**Tugas Praktikum**
**Section 7 - Iterable & Map Data Structure**

1. Program menggabungkan 2 array yang diberikan dan jangan sampai terdapat nama yang sama di data yang sudah tergabung tadi.
   **Sample Test Cases**
   input : ['kazuya', 'jin', 'lee'], ['kazuya', 'feng']
   output : ['kazuya', 'jin', 'lee', 'feng']

```java
package menggabungkan.pkg2.array;

import java.util.*;

public class Menggabungkan2Array {

    public static void main(String[] args) {
        String[] arr1 = {"kazuya", "jin", "lee"};
        String[] arr2 = {"kazuya", "feng"};
        String[] mergedArr = mergeArrays(arr1, arr2);
        System.out.println(Arrays.toString(mergedArr));
    }

    public static String[] mergeArrays(String[] arr1, String[] arr2){
        Set<String> set = new LinkedHashSet<>(Arrays.asList(arr1));
        set.addAll(Arrays.asList(arr2));
        return set.toArray(new String[0]);
    }

}
```

```
run:
[kazuya, jin, lee, feng]
BUILD SUCCESSFUL (total time: 0 seconds)
```

   input : ['lee', 'jin'], ['kazuya', 'panda']
   output : ['lee', 'jin', 'kazuya', 'panda']

```java
    public static void main(String[] args) {
        String[] arr1 = {"lee", "jin"};
        String[] arr2 = {"kazuya", "panda"};
        String[] mergedArr = mergeArrays(arr1, arr2);
        System.out.println(Arrays.toString(mergedArr));
    }
```

```
Output - Menggabungkan 2 Array (run)
run:
[lee, jin, kazuya, panda]
BUILD SUCCESSFUL (total time: 0 seconds)
```

2. Buat program sesuai dengan deskripsi di bawah. Input merupakan variabel string berisi kumpulan angka. Output merupakan list / array berisi angka yang hanya muncul 1 kali pada input.

   **Sample Test Cases**

   input : “76523752”

   output : [6, 3]

```java
package uniquenumbers;
import java.util.*;
public class UniqueNumbers {
    public static void main(String[] args) {
        String input = "76523752";
        int[] uniqueNumbers = findUniqueNumbers(input);
        System.out.println(Arrays.toString(uniqueNumbers));
    }

    public static int[] findUniqueNumbers(String input) {
        Map<Character, Integer> counts = new HashMap<>();
        // Menghitung jumlah kemunculan setiap angka di input
        for (char c : input.toCharArray()) {
            counts.put(c, counts.getOrDefault(c, 0) + 1);
        }
        // Membuat list untuk menampung angka-angka yang hanya muncul satu kali
        List<Integer> unique = new ArrayList<>();
        for (char c : input.toCharArray()) {
            if (counts.get(c) == 1) {
                unique.add(Character.getNumericValue(c));
            }
        }
        // Mengubah list menjadi array
        int[] uniqueNumbers = new int[unique.size()];
        for (int i = 0; i < uniqueNumbers.length; i++) {
            uniqueNumbers[i] = unique.get(i);
        }
        return uniqueNumbers;
    }
}
```

```
Output - UniqueNumbers (run)
run:
[6, 3]
BUILD SUCCESSFUL (total time: 0 seconds)
```

input : “1122”
output : []

```
3   public class UniqueNumbers {
4       public static void main(String[] args) {
5           String input = "1122";
6           int[] uniqueNumbers = findUniqueNumbers(input);
7           System.out.println(Arrays.toString(uniqueNumbers));
8       }
9
```

```
Output - UniqueNumbers (run)
run:
[]
BUILD SUCCESSFUL (total time: 0 seconds)
```

3. Given an array of sorted numbers and a target sum, find a pair in the array whose sum is equal to the given target. Write a function to return the indices of the two numbers (i.e. the pair) such that they add up to the given target.
   **Challenges** :
   Solve with linear complexity O(n), **not** O(n^2) if you can!
   **Sample Test Cases**
   input : [1, 2, 3, 4, 6], target=6
   output : [1, 3]
   Explanation: The numbers at index 1 and 3 add up to 6: 2+4=6

Start Page | UniqueNumbers.java | TwoSum.java

Source | History

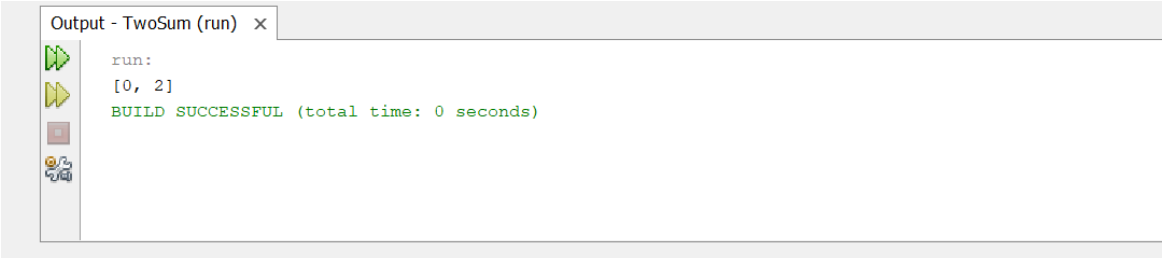
```java
package twosum;

import java.util.*;

public class TwoSum {
    public static void main(String[] args) {
        int[] nums = {1, 2, 3, 4, 6};
        int target = 6;
        int[] result = findTwoSum(nums, target);
        System.out.println(Arrays.toString(result));
    }

    public static int[] findTwoSum(int[] nums, int target) {
        int left = 0;
        int right = nums.length - 1;

        while (left < right) {
            int sum = nums[left] + nums[right];

            if (sum == target) {
                return new int[] { left, right };
            } else if (sum < target) {
                left++;
            } else {
                right--;
            }
        }

        throw new IllegalArgumentException("No two sum solution");
    }
}
```

