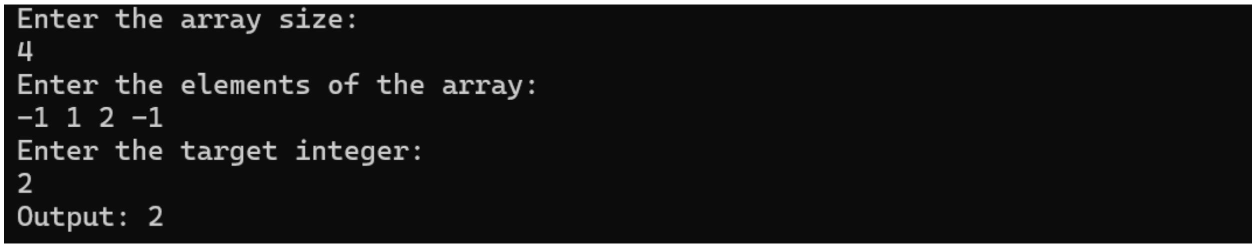
PRACTICE SET – 7 **20-11-2024**

DSA

1.3sum closet

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

public class Problem1 {
  public int threeSumClosest(int[] nums, int target) {
    Arrays.sort(nums);
    int closestSum = Integer.MAX_VALUE / 2;
    for (int i = 0; i < nums.length - 2; i++) {
      int left = i + 1, right = nums.length - 1;
      while (left < right) {
        int currentSum = nums[i] + nums[left] + nums[right];
        if (Math.abs(target - currentSum) < Math.abs(target - closestSum)) {
          closestSum = currentSum;
        }
        if (currentSum < target) {
          left++;
        } else if (currentSum > target) {
          right--;
        } else {
          return currentSum;
        }
      }
    }
    return closestSum;
  }
```

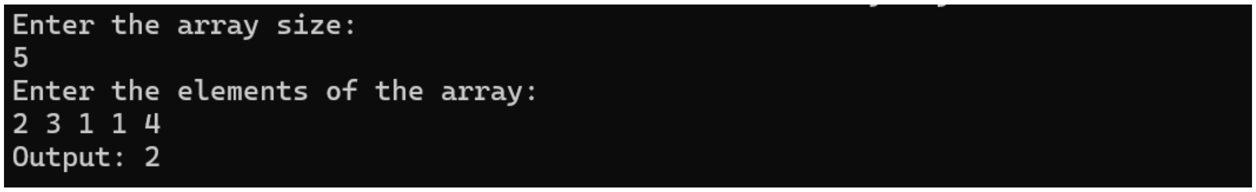
```
public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);

    System.out.println("Enter the array size:");
    int n = scanner.nextInt();

    int[] nums = new int[n];

    System.out.println("Enter the elements of the array:");
    for (int i = 0; i < n; i++) {
        nums[i] = scanner.nextInt();
    }

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

    Problem1 solver = new Problem1();
    int result = solver.threeSumClosest(nums, target);
    System.out.println("Output: " + result);

    scanner.close();
}
```



```
Enter the array size:
4
Enter the elements of the array:
-1 1 2 -1
Enter the target integer:
2
Output: 2
```

2.Jump Game 2

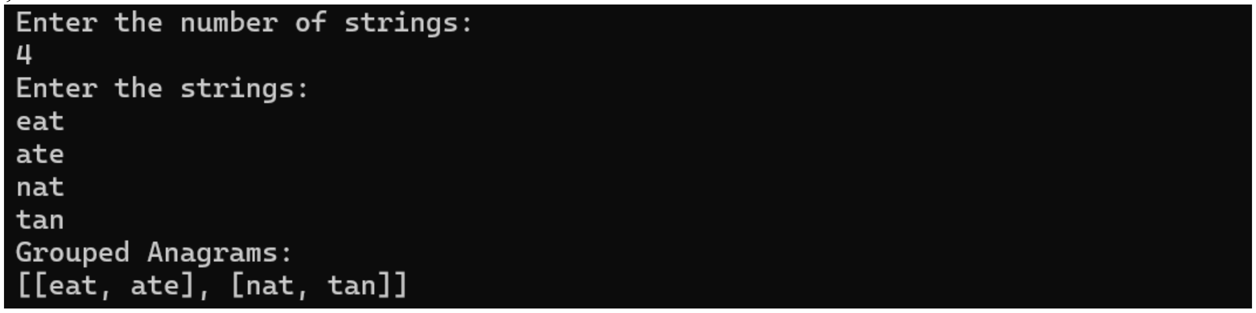
```
import java.util.Scanner;

public class Problem2 {
    public int jump(int[] nums) {
        int jumps = 0;
        int currentEnd = 0;
        int farthest = 0;

        for (int i = 0; i < nums.length - 1; i++) {
            farthest = Math.max(farthest, i + nums[i]);

            if (i == currentEnd) {
                jumps++;
                currentEnd = farthest;
            }
        }

        return jumps;
    }

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

        System.out.println("Enter the array size:");
        int n = scanner.nextInt();

