; j < arr.length - 1 - i; j++) {
                if (arr[j] > arr[j + 1]) {
                    int temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;
                }
            }
        }
        System.out.print("Sorted array: ");
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }
}
```

15. Counting Even and Odd Elements in an Array

```
public class CountEvenOdd {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5, 6};
        int evenCount = 0, oddCount = 0;
        for (int num : arr) {
            if (num % 2 == 0) {
                evenCount++;
            } else {
                oddCount++;
            }
        }
        System.out.println("Even count: " + evenCount);
        System.out.println("Odd count: " + oddCount);
    }
}
```

16. Merging Two Arrays

```
public class MergeArrays {
    public static void main(String[] args) {
        int[] arr1 = {1, 3, 5};
        int[] arr2 = {2, 4, 6};
        int[] merged = new int[arr1.length + arr2.length];

        for (int i = 0; i < arr1.length; i++) {
            merged[i] = arr1[i];
        }
        for (int i = 0; i < arr2.length; i++) {
            merged[arr1.length + i] = arr2[i];
        }

        System.out.print("Merged array: ");
        for (int num : merged) {
            System.out.print(num + " ");
        }
    }
}
```

17. Finding the Sum of All Elements in an Array

```
public class ArraySum {
    public static void main(String[] args) {
        int[] arr = {5, 10, 15, 20, 25};
        int sum = 0;
        for (int num : arr) {
            sum += num;
        }
        System.out.println("Sum of all elements: " + sum);
    }
}
```

18. Copying an Array

```
public class CopyArray {
    public static void main(String[] args) {
        int[] original = {1, 2, 3, 4, 5};
        int[] copy = new int[original.length];

        for (int i = 0; i < original.length; i++) {
            copy[i] = original[i];
        }

        System.out.print("Copied array: ");
        for (int num : copy) {
            System.out.print(num + " ");
        }
    }
}
```

19. Removing Duplicates from an Array

```
import java.util.Arrays;

public class RemoveDuplicates {
    public static void main(String[] args) {
        int[] arr = {1, 2, 2, 3, 4, 4, 5};
        Arrays.sort(arr);
        int[] temp = new int[arr.length];
        int j = 0;

        for (int i = 0; i < arr.length - 1; i++) {
            if (arr[i] != arr[i + 1]) {
                temp[j++] = arr[i];
            }
        }
        temp[j++] = arr[arr.length - 1];

        int[] uniqueArr = Arrays.copyOf(temp, j);
        System.out.print("Array after removing duplicates: ");
        for (int num : uniqueArr) {
            System.out.print(num + " ");
        }
    }
}
```

20. Binary Search in a Sorted Array

```
import java.util.Arrays;

public class BinarySearch {
    public static void main(String[] args) {
        int[] arr = {3, 1, 4, 2, 5};
        Arrays.sort(arr);  // Array must be sorted for binary search
        int target = 4;
        int result = binarySearch(arr, target);

        if (result == -1) {
            System.out.println("Element not found.");
        } else {
            System.out.println("Element found at index: " + result);
        }
    }

    public static int binarySearch(int[] arr, int target) {
        int left = 0, right = arr.length - 1;
        while (left <= right) {
            int mid = left + (right - left) / 2;
            if (arr[mid] == target) {
                return mid;
            }
            if (arr[mid] < target) {
                left = mid + 1;
            } else {
                right = mid - 1;
            }
        }
        return -1;
    }
}
```

21. Finding the Second Largest Element in an Array

```
public class SecondLargest {
    public static void main(String[] args) {
        int[] arr = {4, 7, 1, 9, 3};
        int largest = Integer.MIN_VALUE, secondLargest =
Integer.MIN_VALUE;

        for (int num : arr) {
            if (num > largest) {
                secondLargest = largest;
                largest = num;
            } else if (num > secondLargest && num != largest) {
                secondLargest = num;
            }
        }
        System.out.println("Second largest element: " +
secondLargest);
    }
}
```

22. Inserting an Element at a Specific Position in an Array

```
public class InsertElement {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5};
        int element = 10;
        int position = 3;

        int[] newArr = new int[arr.length + 1];

        for (int i = 0, j = 0; i < newArr.length; i++) {
            if (i == position) {
                newArr[i] = element;
            } else {
                newArr[i] = arr[j++];
            }
        }

        System.out.print("Array after insertion: ");
        for (int num : newArr) {
            System.out.print(num + " ");
        }
    }
}
```

23. Deleting an Element from a Specific Position in an Array

