1. Maximum Subarray Sum – Kadane's Algorithm:

```java
import java.util.Scanner;

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

        int length;
        System.out.print("Enter the length: ");
        length = scanner.nextInt();

        int[] nums = new int[length];
        System.out.print("Enter the array: ");
        for (int i = 0; i < length; i++) {
            nums[i] = scanner.nextInt();
        }

        int currSum = nums[0];
        int maxSum = nums[0];

        for (int i = 1; i < length; i++) {
            currSum = Math.max(currSum + nums[i], nums[i]);
            maxSum = Math.max(maxSum, currSum);
        }

        System.out.printf("Maximum subarray sum: %d\n", maxSum);

        scanner.close();
    }
}

```

Output

```
PS D:\Professional\Programming\Java> cd
ubarray_sum }
Enter the length: 7
Enter the array: 2 3 -8 7 -1 2 3
Maximum subarray sum: 11
PS D:\Professional\Programming\Java>
```

2. Maximum Product Subarray:

```java
import java.util.Scanner;

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

        int length;
        System.out.print("Enter the length: ");
        length = scanner.nextInt();

        int[] nums = new int[length];
        System.out.print("Enter the array: ");
        for (int i = 0; i < length; i++) {
            nums[i] = scanner.nextInt();
        }

        int maxProduct = nums[0];
        int minProduct = nums[0];
        int result = nums[0];

        for (int i = 1; i < nums.length; i++) {
            if (nums[i] == 0) {
                maxProduct = 1;
                minProduct = 1;
                continue;
            }
            if (nums[i] < 0) {
                int temp = maxProduct;
                maxProduct = minProduct;
                minProduct = temp;
            }
            maxProduct = Math.max(nums[i], maxProduct * nums[i]);
            minProduct = Math.min(nums[i], minProduct * nums[i]);
            result = Math.max(result, maxProduct);
        }

        System.out.printf("Maximum subarray product: %d\n", result);

        scanner.close();
    }
}

```

Output:

```
PS D:\Professional\Programming\Java>
ax_subarray_product }
Enter the length: 6
Enter the array: -2 6 -3 -10 0 2
Maximum subarray product: 180
```

3. Search in a rotated sorted array:

```java
import java.util.Scanner;

public class search_rotated_sorted_array {
    private static int binarySearch(int[] nums, int left, int right, int target) {
        if (left > right) {
            return -1;
        }
        int mid = (int) (left + right) / 2;
        if (nums[mid] == target) {
            return mid;
        } else {
            int leftSide = binarySearch(nums, left, mid - 1, target);
            int rightSide = binarySearch(nums, mid + 1, right, target);
            if (leftSide != -1) {
                return leftSide;
            } else {
                return rightSide;
            }
        }
    }

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

        int length;
        System.out.print("Enter the length: ");
        length = scanner.nextInt();

        int[] nums = new int[length];
        System.out.print("Enter the array: ");
        for (int i = 0; i < length; i++) {
            nums[i] = scanner.nextInt();
        }

        int target;
        System.out.print("Enter the target: ");
        target = scanner.nextInt();

        int index = binarySearch(nums, 0, length - 1, target);

        if (index == -1) {
            System.out.println("Target not found: -1");
        } else {
            System.out.printf("Target found at index: %d\n", index);
        }

        scanner.close();
    }
}

```

Output

```
PS D:\Professional\Programming\Java> cd
rch_rotated_sorted_array.java } ; if ($?
Enter the length: 7
Enter the array: 4 5 6 7 0 1 2
Enter the target: 0
Target found at index: 4
PS D:\Professional\Programming\Java>
```

4. Container with Most Water:

```java
import java.util.Scanner;

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

        int length;
        System.out.print("Enter the length: ");
        length = scanner.nextInt();

        int[] nums = new int[length];
        System.out.print("Enter the array: ");
        for (int i = 0; i < length; i++) {
            nums[i] = scanner.nextInt();
        }

        int left = 0, right = length - 1, result = Integer.MIN_VALUE;

        while (left < right) {
            int width = right - left;
            int height = Math.min(nums[left], nums[right]);
            result = Math.max(result, width * height);
            if (nums[left] <= nums[right]) {
                left += 1;
            } else {
                right -= 1;
            }
        }

        System.out.printf("Area of the container with most water: %d\n", result);

        scanner.close();
    }
}