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

        Problem2 solver = new Problem2();
        int result = solver.jump(nums);
        System.out.println("Output: " + result);

        scanner.close();
    }
}
```



```
Enter the array size:
5
Enter the elements of the array:
2 3 1 1 4
Output: 2
```

3.Group Anagrams

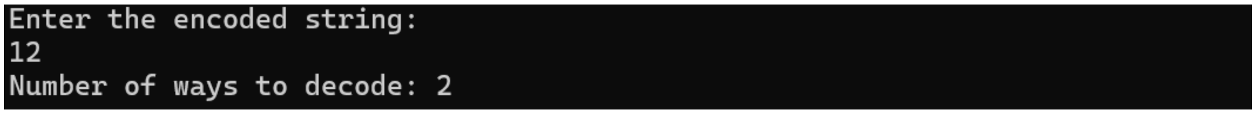
```
import java.util.*;

public class Problem3 {
    public List<List<String>> groupAnagrams(String[] strs) {
        Map<String, List<String>> anagramMap = new HashMap<>();

        for (String str : strs) {
            char[] charArray = str.toCharArray();
            Arrays.sort(charArray);
            String sortedStr = new String(charArray);

            if (!anagramMap.containsKey(sortedStr)) {
                anagramMap.put(sortedStr, new ArrayList<>());
            }
            anagramMap.get(sortedStr).add(str);
        }

        return new ArrayList<>(anagramMap.values());
    }

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

        System.out.println("Enter the number of strings:");
        int n = scanner.nextInt();
        scanner.nextLine();  // consume the newline

        String[] strs = new String[n];

        System.out.println("Enter the strings:");
        for (int i = 0; i < n; i++) {
            strs[i] = scanner.nextLine();
        }

        Problem3 solver = new Problem3();
        List<List<String>> result = solver.groupAnagrams(strs);

        System.out.println("Grouped Anagrams:");
        System.out.println(result);
        scanner.close();
    }
}
```



```
Enter the number of strings:
4
Enter the strings:
eat
ate
nat
tan
Grouped Anagrams:
[[eat, ate], [nat, tan]]
```

4.Decode ways

```
import java.util.Scanner;


public class Problem4 {
    public int numDecodings(String s) {
        if (s == null || s.length() == 0 || s.charAt(0) == '0') {
            return 0;
        }


        int n = s.length();
        int[] dp = new int[n + 1];
        dp[0] = 1;
        dp[1] = 1;


        for (int i = 2; i <= n; i++) {
            if (s.charAt(i - 1) != '0') {
                dp[i] += dp[i - 1];
            }


            int twoDigit = Integer.parseInt(s.substring(i - 2, i));
            if (twoDigit >= 10 && twoDigit <= 26) {
                dp[i] += dp[i - 2];
            }
        }

        return dp[n];
    }

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

        System.out.println("Enter the encoded string:");
        String s = scanner.next();

        Problem4 solver = new Problem4();
        int result = solver.numDecodings(s);

        System.out.println("Number of ways to decode: " + result);

        scanner.close();
    }
}
```

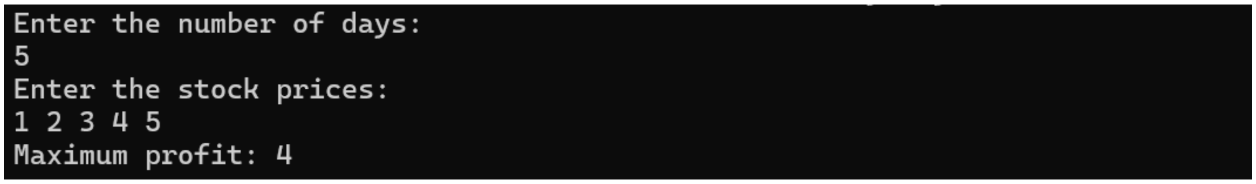


```
Enter the encoded string:
12
Number of ways to decode: 2
```

5.Best time to buy and sell stock 2

