Google

Q Find number of triplets

i, j, k { indices }

such that

$i < j < k$

&

$arr[i] < arr[j] < arr[k]$

	0	1	2	3	4
arr :	3	4	6	9	2

i	j	k
0	1	2
0	2	3
1	2	3
0	1	3

$\boxed{ans = 4}$

	0	1	2	3	4
arr :	2	6	9	4	10

0	1	2
0	1	4
0	2	4
0	3	4
1	2	4

$\boxed{ans = 5}$

Basic :- consider all possible triplets

$i < j < k$

arr[i] < arr[j]

```
cnt=0;
for( i=0; i<n; i++)
{
    for( j=i+1; j<n; j++)
    {
        if( arr[i] >= arr[j]) continue;
            for(int k = j+1; k<n; k++)
            {
                if( arr[k] > arr[j])
                    cnt++;
            }
    }
}
```

$O(N^3)$

we can't sort over here.

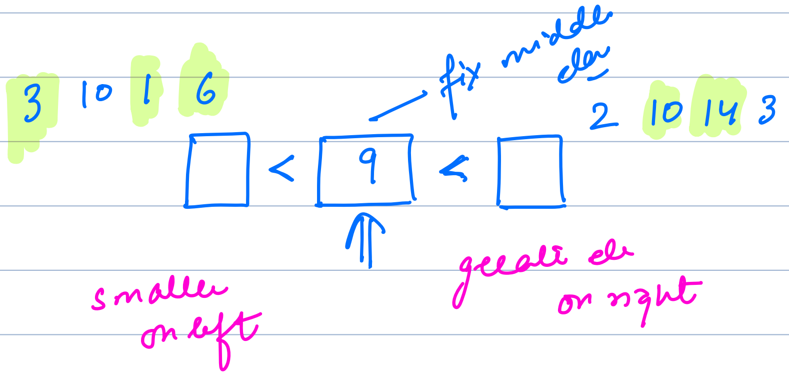
N^3 ✓

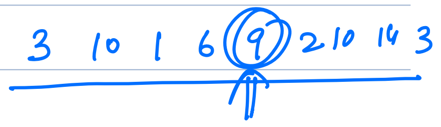
↓

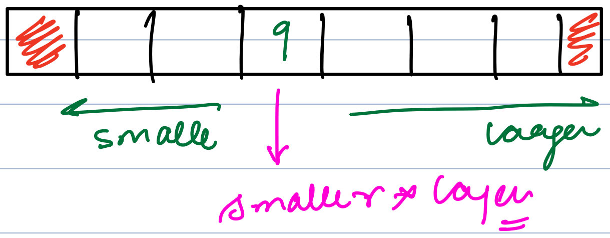
$N^2 \log_2 N$

↓

N^2







0	1	2	3	4	5	6
3	6	9	12	16	11	1
←	↑					

smaller	1	2
larger	4	3
	4	

$\boxed{1 \rightarrow n-2}$ — treating each element as middle

```
for( i=1; i< n-1; i++)
{           // i^th → middle

            ls =0;
            for( j=0; j< i; j++)
            {    if( arr[j] < arr[i])
                          ls++;
            }
            rs =0;
            for( j = i+1; j<n; j++)
            {         ____
            }
            ans += ls*rs;
}
```

$\boxed{T.C: O(N^2)}$

↓

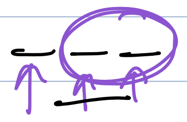
T.C: nlogn

BBST

takeaway - $O(N^3)$ ✓

↓

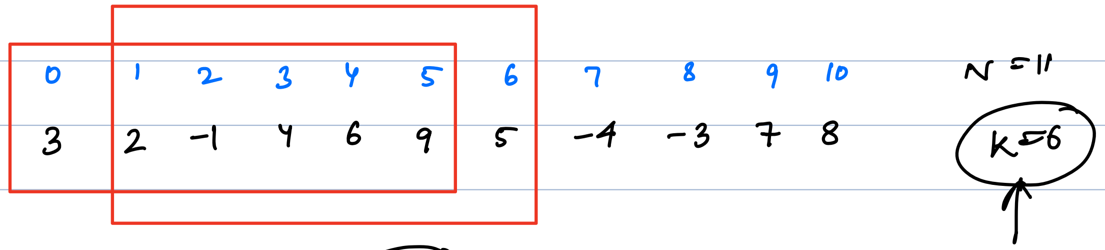
$O(N^2)$



— $O(N)$ time

Q array of size N.

Print (start, end) indices of all subarrays of size K.



0	1	2	3	4	5	6	7	8	9	10
3	2	-1	4	6	9	5	-4	-3	7	8

N = 11

$\boxed{k=6}$

$\boxed{k=6}$	$\boxed{k=3}$	$\boxed{k}$
0 - 5	0 - 2	✓ 0 → k-1
1 - 6	⋮	1 → k
⋮	⋮	2 → k+1
⋮	⋮	3 → k+2
		⋮
5 - 10	8 - 10	✓ $N-k \xrightarrow{k} N-1$
6 sub	9 subarr	

$\boxed{n-k+1}$ ⇒ total count of subarr of size k

$b - a + 1 = k$

$(N-1) - a + 1 = k$

$\boxed{a = N-k}$

```
int s = 0;
int e = k-1;        } start & end pt of first sub of size k
  while ( e<N )
  {
       print (s,e);
         s++; e++;
  }
```

⇒ consider all possible subarr of size k

Q Find the max subarray sum of size k.

