

Name - Anisha Raj
Roll no. - 2410031455
Section - 2CSE8

Java Experiment-2

The screenshot displays the GeeksforGeeks IDE interface for solving the "Chocolate Distribution Problem". The browser address bar shows the URL: [geeksforgeeks.org/problems/chocolate-distribution-problem3825/1](https://www.geeksforgeeks.org/problems/chocolate-distribution-problem3825/1).

The IDE features a top navigation bar with a search bar, a "Get 90% Refund" banner, and links to Courses, Tutorials, Practice, and Jobs. The left sidebar contains tabs for Problem, Editorial, Submissions, and Comments. The main editor area shows a Java code snippet for the "Chocolate Distribution Problem".

Compilation Results:

Compilation Completed

Case 1

Input: `3 4 1 9 56 7 9 12`
`5`

Your Output: `6`

Expected Output: `6`

Java Code:

```
1- import java.util.*;
2-
3- class Solution {
4-
5-     public int findMinDiff(ArrayList<Integer> arr, int m) {
6-
7-         if (m == 0 || arr.size() == 0)
8-             return 0;
9-
10-        if (arr.size() < m)
11-            return -1;
12-
13-        Collections.sort(arr);
14-
15-        int minDiff = Integer.MAX_VALUE;
16-
17-        for (int i = 0; i + m - 1 < arr.size(); i++) {
18-            int diff = arr.get(i + m - 1) - arr.get(i);
19-            minDiff = Math.min(minDiff, diff);
20-        }
21-
22-        return minDiff;
23-    }
24-}
25-
```

At the bottom right, there are buttons for "Custom Input", "Compile & Run", and "Submit".

Name - Anisha Raj
Roll no. - 2410031455
Section - 2CSE8

Java Experiment-2

The screenshot displays the GeeksforGeeks website interface. The browser tab shows the URL `geeksforgeeks.org/problems/smallest-subarray-with-sum-greater-than-x5651/1`. The page title is "Smallest subarray with sum greater than x". The difficulty is marked as "Easy", with an accuracy of 37.07%, 155K+ submissions, 2 points, and an average time of 20m.

Examples:

- Input: `x = 51, arr[] = [1, 4, 45, 6, 0, 19]`
Output: 3
Explanation: Minimum length subarray is [4, 45, 6]
- Input: `x = 100, arr[] = [1, 10, 5, 2, 7]`
Output: 0
Explanation: No subarray exist

Constraints:

- $1 \leq \text{arr.size}, x \leq 10^5$
- $0 \leq \text{arr}[] \leq 10^4$

Expected Complexities

Company Tags

The code editor on the right shows the following Java code:

```
1 class Solution {
2
3     public int smallestSubWithSum(int x, int[] arr) {
4
5         int n = arr.length;
6         int minLength = Integer.MAX_VALUE;
7
8         int start = 0, currSum = 0;
9
10        for (int end = 0; end < n; end++) {
11
12            currSum += arr[end];
13
14            while (currSum > x) {
15                minLength = Math.min(minLength, end - start + 1);
16                currSum -= arr[start];
17                start++;
18            }
19        }
20
21        return (minLength == Integer.MAX_VALUE) ? 0 : minLength;
22    }
23 }
24
```

At the bottom right, there are buttons for "Custom Input", "Compile & Run", and "Submit".

Name - Anisha Raj
Roll no. - 2410031455
Section - 2CSE8

Java Experiment-2

The screenshot displays the GeeksforGeeks website interface. The browser tabs at the top include 'Chocolate Distribution Problem', 'Smallest subarray with sum', and 'Three way partitioning | Practice'. The address bar shows the URL 'geeksforgeeks.org/problems/three-way-partitioning/1'. The website header features a search bar, a 'Get 90% Refund' banner, and navigation links for 'Courses', 'Tutorials', 'Practice', and 'Jobs'. The user profile icon shows a rating of '99+'.

The main content area is titled 'Three way partitioning' with a difficulty level of 'Easy', an accuracy of '41.58%', and '187K+' submissions. The problem description states: 'Given an array and a range a, b. The task is to partition the array around the range such that the array is divided into three parts. 1) All elements smaller than a come first. 2) All elements in range a to b come next. 3) All elements greater than b appear in the end. The individual elements of three sets can appear in any order. You are required to return the modified array. Note: The generated output is true if you modify the given array successfully. Otherwise false. Geeky Challenge: Solve this problem in O(n) time complexity.'