```

Output

```
PS D:\Professional\Programming\Java> cd "d:
ontainer_most_water }
Enter the length: 5
Enter the array: 3 1 2 4 5
Area of the container with most water: 12
PS D:\Professional\Programming\Java>
```

5. Factorial of a large number:

```java
import java.math.BigInteger;
import java.util.Scanner;

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

        long input;
        System.out.print("Enter the number: ");
        input = scanner.nextLong();

        BigInteger result = BigInteger.ONE;

        for (long i = 2; i <= input; i++) {
            BigInteger bigint = new BigInteger(Long.toString(i));
            result = result.multiply(bigint);
        }

        System.out.printf("Factorial of %d is %s", input, result);

        scanner.close();
    }
}

```

Output

```
0c7\bin' 'factorial_large_number'
Enter the number: 100
Factorial of 100 is 93326215443944152681699238856266700490715968264381621468592963895217599993229915608941463976156518286253697920827223758251185210916864000000000000000000000000
PS D:\Professional\Programming\Java>
```

6. Trapping Rain Water:

```java
import java.util.Scanner;

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

        int length;
        System.out.print("Enter the length: ");
        length = scanner.nextInt();

        int[] array = new int[length];
        System.out.print("Enter the heights of the bars: ");
        for (int i = 0; i < length; i++) {
            array[i] = scanner.nextInt();
        }

        int leftMax = array[0];
        int rightMax = array[length - 1];
        int lpointer = 1;
        int rpointer = length - 2;
        int water = 0;

        while (lpointer <= rpointer) {
            if (leftMax <= rightMax) {
                water += Math.max(leftMax - array[lpointer], 0);
                leftMax = Math.max(leftMax, array[lpointer]);
                lpointer += 1;
            } else {
                water += Math.max(rightMax - array[rpointer], 0);
                rightMax = Math.max(rightMax, array[rpointer]);
                rpointer -= 1;
            }
        }

        System.out.printf("Amount of water stored: %d", water);

        scanner.close();
    }
}

```

Output

```
0c7\bin' 'trapping_rainwater'
Enter the length: 5
Enter the heights of the bars: 3 0 2 0 4
Amount of water stored: 7
PS D:\Professional\Programming\Java>
```

7. Chocolate Distribution Problem:

```java
import java.util.Arrays;
import java.util.Scanner;

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

        int length;
        System.out.print("Enter the length: ");
        length = scanner.nextInt();

        int[] chocolates = new int[length];
        System.out.print("Enter the packets: ");
        for (int i = 0; i < length; i++) {
            chocolates[i] = scanner.nextInt();
        }

        int students;
        System.out.print("Enter the number of students: ");
        students = scanner.nextInt();

        Arrays.sort(chocolates);
        int left = 0, right = students - 1;
        int diff = Integer.MAX_VALUE;

        if (length < students) {
            System.out.println("Not enough chocolates");
        } else {
            while (right < length) {
                diff = Math.min(chocolates[right] - chocolates[left], diff);
                right++;
                left++;
            }
        }

        System.out.printf("Minimized difference: %d", diff);

        scanner.close();
    }
}

