DESIGN AND ANALYSIS OF ALGORITHMS LABORATORY
(MA39203)
Department of Mathematics
Indian Institute of Technology, Kharagpur
Assignment : 06
Date : 27/08/2025

1. Implement **Randomized Select** to find the k-th smallest element of a given unsorted array.

Example: arr = [10, 25, 3, 1, 15, 11], $k = 3$ Output: 10

2. You are given an integer k and an array A of size n, which represents a data stream of integers arriving one by one. For each position i (where $i \geq k$), find the k-th largest element among the first i elements of the stream. Return the results in an array B of size $n - k + 1$, where: B$[i - k]$ = the k-th largest element in the prefix A$[0, 1, ..., i - 1]$.

Example: Input: $n = 8, k = 3$, A = [4, 5, 8, 2, 3, 5, 10, 9] Output: [4, 4, 4, 5, 5, 8]

3. You are given an array A of size n, which represents a data stream of integers arriving one by one. For each prefix of the stream (from the first element up to the i-th element), compute the median of that prefix. Return the results in an array B of size n, where $B[i]$ is median of the prefix A$[0..i]$.

Example: Input: $n = 6$, A = [4, 5, 8, 2, 3, 1] Output: [4, 4.5, 5, 4.5, 4, 3.5]

Use *iostream* header only for the following questions 4, 5 and 6.

4. Given an array of integers arr[ ] representing a permutation (i.e., all elements are unique and arranged in some order), find the next lexicographically greater permutation by rearranging the elements of the array. If such a permutation does not exist (i.e., the array is the last possible permutation), rearrange the elements to form the lowest possible order (i.e., sorted in ascending order).

Example 1: arr[] = [2, 4, 1, 7, 5, 0] Output: [2, 4, 5, 0, 1, 7]
Explanation: The next lexicographically greater arrangement of 241750 is 245017

Example 2: Input: arr[ ] = [3, 2, 1] Output: [1, 2, 3]
Explanation: This is the last permutation, so we return the lowest possible permutation.

5. Create a Linked List class that supports construction of a linked list from a given input array of integers. Write a sort function that takes the head of the linked list as an argument and sorts the list in ascending order. The sorting algorithm must run in $O(n \log n)$ time complexity and use $O(\log n)$ space complexity.

Also add a print function that prints the integers in the linked list in order, separated by spaces.

Example: head = [3, 1, 4, 2] Output: 1 2 3 4

6. Implement **Bucket Sort** for an input array of floating-point numbers where all values lie in the range [0,1). Each bucket should be represented by a linked list. Modify your Linked List class and the sort function from the previous problem accordingly, also add a *tail* pointer to your Linked List class so that new elements can be inserted in $O(1)$ time.