1. Rotate array by k steps

import java.util.Arrays;

public class ArrayRotation {

*// Function to rotate the array to the right by k steps*

public static void rotateArray(int[] arr, int k) {

int n = arr.length;

*// If k is greater than the length of the array, reduce it to avoid unnecessary rotations*

k = k % n;

*// Reverse the whole array*

reverseArray(arr, 0, n - 1);

*// Reverse the first k elements*

reverseArray(arr, 0, k - 1);

*// Reverse the remaining elements*

reverseArray(arr, k, n - 1);

}

*// Function to reverse a part of the array*

public static void reverseArray(int[] arr, int start, int end) {

while (start < end) {

*// Swap the elements*

int temp = arr[start];

arr[start] = arr[end];

arr[end] = temp;

*// Move the pointers towards the center*

start++;

end--;

}

}

*// Function to print the array*

public static void printArray(int[] arr) {

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

System.out.print(arr[i] + " ");

}

System.out.println();

}

public static void main(String[] args) {

*// Example array*

int[] arr = {1, 2, 3, 4, 5, 6, 7};

*// Number of positions to rotate*

int k = 3;

System.out.println("Original array:");

printArray(arr);

*// Rotate the array to the right by k steps*

rotateArray(arr, k);

System.out.println("Array after rotating right by " + k + " steps:");

printArray(arr);

}

}

1. Take 2 numbers and find gcd (greatest common divisor)

public class GCD {

*// Method to calculate GCD using Euclidean algorithm*

public static int gcd(int a, int b) {

*// Loop until one of the numbers becomes zero*

while (b != 0) {

int temp = b;

b = a % b; *// Remainder of a divided by b*

a = temp; *// Update a to the value of b*

}

return a; *// a will be the GCD when b becomes zero*

}

public static void main(String[] args) {

int num1 = 56;

int num2 = 98;

*// Call the gcd method and print the result*

System.out.println("GCD of " + num1 + " and " + num2 + " is: " + gcd(num1, num2));

}

}