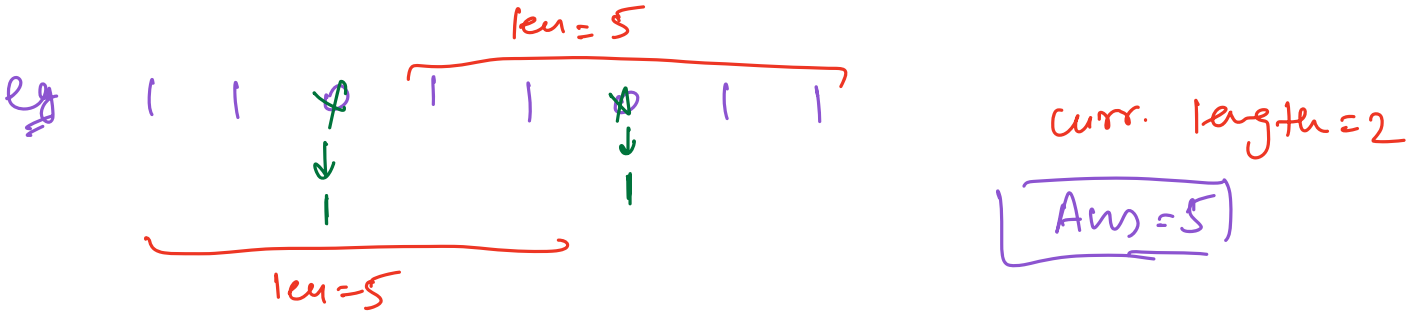
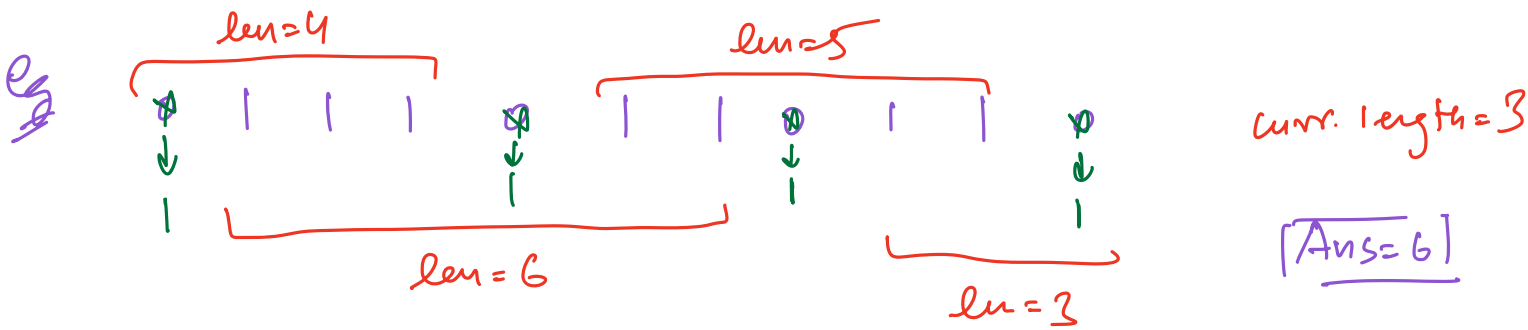
Array: Important Problems

Question 1

Given a binary array (a[i] is either 0 or 1), we can atmost (<=1) replace a single 0 or 1.

find max consecutive 1's you can get in the array.





