Date....................

TYPES OF PROBLEMS IN SLIDING WINDOW TECHNIQUE

Fixed size window

- Max/Min subarray of size K
- first -ve in every window of size K
- Count occurrence of anagrams
- Max of all subarrays of size K
- Max of min for every size K

Variable Size Window

- Largest/smallest subarray with sum K
- Longest substring with K distinct characters
- Length of longest substring with no repeating characters
- Pick Toy
- Minimum Window Substring

SLIDING WINDOW TECHNIQUE

arr= { 2 , 3 , 5 , 2 , 9 , 7 , 1 }

2   3   5   2   9   7   1

Q. Find the largest sum subarray of fixed size.

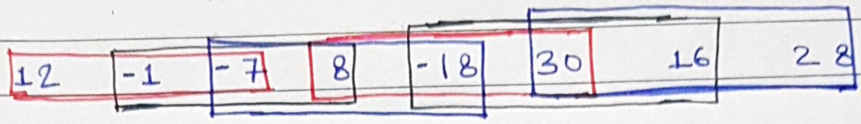
k=3

```
int maximumSumSubarray(vector<int> &v, int k) {
        int p1=0, p2=0, sum=0, tmp=0;

        while(p2 != v.size()) {

            while(p2-p1 != k-1) {
                tmp +=v[p2];
                p2++;
            }

            tmp+= v[p2];
            sum= max(tmp, sum);
            tmp -= v[p1];
            p1++;
            p2++;
    }
    return sum;
```

RUNS TO MAKE A WINDOW OF SIZE 'K'

FIRST NEGATIVE INTEGER IN EVERY WINDOW OF SIZE 'K'

arr =



Window Size = 3

```
void FirstNegativeNumberInWindow(vector<int> &arr, vector<int>&ans,
                                                    int k) {


    int p1=0, p2=0;
    queue <int> q;

    while( p2 != arr.size()) {

        while (p2-p1 != k-1) {
            if (arr[p2] <0)
                q.push(arr[p2]);

            p2++;
        }

        if (arr[p2] <0)
            q.push(arr[p2]);

        if( q.empty())
            ans.push-back (0);
        else
            ans.push-back (q.front());

        if(arr[p1]= q.front())
                q.pop();
        p1++;
        p2++;
    }
```