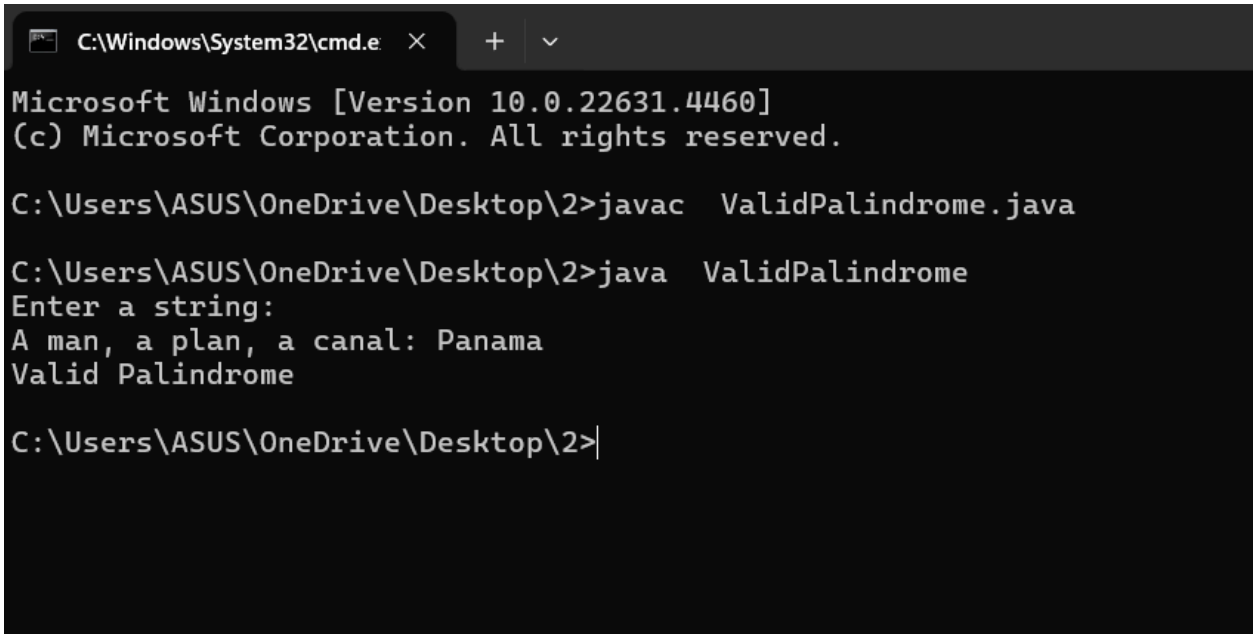
1)valid palindrome

```
import java.util.Scanner;

public class ValidPalindrome {
    public static boolean isPalindrome(String s) {
        int left = 0, right = s.length() - 1;
        while (left < right) {
            while (left < right && !Character.isLetterOrDigit(s.charAt(left)))
left++;
            while (left < right && !Character.isLetterOrDigit(s.charAt(right)))
right--;
            if (Character.toLowerCase(s.charAt(left)) !=
Character.toLowerCase(s.charAt(right))) return false;
            left++;
            right--;
        }
        return true;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a string:");
        String input = scanner.nextLine();
        System.out.println(isPalindrome(input) ? "Valid Palindrome" : "Not a
Palindrome");
    }
}
```