```

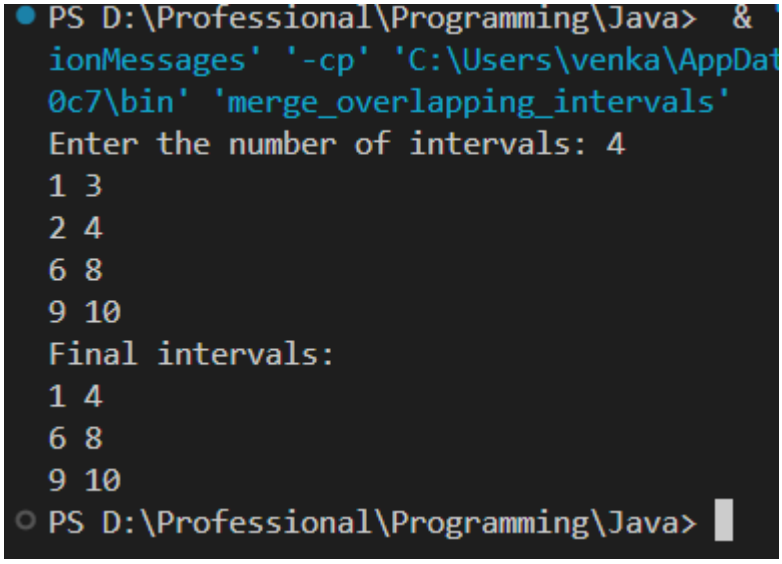
Output

```
PS D:\Professional\Programming\Java>  &
ionMessages' '-cp' 'C:\Users\venka\AppDa
0c7\bin' 'chocolate_distribution'
Enter the length: 7
Enter the packets: 7 3 2 4 9 12 56
Enter the number of students: 3
Minimized difference: 2
PS D:\Professional\Programming\Java>
```

8. Merge Overlapping Intervals:

```java
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.Scanner;
import java.util.Stack;

class Interval {
    int start;
    int end;

    Interval(int start_, int end_) {
        start = start_;
        end = end_;
    }
}

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

        int length;
        System.out.print("Enter the number of intervals: ");
        length = scanner.nextInt();

        List<Interval> intervals = new ArrayList<>();
        for (int i = 0; i < length; i++) {
            int start = scanner.nextInt();
            int end = scanner.nextInt();
            intervals.add(new Interval(start, end));
        }

        Collections.sort(intervals, (a, b) -> {
            if (a.start != b.start) {
                return a.start - b.start;
            } else {
                return a.end - b.end;
            }
        });

        Stack<Interval> stack = new Stack<>();
        for (Interval interval : intervals) {
            if (stack.empty()) {
                stack.push(interval);
            } else {
                Interval last = stack.pop();
                if (interval.start < last.end) {
                    int newStart = Math.min(last.start, interval.start);
                    int newEnd = Math.max(last.end, interval.end);
                    stack.push(new Interval(newStart, newEnd));
                } else {
                    stack.push(last);
                    stack.push(interval);
                }
            }
        }

        System.out.println("Final intervals: ");
        List<Interval> list = new ArrayList<>(stack);
        for (Interval interval : list) {
            System.out.println(interval.start + " " + interval.end);
        }

        scanner.close();
    }
}

```

Output


PS D:\Professional\Programming\Java> & '
ionMessages' '-cp' 'C:\Users\venka\AppDat
0c7\bin' 'merge_overlapping_intervals'
Enter the number of intervals: 4
1 3
2 4
6 8
9 10
Final intervals:
1 4
6 8
9 10
PS D:\Professional\Programming\Java>

9. A Boolean Matrix Question:

```java
import java.util.HashSet;
import java.util.Scanner;
import java.util.Set;

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

        int rows, cols;
        System.out.print("Enter the number of rows: ");
        rows = scanner.nextInt();
        System.out.print("Enter the number of columns: ");
        cols = scanner.nextInt();

        int[][] matrix = new int[rows][cols];
        System.out.println("Enter the matrix: ");
        for (int i = 0; i < rows; i++) {
            System.out.printf("Row %d: ", i + 1);
            for (int j = 0; j < cols; j++) {
                matrix[i][j] = scanner.nextInt();
            }
        }

        Set<Integer> changeRows = new HashSet<>();
        Set<Integer> changeCols = new HashSet<>();

        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                if (matrix[i][j] == 1) {
                    changeRows.add(i);
                    changeCols.add(j);
                }
            }
        }

        for (int row : changeRows) {
            for (int i = 0; i < cols; i++) {
                matrix[row][i] = 1;
            }
        }

        for (int col : changeCols) {
            for (int i = 0; i < rows; i++) {
                matrix[i][col] = 1;
            }
        }

        System.out.println("\nResult matrix:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                System.out.print(matrix[i][j] + " ");
            }
            System.out.println();
        }

        scanner.close();
    }
}