Idea:

for each zero:

1. count consecutive 1's in the left = l
2. " " " right = r
3. len = l + r + 1

Code

```
int Replace (a[]) {
    n = a.length
    c = 0
    for(i=0; i<n; ++i)
        c += a[i]
```

if a[i] == 1 ⇒ c+1
else ⇒ no change

```
    if (c == n)
        return n

    ans = 0
    for (i=0; i<n; ++i) {
        if (a[i] == 0) {
            l = 0, r = 0
            for (j = i-1; j >= 0; --j) {
                if (a[j] == 1)
                    l++
                else
                    break
            }
            for (j = i+1; j < n; ++j) {
                if (a[j] == 1)
                    r++
                else
                    break
            }
            ans = max(ans, l+r+1)
        }
    }
    return ans }
```

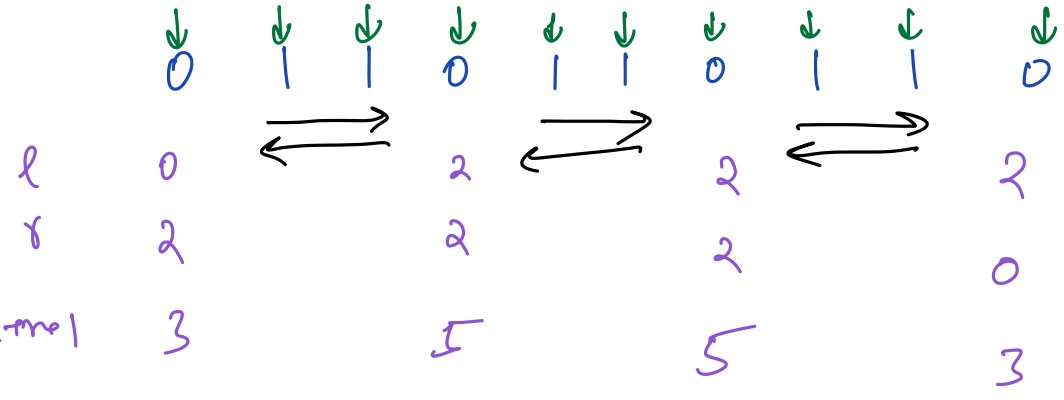
consecutive 1's in the left

consecutive 1's in the right

O(N)

TC: O(N²)

SC: O(1)



max
3N iterations

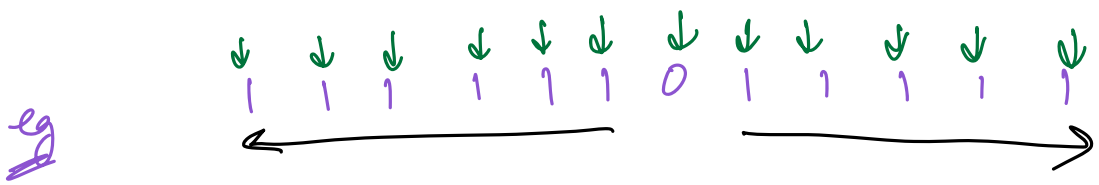
Ans = 5

TC discussion

```
for (i=0; i<3; ++i) {
    for (j=0; j<n; ++j) {
        print(a[j])
    }
}
```

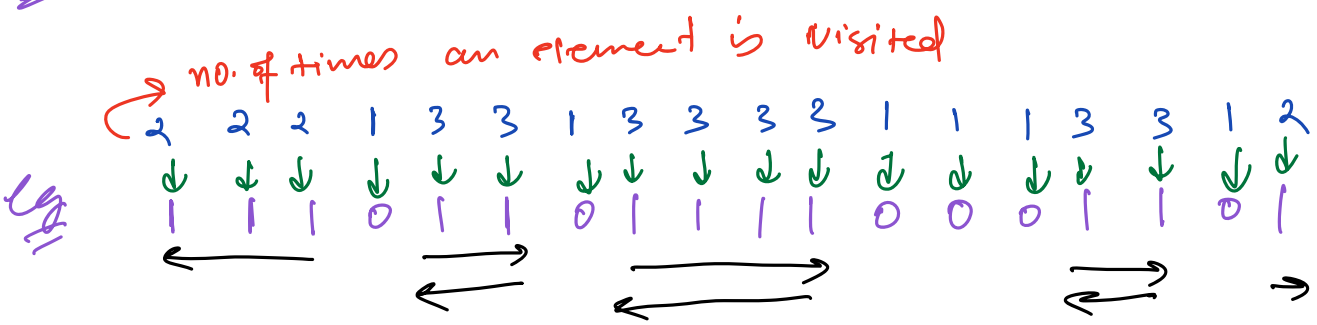
→ 3N iterations

TC: O(N)



