22/11/2023
Wednesday

* **Reverse Queue -**

I/P - 10 20 30 40 50

O/P - 50 40 30 20 10

Here, we are using stack to reverse the queue-

Step 1: Queue ke saare elements ko stack mei daaldo. Vo ab reverse hogye honge.

Step 2: Stack mei se saare elements ko firse queue mei push krvado.

Code -

```
class Solution {
    public:
    queue<int> rev(queue<int> q) {
      stack<int> st;
    // queue se stack mei push
      while (!q.empty()) {
        int temp = q.front();
        q.pop();
        st.push(temp);
      }
    // stack se firse queue mei push
      while (!st.empty()) {
        int temp = st.top();
        st.pop();
        q.push(temp);
      }
    return q; } };
```

Here, we are using recursion to reverse the queue -

- Ek element ko reverse krdo queue ke, rest recursion will handle.

Code -

```
class Solution{
    public:
    void reverse(queue<int> &q){

        // base case
        if(q.empty()){
            return;
        }
        // one case
        int temp = q.front();
        q.pop();
        // reverseing, recursive call
        reverse(q);
        // BT
        q.push(temp);
    }

    queue<int> rev(queue<int> q){
            reverse(q);
            return q;
        }
};
```

__/__/____

* Reverse first K elements of Queue –

I/P – n = 5 K = 3
1 2 3 4 5

O/P – 3 2 1 4 5

I/P – n = 7 K = 4
1 2 3 4 5 6 7

O/P – 4 3 2 1 5 6 7

→ Assuming, K is always less than or equals n. K elements ko reverse krdo, rest elements ko as it is return krdo.

Approach –

Step 1: Push first K elements into the stack.

Step 2: Push all K elements in queue again.

Step 3: First pop, then push (n-K) elements into queue again.

Code -

```
queue<int> modifyQueue (queue<int>q, int k){

    stack<int>st;

    for (int i=0; i<k; i++){
        int element = q.front();
        q.pop();
        st.push(element);
    }

    while (!st.empty()){
        int element = st.top();
        st.pop();
        q.push(element);
    }

    int n= q.size();
    for (int i=0; i<(n-k); i++){
        int element= q.front();
        q.pop();
        q.push(element);
    }
    return q;
}
```

__/__/____

* **Interleave the first half of the Queue with the second half -**

→ Given queque of N integers of even length.

I/P - 1 2 3 4 5 6

O/P - 1 4 2 5 3 6

I/P - 7 4 2 9 8 3 6 5

O/P - 7 8 4 3 2 6 9 5

Approach -

Step 1: Break the queue into half. Now, we have two queues.

Step 2: Now, ek first queue ka element insert kro and ek second queue ka until they get emptied.

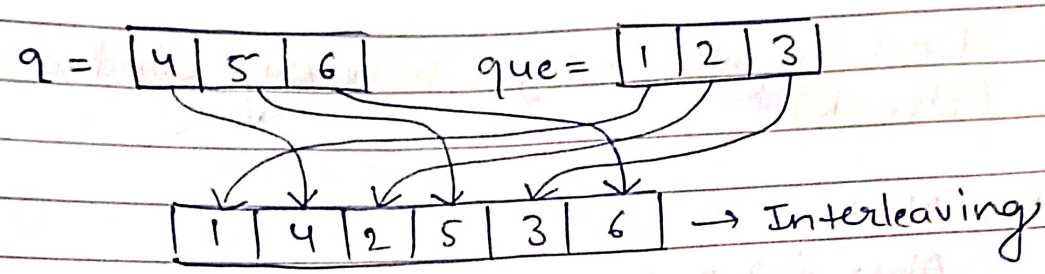
let we have a queue q -

q = | ~~1~~ | ~~2~~ | ~~3~~ | 4 | 5 | 6 |

Humne ek second queue que bnai -

que = | 1 | 2 | 3 |

and the queue q is - | 4 | 5 | 6 |



Code-

```
class Solution {
    public:
vector<int> rearrangeQueue (queue <int> q) {

    queue <int> que;
    int pos = q.size() /2 +1;

    while (pos != 1) {
        pos--;
        int temp= q.front();
        q.pop();
        que.push (temp);
       }

    vector <int> ans;
    while (! q.empty()) {
      int temp = que.front();
      ans.push_back (temp);
      que.pop();

        temp = q.front();
        ans.push_back (temp);
        q.pop();
      }
   return ans;
  }
};
```

__/__/____

* First negative integer in every window of size K.