```

Output

```
PS D:\Professional\Programming\Java>  & 'C:\P
ionMessages' '-cp' 'C:\Users\venka\AppData\Ro
0c7\bin' 'boolean_matrix_question'
Enter the number of rows: 3
Enter the number of columns: 6
Enter the matrix:
Row 1: 1 0 0 0 0 0
Row 2: 0 0 0 0 0 0
Row 3: 0 0 0 0 1 0

Result matrix:
1 1 1 1 1 1
1 0 0 0 1 0
1 1 1 1 1 1
PS D:\Professional\Programming\Java>
```

10. Spiral Matrix:

```java
/*
 *
 * Test Cases:
 * 1. Row matrix
 * 2. Column matrix
 * 3. m = 2 or n = 2 matrix
 * 4. m == n > 2 square matrix
 * 5. m > n rectangle matrix
 * 6. m < n rectangle matrix
 *
 */

import java.util.ArrayList;
import java.util.List;

public class spiral_matrix {

    class Solution {
        public List<Integer> spiralOrder(int[][] matrix) {
            int rows = matrix.length;
            int columns = matrix[0].length;
            int rowStart = 0;
            int rowEnd = matrix.length - 1;
            int colStart = 0;
            int colEnd = matrix[0].length - 1;
            int rowLimit = (int) rows / 2;
            int colLimit = (int) columns / 2;
            int row = rowStart;
            int col = colStart;
            boolean reverse = false;
            List<Integer> elements = new ArrayList<>();

            if (rows == 1) {
                for (int i = 0; i < columns; i++) {
                    elements.add(matrix[0][i]);
                }
                return elements;
            } else if (columns == 1) {
                for (int i = 0; i < rows; i++) {
                    elements.add(matrix[i][0]);
                }
                return elements;
            } else if (rows == 2 || columns == 2) {
                colLimit = 0;
                rowLimit = 1;
            } else if (columns == rows && rows % 2 == 0) {
                rows -= 1;
                columns -= 1;
                colLimit -= 1;
```

```
        } else if (rows > columns) {
            if (columns % 2 == 0) {
                colLimit -= 1;
            } else if (rows - columns > 1) {
                rowLimit += 1;
            }
        } else if (rows < columns) {
            int rowReduce = (int) rows / 2;
            int colReduce = (int) columns / 2;
            boolean parity1 = rows % 2 == 0;
            boolean parity2 = columns % 2 == 0;
            int diff = Math.abs(colReduce - rowReduce);
            if (!parity1 && !parity2) {
                colLimit += diff;
            } else if (!parity1 && parity2) {
                colLimit += diff;
                colLimit -= 1;
            } else {
                colLimit -= diff;
                colLimit -= 1;
            }
        }

        while (rowStart <= rowEnd && colStart <= colEnd) {
            elements.add(matrix[row][col]);
            if (row == rowLimit && col == colLimit)
                break;
            if (!reverse) {
                if (col < colEnd) {
                    col += 1;
                } else if (row < rowEnd) {
                    row += 1;
                } else {
                    reverse = true;
                    col -= 1;
                }
            } else {
                if (col > colStart) {
                    col -= 1;
                } else if (row > rowStart + 1) {
                    row -= 1;
                } else {
                    reverse = false;
                    rowStart += 1;
                    colStart += 1;
                    rowEnd -= 1;
                    colEnd -= 1;
                    row = rowStart;
                    col = colStart;
                }
            }
        }
```

```
            return elements;
        }
    }

    public static void main(String[] args) {
        spiral_matrix outer = new spiral_matrix();
        Solution solution = outer.new Solution();

        int[][] matrix = {
                { 1, 2, 3, 4 },
                { 5, 6, 7, 8 },
                { 9, 10, 11, 12 },
                { 13, 14, 15, 16 }
        };

        List<Integer> result = solution.spiralOrder(matrix);
        System.out.println("Spiral Order: " + result);
    }

}
```