0	1	2	3	4	5	6	7	8	9
−3	4	−2	5	3	−2	8	2	−1	4

$k=5$

s	e	sum
0	4	7
1	5	8
2	6	12
3	7	16
4	8	10
5	9	11

```
s=0;
e=k-1;
while(e<N)
{
        //s,e     sum=0;
        for( i=s; i<=e; i++)
            sum+=arr[i];
        ans=max( ans, sum);
}
```

prefix sum

T.C: O(N)
S.C: O(N)

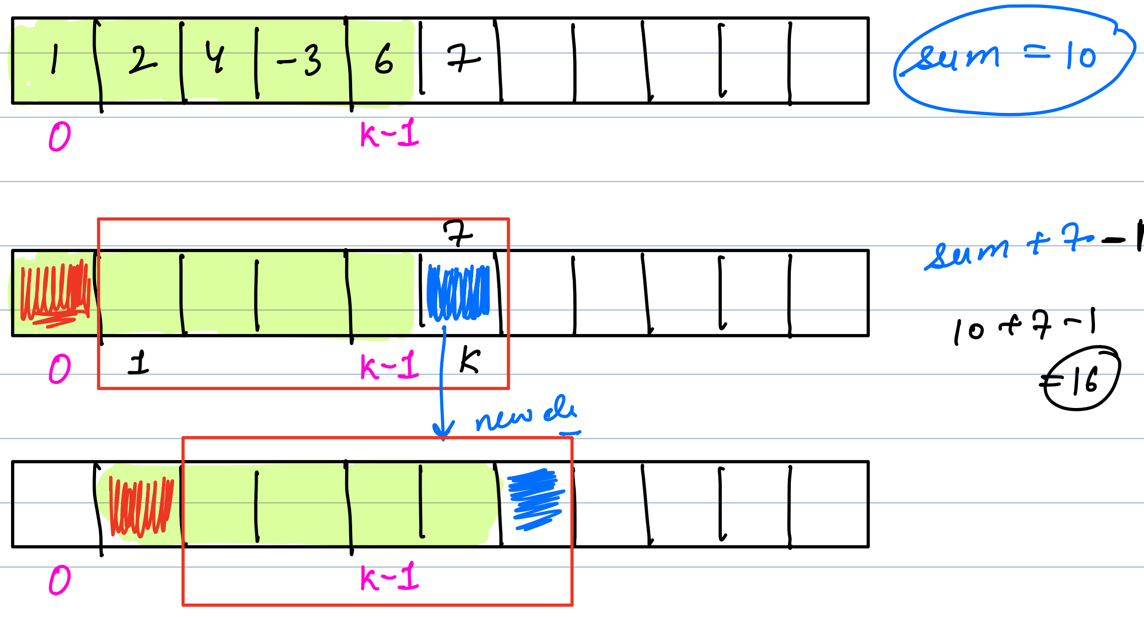
T.C: $(n-k+1) * k$

$k=1$: $(n-1+1)*1$ → $O(n)$

$k=N$: $(\cancel{n-n}+1)*n$ → $O(n)$

$k=n/2$: $(n-n/2+1)*n/2 \approx \frac{n^2}{4}$ → $O(n^2)$

k size
subar



0 ⟶ k-1   iterate & calculate = sum

1 ⟶ K   sum + arr[k] - arr[0] = sum

2 ⟶ k+1   sum + arr[k+1] - arr[1]

⋮

N-K ⟶ N-1   sum + arr[N-1] - arr[N-K-1]

$$\boxed{\text{sum} + \text{arr}[e] - \text{arr}[s-1]}$$

```
sum = 0;    ans = -∞;
    // calculate for first subarray
    for( i=0; i<k; i++)
    {
            sum+=arr[i];
    }
    ans = max(ans, sum);      // ans =sum; ✓
   { s = 1;
     e = k;
     while( e<n)
     {
            sum = sum + arr[e] - arr[s-1];
            ans = max(ans,sum);
            s++;
            e++;
     }
```

K

N-K

~~K~~+N-~~K~~

T.C: O(N)

S.C: O(1)