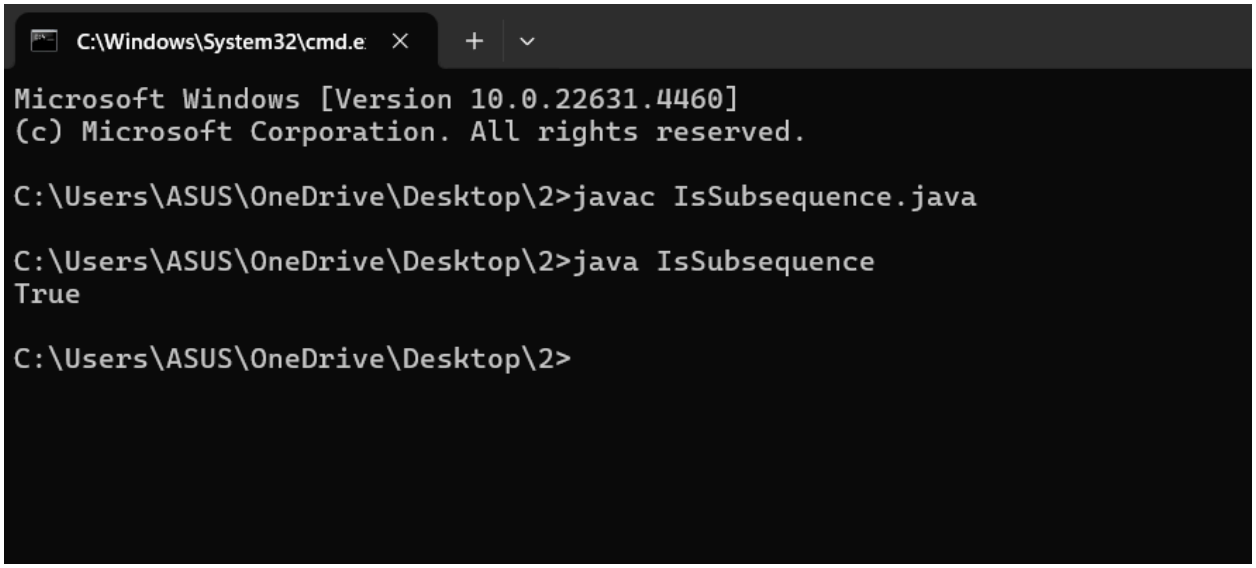


Time:O(n)

2)Is Subsequence

```
public class IsSubsequence {
    public static boolean isSubsequence(String s, String t) {
        int i = 0, j = 0;
        while (i < s.length() && j < t.length()) {
            if (s.charAt(i) == t.charAt(j)) i++;
            j++;
        }
        return i == s.length();
    }

    public static void main(String[] args) {
        String s = "abc";
        String t = "ahbgdc";
        System.out.println(isSubsequence(s, t) ? "True" : "False");
    }
}
```