```
import java.util.Scanner;

public class Problem5 {
    public int maxProfit(int[] prices) {
```

```
    int profit = 0;

    for (int i = 1; i < prices.length; i++) {
        if (prices[i] > prices[i - 1]) {
            profit += prices[i] - prices[i - 1];
        }
    }

    return profit;
}

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

    System.out.println("Enter the number of days:");
    int n = scanner.nextInt();

    int[] prices = new int[n];

    System.out.println("Enter the stock prices:");
    for (int i = 0; i < n; i++) {
        prices[i] = scanner.nextInt();
    }

    Problem5 solver = new Problem5();
    int result = solver.maxProfit(prices);

    System.out.println("Maximum profit: " + result);

    scanner.close();
    }
}
```

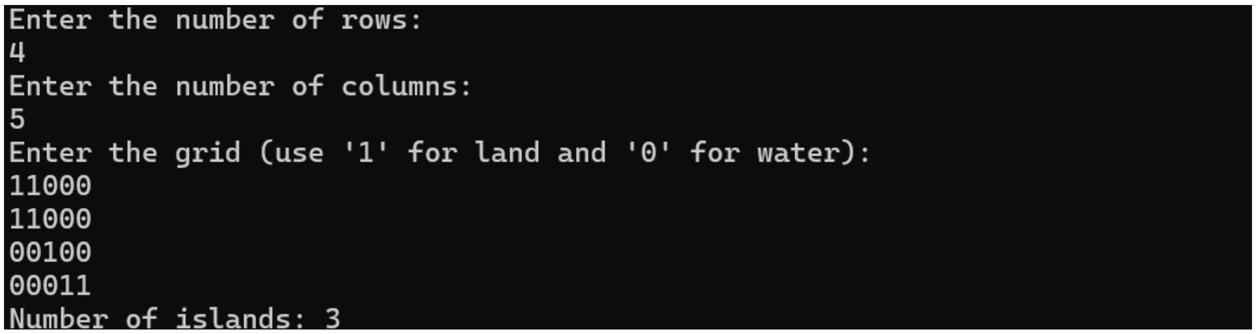


```
Enter the number of days:
5
Enter the stock prices:
1 2 3 4 5
Maximum profit: 4
```

6.Number of Islands

```
import java.util.Scanner;

public class Problem6 {
    public int numIslands(char[][] grid) {
        if (grid == null || grid.length == 0) {
            return 0;
        }

        int m = grid.length;
        int n = grid[0].length;
        int islandCount = 0;

        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                if (grid[i][j] == '1') {
                    islandCount++;
                    dfs(grid, i, j, m, n);
                }
            }
        }
    return islandCount;
}

private void dfs(char[][] grid, int i, int j, int m, int n) {
    if (i < 0 || j < 0 || i >= m || j >= n || grid[i][j] == '0') {
        return;
    }

    grid[i][j] = '0';

    dfs(grid, i + 1, j, m, n);  // down
    dfs(grid, i - 1, j, m, n);  // up
    dfs(grid, i, j + 1, m, n);  // right
    dfs(grid, i, j - 1, m, n);  // left
}

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

    System.out.println("Enter the number of rows:");
    int m = scanner.nextInt();
    System.out.println("Enter the number of columns:");
    int n = scanner.nextInt();

    char[][] grid = new char[m][n];

    System.out.println("Enter the grid (use '1' for land and '0' for water):");
    for (int i = 0; i < m; i++) {
        String row = scanner.next();
```

```
            grid[i] = row.toCharArray();
        }

        Problem6 solver = new Problem6();
        int result = solver.numIslands(grid);

        System.out.println("Number of islands: " + result);

        scanner.close();
    }}
```



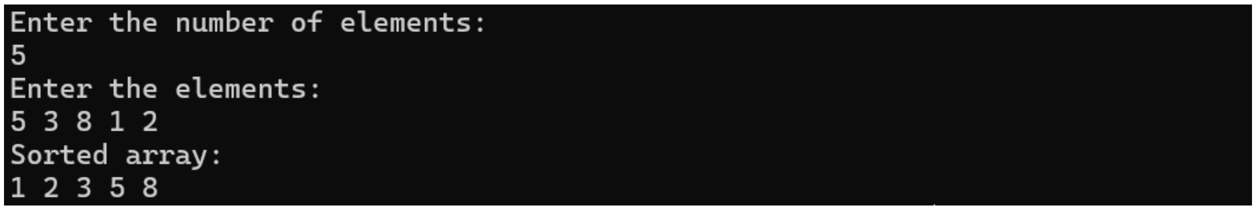
```
Enter the number of rows:
4
Enter the number of columns:
5
Enter the grid (use '1' for land and '0' for water):
11000
11000
00100
00011
Number of islands: 3
```

7.QuickSort