Takeaway → whenever subarray size is fixed

↓

sliding window

10: 43

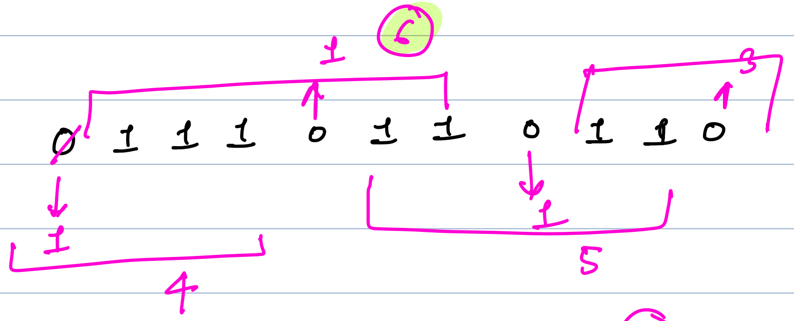
↑

Q// Binary array of size N
↓
0 & 1's

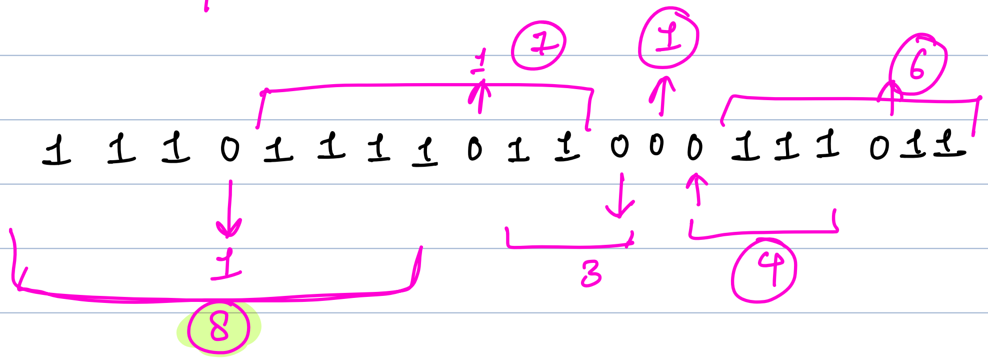
You are allowed to replace atmost one 0 with a 1

length of longest consecutive 1's ?

Ex:



Ex:

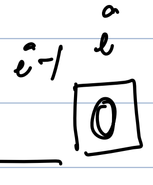


go to every zero $\xrightarrow{\text{com}}$ consecutive 1's ?

↓

length of conse one's on left + length of consecut on right

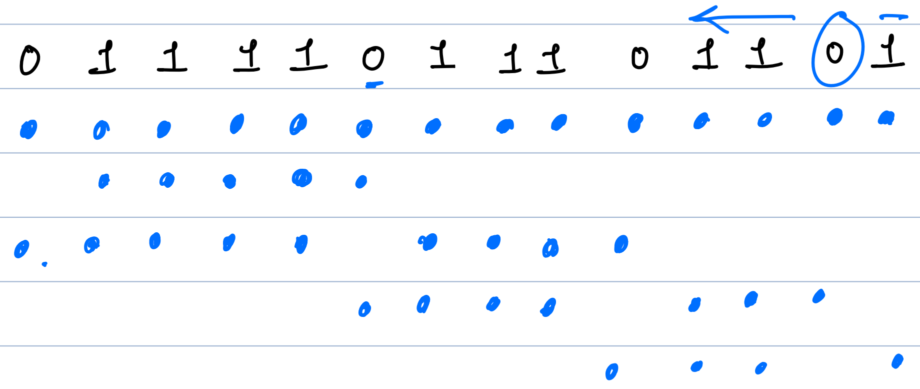
$$\boxed{ls + rs + 1}$$

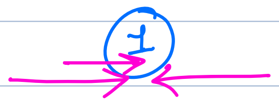


```
ans = 0;

for( int i=0; i<n; i++)
{
        if( arr[i] == 0)
        {
                j=i-1; ls=0;
                while( j>=0 && arr[j] ==1)
                {
                        ls++;
                        j--;
                }
                // calculate rs

                ans = max(ans, ls+rs+1);
        }
}
```

1 1 1 1 1 1

```
if( ans == 0)
        return arr.length();
```

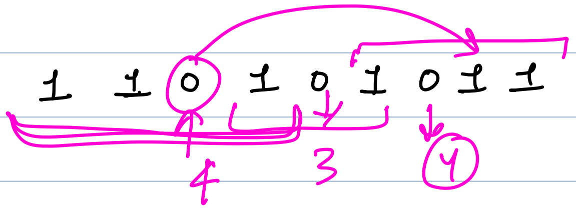




(3N) T.C: O(N)

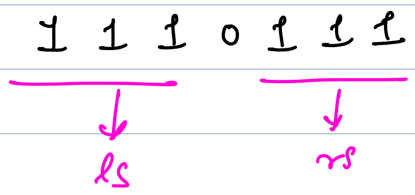
replace 0 with 1

swap atmost one '0' with '1'



ls+rs != total

ans = max(ans, ls+rs+1).



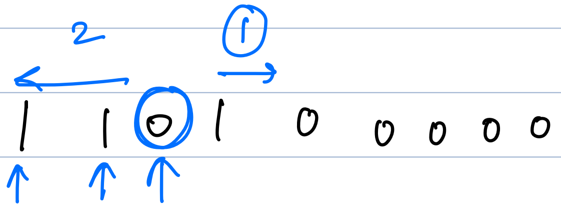
ls+rs == total no of one's

↓ no extra one

↓

ans = max(ans, ls+rs);

extra 1



2
1
1 1 0 1 0 0 0 0 0