

EXPERIMENT FOR PROGRAMMING ABILITY AND LOGIC BUILDING-1 (JAVA)

pdf-1

EXPERIMENT-1

The screenshot shows a browser window for GeeksforGeeks Practice. The URL is <https://www.geeksforgeeks.org/problems/reverse-an-array/1>. The page displays a Java code editor with the following code:

```
1 class Solution {
2     public void reverseArray(int arr[]) {
3         // code here
4         int i=0;
5         int j=arr.length-1;
6         while(i<j){
7             int temp=arr[i];
8             arr[i]=arr[j];
9             arr[j]=temp;
10            i++;
11            j--;
12        }
13    }
14    public static void main(String[] args){
15        int[]arr={1,4,3,2,6,5};
16        System.out.println("Reversed Array:"+ Arrays.toString(arr));
17    }
18 }
```

The 'Compilation Results' section shows 'Problem Solved Successfully' with 1115 / 1115 test cases passed, 1 / 1 attempts correct, and 100% accuracy. The 'Output Window' shows the reversed array [5, 6, 2, 3, 4, 1]. The status bar at the bottom right indicates the date as 27-01-2026.

EXPERIMENT-2

The screenshot shows a browser window for GeeksforGeeks Practice. The URL is <https://www.geeksforgeeks.org/problems/find-minimum-and-maximum-element-in-an-array/1>. The page displays a Java code editor with the following code:

```
1 class Solution {
2     public ArrayList<Integer> getMinMax(int[] arr) {
3         // code Here
4         ArrayList<Integer> result = new ArrayList<>();
5
6         int min = arr[0];
7         int max = arr[0];
8
9         for (int i = 1; i < arr.length; i++) {
10             if (arr[i] < min) {
11                 min = arr[i];
12             }
13             if (arr[i] > max) {
14                 max = arr[i];
15             }
16         }
17
18         result.add(min);
19         result.add(max);
20
21         return result;
22     }
23
24 }
```

The 'Compilation Results' section shows 'Problem Solved Successfully' with 1111 / 1111 test cases passed, 1 / 1 attempts correct, and 100% accuracy. The 'Output Window' shows the minimum and maximum values of the array. The status bar at the bottom right indicates the date as 27-01-2026.

EXPERIMENT-3

The screenshot shows a browser window on the GeeksforGeeks Practice platform. The URL is <https://www.geeksforgeeks.org/problems/kth-smallest-element5635/1>. The code editor contains the following Java code:

```
1 class Solution {
2     public int kthSmallest(int[] arr, int k) {
3         // Code here
4         Arrays.sort(arr);
5         return arr[k-1];
6     }
7 }
```

The 'Compilation Results' section shows 'Test Cases Passed: 1121 / 1121', 'Attempts : Correct / Total: 1 / 1', 'Accuracy: 100%', 'Points Scored: 4 / 4', and 'Time Taken: 0.74'. A message at the top says 'Problem Solved Successfully'.

EXPERIMENT-4

The screenshot shows a browser window on the GeeksforGeeks Practice platform. The URL is <https://www.geeksforgeeks.org/problems/union-of-two-arrays3538/1>. The code editor contains the following Java code:

```
1 class Solution {
2     public static ArrayList<Integer> findUnion(int[] a, int[] b) {
3         // code here
4         HashSet<Integer> set=new HashSet<>();
5         for(int x:a){
6             set.add(x);
7         }
8         for(int x:b){
9             set.add(x);
10        }
11    }
12    return new ArrayList<>(set);
13 }
```

The 'Compilation Results' section shows 'Test Cases Passed: 1111 / 1111', 'Attempts : Correct / Total: 1 / 1', 'Accuracy: 100%', 'Points Scored: 2 / 2', and 'Time Taken: 1.07'. A message at the top says 'Problem Solved Successfully'.