```
import java.util.Scanner;

public class Problem7 {
    public void quickSort(int[] arr, int low, int high) {
        if (low < high) {
            int pi = partition(arr, low, high);
            quickSort(arr, low, pi - 1);
            quickSort(arr, pi + 1, high);
        }
    }
```

```
private int partition(int[] arr, int low, int high) {
    int pivot = arr[high];
    int i = (low - 1);
    for (int j = low; j < high; j++) {
        if (arr[j] < pivot) {
            i++;
            int temp = arr[i];
            arr[i] = arr[j];
            arr[j] = temp;
        }
    }
    int temp = arr[i + 1];
    arr[i + 1] = arr[high];
    arr[high] = temp;
    return i + 1;
}

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

    System.out.println("Enter the number of elements:");
    int n = scanner.nextInt();

    int[] arr = new int[n];

    System.out.println("Enter the elements:");
    for (int i = 0; i < n; i++) {
        arr[i] = scanner.nextInt();
    }
```

```
        Problem7 sorter = new Problem7();
        sorter.quickSort(arr, 0, arr.length - 1);

        System.out.println("Sorted array:");
        for (int num : arr) {
            System.out.print(num + " ");
        }
        scanner.close();
    }
}
```



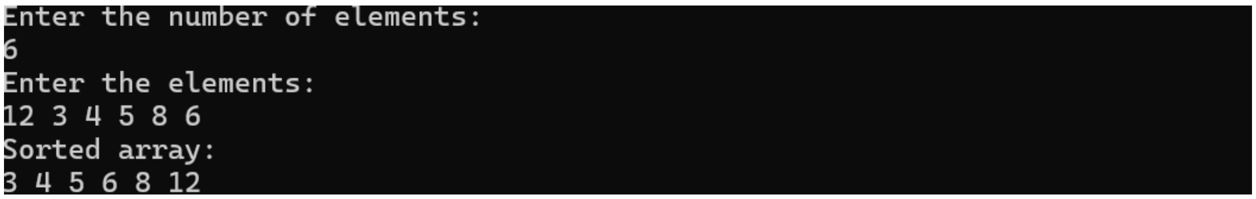
```
Enter the number of elements:
5
Enter the elements:
5 3 8 1 2
Sorted array:
1 2 3 5 8
```

8.Merge Sort

```
import java.util.Scanner;

public class Problem8 {
    public void mergeSort(int[] arr, int left, int right) {
        if (left < right) {
            int mid = left + (right - left) / 2;
            mergeSort(arr, left, mid);
            mergeSort(arr, mid + 1, right);
            merge(arr, left, mid, right);
        }
    }
```

```
private void merge(int[] arr, int left, int mid, int right) {
    int n1 = mid - left + 1;
    int n2 = right - mid;
    int[] leftArr = new int[n1];
    int[] rightArr = new int[n2];

    for (int i = 0; i < n1; ++i) leftArr[i] = arr[left + i];
    for (int i = 0; i < n2; ++i) rightArr[i] = arr[mid + 1 + i];

    int i = 0, j = 0;
    int k = left;
    while (i < n1 && j < n2) {
        if (leftArr[i] <= rightArr[j]) {
            arr[k] = leftArr[i];
            i++;
        } else {
            arr[k] = rightArr[j];
            j++;
        }
        k++;
    }

    while (i < n1) {
        arr[k] = leftArr[i];
        i++;
        k++;
    }

    while (j < n2) {
        arr[k] = rightArr[j];
```

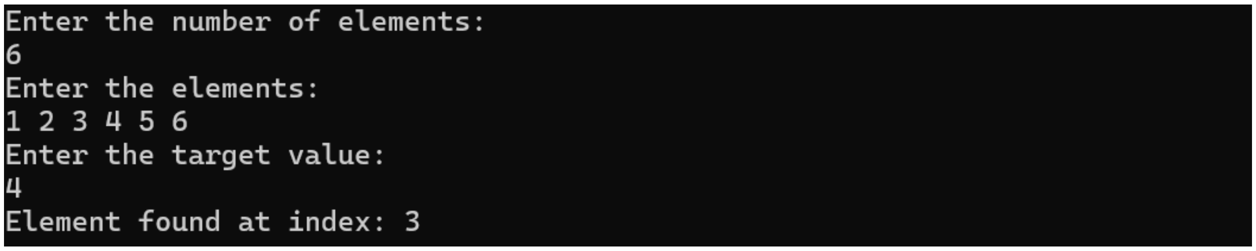
```
            j++;
            k++;
        }
    }