Output - TwoSum (run)

```
run:
[1, 3]
BUILD SUCCESSFUL (total time: 0 seconds)
```

input : [2, 5, 9, 11], target=11
output : [0, 2]
Explanation : The numbers at index 0 and 2 add up to 11: 2+9=11

```java
public class TwoSum {
    public static void main(String[] args) {
        int[] nums = {2, 5, 9, 11};
        int target = 11;
        int[] result = findTwoSum(nums, target);
        System.out.println(Arrays.toString(result));
    }
```




4. Buatlah sebuah program **ArrayUnique** yang menerima 2 parameter berupa array angka. Output adalah program adalah satu array berupa kumpulan angka di array pertama tetapi tidak memiliki duplikasi di di array kedua.
   **Sample Test Case**
   input : [1, 2, 3, 4] dan [1, 3, 5, 10, 16]
   output : [2, 4]

```java
package arrayunique;
import java.util.*;
public class ArrayUnique {
    public static void main(String[] args) {
        int[] arr1 = {1, 2, 3, 4};
        int[] arr2 = {1, 3, 5, 10, 16};
        int[] result = getUniqueElements(arr1, arr2);
        System.out.println(Arrays.toString(result));
    }
    public static int[] getUniqueElements(int[] arr1, int[] arr2) {
        Set<Integer> set = new HashSet<>();
        Set<Integer> duplicates = new HashSet<>();

        for (int num : arr2) {
            set.add(num);
        }

        for (int num : arr1) {
            if (!set.contains(num)) {
                duplicates.add(num);
            }
        }

        int[] result = new int[duplicates.size()];
        int index = 0;
        for (int num : duplicates) {
            result[index++] = num;
        }
        return result;
    }
}
```

```
run:
[2, 4]
BUILD SUCCESSFUL (total time: 0 seconds)
```

input : [3, 8] dan [2, 8]
output : [3]

```java
public class ArrayUnique {
    public static void main(String[] args) {
        int[] arr1 = {3, 8};
        int[] arr2 = {2, 8};
        int[] result = getUniqueElements(arr1, arr2);
        System.out.println(Arrays.toString(result));
    }
```

```
Output - ArrayUnique (run)
run:
[3]
BUILD SUCCESSFUL (total time: 0 seconds)
```

5. Given an array of sorted numbers, remove all duplicates from it. You should not use any extra space; after removing the duplicates in-place return the length of the subarray that has no duplicate in it.

   **Sample Test Case**

   input : [2, 3, 3, 3, 6, 9, 9]

   output : 4

   Explanation : The first four elements after removing the duplicates will be [2, 3, 6, 9].

```
Start Page | RemoveDuplicates.java
Source | History
```

```java
package removeduplicates;

public class RemoveDuplicates {
    public static int remove(int[] arr) {
        if (arr == null || arr.length == 0) {
            return 0;
        }
        int nonDuplicateIndex = 1;
        for (int i = 1; i < arr.length; i++) {
            if (arr[i] != arr[nonDuplicateIndex - 1]) {
                arr[nonDuplicateIndex++] = arr[i];
            }
        }
        return nonDuplicateIndex;
    }
    public static void main(String[] args) {
        int[] arr = {2,3,3,3,6,9,9};
        int length = remove(arr);
        System.out.println("Length of subarray with no duplicates: " + length);
        System.out.print("Array after removing duplicates: ");
        for (int i = 0; i < length; i++) {
            System.out.print(arr[i] + " ");
        }
    }
}
```

```
Output - RemoveDuplicates (run)
run:
Length of subarray with no duplicates: 4
Array after removing duplicates: 2 3 6 9 BUILD SUCCESSFUL (total time: 0 seconds)
```

input : [2, 2, 2, 11]
output : 2
Explanation : The first two elements after removing the duplicates will be [2, 11].

Start Page | RemoveDuplicates.java

Source | History

```java
package removeduplicates;

public class RemoveDuplicates {
    public static int remove(int[] arr) {
        if (arr == null || arr.length == 0) {
            return 0;
        }
        int nonDuplicateIndex = 1;
        for (int i = 1; i < arr.length; i++) {
            if (arr[i] != arr[nonDuplicateIndex - 1]) {
                arr[nonDuplicateIndex++] = arr[i];
            }
        }
        return nonDuplicateIndex;
    }
    public static void main(String[] args) {
        int[] arr = {2,2,2,11};
        int length = remove(arr);
        System.out.println("Length of subarray with no duplicates: " + length);
        System.out.print("Array after removing duplicates: ");
        for (int i = 0; i < length; i++) {
            System.out.print(arr[i] + " ");
        }
    }
}
```

Output - RemoveDuplicates (run)

```
run:
Length of subarray with no duplicates: 2
Array after removing duplicates: 2 11 BUILD SUCCESSFUL (total time: 0 seconds)
```