Examples:

- Input: arr[] = [1, 2, 3, 3, 4], a = 1, b = 2
Output: true
Explanation: One possible arrangement is: {1, 2, 3, 3, 4}. If you return a valid arrangement, output will be true.
- Input: arr[] = [1, 4, 3, 6, 2, 1], a = 1, b = 3
Output: true
Explanation: One possible arrangement is: {1, 3, 2, 1, 4, 6}. If you return a valid arrangement, output will be true.

Constraints:

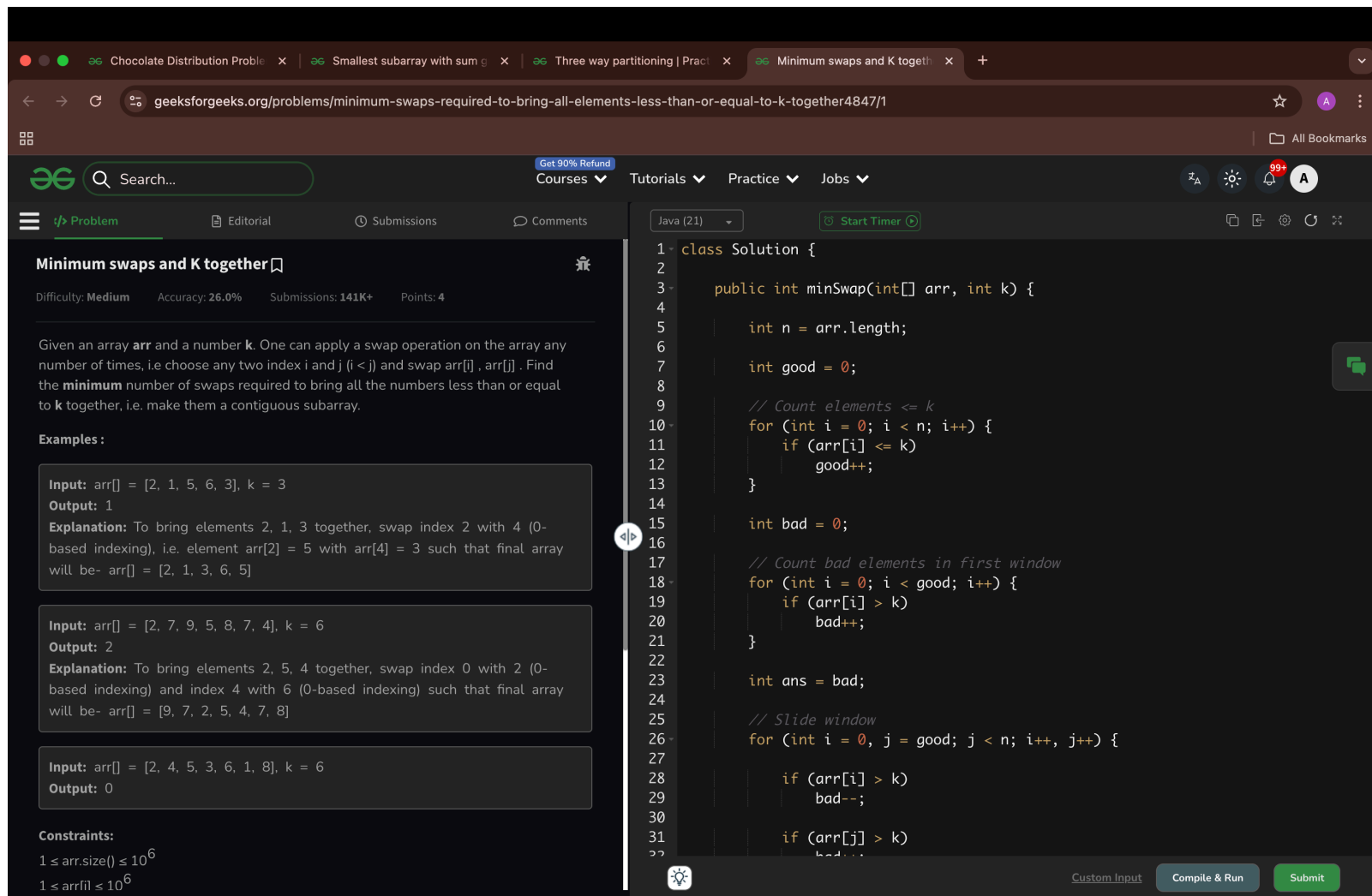
The code editor on the right shows the following Java solution:

```
1 class Solution {
2
3     public void threeWayPartition(int[] arr, int a, int b) {
4
5         int start = 0;
6         int end = arr.length - 1;
7
8         for (int i = 0; i <= end; ) {
9
10            if (arr[i] < a) {
11                swap(arr, i, start);
12                i++;
13                start++;
14            }
15            else if (arr[i] > b) {
16                swap(arr, i, end);
17                end--;
18            }
19            else {
20                i++;
21            }
22        }
23    }
24
25    private void swap(int[] arr, int i, int j) {
26        int temp = arr[i];
27        arr[i] = arr[j];
28        arr[j] = temp;
29    }
30 }
31
```

