```java
import java.util.Random;
import java.util.Scanner;

public class MergeArray {
    public static Random rand = new Random();
    public static int i = 0;
    public static int j = 0;
    public static int k = 0;

    // returns true if a number exist in the given array
    public static boolean exist(int[] arr, int x){
        for (int i = 0; i < arr.length; i++) {
            if (arr[i] == x)
                return true;
        }
        return false;
    }

    // returns a new random array with unique random numbers between min
and max values
    public static int[] getRandomArray(int arrLength, int min, int max){
        int[] arr = new int[arrLength];
        int temp;
        for (int i = 0; i < arr.length; i++)
        {
            do {
                temp = rand.nextInt(min, max);
            }while (exist(arr,temp));
            arr[i] = temp;
        }
        return arr;
    }

    // prints an array
    public static void printArr(int[] arr, int arrLength){
        for (int i = 0; i < arrLength; i++) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();
    }

    // Sorts an array in bubble sort fashion
    public static void bubbleSort(int[] arr) {
        int i = arr.length - 1;
        boolean sorted = false;

        while (!sorted && i > 0) {
            sorted = true;
            for (int j = 0; j < i; j++) {
                if (arr[j] > arr[j + 1]) {
                    swap(arr, j, j + 1);
                    sorted = false;
                }
            }
            i--;
        }
    }

    // swaps 2 numbers in the array
    public static void swap(int[] arr, int x, int y) {
        int temp = arr[x];
        arr[x] = arr[y];
        arr[y] = temp;
    }

    // Sorts an array in selection ךort fashion
    public static void selectionSort(int[] arr){
        int p;
        for (int i = arr.length -1; i > 0 ; i--) {
            p = maxPlaceInArr(arr, i);
            swap(arr, i, p);
        }
    }
    // פעולה שמקבלת מערך ומחזירה את מיקומו של האיבר הכי גדול במערך עד מקום זה
    public static int maxPlaceInArr(int[] arr, int place){
        int max = 0;
        for (int i = 1; i <= place; i++) {
            if (arr[i] > arr[max])
                max = i;
        }
        return max;
    }


    // merges two sorted array into a bigger array and returns it and removes any duplicate numbers
    public static int[] mergeArrs(int[] a, int[] b){
        int[] c = new int[a.length+b.length];
        i=0;
        j=0;
        k=0;
        while (i < a.length && j < b.length ){
            if(!exist(c, a[i])){
                if(!exist(c, b[j])){
                    if(a[i] < b[j])
                    {
                        c[k] = a[i];
                        i++;
                    } else {
                        c[k] = b[j];
                        j++;
                    }
                    k++;
                } else j++;
            } else i++;

        }
        while (i<a.length){
            if(!exist(c, a[i])){
                c[k] = a[i];
                k++;
            }
            i++;
        }
        while (j<b.length){
            if(!exist(c, b[j])){
                c[k] = b[j];
                k++;
            }
            j++;
        }
        return c;
    }

    // merges two sorted array into a bigger array and returns it
    public static int[] mergeArr(int[] a, int[] b){
        int[] c = new int[a.length+b.length];
        i=0;
        j=0;
        k=0;
        while (i < a.length && j < b.length ){
            if(a[i] < b[j])
            {
                c[k] = a[i];
                i++;
            } else {
                c[k] = b[j];
                j++;
            }
            k++;
        }
        while (i<a.length){
            c[k] = a[i];
            i++;
            k++;
        }
        while (j<b.length){
```

```java
            c[k] = b[j];
            j++;
            k++;
        }
        return c;
    }

    // returns a sorted array that consists of numbers who appear in both arrays
    // Function to find the intersection of two sorted arrays
    public static int[] intersectArr(int[] a, int[] b) {
        i = 0;
        j = 0;
        k = 0;
        int[] c = new int[a.length + b.length];
        while (i < a.length && j < b.length) {
            if (a[i] < b[j]) {
                i++;
            } else if (a[i] > b[j]) {
                j++;
            } else {
                c[k] = a[i];
                i++;
                j++;
                k++;
            }
        }
        return c;
    }

    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        System.out.print("enter the size of the first array --> ");
        int n = input.nextInt();
        System.out.print("enter the size of the 2nd array --> ");
        int m = input.nextInt();

        int[] a = getRandomArray(n, 1, 20);
        int[] b = getRandomArray(m, 1, 100);
        printArr(a, a.length);
        printArr(b, b.length);

        bubbleSort(a);
        printArr(a, a.length);
        selectionSort(b);
        printArr(b, b.length);

        int[] c = mergeArr(a, b);
        printArr(c, c.length);

        int[] d = mergeArrs(a,b);
        printArr(d, k);

        int[] e = intersectArr(a,b);
        printArr(e, k);
    }
}
/*
enter the size of the first array --> 7
enter the size of the 2nd array --> 9

1 13 3 16 19 12 6
54 11 6 99 20 43 32 59 9

1 3 6 12 13 16 19
6 9 11 20 32 43 54 59 99

1 3 6 6 9 11 12 13 16 19 20 32 43 54 59 99
1 3 6 9 11 12 13 16 19 20 32 43 54 59 99
6
* */


```