```
public class DeleteElement {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5};
        int position = 2;
        int[] newArr = new int[arr.length - 1];

        for (int i = 0, j = 0; i < arr.length; i++) {
            if (i == position) continue;
            newArr[j++] = arr[i];
        }

        System.out.print("Array after deletion: ");
        for (int num : newArr) {
            System.out.print(num + " ");
        }
    }
}
```

24. Finding the Frequency of Each Element in an Array

```
public class FrequencyCount {
    public static void main(String[] args) {
        int[] arr = {2, 3, 2, 4, 5, 3, 5};
        boolean[] visited = new boolean[arr.length];

        for (int i = 0; i < arr.length; i++) {
            if (!visited[i]) {
                int count = 1;
                for (int j = i + 1; j < arr.length; j++) {
                    if (arr[i] == arr[j]) {
                        visited[j] = true;
                        count++;
                    }
                }
                System.out.println(arr[i] + " occurs " + count + " times");
            }
        }
    }
}
```

25. Finding the Sum of Diagonal Elements in a 2D Array (Matrix)

```
public class DiagonalSum {
    public static void main(String[] args) {
        int[][] matrix = {
            {1, 2, 3},
            {4, 5, 6},
            {7, 8, 9}
        };
        int sum = 0;
        for (int i = 0; i < matrix.length; i++) {
            sum += matrix[i][i];
        }
        System.out.println("Sum of diagonal elements: " + sum);
    }
}
```

26. Transposing a Matrix

```
public class MatrixTranspose {
    public static void main(String[] args) {
        int[][] matrix = {
            {1, 2, 3},
            {4, 5, 6}
        };

        int[][] transpose = new int[matrix[0].length][matrix.length];

        for (int i = 0; i < matrix.length; i++) {
            for (int j = 0; j < matrix[0].length; j++) {
                transpose[j][i] = matrix[i][j];
            }
        }

        System.out.println("Transpose of the matrix:");
        for (int[] row : transpose) {
            for (int val : row) {
                System.out.print(val + " ");
            }
            System.out.println();
        }
    }
}
```

27. Shifting Elements in an Array to the Left

```
public class LeftShiftArray {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5};
        int shiftCount = 2;

        for (int i = 0; i < shiftCount; i++) {
            int first = arr[0];
            for (int j = 0; j < arr.length - 1; j++) {
                arr[j] = arr[j + 1];
            }
            arr[arr.length - 1] = first;
        }
        System.out.print("Array after left shift: ");
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }
}
```

28. Rotating Elements in an Array to the Right

```
public class RightRotateArray {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5};
        int rotateCount = 2;

        for (int i = 0; i < rotateCount; i++) {
            int last = arr[arr.length - 1];
            for (int j = arr.length - 1; j > 0; j--) {
                arr[j] = arr[j - 1];
            }
            arr[0] = last;
        }

        System.out.print("Array after right rotation: ");
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }
}
```

29. Finding the Pair of Elements with a Given Sum

```
public class PairWithSum {
    public static void main(String[] args) {
        int[] arr = {1, 4, 5, 2, 3};
        int targetSum = 6;

        System.out.println("Pairs with sum " + targetSum + ":");
        for (int i = 0; i < arr.length; i++) {
            for (int j = i + 1; j < arr.length; j++) {
                if (arr[i] + arr[j] == targetSum) {
                    System.out.println("(" + arr[i] + ", " + arr[j] + ")");
                }
            }
        }
    }
}
```

30. Counting Positive, Negative, and Zero Elements in an Array

```
public class CountPosNegZero {
    public static void main(String[] args) {
        int[] arr = {3, -1, 0, -3, 5, 0, -2, 2};
        int positiveCount = 0, negativeCount = 0, zeroCount = 0;

        for (int num : arr) {
            if (num > 0) {
                positiveCount++;
            } else if (num < 0) {
                negativeCount++;
            } else {
                zeroCount++;
            }
        }

        System.out.println("Positive count: " + positiveCount);
        System.out.println("Negative count: " + negativeCount);
        System.out.println("Zero count: " + zeroCount);
    }
}
```

31. Reversing an Array

```
public class ReverseArray {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5};
        for (int i = 0; i < arr.length / 2; i++) {
            int temp = arr[i];
            arr[i] = arr[arr.length - i - 1];
            arr[arr.length - i - 1] = temp;
        }
        System.out.print("Reversed Array: ");
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }
}
```

32. Removing Duplicates from an Array