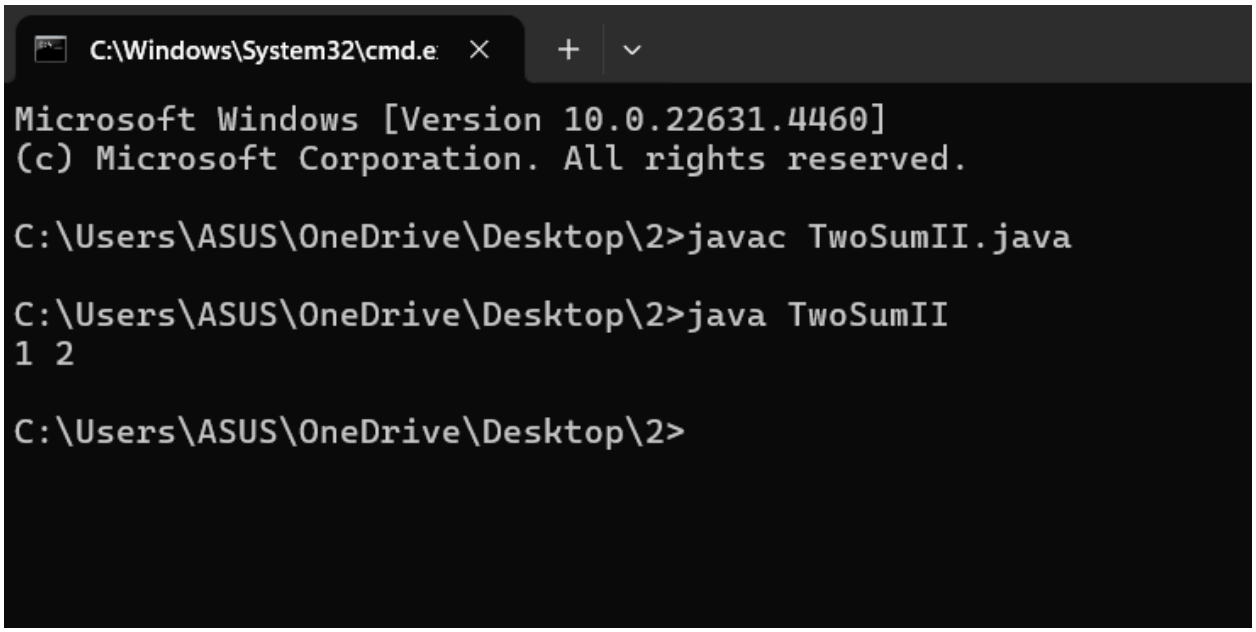


Time:O(n+m)

3)Two sum II-Input Array

```
public class TwoSumII {
    public static int[] twoSum(int[] numbers, int target) {
        int left = 0, right = numbers.length - 1;
        while (left < right) {
            int sum = numbers[left] + numbers[right];
            if (sum == target) return new int[]{left + 1, right + 1};
            else if (sum < target) left++;
            else right--;
        }
        return new int[]{-1, -1};
    }

    public static void main(String[] args) {
        int[] numbers = {2, 7, 11, 15};
        int target = 9;
        int[] result = twoSum(numbers, target);
        System.out.println(result[0] + " " + result[1]);
    }
}
```

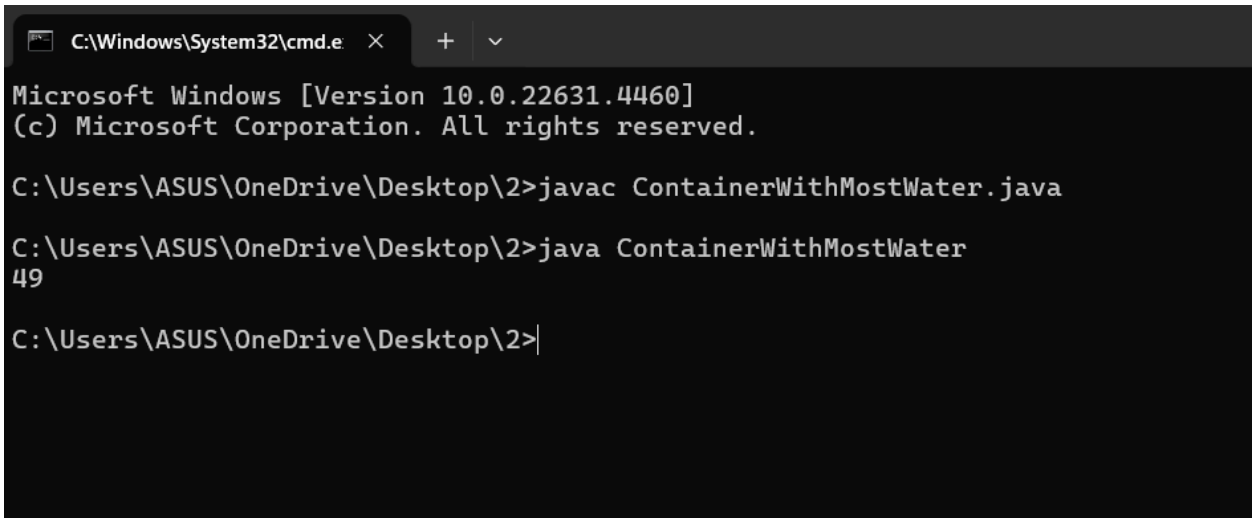


Time:O(n)