EXPERIMENT-5

The screenshot shows a browser window with several tabs open, including "JAVA EXP-1 - Google Docs", "Min and Max in Array | Practice", "Largest in Array | Practice | Geeks", and the current page "geeksforgeeks.org/problems/largest-element-in-array4009/0?utm_source=youtube&utm_medium=collab_striver_ytdescription&utm". The main content is a GeeksforGeeks practice interface.

Output Window: Problem Solved Successfully (1115 / 1115). Attempts: Correct / Total 1 / 1. Accuracy: 100%. Points Scored 1 / 1. Time Taken 0.74. Your Total Score: 14.

Code Editor: Java (21) - Start Timer. The code is as follows:

```
1 class Solution {
2     public static int largest(int[] arr) {
3         // code here
4
5         int max = arr[0];
6         for (int i = 1; i < arr.length; i++) {
7             if (arr[i] > max) {
8                 max = arr[i];
9             }
10        }
11    }
12    return max;
13 }
14 }
```

Buttons at the bottom: Custom Input, Compile & Run, Submit.

EXPERIMENT-6

The screenshot shows a browser window with several tabs open, including "JAVA EXP-1 - Google Docs", "Min and Max in Array | Practice", "Largest in Array | Practice | Geeks", and the current page "geeksforgeeks.org/problems/cyclically-shift-an-array-by-one2614/1". The main content is a GeeksforGeeks practice interface.

Output Window: Problem Solved Successfully (1115 / 1115). Attempts: Correct / Total 1 / 1. Accuracy: 100%. Points Scored 1 / 1. Time Taken 1.09. Your Total Score: 15.

Code Editor: Java (21) - Start Timer. The code is as follows:

```
1 // User function Template for Java
2
3 class Solution {
4     public void rotate(int[] arr) {
5         // code here
6         int n=arr.length;
7         int last =arr[n-1];
8         for(int i=n-1;i>0;i--){
9             arr[i]=arr[i-1];
10        }
11        arr[0]=last;
12    }
13 }
```

Buttons at the bottom: Custom Input, Compile & Run, Submit.

EXPERIMENT-7

A screenshot of a web browser window showing a Java practice session on GeeksforGeeks. The title bar says "JAVA EXP-1 - Google Docs". The main content area shows the problem statement for Kadane's Algorithm, the code submitted by the user, and the results of the submission.

Output Window

Compilation Results

Problem Solved Successfully

Test Cases Passed: **1120 / 1120**

Attempts : Correct / Total: **1 / 1**

Accuracy: 100%

Points Scored: **4 / 4**

Time Taken: **0.65**

Your Total Score: **8**

Solve Next

Java (21) Start Timer

```
1 class Solution {  
2     int maxSubarraySum(int[] arr) {  
3         // Code here  
4         int currentSum=arr[0];  
5         int maxSum=arr[0];  
6         for(int i=1;i<arr.length; i++){  
7             currentSum=Math.max(arr[i],currentSum+arr[i]);  
8             maxSum=Math.max(maxSum,currentSum);  
9         }  
10    }  
11 }  
12 }
```

Custom Input Compile & Run Submit

EXPERIMENT-8

A screenshot of a web browser window showing a LeetCode solution statistics page for the "Search Insert Position" problem. The title bar says "JAVA EXP-1 - Google Docs" and "leetcode.com/problems/search-insert-position/submissions/1915325420/?envType=problem-list-v2&envId=array".

Runtime: 0 ms | Beats 100.00% (Fastest)

Memory: 44.98 MB | Beats 25.13%

Code

```
1 class Solution {  
2     public int searchInsert(int[] nums, int target) {  
3         int left = 0;  
4         int right = nums.length - 1;  
5         int mid = left + (right - left) / 2;  
6         while (left <= right) {  
7             if (nums[mid] == target) {  
8                 return mid;  
9             } else if (nums[mid] < target) {  
10                 left = mid + 1;  
11             } else {  
12                 right = mid - 1;  
13             }  
14             mid = left + (right - left) / 2;  
15         }  
16         return left;  
17     }  
18 }
```

Testcase: Accepted | Runtime: 0 ms

Input:
nums = [1,3,5,6]
target =