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

        System.out.println("Enter the number of elements:");
        int n = scanner.nextInt();

        int[] arr = new int[n];

        System.out.println("Enter the elements:");
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }

        Problem8 sorter = new Problem8();
        sorter.mergeSort(arr, 0, arr.length - 1);

        System.out.println("Sorted array:");
        for (int num : arr) {
            System.out.print(num + " ");
        }
        scanner.close();
    }
}
```



```
Enter the number of elements:
6
Enter the elements:
12 3 4 5 8 6
Sorted array:
3 4 5 6 8 12
```

9.Ternary Search

```
import java.util.Scanner;

public class Problem9 {
    public int ternarySearch(int[] arr, int left, int right, int target) {
        if (right >= left) {
            int mid1 = left + (right - left) / 3;
            int mid2 = right - (right - left) / 3;

            if (arr[mid1] == target) return mid1;
            if (arr[mid2] == target) return mid2;

            if (target < arr[mid1]) return ternarySearch(arr, left, mid1 - 1, target);
            if (target > arr[mid2]) return ternarySearch(arr, mid2 + 1, right, target);
            return ternarySearch(arr, mid1 + 1, mid2 - 1, target);
        }
        return -1;
    }

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

        System.out.println("Enter the number of elements:");
        int n = scanner.nextInt();
        int[] arr = new int[n];

        System.out.println("Enter the elements:");
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }

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

        Problem9 searcher = new Problem9();
        int result = searcher.ternarySearch(arr, 0, arr.length - 1, target);

        System.out.println("Element found at index: " + result);
        scanner.close();
    }
}
```



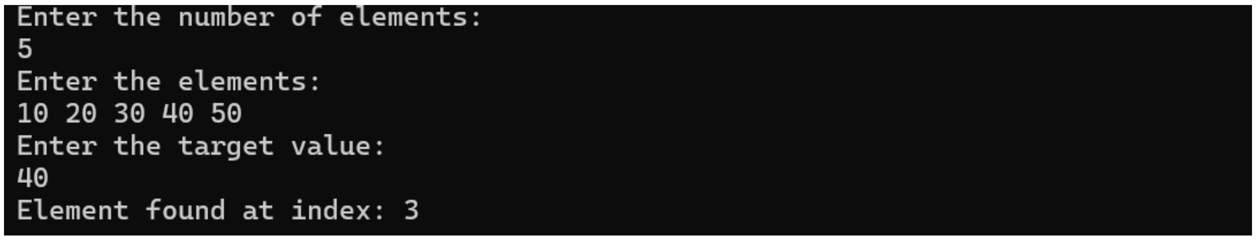
```
Enter the number of elements:
6
Enter the elements:
1 2 3 4 5 6
Enter the target value:
4
Element found at index: 3
```

10.Interpolation Search

```
import java.util.Scanner;

public class Problem10 {
```

```
public int interpolationSearch(int[] arr, int left, int right, int target) {
    if (left <= right && target >= arr[left] && target <= arr[right]) {
        int pos = left + ((right - left) / (arr[right] - arr[left])) * (target - arr[left]);
        if (arr[pos] == target) return pos;
        if (arr[pos] < target) return interpolationSearch(arr, pos + 1, right, target);
        if (arr[pos] > target) return interpolationSearch(arr, left, pos - 1, target);
    }
    return -1;
}

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

    System.out.println("Enter the number of elements:");
    int n = scanner.nextInt();

    int[] arr = new int[n];

    System.out.println("Enter the elements:");
    for (int i = 0; i < n; i++) {
        arr[i] = scanner.nextInt();
    }

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

    Problem10 searcher = new Problem10();
    int result = searcher.interpolationSearch(arr, 0, arr.length - 1, target);

    System.out.println("Element found at index: " + result);
```

```
        scanner.close();
    }
}
```



```
Enter the number of elements:
5
Enter the elements:
10 20 30 40 50
Enter the target value:
40
Element found at index: 3
```