4)Container with most Water

```
public class ContainerWithMostWater {
    public static int maxArea(int[] height) {
        int left = 0, right = height.length - 1, max = 0;
        while (left < right) {
            int area = Math.min(height[left], height[right]) * (right - left);
            max = Math.max(max, area);
            if (height[left] < height[right]) left++;
            else right--;
        }
        return max;
    }

    public static void main(String[] args) {
        int[] height = {1, 8, 6, 2, 5, 4, 8, 3, 7};
        System.out.println(maxArea(height));
    }
}
```



Time:O(n)

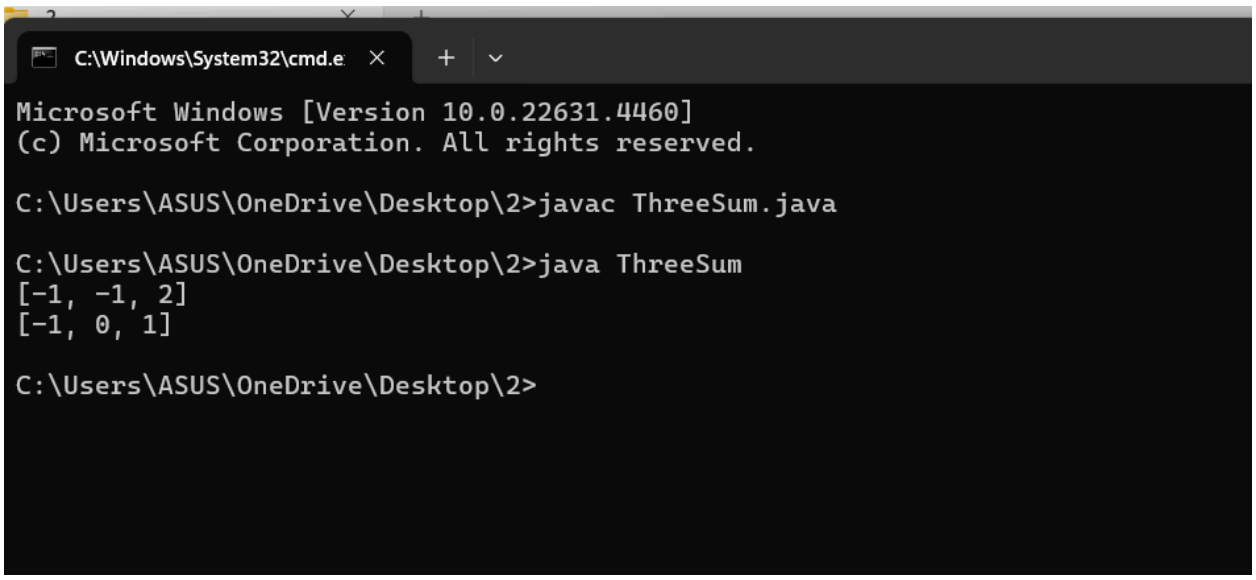
5)3sum

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

public class ThreeSum {
    public static List<List<Integer>> threeSum(int[] nums) {
        List<List<Integer>> result = new ArrayList<>();
        Arrays.sort(nums);
        for (int i = 0; i < nums.length - 2; i++) {
            if (i > 0 && nums[i] == nums[i - 1]) continue;
            int left = i + 1, right = nums.length - 1;
            while (left < right) {
                int sum = nums[i] + nums[left] + nums[right];
                if (sum == 0) {
                    result.add(Arrays.asList(nums[i], nums[left], nums[right]));
                    while (left < right && nums[left] == nums[left + 1]) left++;
                    while (left < right && nums[right] == nums[right - 1]) right--;
                    left++;
                    right--;
```

```
            } else if (sum < 0) left++;
            else right--;
        }
    }
    return result;
  }

  public static void main(String[] args) {
    int[] nums = {-1, 0, 1, 2, -1, -4};
    List<List<Integer>> result = threeSum(nums);
    for (List<Integer> triplet : result) {
      System.out.println(triplet);
    }
  }
}
```



Time:O(n^2)