max
2N iterations

eg

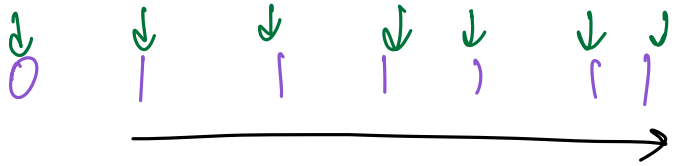


eg

all elements are visited <= 3 times

total iterations <= 3N

TC : O(N)



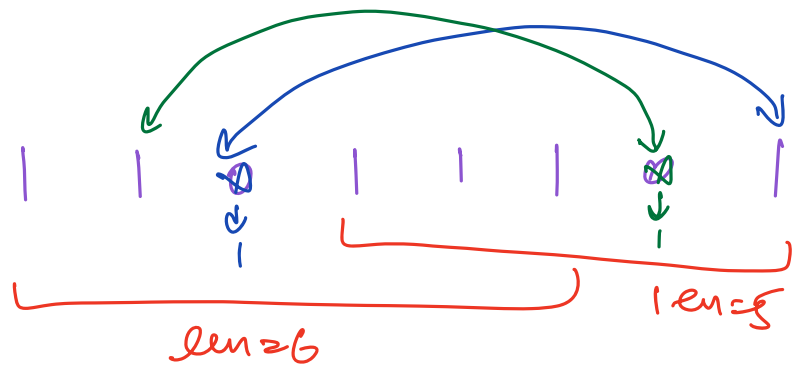
Lesson: If there is a "break" statement in any loop, then calculate TC carefully.

Question 1 - Part 2

Given a binary array, we can at most **swap** single 0 with 1.

find max consecutive 1's.

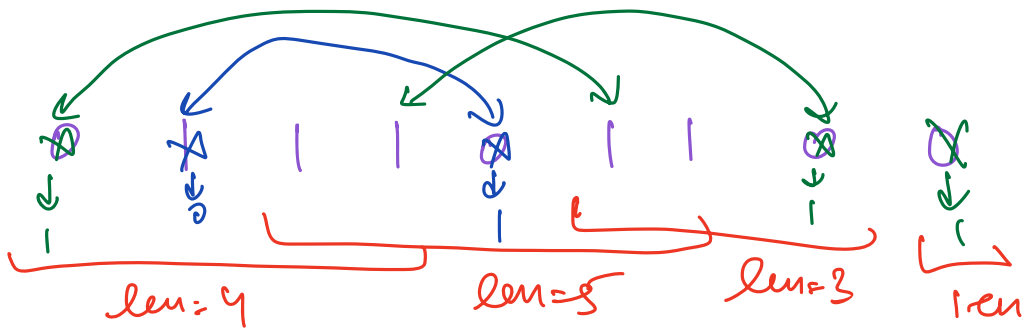
Eg



curr. length = 3

Ans = 6

Eg



curr. length = 3

Ans = 5

Part 2 is different from Part 1 in only 1 case:

your ans can't be greater than total 1's in the array.