Output

```
PS D:\Professional\Programming\Java>  & 'C:\Program Files\Eclipse Adopti
ionMessages' '-cp' 'C:\Users\venka\AppData\Roaming\Code\User\workspaceSt
0c7\bin' 'spiral_matrix'
Spiral Order: [1, 2, 3, 4, 8, 12, 16, 15, 14, 13, 9, 5, 6, 7, 11, 10]
PS D:\Professional\Programming\Java>
```

11. Valid Parentheses:

```java
import java.util.Scanner;
import java.util.Stack;

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

        System.out.print("Enter the parantheses expression: ");
        String input = scanner.next();

        Stack<Character> stack = new Stack<>();

        for (char letter : input.toCharArray()) {
            if (stack.empty()) {
                stack.push(letter);
                continue;
            }
            char top = stack.peek();
            if (top == '(' && letter == ')') {
                stack.pop();
            } else {
                stack.push(letter);
            }
        }

        if (stack.empty()) {
            System.out.println("Answer: Balanced");
        } else {
            System.out.println("Answer: Not balanced");
        }

        scanner.close();
    }
}

```

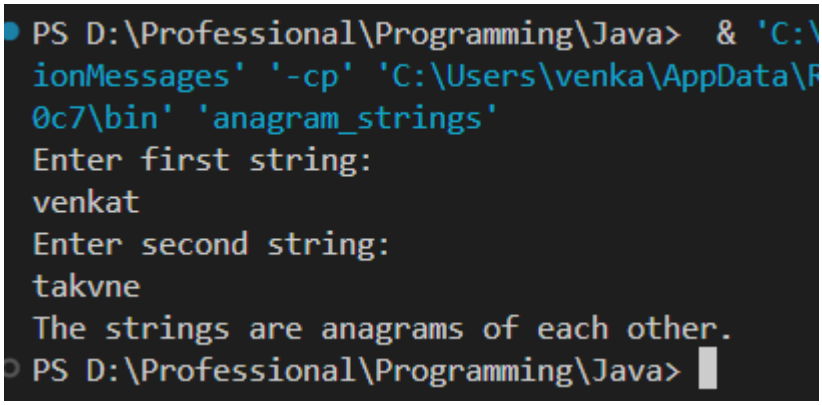
Output

```
PS D:\Professional\Programming\Java>  & 'C:\Progr
ionMessages' '-cp' 'C:\Users\venka\AppData\Roamir
0c7\bin' 'valid_parantheses'
Enter the parantheses expression: ((()))()
Answer: Balanced
PS D:\Professional\Programming\Java>  & 'C:\Progr
ionMessages' '-cp' 'C:\Users\venka\AppData\Roamir
0c7\bin' 'valid_parantheses'
Enter the parantheses expression: ()(()())())
Answer: Not balanced
PS D:\Professional\Programming\Java>
```

12. Check Whether Anagrams:

```java
import java.util.HashMap;
import java.util.Scanner;

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

        System.out.println("Enter first string: ");
        String str1 = scanner.next();

        System.out.println("Enter second string: ");
        String str2 = scanner.next();

        HashMap<Character, Integer> map1 = new HashMap<>();
        HashMap<Character, Integer> map2 = new HashMap<>();

        for (char letter : str1.toCharArray()) {
            map1.putIfAbsent(letter, 0);
            map1.put(letter, map1.get(letter) + 1);
        }

        for (char letter : str2.toCharArray()) {
            map2.putIfAbsent(letter, 0);
            map2.put(letter, map2.get(letter) + 1);
        }

        boolean result = true;

        for (char key : map1.keySet()) {
            if (!map2.containsKey(key) || map1.get(key) != map2.get(key)) {
                result = false;
                break;
            }
        }

        if (result) {
            System.out.println("The strings are anagrams of each other.");
        } else {
            System.out.println("The strings are not anagrams of each other.");
        }

        scanner.close();
    }
}

```

Output


PS D:\Professional\Programming\Java> & 'C:\
ionMessages' '-cp' 'C:\Users\venka\AppData\R
0c7\bin' 'anagram_strings'
Enter first string:
venkat
Enter second string:
takvne
The strings are anagrams of each other.
PS D:\Professional\Programming\Java>