```
import java.util.Arrays;

public class RemoveDuplicates {
    public static void main(String[] args) {
        int[] arr = {1, 2, 2, 3, 4, 4, 5};
        int[] uniqueArr = Arrays.stream(arr).distinct().toArray();
        System.out.print("Array after removing duplicates: ");
        for (int num : uniqueArr) {
            System.out.print(num + " ");
        }
    }
}
```

33. Finding the Intersection of Two Arrays

```
import java.util.HashSet;

public class ArrayIntersection {
    public static void main(String[] args) {
        int[] arr1 = {1, 2, 3, 4, 5};
        int[] arr2 = {3, 4, 5, 6, 7};
        HashSet<Integer> set = new HashSet<>();

        for (int i : arr1) {
            set.add(i);
        }

        System.out.print("Intersection: ");
        for (int i : arr2) {
            if (set.contains(i)) {
                System.out.print(i + " ");
            }
        }
    }
}
```

34. Merging Two Sorted Arrays

```
import java.util.Arrays;

public class MergeSortedArrays {
    public static void main(String[] args) {
        int[] arr1 = {1, 3, 5, 7};
        int[] arr2 = {2, 4, 6, 8};
        int[] merged = new int[arr1.length + arr2.length];

        int i = 0, j = 0, k = 0;
        while (i < arr1.length && j < arr2.length) {
            merged[k++] = (arr1[i] < arr2[j]) ? arr1[i++] : arr2[j++];
        }
        while (i < arr1.length) merged[k++] = arr1[i++];
        while (j < arr2.length) merged[k++] = arr2[j++];

        System.out.print("Merged Array: ");
        for (int num : merged) {
            System.out.print(num + " ");
        }
    }
}
```

35. Finding the Maximum Product of Two Elements

```
public class MaxProduct {
    public static void main(String[] args) {
        int[] arr = {1, 20, 3, 4, 5};
        int max1 = Integer.MIN_VALUE, max2 =
Integer.MIN_VALUE;

        for (int num : arr) {
            if (num > max1) {
                max2 = max1;
                max1 = num;
            } else if (num > max2) {
                max2 = num;
            }
        }
        System.out.println("Maximum product of two elements: " +
(max1 * max2));
    }
}
```

36. Finding Missing Number in a Sequence (1 to N)

```
public class MissingNumber {
    public static void main(String[] args) {
        int[] arr = {1, 2, 4, 5, 6}; // Missing 3
        int n = arr.length + 1;
        int totalSum = n * (n + 1) / 2;

        for (int num : arr) {
            totalSum -= num;
        }
        System.out.println("Missing number: " + totalSum);
    }
}
```

37. Counting the Number of Even and Odd Elements

```
public class CountEvenOdd {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5, 6};
        int evenCount = 0, oddCount = 0;

        for (int num : arr) {
            if (num % 2 == 0) {
                evenCount++;
            } else {
                oddCount++;
            }
        }
        System.out.println("Even count: " + evenCount);
        System.out.println("Odd count: " + oddCount);
    }
}
```

38. Finding the Longest Sequence of Consecutive Numbers

```
import java.util.HashSet;

public class LongestConsecutiveSequence {
    public static void main(String[] args) {
        int[] arr = {1, 9, 3, 10, 4, 20, 2};
        HashSet<Integer> set = new HashSet<>();
        int maxLength = 0;

        for (int num : arr) set.add(num);

        for (int num : arr) {
            if (!set.contains(num - 1)) {
                int currentNum = num;
                int length = 1;
                while (set.contains(currentNum + 1)) {
                    currentNum++;
                    length++;
                }
                maxLength = Math.max(maxLength, length);
            }
        }
        System.out.println("Length of longest consecutive sequence: 
" + maxLength);
    }
}
```

39. Checking if an Array is a Palindrome

```
public class PalindromeArray {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 2, 1};
        boolean isPalindrome = true;

        for (int i = 0; i < arr.length / 2; i++) {
            if (arr[i] != arr[arr.length - i - 1]) {
                isPalindrome = false;
                break;
            }
        }
        System.out.println("Is array a palindrome? " + isPalindrome);
    }
}
```

40. Splitting an Array into Two Equal Sum Subarrays

```
public class SplitArrayEqualSum {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 5, 2};
        int leftSum = 0, rightSum = 0;

        for (int num : arr) rightSum += num;

        boolean canSplit = false;
        for (int i = 0; i < arr.length; i++) {
            leftSum += arr[i];
            rightSum -= arr[i];

            if (leftSum == rightSum) {
                canSplit = true;
                System.out.println("Array can be split at index: " + i);
                break;
            }
        }

        if (!canSplit) {
            System.out.println("Array cannot be split into equal sum
subarrays.");
        }
    }
}
```