Here are some more complex Java programs that utilize arrays and ArrayLists, demonstrating various advanced operations and concepts:

1. \*\*Matrix Multiplication\*\*

```java

public class MatrixMultiplication {

public static void main(String[] args) {

int[][] matrixA = {

{1, 2, 3},

{4, 5, 6}

};

int[][] matrixB = {

{7, 8},

{9, 10},

{11, 12}

};

int[][] result = new int[matrixA.length][matrixB[0].length];

for (int i = 0; i < matrixA.length; i++) {

for (int j = 0; j < matrixB[0].length; j++) {

for (int k = 0; k < matrixA[0].length; k++) {

result[i][j] += matrixA[i][k] \* matrixB[k][j];

}

}

}

System.out.println("Resultant Matrix:");

for (int[] row : result) {

for (int val : row) {

System.out.print(val + " ");

}

System.out.println();

}

}

}

```

2. \*\*Finding the Second Largest Element in an Array\*\*

```java

public class SecondLargest {

public static void main(String[] args) {

int[] numbers = {12, 35, 1, 10, 34, 1};

int first = Integer.MIN\_VALUE;

int second = Integer.MIN\_VALUE;

for (int number : numbers) {

if (number > first) {

second = first;

first = number;

} else if (number > second && number != first) {

second = number;

}

}

if (second == Integer.MIN\_VALUE) {

System.out.println("There is no second largest element.");

} else {

System.out.println("The second largest element is: " + second);

}

}

}

```

3. \*\*Count Frequency of Each Element in an Array\*\*

```java

import java.util.HashMap;

public class FrequencyCount {

public static void main(String[] args) {

int[] numbers = {1, 3, 2, 3, 1, 4, 5, 2};

HashMap<Integer, Integer> frequencyMap = new HashMap<>();

for (int number : numbers) {

frequencyMap.put(number, frequencyMap.getOrDefault(number, 0) + 1);

}

System.out.println("Frequency of elements:");

for (Integer key : frequencyMap.keySet()) {

System.out.println(key + ": " + frequencyMap.get(key));

}

}

}

```

4. \*\*Rotate an Array Left by K Positions\*\*

```java

import java.util.Arrays;

public class RotateArray {

public static void main(String[] args) {

int[] arr = {1, 2, 3, 4, 5};

int k = 2; // Number of positions to rotate

k = k % arr.length; // In case k is greater than array length

reverse(arr, 0, arr.length - 1);

reverse(arr, 0, arr.length - k - 1);

reverse(arr, arr.length - k, arr.length - 1);

System.out.println("Rotated array: " + Arrays.toString(arr));

}

private static void reverse(int[] arr, int start, int end) {

while (start < end) {

int temp = arr[start];

arr[start] = arr[end];

arr[end] = temp;

start++;

end--;

}

}

}

```

5. \*\*Merge Two Sorted Arrays\*\*

```java

import java.util.Arrays;

public class MergeSortedArrays {

public static void main(String[] args) {

int[] array1 = {1, 3, 5};

int[] array2 = {2, 4, 6};

int[] mergedArray = new int[array1.length + array2.length];

int i = 0, j = 0, k = 0;

while (i < array1.length && j < array2.length) {

if (array1[i] < array2[j]) {

mergedArray[k++] = array1[i++];

} else {

mergedArray[k++] = array2[j++];

}

}

while (i < array1.length) mergedArray[k++] = array1[i++];

while (j < array2.length) mergedArray[k++] = array2[j++];

System.out.println("Merged sorted array: " + Arrays.toString(mergedArray));

}

}

```

6. \*\*Implementing a Dynamic ArrayList\*\*

```java

import java.util.Arrays;

public class DynamicArrayList<T> {

private Object[] elements;

private int size;

private static final int DEFAULT\_CAPACITY = 10;

public DynamicArrayList() {

elements = new Object[DEFAULT\_CAPACITY];

}

public void add(T element) {

ensureCapacity();

elements[size++] = element;

}

@SuppressWarnings("unchecked")

public T get(int index) {

if (index >= size || index < 0)

throw new IndexOutOfBoundsException("Index: " + index + ", Size: " + size);

return (T) elements[index];

}

private void ensureCapacity() {

if (size == elements.length) {

elements = Arrays.copyOf(elements, size \* 2);

}

}

public int size() {

return size;

}

public static void main(String[] args) {

DynamicArrayList<String> list = new DynamicArrayList<>();

list.add("Hello");

list.add("World");

System.out.println("Element at index 0: " + list.get(0));

System.out.println("Size of the list: " + list.size());

}

}

```

7. \*\*Find All Pairs in an Array That Sum to a Specific Value\*\*

```java

import java.util.HashSet;

public class PairSumFinder {

public static void main(String[] args) {

int[] numbers = {1, 5, 7, -1, 5};

int targetSum = 6;

HashSet<Integer> set = new HashSet<>();

System.out.println("Pairs with sum " + targetSum + ":");

for (int number : numbers) {

int complement = targetSum - number;

if (set.contains(complement)) {

System.out.println("(" + complement + ", " + number + ")");

}

set.add(number);

}

}

}

```

8. \*\*Transpose a Matrix\*\*

```java

public class MatrixTranspose {

public static void main(String[] args) {

int[][] matrix = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

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

for (int i = 0; i < matrix.length; i++) {

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

transposedMatrix[j][i] = matrix[i][j];

}

}

System.out.println("Transposed Matrix:");

for (int[] row : transposedMatrix) {

for (int val : row) {

System.out.print(val + " ");

}

System.out.println();

}

}

}

```

9. \*\*Find Missing Number in an Array\*\*

```java

public class MissingNumberFinder {

public static void main(String[] args) {

int[] numbers = {3, 7, 1, 2, 8}; // Missing number is '4'

int n = numbers.length + 1; // Total numbers should be n

// Calculate expected sum of first n natural numbers

int expectedSum = n \* (n + 1) / 2;

// Calculate actual sum of the given numbers

int actualSum = 0;

for (int number : numbers) actualSum += number;

// The missing number is the difference

System.out.println("The missing number is: " + (expectedSum - actualSum));

}

}

```

10. \*\*Kth Smallest Element in an Unsorted Array\*\*

```java

import java.util.Arrays;

public class KthSmallestElementFinder {

public static void main(String[] args) {

int[] arr = {7, 10, 4, 3, 20, 15};

int k = 3; // Find the kth smallest element

Arrays.sort(arr); // Sort the array

System.out.println(k + "th smallest element is: " + arr[k - 1]);

}

}

```

These programs cover a range of more complex operations involving arrays and ArrayLists in Java. They include matrix manipulations and algorithms that are commonly used in programming challenges and real-world applications.

Citations:

[1] https://gyansetu-core-java-for-java.gitbook.io/project/untitled-1/creating-and-using-list-set-and-deque-implementations/arraylist

[2] https://www.javatpoint.com/java-arraylist

[3] https://www.javatpoint.com/array-in-java

[4] https://www.shiksha.com/online-courses/articles/array-programs-in-java/

[5] https://docs.oracle.com/javase/tutorial/java/nutsandbolts/arrays.html

[6] https://www.w3schools.com/java/java\_arrays.asp

[7] https://www.geeksforgeeks.org/array-manipulation-and-sum/

[8] https://www.w3resource.com/java-exercises/collection/array-list.php