DESIGN AND ANALYSIS OF ALGORITHMS LABORATORY (MA39203)

Department of Mathematics
Indian Institute of Technology, Kharagpur
Assignment : 07
Date : 03/09/2025

1. Implement the Median of Medians algorithm to find the K-th smallest element in a given array.
Example: Input: arr[ ] = [7, 10, 4, 11, 8, 3, 20, 15], $K = 3$ Output: 7

2. Given a string, find the length of its longest substring without any repeating character.
Example: Input: s =“abcabcbb” Output: 3 (“abc” is the longest substring)

3. Given an integer array arr[ ] and an integer k, find the number of non-empty subarrays that have a sum divisible by k.
Example: Input: arr[ ] = {-1, 2, 9}, $k = 2$; Output: 2
Explanation: {2} and {-1, 2, 9} are the two subarrays with sum divisible by 2.

4. Given two strings s and t, return the minimum window substring of s such that every character in t (including duplicates) is included in the window. If there is no such substring, return the empty string “”.
Example 1: Input: s = “ADOBECODEBANC”, t = “ABC” Output: “BANC”
Explanation: The minimum window substring “BANC” includes ‘A’, ‘B’, and ‘C’ from string t.
Example 2: Input: s = “ab”, t = “aa” Output: “”
Explanation: Both ‘a’s from t must be included in the window. Since s only has one ‘a’, return empty string.

5. Given an array arr[ ] consisting of n integers, print all the array elements that occur strictly more than $\lfloor \frac{n}{3} \rfloor$ times.
Example 1: Input: $n = 8$, arr[ ] = {2, 2, 3, 1, 3, 2, 1, 1} Output: 1 2
Explanation: The frequency of 1 and 2 is 3, which is more than $\lfloor \frac{8}{3} \rfloor = 2$.
Example 2: $n = 3$, arr[ ]={2, 3, 5} Output: No element found
Explanation: Each element in the array has frequency 1, which is not greater than $\lfloor \frac{3}{3} \rfloor = 1$.

6. Implement two **hash tables**, one using **chaining** for collision handling, another one using **open addressing** (linear probing). Both of them should support insert, search and delete operations. Use the hash function $h(k) = k \bmod 13$.

In Open Addressing (Linear Probing), each slot of the table holds at most one key. On collision, probe sequentially (i.e., if $h(k)$ is non-empty, try the slot $(h(k) + 1) \bmod 13$, next $(h(k) + 2) \bmod 13$, and so on) until an empty slot is found.