Code

```
int Swap( a[]) {
    n = a.length
    c=0
    for(i=0; i<n; ++i)        } if a[i]==1  => c+1
        c+= a[i]                  else      => no change

    if (c==n)
        return n

    ans=0

    for(i=0; i<n; ++i) {
        if(a[i]==0) {
            l=0, r=0
            for(j=i-1; j>=0; --j) {
                if(a[j]==1)
                    l++
                else
                    break
            }
```

consecutive 1's in the left

consecutive 1's in the right

```
for (j = i+1 ; j < n ; ++j) {
    if (a[j] == 1)
        r++
    else
        break
}
ans = max(ans, l + r - 1)
}
}
if (ans > c) {
    ans = c
}
return ans
}
```

→ ans can't be greater than total 1's in array

Question 3

Given a[N] elements, calculate no. of triplets i, j, K such that $i < j < K$ and $a[i] < a[j] < a[K]$

↳ indices of array

Eg A = 2 6 9 4 10
0 1 2 3 4

i < j < K	a[i] < a[j] < a[K]
(0 < 1 < 2)	2 < 6 < 9
(0 < 1 < 4)	2 < 6 < 10
(0 < 3 < 4)	2 < 4 < 10
(0 < 2 < 4)	2 < 9 < 10
(1 < 2 < 4)	6 < 9 < 10

Ans = 5

Code

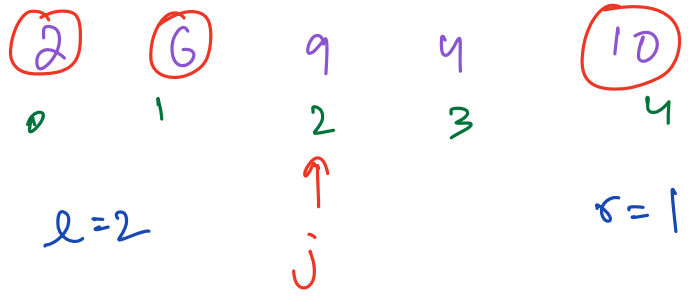
```
int triplets(a[]) {
    n = a.length
    ans = 0
    for(i=0; i<n; ++i) {
        for(j=i+1; j<n; ++j) {
            for(K=j+1; K<n; ++j) {
                // triplet (i,j,K)  i<j<K
                if(a[i]<a[j] && a[j]<a[K])
                    ans++
            }
        }
    }
    return ans
}
```

TC : O(N^3)

SC : O(1)

OPTIMIZE

Hint: In how many triplets, index 2 is the middle element?



(0, 2, 4)

(1, 2, 4)

⇒ 2 × 1 = 2

Idea: for every element a[i]:

→ get no. of elements less than a[i] in left = l

→ get no. of elements greater than a[i] in right = r

→ no. of triplets with a[i] as middle = l × r

Code

```
int tripleks(a[]) {
    n = a.length
    ans = 0
    for (j=1; j<n; ++j) {  → middle index
        l=0, r=0
        for (i=j-1; i>=0; --i) {  → first index
            if (a[i] < a[j])
                ++l
        }
        for (k=j+1; k<n; ++k) {  → last index
            if (a[j] < a[k])
                ++r
        }
        count = l*r
        ans += count
    }
    return ans
}
```

TC: $O(N^2)$

SC: $O(1)$

2 5 1 9 8

j

l=0 r=2

(2,3,5)
(2,3,4)
(1,3,5)
(1,3,4)
(0,3,5)
(0,3,4)

(1,2,5)
(1,2,4)
(1,2,3)