At the bottom right, there are buttons for 'Custom Input', 'Compile & Run', and 'Submit'.

Name - Anisha Raj
Roll no. - 2410031455
Section - 2CSE8

Java Experiment-2



The screenshot shows the GeeksforGeeks website interface. The browser tabs at the top include "Chocolate Distribution Problem", "Smallest subarray with sum", "Three way partitioning | Pract", and "Minimum swaps and K together". The address bar shows the URL: [geeksforgeeks.org/problems/minimum-swaps-required-to-bring-all-elements-less-than-or-equal-to-k-together4847/1](https://www.geeksforgeeks.org/problems/minimum-swaps-required-to-bring-all-elements-less-than-or-equal-to-k-together4847/1).

The page title is "Minimum swaps and K together". The difficulty is "Medium", accuracy is "26.0%", submissions are "141K+", and points are "4".

Problem Description: Given an array `arr` and a number `k`. One can apply a swap operation on the array any number of times, i.e. choose any two index `i` and `j` ($i < j$) and swap `arr[i]` , `arr[j]` . Find the **minimum** number of swaps required to bring all the numbers less than or equal to `k` together, i.e. make them a contiguous subarray.

Examples :

- Input:** `arr[] = [2, 1, 5, 6, 3], k = 3`
Output: 1
Explanation: To bring elements 2, 1, 3 together, swap index 2 with 4 (0-based indexing), i.e. element `arr[2] = 5` with `arr[4] = 3` such that final array will be- `arr[] = [2, 1, 3, 6, 5]`
- Input:** `arr[] = [2, 7, 9, 5, 8, 7, 4], k = 6`
Output: 2
Explanation: To bring elements 2, 5, 4 together, swap index 0 with 2 (0-based indexing) and index 4 with 6 (0-based indexing) such that final array will be- `arr[] = [9, 7, 2, 5, 4, 7, 8]`
- Input:** `arr[] = [2, 4, 5, 3, 6, 1, 8], k = 6`
Output: 0

Constraints:

- $1 \leq \text{arr.size()} \leq 10^6$
- $1 \leq \text{arr}[i] \leq 10^6$

The code editor on the right shows the following Java solution:

```
1 class Solution {
2
3     public int minSwap(int[] arr, int k) {
4
5         int n = arr.length;
6
7         int good = 0;
8
9         // Count elements <= k
10        for (int i = 0; i < n; i++) {
11            if (arr[i] <= k)
12                good++;
13        }
14
15        int bad = 0;
16
17        // Count bad elements in first window
18        for (int i = 0; i < good; i++) {
19            if (arr[i] > k)
20                bad++;
21        }
22
23        int ans = bad;
24
25        // Slide window
26        for (int i = 0, j = good; j < n; i++, j++) {
27
28            if (arr[i] > k)
29                bad--;
30
31            if (arr[j] > k)
32                bad++;
```

Name - Anisha Raj
Roll no. - 2410031455
Section - 2CSE8

Java Experiment-2

The screenshot displays the GeeksforGeeks website interface. The browser's address bar shows the URL `geeksforgeeks.org/problems/find-the-median0527/1`. The page title is "Median of an Array". The problem description states: "Given an array `arr[]` of integers, calculate the median." It includes three examples with their inputs, outputs, and explanations. The constraints are $1 \leq \text{arr.size()} \leq 10^5$ and $1 \leq \text{arr}[i] \leq 10^5$. A "Try more examples" button is visible. The right side of the page shows a Java code editor with the following code:

```
1 import java.util.Arrays;
2
3 class Solution {
4
5     public double findMedian(int[] arr) {
6
7         Arrays.sort(arr);
8         int n = arr.length;
9
10        if (n % 2 != 0) {
11            return arr[n / 2];
12        }
13
14        return (arr[n / 2] + arr[(n / 2) - 1]) / 2.0;
15    }
16 }
17
```

At the bottom right, there are buttons for "Custom Input", "Compile & Run", and "Submit".

