DSA SERIES

- Learn Coding

Topic to be Covered today

Deque

LETS START TODAY'S LECTURE

641. Design Circular Deque

```
class MyCircularDeque {
public:
    int* arr;
    int front, rear, capacity, size;

    MyCircularDeque(int k) {
        capacity = k;
        arr = new int[capacity];
        front = -1;
        rear = 0;
        size = 0;
    }

    bool insertFront(int value) {
        if(isFull()){
            return false;
        }

        if(front == -1){
            front=rear=0;
```

```
  } else{
        front = (front+capacity-1)%capacity;
    }

    arr[front]= value;
    size++;
    return true;
}

bool insertLast(int value) {
     if(isFull()){
        return false;
    }

    if(front == -1){
        front=rear=0;
    } else{
       rear = (rear+1)%capacity;
    }

    arr[rear]= value;
    size++;
```

```
return true;
}

bool deleteFront() {
    if(isEmpty()){
        return false;
    }

    if(front == rear){
        front=rear=-1;
    } else{
        front = (front+1)%capacity;
    }
    size--;
    return true;
}

bool deleteLast() {
     if(isEmpty()){
        return false;
    }

    if(front == rear){
        front=rear=-1;
```

```
    } else{
        rear = (rear+capacity-1)%capacity;
    }
    size--;
    return true;
}

int getFront() {
    if(isEmpty()){
        return -1;
    }
    return arr[front];
}

int getRear() {
     if(isEmpty()){
        return -1;
    }

    return arr[rear];
}
```

```
    bool isEmpty() {
        return size==0;
    }

    bool isFull() {
        return size==capacity;
    }
};
```

239. Sliding Window Maximum

```
class Solution {
public:
    vector<int> maxSlidingWindow(vector<int>& nums, int k) {
        deque<int> dq;

        vector<int> result;
        int n = nums.size();

        for(int i =0;i<n;i++){
            // remove the outer window element
            if(!dq.empty() && dq.front()==i-k){
                dq.pop_front();
            }

            // Remove smaller element
            while(!dq.empty() && nums[dq.back()]<nums[i]){
                dq.pop_back();
            }
```

```
  // push current element index
            dq.push_back(i);

            // put the element in the result
            if(i>=k-1){
                result.push_back(nums[dq.front()]);
            }
        }

        return result;
    }
};
```

Deque Implementation using the STL

```
#include<iostream>
#include<deque>
using namespace std;
int main()
{

    deque<int> dq;

    dq.push_back(10); // 10
    dq.push_front(20); // 20 10
    dq.push_back(30); // 20 10 30
    dq.push_front(40); // 40 20 10 30

    cout<< "Deque elements are :";
    for(int x : dq){
        cout<<x<<" ";
    }
    cout<<endl;
```

```
    cout<<dq.at(1);

return 0;
}


/*

push_front(x)
push_back(x)
pop_front()
pop_back()
front()
back()
size()
empty()
clear()
at(index)
begin()
end()


*/
```

Learn coding

THANK YOU