I/P → N = 5
A[] = {-8, 2, 3, -6, 10}
K = 2

O/P → -8 0 -6 -6

I/P → N = 8
A[] = {-12, -1, -7, 8, -15, 30, 16, 28}
K = 3

O/P → -1 -1 -7 -15 -15 0

Approach -

Sliding window ka concept lg rha hei isme.

- First window ko process kro and store ans.

First window ko alg se isliye process kr. rhe so that deque mei kuch initial elements aa jaye i.e. deque mei initial state maintain krni thi isliye kiya.

- Remaining windows process krlo.

Note - Normal queue se bhi ho skta hai but deque se kr rhe hai coz deque mei dono taraf se push & pop ki accessibility hai.

Sliding window

- First K elements process - first window
- Remaining windows process
  - Removal
  - Addition

A[] =

-8	2	3	-6	10
0	1	2	3	4

No. of windows = N-K+1 = 5-2+1 = 4

Maintain a deque. Deque mei element ko tbhi daalna if the element is -ve. Index daalna hai -ve element ka deque mei.

Initial deque -

deque
0

$\omega_1 = -8\ \ 2 \rightarrow -8$
$\omega_2 = 2\ \ 3 \rightarrow 0$
$\omega_3 = 3\ \ -6 \rightarrow -6$
$\omega_4 = -6\ \ 10 \rightarrow -6$

For every next window, ek element remove ho rha hai and ek element add ho rha hai.

If !dq.empty(), then us index pr jo element hai use ans vector mei push krdo.

ans

-8	0	-6	-6

__/__/__

w_1 → | -8 | 2 |
0 1

Deque → | 0 | |

Ans → | -8 |

w_2 → | 2 | 3 |
1 2

w_2 mei 0 index wala element nhi aara, that's why use pop out kr diya.

Deque → | ~~0~~ | |

w_2 mei koi element -ve nhi tha that's why kuch push bhi nhi kiya. Empty deque ke case mei ans mei '0' push krna hai.

Ans → | -8 | 0 |

w_3 → | 3 | -6 |
2 3

Deque → | 3 | |

Ans → | -8 | 0 | -6 |

w_4 → | -6 | 10 |
3 4

Deque → | 3 | |

w_4 mei 3rd index wala element aara hai, that's why pop nhi kiya.

Ans → | -8 | 0 | -6 | -6 |

Ab deque ke front mei 3rd index hai, so A[3] push kr diya.

Code –

```
vector<long long> printFirstNegativeInteger(
long long int A[], long long int N, long long int K){

    vector<long long> ans;
    deque<int> dq;

  //process first k elements - first window
   for(int index=0; index<K; index++){
        int element = A[index];
        if(element<0){
            dq.push_back(index);
          }
      }

  //process rest of the elements
   for(int index=K; index<N; index++){
        if(dq.empty()){
            ans.push_back(0); }
        else{
            ans.push_back(A[dq.front()]);
          }

    //removal
    if(index-dq.front()>=K && !dq.empty()){
          dq.pop_front();
        }

    //addition
    if(A[index]<0){
          dq.push_back(index);
        }
  }
```

// Pichli loop last window ka ans nhi degi coz
// condition false ho jayegi isliye usko alg se
// push krva liya.

```
if (dq.empty()){
    ans.push_back(0);
  }
else{
    ans.push_back(A[dq.front()]);
  }
return ans;
}
```

Important pattern -

Answer input me hi lie kr rha hota hai whether array or string but hum kuch operations bchane ke liye queue ka use krte hai coz queue mei agr koi element answer nhi hai to hum use discard bhi kr skte hai.