Name - Anisha Raj
Roll no. - 2410031455
Section - 2CSE8

Java Experiment-2

The screenshot displays the GeeksforGeeks website interface. The browser's address bar shows the URL: `geeksforgeeks.org/problems/spirally-traversing-a-matrix-1587115621/1`. The page title is "Spirally traversing a matrix". The problem description states: "You are given a rectangular matrix `mat[][]` of size `n x m`, and your task is to return an array while traversing the matrix in spiral form." The difficulty is "Medium", accuracy is "35.2%", and there are "343K+" submissions for "4" points.

Examples:

Input: `mat[][] = [[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12], [13, 14, 15, 16]]`
Output: `[1, 2, 3, 4, 8, 12, 16, 15, 14, 13, 9, 5, 6, 7, 11, 10]`
Explanation:

Example of matrix in spiral form

Matrix:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Output: 1, 2, 3, 4, 8, 12, 16, 15, 14, 13, 9, 5, 6, 7, 11, 10

Input: `mat[][] = [[1, 2, 3, 4, 5, 6], [7, 8, 9, 10, 11, 12], [13, 14, 15, 16, 17, 18]]`
Output: `[1, 2, 3, 4, 5, 6, 12, 18, 17, 16, 15, 14, 13, 7, 8, 9, 10, 11]`
Explanation: Applying same technique as shown above.

Input: `mat[][] = [[32, 44, 27, 23], [54, 28, 50, 62]]`

The right side of the image shows the Java code for the solution:

```
1 import java.util.*;
2
3 class Solution {
4
5     public ArrayList<Integer> spirallyTraverse(int[][] mat) {
6
7         ArrayList<Integer> result = new ArrayList<>();
8
9         int n = mat.length;
10        int m = mat[0].length;
11
12        int top = 0, bottom = n - 1;
13        int left = 0, right = m - 1;
14
15        while (top <= bottom && left <= right) {
16
17            // Left -> Right
18            for (int i = left; i <= right; i++) {
19                result.add(mat[top][i]);
20            }
21            top++;
22
23            // Top -> Bottom
24            for (int i = top; i <= bottom; i++) {
25                result.add(mat[i][right]);
26            }
27            right--;
28
29            // Right -> Left
30            if (top <= bottom) {
31                for (int i = right; i >= left; i--) {
32                    result.add(mat[bottom][i]);
33                }
34            }
35            bottom--;
36        }
37        return result;
38    }
39 }
```

At the bottom, there are buttons for "Custom Input", "Compile & Run", and "Submit".

Name - Anisha Raj
Roll no. - 2410031455
Section - 2CSE8

Java Experiment-2

Problem List

74. Search a 2D Matrix

Medium

Topics

Companies

You are given an $m \times n$ integer matrix `matrix` with the following two properties:

- Each row is sorted in non-decreasing order.
- The first integer of each row is greater than the last integer of the previous row.

Given an integer `target`, return `true` if `target` is in `matrix` or `false` otherwise.

You must write a solution in $O(\log(m * n))$ time complexity.

Example 1:

1	3	5	7
10	11	16	20
23	30	34	60

Input: `matrix = [[1,3,5,7],[10,11,16,20],[23,30,34,60]]`, `target = 3`
Output: `true`

Example 2:

1	3	5	7
---	---	---	---

Code

Java

```
1 class Solution {
2
3     public boolean searchMatrix(int[][] matrix, int target) {
4
5         int m = matrix.length;
6         int n = matrix[0].length;
7
8         int left = 0;
9         int right = m * n - 1;
10
11         while (left <= right) {
12
13             int mid = left + (right - left) / 2;
14
15             // Convert 1D index to 2D coordinates
16             int row = mid / n;
17             int col = mid % n;
18
19             int value = matrix[row][col];
20
21             if (value == target) {
22                 return true;
23             }
24             else if (value < target) {
25                 left = mid + 1;
26             }
27             else {
28                 right = mid - 1;
29             }
30         }
31
32         return false;
33     }
34 }
35
```

