

## **Experiment List for Programming Ability and Logic Building-2**

Name : Ikshu Goel

Batch : 2CSE5

Roll Number : 2410030774

Week : 19/1/26 to 25/1/26

Lecture : 1 & 2

### **EXPERIMENT 1**

Given an array `arr[]` of positive integers, where each value represents the number of chocolates in a packet. Each packet can have a variable number of chocolates. There are  $m$  students, the task is to distribute chocolate packets among  $m$  students such that -

- i. Each student gets exactly one packet.

ii. The difference between maximum number of chocolates given to a student and minimum number of chocolates given to a student is minimum and return that minimum possible difference.

Examples:

Input: arr = [3, 4, 1, 9, 56, 7, 9, 12], m = 5

Output: 6

Explanation: The minimum difference between maximum chocolates and minimum chocolates is  $9 - 3 = 6$  by choosing following m packets :[3, 4, 9, 7, 9].

Constraints:

$1 \leq m \leq \text{arr.size} \leq 105$

$1 \leq \text{arr}[i] \leq 109$

The screenshot shows the GeeksforGeeks IDE interface. On the left, the 'Output Window' displays 'Compilation Results' for 'Custom Input'. It shows 'Compilation Completed' for 'Case 1'. The input is '3 4 1 9 56 7 9 12' and '5'. The 'Your Output' is '6' and the 'Expected Output' is '6'. On the right, the code editor shows a Java solution for the problem. The code is as follows:

```
1 import java.util.*;
2
3 class Solution {
4     public long findMinDiff(ArrayList<Integer> arr, int m) {
5
6         int n = arr.size();
7         if (m > n)
8             return -1;
9
10        Collections.sort(arr);
11
12        long minDiff = Long.MAX_VALUE;
13
14        for (int i = 0; i <= n - m; i++) {
15            long diff = arr.get(i + m - 1) - arr.get(i);
16
17            if (diff < minDiff) {
18                minDiff = diff;
19            }
20        }
21
22        return minDiff;
23    }
24 }
25
26
```

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

## EXPERIMENT 2

Given a number  $x$  and an array of integers  $arr$ , find the smallest subarray with sum greater than the given value. If such a subarray do not exist return 0 in that case.

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:

$1 \leq arr.size, x \leq 10^5$

$0 \leq arr[i] \leq 10^4$

Chocolate Distribution Problem x Smallest subarray with sum x

geeksforgeeks.org/problems/smallest-subarray-with-sum-g

To exit full screen, press and hold **esc**

Search...

Courses Tutorials Practice Jobs

Problem Editorial Submissions Comments

Output Window

Compilation Results Custom Input

Compilation Completed

Case 1

Input:

x =

51

arr[] =

1 4 45 6 0 19

Your Output:

3

Expected Output:

3

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

Custom Input Compile & Run Submit

## EXPERIMENT 3

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:

$1 \leq \text{arr.size()} \leq 10^6$

$1 \leq \text{array}[i], a, b \leq 10^9$

The screenshot shows the GeeksforGeeks website interface. The top navigation bar includes links for Courses, Tutorials, Practice, and Jobs. The main content area is divided into two sections: the left sidebar and the right main editor.

**Left Sidebar (Output Window):**

- Compilation Results** (Custom Input)
- Compilation Completed**
- Case 1**
- Input:**  
arr[] = 1 2 3 4  
a = 1  
b = 2
- Your Output:** true
- Expected Output:** true

**Right Main Editor (Java (21)):**

```
1- class Solution {
2-     public void threeWayPartition(int[] arr, int a, int b) {
3-         int low = 0, mid = 0, high = arr.length - 1;
4-
5-         while (mid <= high) {
6-             if (arr[mid] < a) {
7-                 swap(arr, low++, mid++);
8-             } else if (arr[mid] > b) {
9-                 swap(arr, mid, high--);
10-            } else {
11-                mid++;
12-            }
13-        }
14-    }
15-
16-    void swap(int[] arr, int i, int j) {
17-        int temp = arr[i];
18-        arr[i] = arr[j];
19-        arr[j] = temp;
20-    }
21- }
22- }
```

## EXPERIMENT 4

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:  $\text{arr}[] = [2, 1, 5, 6, 3]$ ,  $k = 3$

Output: 1

Explanation: To bring elements 2, 1, 3 together, swap index 2 with 4 (0-based indexing), i.e. element  $\text{arr}[2] = 5$  with  $\text{arr}[4] = 3$  such that final array will be-  $\text{arr}[] = [2, 1, 3, 6, 5]$

Input:  $\text{arr}[] = [2, 7, 9, 5, 8, 7, 4]$ ,  $k = 6$

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-  $\text{arr}[] = [9, 7, 2, 5, 4, 7, 8]$

Input:  $\text{arr}[] = [2, 4, 5, 3, 6, 1, 8]$ ,  $k = 6$

Output: 0

Constraints:

$1 \leq \text{arr.size}() \leq 10^6$

$1 \leq \text{arr}[i] \leq 10^6$

$1 \leq k \leq 10^6$

The screenshot shows the GeeksforGeeks online IDE interface. The browser tabs include "Chocolate Distribution Problem", "Smallest subarray with sum...", "Three way partitioning | Practice", and "Minimum swaps and K together". The URL bar shows "geeksforgeeks.org/problems/minimum-swaps-required-to-s...". The IDE has a top navigation bar with "Courses", "Tutorials", "Practice", and "Jobs". The main editor displays a Java solution for the "Minimum swaps required to sort an array" problem. The code is as follows:

```
1 class Solution {
2     int minSwap(int[] arr, int k) {
3         int n = arr.length;
4
5         int count = 0;
6         for (int x : arr)
7             if (x <= k) count++;
8
9         int bad = 0;
10        for (int i = 0; i < count; i++)
11            if (arr[i] > k) bad++;
12
13        int ans = bad;
14        for (int i = 0, j = count; j < n; i++, j++) {
15            if (arr[i] > k) bad--;
16            if (arr[j] > k) bad++;
17            ans = Math.min(ans, bad);
18        }
19        return ans;
20    }
21 }
22
23
```

The "Output Window" on the left shows "Compilation Results" for "Custom Input". It indicates "Compilation Completed" for "Case 1". The input is "arr[] = 2 1 5 6 3" and "k = 3". The "Your Output" is "1" and the "Expected Output" is "1". At the bottom right, there are buttons for "Custom Input", "Compile & Run", and "Submit".

## EXPERIMENT 5

Given an array `arr[]` of positive integers. Return true if all the array elements are palindrome otherwise, return false.



Examples:

Input: `arr[] = [111, 222, 333, 444, 555]`

Output: `true`

Explanation:

`arr[0] = 111`, which is a palindrome number.

`arr[1] = 222`, which is a palindrome number.

`arr[2] = 333`, which is a palindrome number.

`arr[3] = 444`, which is a palindrome number.

`arr[4] = 555`, which is a palindrome number.

As all numbers are palindrome so This will return true. Input: `arr[] = [121, 131, 20]`

Output: `false`

Explanation: 20 is not a palindrome hence the output is false.

Expected Time Complexity:  $O(n \log n)$

Expected Space Complexity:  $O(1)$

Constraints:

$1 \leq \text{arr.size} \leq 20$

$1 \leq \text{arr}[i] \leq 10^5$

The screenshot shows the GeeksforGeeks online IDE interface. The browser tabs at the top include "Chocolate Distribution Problem", "Smallest subarray with sum", "Three way partitioning | Practice", "Minimum swaps and K together", and "Palindromic Array | Practice". The URL in the address bar is <https://www.geeksforgeeks.org/problems/palindromic-array-1587115620>. A tooltip above the IDE says "To exit full screen, press and hold esc".

The IDE has a top navigation bar with "Problem", "Editorial", "Submissions", and "Comments" tabs. The "Problem" tab is active, showing the "Output Window" on the left and the code editor on the right.

**Output Window:**

- Compilation Results:** Custom Input
- Compilation Completed**
- Case 1:**
- Input:** 111 222 333 444 555
- Your Output:** true
- Expected Output:** true

**Code Editor:**

```
1 class Solution {
2
3     boolean isPalinArray(int[] arr) {
4         for (int num : arr) {
5             if (!isPalindrome(num)) {
6                 return false;
7             }
8         }
9         return true;
10    }
11
12    boolean isPalindrome(int num) {
13        int rev = 0;
14        int temp = num;
15
16        while (temp > 0) {
17            rev = rev * 10 + temp % 10;
18            temp /= 10;
19        }
20        return rev == num;
21    }
22 }
23
```

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

## EXPERIMENT 6

Given an array `arr[]` of integers, calculate the median.

Examples:

Input: `arr[] = [90, 100, 78, 89, 67]`

Output: 89

Explanation: After sorting the array middle element is the median

Input: `arr[] = [56, 67, 30, 79]`

Output: 61.5

Explanation: In case of even number of elements, average of two middle elements is the median.

Input: `arr[] = [1, 2]`

Output: 1.5

Explanation: The average of both elements will result in 1.5.

Constraints:

$1 \leq \text{arr.size()} \leq 105$

$1 \leq \text{arr}[i] \leq 105$

The screenshot shows the GeeksforGeeks online IDE interface. The browser address bar displays `geeksforgeeks.org/problems/find-the-median0527/1`. The IDE has a dark theme. On the left, the 'Output Window' is open, showing 'Compilation Results' for 'Custom Input'. It indicates 'Compilation Completed' for 'Case 1'. The input is `arr` with the value `90 100 78 89 67`. The 'Your Output' is `89`, and the 'Expected Output' is also `89`. On the right, the code editor shows a Java solution for finding the median of an array. The code is as follows:

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

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

## EXPERIMENT 7

You are given a rectangular matrix `mat[][]` of size  $n \times m$ , and your task is to return an array while traversing the matrix in spiral form.

