

AGENDA

Queue:

- Introduction (FIFO)
- Implement Queue using 2 stacks [only discuss]
- Reverse a queue
- Generate binary numbers from 1 to N
- First non-repeating character in a stream of characters

Deque

- Introduction
- Sliding window maximum

Reverse a Queue

Given a queue **q** containing integer elements, your task is to reverse the queue.

Examples:

Input: `q[] = [4 3 1 10 2 6]`

Output: `[6, 2, 10, 1, 3, 4]`

Explanation: After reversing the given elements of the queue, the resultant queue will be 6 2 10 1 3 4.

Input: `q[] = [4 3 2 1]`

Output: `[1, 2, 3, 4]`

Explanation: After reversing the given elements of the queue, the resultant queue will be 1 2 3 4.

Input: `q[] = [7, 9, 5, 12, 8]`

Output: `[8, 12, 5, 9, 7]`

Explanation: After reversing the given elements of the queue, the resultant queue will be 8, 12, 5, 9, 7.

Generate binary numbers from 1 to N

Given a number **n**. The task is to generate all **binary numbers with decimal values** from **1 to n**.

Examples:

Input: n = 2

Output: ["1", "10"]

Explanation: Binary numbers from 1 to 2 are 1 and 10.

Input: n = 5

Output: ["1", "10", "11", "100", "101"]

Explanation: Binary numbers from 1 to 5 are 1 , 10 , 11 , 100 and 101.

First non-repeating character in a stream of characters

Given an input stream **s** consisting only of lowercase alphabets. While reading characters from the stream, you have to tell which character has appeared only once in the stream upto that point. If there are many characters that have appeared only once, you have to tell which one of them was the first one to appear. If there is no such character then append '#' to the answer.

NOTE:

1. You need to find the answer for every i ($0 \leq i < n$)
2. In order to find the solution for every i you need to consider the string from starting position till the i th position.

Examples:

Input: `s = "aabc"`

Output: `"a#bb"`

Explanation: For every i th character we will consider the string from index 0 till index i first non repeating character is as follow- "a" - first non-repeating character is 'a' "aa" - no non-repeating character so '#' "aab" - first non-repeating character is 'b' "aabc" - there are two non repeating characters 'b' and 'c', first non-repeating character is 'b' because 'b' comes before 'c' in the stream.

Input: `s = "zz"`

Output: `"z#"`

Explanation: For every character first non repeating character is as follow- "z" - first non-repeating character is 'z' "zz" - no non-repeating character so '#'

Input: `s = "bb"`

Output: `"b#"`

Explanation: For every character first non repeating character is as follow- "b" - first non-repeating character is 'b' "bb" - no non-repeating character so '#'

Sliding window maximum

Given an array **arr[]** of integers and an integer **k**, your task is to find the maximum value for each contiguous subarray of size **k**. The output should be an array of maximum values corresponding to each contiguous subarray.

Examples:

Input: arr[] = [1, 2, 3, 1, 4, 5, 2, 3, 6], k = 3

Output: [3, 3, 4, 5, 5, 5, 6]

Explanation:

1st contiguous subarray = [1 2 3] max = 3

2nd contiguous subarray = [2 3 1] max = 3

3rd contiguous subarray = [3 1 4] max = 4

4th contiguous subarray = [1 4 5] max = 5

5th contiguous subarray = [4 5 2] max = 5

6th contiguous subarray = [5 2 3] max = 5

7th contiguous subarray = [2 3 6] max = 6

Input: arr[] = [8, 5, 10, 7, 9, 4, 15, 12, 90, 13], k = 4

Output: [10, 10, 10, 15, 15, 90, 90]

Explanation:

1st contiguous subarray = [8 5 10 7], max = 10

2nd contiguous subarray = [5 10 7 9], max = 10

3rd contiguous subarray = [10 7 9 4], max = 10

4th contiguous subarray = [7 9 4 15], max = 15

5th contiguous subarray = [9 4 15 12], max = 15

6th contiguous subarray = [4 15 12 90], max = 90

7th contiguous subarray = [15 12 90 13], max = 90

Input: arr[] = [5, 1, 3, 4, 2, 6], k = 1

Output: [5, 1, 3, 4, 2, 6]

Explanation:

When k = 1, each element in the array is its own subarray, so the output is simply the same array