13. Longest palindromic Substring:

```java
public class longest_palindrome {
    class Solution {
        public String longestPalindrome(String s) {
            if (s == null || s.length() == 0) {
                return "";
            }

            int start = 0;
            int end = 0;

            for (int i = 0; i < s.length(); i++) {
                int odd = expandAroundCenter(s, i, i);
                int even = expandAroundCenter(s, i, i + 1);
                int max_len = Math.max(odd, even);

                if (max_len > end - start) {
                    start = i - (max_len - 1) / 2;
                    end = i + max_len / 2;
                }
            }

            return s.substring(start, end + 1);
        }

        private int expandAroundCenter(String s, int left, int right) {
            while (left >= 0 && right < s.length() && s.charAt(left) == s.charAt(right)) {
                left--;
                right++;
            }
            return right - left - 1;
        }
    }

    public static void main(String[] args) {
        longest_palindrome outer = new longest_palindrome();
        Solution solution = outer.new Solution();

        String s = "babad";
        String result = solution.longestPalindrome(s);
        System.out.println("Longest Palindromic Substring: " + result);
    }

}

```

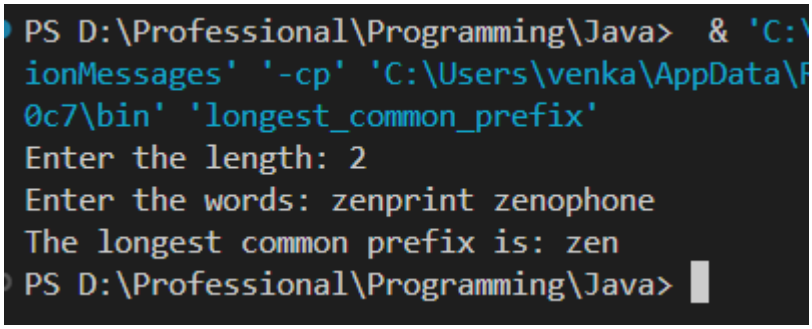
Output

```
PS D:\Professional\Programming\Java>  & 'C:\P
ionMessages' '-cp' 'C:\Users\venka\AppData\Ro
0c7\bin' 'longest_palindrome'
Longest Palindromic Substring: aba
PS D:\Professional\Programming\Java>
```

14. Longest Common Prefix:

```java
import java.util.Arrays;
import java.util.Scanner;

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

        int length;
        System.out.print("Enter the length: ");
        length = scanner.nextInt();

        String[] words = new String[length];
        System.out.print("Enter the words: ");
        for (int i = 0; i < length; i++) {
            words[i] = scanner.next();
        }

        Arrays.sort(words);

        String first = words[0];
        String last = words[length - 1];
        String prefix = new String();

        for (int i = 0; i < Math.min(first.length(), last.length()); i++) {
            if (first.charAt(i) == last.charAt(i)) {
                prefix += first.charAt(i);
            }
        }

        if (prefix.isEmpty()) {
            System.out.println("No common prefix");
        } else {
            System.out.println("The longest common prefix is: " + prefix);
        }

        scanner.close();
    }
}

