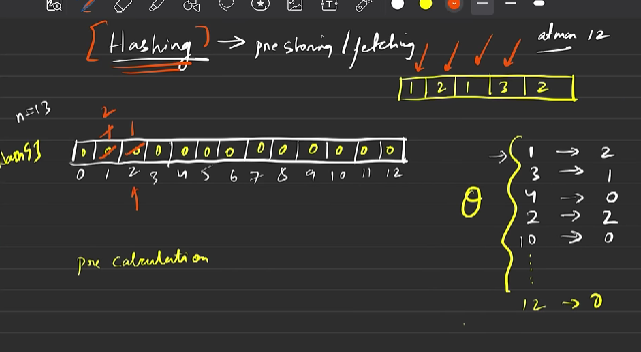
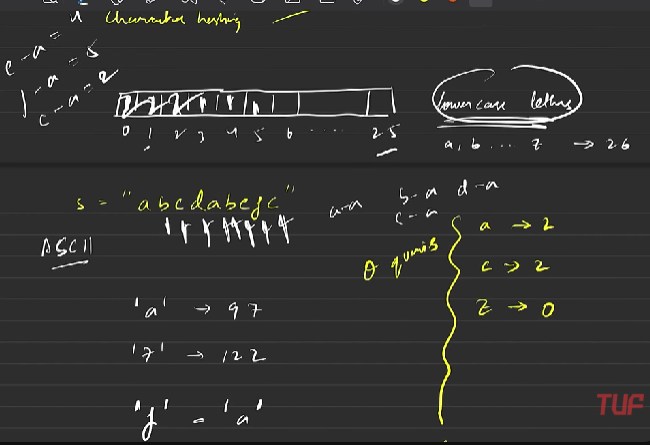
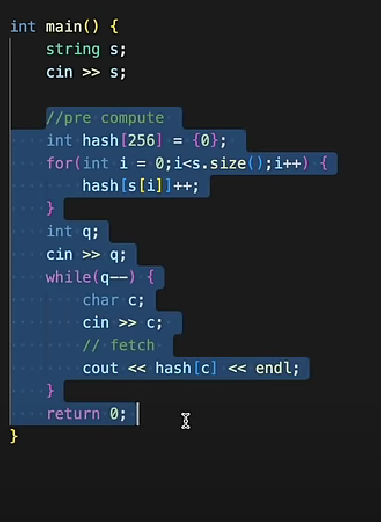
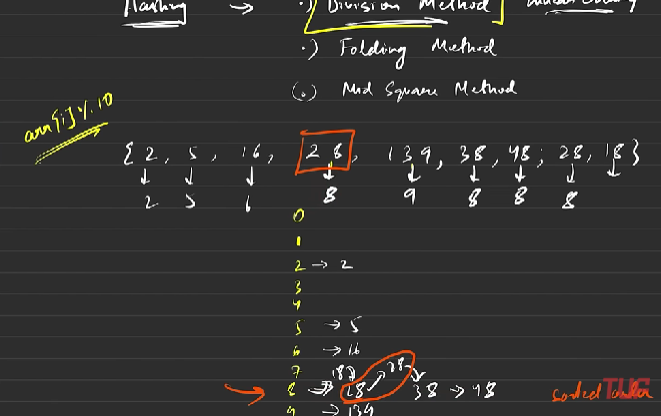
# Hashing

* Counting frequency of an element in array –(Number Hashing)
  + Brute Force Approach –
    - Input the element and match it with each element while traversing through the array .
    - If it matches increase counter by one.
    - TC – length of array \* O(n) - if I need to find the frequency of each element of array having n numbers.
  + Optimised Approach – Hash Table / Frequency Table
    - Prestoring and fetching something
    - Where in the array index=the element in original array and element at index=no. of its occurrences in original array.
    - Iterate from 0 to n and as soon as any element comes add one to its count on that index in the hash array
    - TC – O(n) – and we can get frequency of occurrence of each element in single go.
    - 
    - inside main u can declare array of max 10 raise to power 6 of numbers (bool - 10 raise to power 7)
    - outside it can be of 10 raise to power 7 -> int arr[1e7] -> (bool - 10 raise to power 8)
    - otherwise segmentation array.
  + Problem of limited memory of array in main
    - Solution- use map or unordered map from STL.
    - Key value format where key is element in original array and value is its frequency.
* Character Hashing – how many times certain character appear in string.
  + Brute Force Approach
  + Optimised Approach –
    - 256 characters in total including uppercase lowercase and symbols.
    - 
    - 
    - Map <char,int> can also be used to sstore frequencies so that there is no need of conversion into ASCII.
* Important –
  + Map – TC - O(log n) for both storing and fetching.
  + Unordered Map
    - Best and Average Case – TC – O(1)
    - Worst case – TC – O(n) – (COLLISION)
    - First preference should be unordered map.
    - Switch to map only if TLE (Time Limit Exceeded)
* Hashing –
  + Division Method –
    - 
    - Problem - for eg. You have array = [1,2,3,45,456];
      * So you need to create an hash array with hash[457].
      * But the condition is u can have hash table with max length 10.
      * Therefor division method comes in where each element is modularised by 10 and stored at that particular index, Lessens time.
      * However, ,many elements can have same modulus so they are stored at that particular index in chained fashion in sorted order.
      * Also there can be a situation where all elements are stored at same index -> COLLISION.
      * COLLISION – worst case scenario – O(n)
  + Folding Method
  + Mid Square Method
* In Map any ds can pe key like pair etc.
* In Unordered Map only limited like int,char.