Examples:

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]]`

Output: `[32, 44, 27, 23, 62, 50, 28, 54]`

Explanation: Applying same technique as shown above, output will be `[32, 44, 27, 23, 62, 50, 28, 54]`.

Constraints:

$1 \leq n, m \leq 1000$

$0 \leq \text{mat}[i][j] \leq 100$

geeksforgeeks.org/problems/spirally-traversing-a-matrix-15

To exit full screen, press and hold **esc**

Search...

Courses Tutorials Practice Jobs

Problem Editorial Submissions Comments

Output Window

Compilation Results Custom Input

Compilation Completed

Case 1

Input:

n = 4

m = 4

mat =

1	2	3	4
5	6	7	8

Your Output:

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

Expected Output:

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

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

Custom Input Compile & Run Submit

## EXPERIMENT 8

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:**

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

**Output:** true

**Example 2:**

**Input:** matrix = [[1,3,5,7],[10,11,16,20],[23,30,34,60]], target = 13

**Output:** false

**74. Search a 2D Matrix**

Medium

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:**

Input: `matrix = [[1,3,5,7],[10,11,16,20],[23,30,34,60]]`, `target = 13`  
Output: `false`

```

1 class Solution {
2     public boolean searchMatrix(int[][] matrix, int target) {
3         int n = matrix.length;
4         int m = matrix[0].length;
5
6         int low = 0, high = n * m - 1;
7
8         while (low <= high) {
9             int mid = (low + high) / 2;
10            int val = matrix[mid / m][mid % m];
11
12            if (val == target) return true;
13            else if (val < target) low = mid + 1;
14            else high = mid - 1;
15        }
16        return false;
17    }
18 }

```

Accepted Runtime: 0 ms

Case 1 Case 2

Input

matrix =  
[[1,3,5,7],[10,11,16,20],[23,30,34,60]]

target =

## EXPERIMENT 9

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:  $\text{mat} = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 6 & 9 \\ 3 & 6 & 9 \end{bmatrix}$

Output: 5

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

Input:  $\text{mat} = \begin{bmatrix} 2 & 4 & 9 \\ 3 & 6 & 7 \\ 4 & 7 & 10 \end{bmatrix}$

Output: 6

Explanation: Sorting matrix elements gives us [2, 3, 4, 4, 6, 7, 7, 9, 10].

Hence, 6 is median. Input:  $\text{mat} = \begin{bmatrix} 3 \\ 4 \\ 8 \end{bmatrix}$

Output: 4

Explanation: Sorting matrix elements gives us [3, 4, 8]. Hence, 4 is median.

Constraints:

$1 \leq n, m \leq 400$

$1 \leq \text{mat}[i][j] \leq 2000$



The screenshot displays the GeeksforGeeks online IDE interface. On the left, the 'Output Window' shows the 'Compilation Results' for 'Case 1'. The input is defined as  $n = 3$ ,  $m = 3$ , and a 3x3 matrix  $mat[][] = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 6 & 9 \end{bmatrix}$ . The 'Your Output' is 5, and the 'Expected Output' is also 5. The right panel shows the Java code for the 'median' function, which uses a list to collect all elements and then sorts them to find the median.

```

1- import java.util.*;
2-
3- class Solution {
4-     int median(int[][] matrix) {
5-         int r = matrix.length;
6-         int c = matrix[0].length;
7-
8-         ArrayList<Integer> list = new ArrayList<>();
9-
10-        for (int i = 0; i < r; i++) {
11-            for (int j = 0; j < c; j++) {
12-                list.add(matrix[i][j]);
13-            }
14-        }
15-
16-        Collections.sort(list);
17-        return list.get(list.size() / 2);
18-    }
19- }
20-

```

## EXPERIMENT 10

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:

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

$0 \leq \text{arr}[i][j] \leq 1$

geeksforgeeks.org/problems/row-with-max-1s0023/1

To exit full screen, press and hold **esc**

Search...

Courses Tutorials Practice Jobs

Problem Editorial Submissions Comments

Output Window

Compilation Results Custom Input

Compilation Completed

Case 1

Input:

arr[][] =

`[[0,1,1,1], [0,0,1,1], [1,1,1,1], [0,0,0,0]]`

Your Output:

2

Expected Output:

2

Java (21)

Start Timer

```
1 class Solution {
2     int rowWithMaxis(int[][] arr) {
3         int n = arr.length;
4         int m = arr[0].length;
5
6         int maxRow = -1;
7         int maxCount = 0;
8
9         for (int i = 0; i < n; i++) {
10            int count = 0;
11            for (int j = 0; j < m; j++) {
12                if (arr[i][j] == 1) count++;
13            }
14            if (count > maxCount) {
15                maxCount = count;
16                maxRow = i;
17            }
18        }
19        return maxRow;
20    }
21 }
22
23
```

Custom Input Compile & Run Submit