ARRAYS - 01

1)

```java
package arrays_01;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

public class TargetIndices {
    static List<List<Integer>> twoSum(int[] nums, int target) {
        Arrays.sort(nums);
        List<List<Integer>> res = new ArrayList<>();
        int l = 0, h = nums.length - 1;
        while (l < h) {
            int sum = nums[l] + nums[h];
            if (sum < target) {
                l++;
            } else if (sum > target) {
                h--;
            } else {
                res.add(Arrays.asList(l, h));
                l++;
                h--;
                while (l < h && nums[l] == nums[l - 1]) {
                    l++;
                }
            }
        }
        return res;
    }

    public static void main(String[] args) {
        int nums[] = {2, 7, 11, 15};
        int target = 9;
        List<List<Integer>> ans = twoSum(nums, target);
        System.out.println(ans);
    }
}
```

2)

```java
package arrays_01;
import java.util.*;
public class TargetIndices {
          static int removeElement(int[] nums, int val) {
              int n = nums.length;
              int left = 0;
              for (int i = 0; i < n; i++) {
                  if (nums[i] != val) {
                      nums[left] = nums[i];
                      left++;
                  }
              }

              return left;
```

```
        }

        public static void main(String[] args) {
            int[] nums = {3, 2, 2, 3};
            int val = 3;
            int newSize = removeElement(nums, val);
            System.out.println(newSize);
            System.out.println(Arrays.toString(nums));
        }
    }
```

3)

```
package arrays_01;

public class TargetIndex {
    public static int binarySearch(int[] arr, int target) {
        int left = 0;
        int right = arr.length - 1;
        int c = 0;
        while (left <= right || left>right) {
            int mid = left + (right - left) / 2;

            if (arr[mid] == target) {
             c = 1;
                return mid;
            } else if (arr[mid] < target) {
                left = mid + 1;
            } else {
                right = mid - 1;
            }
            if(left>right)
            {
             c = mid+1;
             break;
            }
        }
        return c;
    }

    public static void main(String[] args) {
        int[] arr = {1, 3, 5, 7, 9};
        int target = 4;
        int index = binarySearch(arr, target);

        if (index != 0) {
            System.out.println("Element found at index " + index);
        } else {
            System.out.println("Element not found in the array");
        }
    }
}
```

4)

```java
package arrays_01;
import java.util.*;

public class AddedArray {
	static int[] doAdd(int[] nums)
	{
		try
		{
		if(nums[nums.length - 1] != 9)
		{
			nums[nums.length - 1] += 1;
		}
		else
		{
			int i = nums.length - 1;
			while(nums[i]>=9)
			{
				nums[i]=0;
				nums[i-1]+=1;
				if(nums[i-1]<=9)
				{
					break;
				}
				int a = nums[i]-10;
				if(nums[i]>=10)
				{
					nums[i-1]+=a;
				}
				i--;
			}
		}
		}
		catch(ArrayIndexOutOfBoundsException e)
		{
			int one[] = {1};
			int[] merged = new int[one.length + nums.length];
		  System.arraycopy(one, 0, merged, 0, one.length);
		  System.arraycopy(nums, 0, merged, one.length, nums.length);
		  return merged;
		}
		return nums;
	}
	public static void main(String[] args) {
		int nums[] = {9,9,9,9};
		int[] ans = doAdd(nums);
		System.out.println(Arrays.toString(ans));

	}

}
```

5)

```java
package arrays_01;

import java.util.Arrays;
```

```java
public class MergeSortedArray {
    public static int[] merge(int[] nums1, int m, int[] nums2, int n) {
        int[] nums1_c = new int[m];
        for(int i=0;i<m;i++)
        {
            nums1_c[i] = nums1[i];
        }
        int p1=0,p2=0;
        for(int k=0;k<nums1.length;k++)
        {
            if(p2>=n || (p1<m && nums1_c[p1]<nums2[p2]))
            {
                nums1[k] = nums1_c[p1++];
            }
            else
            {
                nums1[k] = nums2[p2++];
            }
        }
        return nums1;
    }
    public static void main(String[] args)
    {
       int nums1[] = {1,2,3,0,0,0};
       int m = 3;
       int nums2[] = {2,5,6};
       int n = 3;
       int[] ans = merge(nums1,m,nums2,n);
       System.out.println(Arrays.toString(ans));

    }
}
```

6)

```java
class Solution {
    public int findDuplicate(int[] nums) {
        Set<Integer> seen = new HashSet<Integer>();
        for(int num: nums)
        {
            if(seen.contains(num))
            {
                return num;
            }
            seen.add(num);
        }
        return -1;
    }
}
```

7)

```java
package arrays_01;
import java.util.*;
public class removeZero {
    public static void moveZeroes(int[] nums) {
        int n = nums.length;
        int left = 0;
        for (int i = 0; i < n; i++) {
            if (nums[i] != 0) {
                nums[left] = nums[i];
                left++;
            }
        }
        for (int i = left; i < n; i++) {
            nums[i] = 0;
        }
    }

    public static void main(String[] args) {
        int[] nums = {0, 1, 0, 3, 12};
        moveZeroes(nums);
        System.out.println("Modified Array: " + Arrays.toString(nums));
    }
}
```

8)

```java
package arrays_01;
import java.util.*;
public class missingDuplicate {
    public int[] findDuplicate(int[] nums) {
        Set<Integer> seen = new HashSet<Integer>();
        int d = 0,m=0;
        for(int num: nums)
        {
            if(seen.contains(num))
            {
                d =num;
                break;
            }
            seen.add(num);
        }
        for(int i=0;i<nums.length+1;i++)
        {
            if(!seen.contains(i))
            {
                m = i;
            }
        }
        int arr[] = {d,m};
        return arr;
    }
}
```