Saved

Ln 35, Col 1

17.7K

346

239 Online

Testcase

Test Result

Name - Anisha Raj
Roll no. - 2410031455
Section - 2CSE8

Java Experiment-2

The screenshot displays the GeeksforGeeks website interface. The browser's address bar shows the URL `geeksforgeeks.org/problems/median-in-a-row-wise-sorted-matrix1527/1`. The page title is "Median in a row-wise sorted Matrix". The problem description states: "Given a **row-wise sorted** matrix `mat[][]` of size `n*m`, where the number of rows and columns is always **odd**. Return the **median** of the matrix."

Examples:

Input: `mat[][] = [[1, 3, 5], [2, 6, 9], [3, 6, 9]]`
Output: 5
Explanation: Sorting matrix elements gives us [1, 2, 3, 3, 5, 6, 6, 9, 9]. Hence, 5 is median.

Input: `mat[][] = [[2, 4, 9], [3, 6, 7], [4, 7, 10]]`
Output: 6
Explanation: Sorting matrix elements gives us [2, 3, 4, 4, 6, 7, 7, 9, 10]. Hence, 6 is median.

Input: `mat = [[3], [4], [8]]`
Output: 4
Explanation: Sorting matrix elements gives us [3, 4, 8]. Hence, 4 is median.

Constraints:

The right side of the image shows the Java solution code in a dark-themed editor. The code is as follows:

```
1 class Solution {
2
3     int median(int[][] matrix) {
4
5         int r = matrix.length;
6         int c = matrix[0].length;
7
8         int low = 1;
9         int high = 2000; // according to constraints
10
11         while (low < high) {
12
13             int mid = (low + high) / 2;
14             int count = 0;
15
16             for (int i = 0; i < r; i++) {
17                 count += countSmallerThanOrEqual(matrix[i], mid);
18             }
19
20             if (count < (r * c + 1) / 2)
21                 low = mid + 1;
22             else
23                 high = mid;
24         }
25
26         return low;
27     }
28
29     int countSmallerThanOrEqual(int[] row, int target) {
30
31         int left = 0, right = row.length - 1;
32     }
```

At the bottom right, there are buttons for "Custom Input", "Compile & Run", and "Submit".

Name - Anisha Raj
Roll no. - 2410031455
Section - 2CSE8

Java Experiment-2

The screenshot displays the GeeksforGeeks website interface. The browser's address bar shows the URL `geeksforgeeks.org/problems/row-with-max-1s0023/1`. The page title is "Row with max 1s". The problem description states: "You are given a 2D binary array `arr[][]` consisting of only 1s and 0s. Each row of the array is sorted in non-decreasing order. Your task is to find and return the index of the first row that contains the maximum number of 1s. If no such row exists, return -1."

Note:

- The array follows 0-based indexing.
- The number of rows and columns in the array are denoted by `n` and `m` respectively.

Examples:

Input: `arr[][] = [[0,1,1,1], [0,0,1,1], [1,1,1,1], [0,0,0,0]]`
Output: 2
Explanation: Row 2 contains the most number of 1s (4 1s). Hence, the output is 2.

Input: `arr[][] = [[0,0], [1,1]]`
Output: 1
Explanation: Row 1 contains the most number of 1s (2 1s). Hence, the output is 1.

Input: `arr[][] = [[0,0], [0,0]]`
Output: -1
Explanation: No row contains any 1s, so the output is -1.

Constraints:
 $1 \leq \text{arr.size}(), \text{arr}[i].\text{size}() \leq 10^3$

The right side of the image shows the Java code for the solution:

```
1 class Solution {
2
3     int rowWithMax1s(int arr[][]){
4
5         int n = arr.length;
6         int m = arr[0].length;
7
8         int rowIndex = -1;
9         int j = m - 1; // Start from top-right corner
10
11         for (int i = 0; i < n; i++) {
12
13             while (j >= 0 && arr[i][j] == 1) {
14                 j--;
15                 rowIndex = i; // Update row index
16             }
17
18         }
19
20         return rowIndex;
21     }
22 }
```

At the bottom right, there are buttons for "Custom Input", "Compile & Run", and "Submit".