```

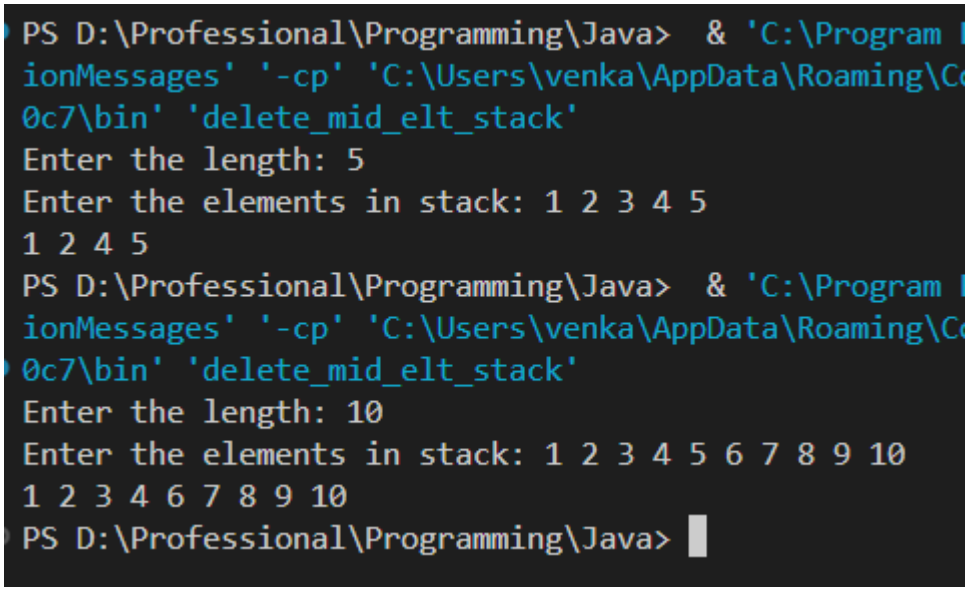
Output


PS D:\Professional\Programming\Java> & 'C:\
ionMessages' '-cp' 'C:\Users\venka\AppData\R
0c7\bin' 'longest_common_prefix'
Enter the length: 2
Enter the words: zenprint zenophone
The longest common prefix is: zen
PS D:\Professional\Programming\Java>

15. Delete the Middle Element of a Stack:

```java
import java.util.Stack;
import java.util.Scanner;

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

        int length;
        System.out.print("Enter the length: ");
        length = scanner.nextInt();

        Stack<Integer> stack1 = new Stack<>();
        System.out.print("Enter the elements in stack: ");
        for (int i = 0; i < length; i++) {
            int elt = scanner.nextInt();
            stack1.push(elt);
        }

        int mid = (int) length / 2;
        int top = -1;

        Stack<Integer> stack2 = new Stack<>();
        while (!stack1.empty()) {
            top += 1;
            int elt = stack1.pop();
            if (top != mid) {
                stack2.push(elt);
            }
        }

        while (!stack2.empty()) {
            System.out.print(stack2.pop() + " ");
        }

        scanner.close();
    }
}

```

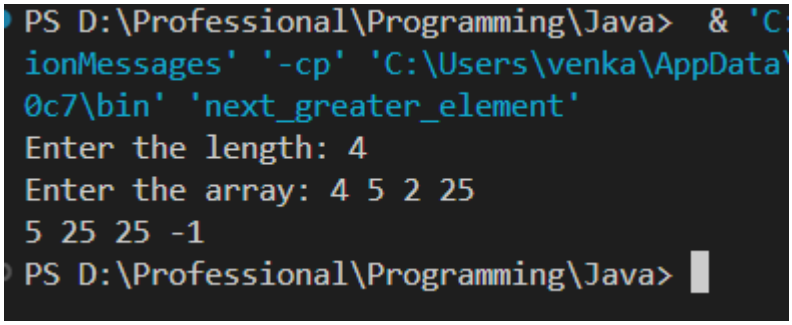
Output


PS D:\Professional\Programming\Java> & 'C:\Program
ionMessages' '-cp' 'C:\Users\venka\AppData\Roaming\C
0c7\bin' 'delete_mid_elt_stack'
Enter the length: 5
Enter the elements in stack: 1 2 3 4 5
1 2 4 5
PS D:\Professional\Programming\Java> & 'C:\Program
ionMessages' '-cp' 'C:\Users\venka\AppData\Roaming\C
0c7\bin' 'delete_mid_elt_stack'
Enter the length: 10
Enter the elements in stack: 1 2 3 4 5 6 7 8 9 10
1 2 3 4 6 7 8 9 10
PS D:\Professional\Programming\Java>