eg

	4	1	2	6	9	7
	0	1	2	3	4	5
l	0	0	1	3	4	4
r	3	4	3	2	0	0
l+r	0	0	3	6	0	0

ans = 3+6
= 9

$O(N^2)$ ⟶

$O(N \log N)$
- Balanced Binary Search Tree
- Segment trees

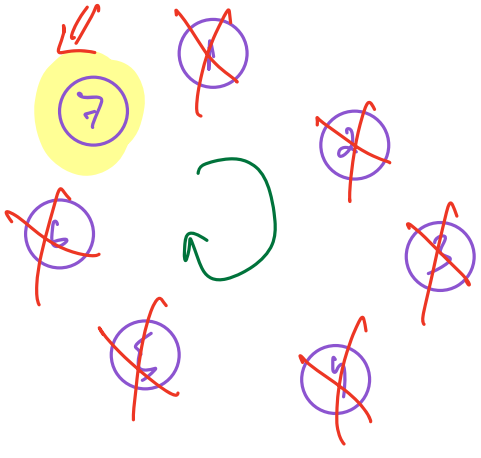
Don't Worry

Question 4 - Josephus Problem

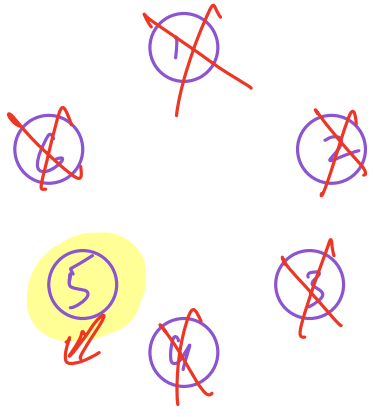
N people are standing in circle. Person 1 has a knife. He kills next person in clockwise direction and passes on the knife to the next person in clockwise direction.

Repeat this until only 1 person is alive. find the last man standing?

eg N = 7



N = 6

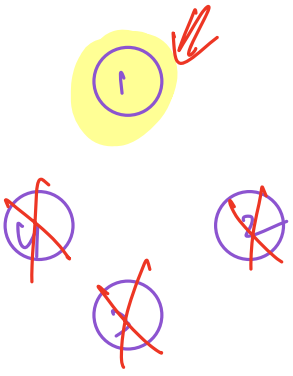


Observations:

1. last odd will be prime
   - if N is odd → N
   - if N is even → N-1
2. largest prime <= N

