1. Implement your templated unordered_set class with following functions using linear probing as collision handling scheme
    a. Constructor
    b. bool contains() const {}
    c. void insert(const key & k) {}
    d. void erase(const key &k) {} // Throw error if key not present
    e. void clear() {}
    f. bool isEmpty() const {}
    g. private: expandAndRehash()
2. Given two arrays: arr1[0..m-1] and arr2[0..n-1]. Find whether arr2[] is a subset of arr1[] or not. Both the arrays are not in sorted order. In O(max(n,m))
3. Given two strings S and T. Write a function that returns the minimum length substring in S which contains all characters in T. in O(n)
4. Given an array of N integers print pair of elements which sum to X. in O(n)
5. You are given with an array of integer contain number in no particular order. Write a program find the longest possible sequence of consecutive numbers using the numbers from the array. Best solution takes O(n) time. e.g. Input = [2, 12, 9, 16, 10, 5, 3, 20, 25, 11, 1, 8, 6] Output = [8, 9, 10, 11, 12] Input = [15, 13, 23, 21, 19, 11, 16] Output = [15, 16]
6. You are given an array of N positive integers, A1, A2 ,…, AN. Also, given a Q updates of form: - i j: Update Ai = j. 1 ≤ i ≤ N. Perform all updates and after each such update report mode of the array. Mode is the most frequently occurring element on the array. - If multiple modes are possible, return the smallest one. Input would be N – Number of integers in the array, Q – Number of Queries, followed by N integers of the array and then Q rows of two integers each representing i and j. Output – Q integers which are modes after each update query.

    Sample Input:

    5
    2,2,2,3,3
    3
    1,3
    5,4,
    2,4

    Sample Ouput:

    3

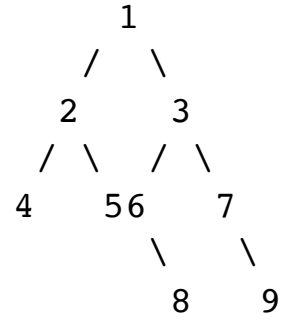
2
3

Explanation:
A = [3, 2, 2, 3, 3] after 1st update.
3 is mode.

A = [3, 2, 2, 3, 4] after 2nd update.
2 and 3 both are mode. Return smaller i.e. 2.

A = [3, 4, 2, 3, 4] after 3rd update.
3 and 4 both are mode. Return smaller i.e. 3.

7. Print a Binary Tree in Vertical Order



The output of print this tree vertically will be:

4
2
1 5 6
3 8
7
9

8. Given an unsorted array that may contain duplicates. Also given a number k which is smaller than size of array. Write a function that returns true if array contains duplicates within k distance.
Example:

Input: k = 3, arr[] = {1, 2, 3, 4, 1, 2, 3, 4}
Output: false
All duplicates are more than k distance away.

Input: k = 3, arr[] = {1, 2, 3, 1, 4, 5}

Output: true
1 is repeated at distance 3