16. Next Greater Element:

```java
import java.util.Scanner;
import java.util.Stack;

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

        int length;
        System.out.print("Enter the length: ");
        length = scanner.nextInt();

        int[] array = new int[length];
        System.out.print("Enter the array: ");
        for (int i = 0; i < length; i++) {
            array[i] = scanner.nextInt();
        }

        Stack<Integer> stack = new Stack<>();
        int[] answer = new int[length];

        for (int i = length - 1; i > -1; i--) {
            while (!stack.empty()) {
                if (stack.peek() > array[i]) {
                    answer[i] = stack.peek();
                    break;
                } else {
                    stack.pop();
                }
            }
            if (stack.empty()) {
                answer[i] = -1;
            }
            stack.push(array[i]);
        }

        for (int elt : answer) {
            System.out.print(elt + " ");
        }

        scanner.close();
    }
}

```

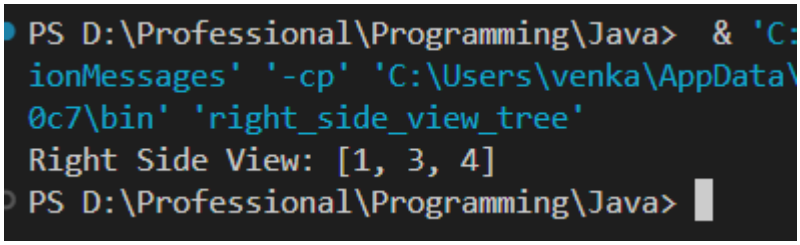
Output


PS D:\Professional\Programming\Java> & 'C:
ionMessages' '-cp' 'C:\Users\venka\AppData\
0c7\bin' 'next_greater_element'
Enter the length: 4
Enter the array: 4 5 2 25
5 25 25 -1
PS D:\Professional\Programming\Java>

17. Right Side View of a Tree:

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

public class right_side_view_tree {
    private List<Integer> list = new ArrayList<>();

    private void traversal(TreeNode root, int index) {
        if (root == null) {
            return;
        }
        if (index < list.size()) {
            list.set(index, root.val);
        } else {
            list.add(root.val);
        }
        traversal(root.left, index + 1);
        traversal(root.right, index + 1);
    }

    public List<Integer> rightSideView(TreeNode root) {
        traversal(root, 0);
        return list;
    }

    public static void main(String[] args) {
        TreeNode root = new TreeNode(1);
        root.left = new TreeNode(2);
        root.right = new TreeNode(3);
        root.left.right = new TreeNode(5);
        root.right.right = new TreeNode(4);

        right_side_view_tree solution = new right_side_view_tree();
        List<Integer> result = solution.rightSideView(root);

        System.out.println("Right Side View: " + result);
    }

}

```

Output


PS D:\Professional\Programming\Java> & 'C:
ionMessages' '-cp' 'C:\Users\venka\AppData\
0c7\bin' 'right_side_view_tree'
Right Side View: [1, 3, 4]
PS D:\Professional\Programming\Java>

18. Maximum height or depth of a tree:

```java
class TreeNode {
    int val;
    TreeNode left;
    TreeNode right;

    TreeNode(int val) {
        this.val = val;
        this.left = null;
        this.right = null;
    }
};

public class max_height_tree {
    private static int traversal(TreeNode root, int level) {
        if (root == null) {
            return level;
        }
        int leftSide = traversal(root.left, level + 1);
        int rightSide = traversal(root.right, level + 1);
        return Math.max(leftSide, rightSide);
    }

    public static int maxDepth(TreeNode root) {
        int height = traversal(root, 0);
        return height;
    }

    public static void main(String[] args) {
        TreeNode root = new TreeNode(1);
        root.left = new TreeNode(2);
        root.right = new TreeNode(3);
        root.left.left = new TreeNode(4);
        root.left.right = new TreeNode(5);
        root.right.right = new TreeNode(6);
        root.left.left.left = new TreeNode(7);

        int result = maxDepth(root);
        System.out.println("Max Depth of Tree: " + result);
    }

}

```

Output

```
PS D:\Professional\Programming\Java>  & 'C
ionMessages' '-cp' 'C:\Users\venka\AppData
0c7\bin' 'max_height_tree'
Max Depth of Tree: 4
PS D:\Professional\Programming\Java>
```