if N=4 all observations are wrong

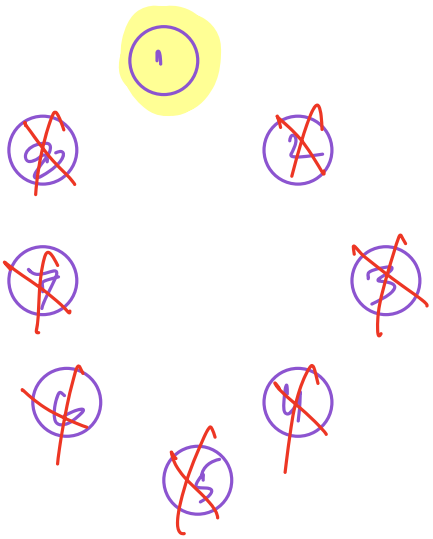
N=4



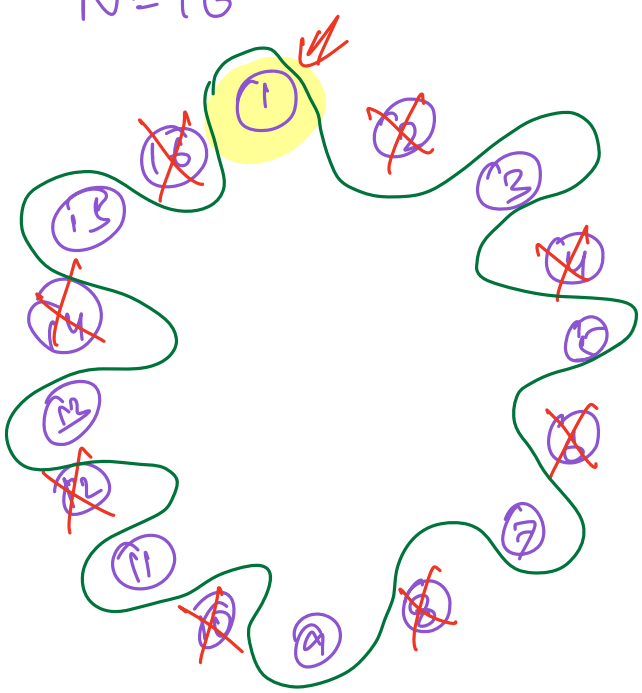
N=1

N=2

N=8



N=16

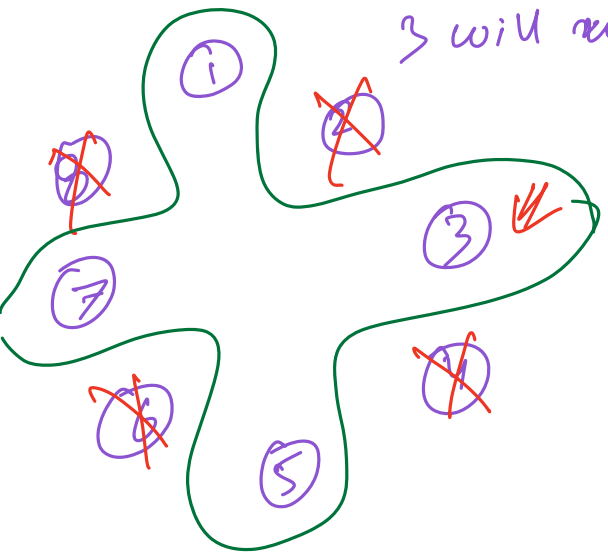


Observation

if N is power of 2

if 1 starts → 1 wins

N=8 (if we start from 3, 3 will win)



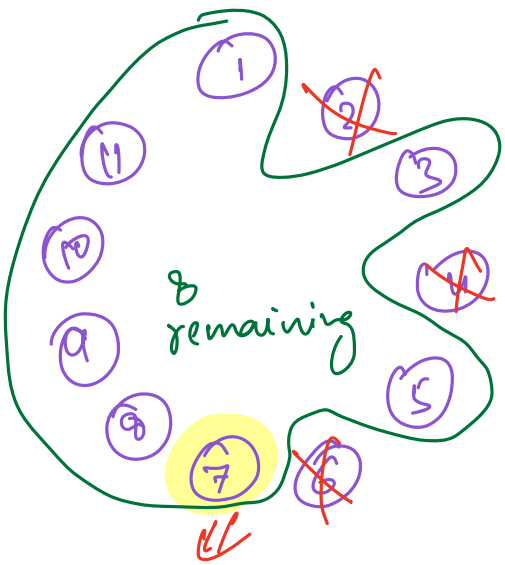
Observation

if N is power of 2,

whoever starts, wins the game.

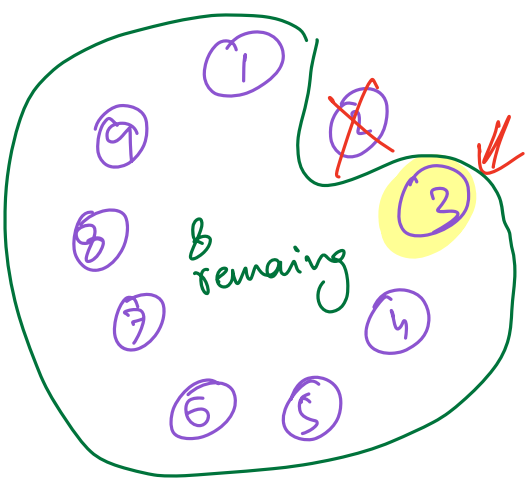
N=11 $\xrightarrow{3 \text{ kill}}$ 8

$1 + 3 \times 2 = 7$



N=9 $\xrightarrow{1 \text{ kill}}$ 8

$1 + 2 \times 1 = 3$



for N = 100 ?

$2^6 = 64$ $2^7 = 128$

100 - 64 = Kills $\Rightarrow$ Kills = 36

ans = 1 + 2 × kills

= 1 + 2 × 36 = 73

Code

N = 100 $\log_2 N$ $\log_2 100$

6.xxx

Kills = N - $2^{\lfloor \log_2 N \rfloor}$ → floor value

ans = 1 + 2 × kills

TC : O(1)

SC : O(1)