Today's Content

Interview Problems

→ max consecutive 1s by

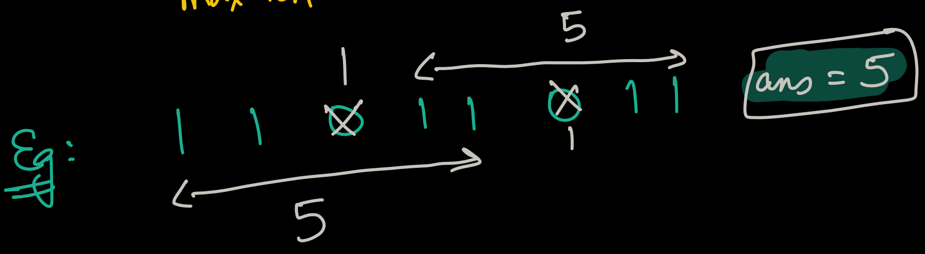
(a) Atmost 1 replace

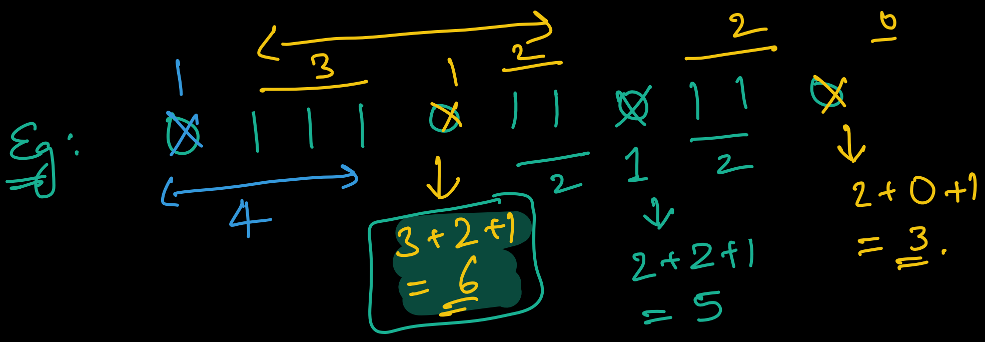
(b) Atmost 1 swap

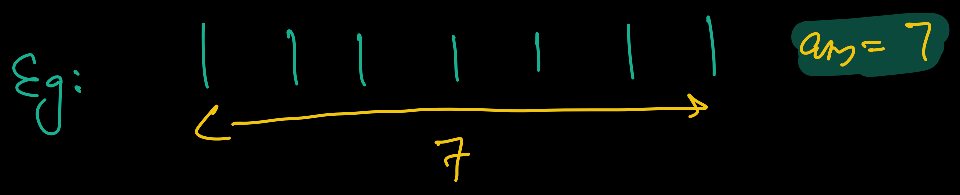
→ Count triplets ← GS

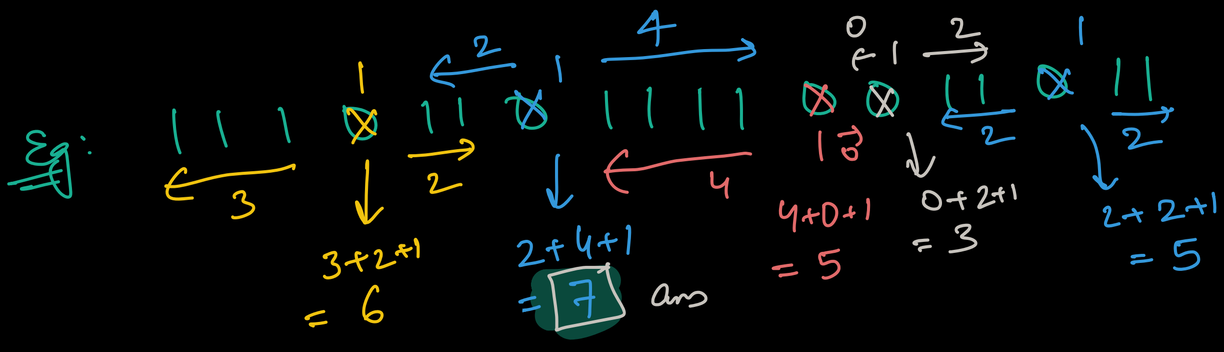
→ Josepheus Problem

Q1: Given a binary a[] {0,1}. We can atmost replace a single 0 with 1. Find max len consecutive 1s we can get in a[].









idea: Brute force Soln

→ for every zero encountered

(a) Count no. of 1s on the left = l

(b) Count no. of 1s on the right = r

(c) if(l+r+1 > ans) { ans = l+r+1 }

Edge case: if all are 1s
↳ return n

Pseudocode:

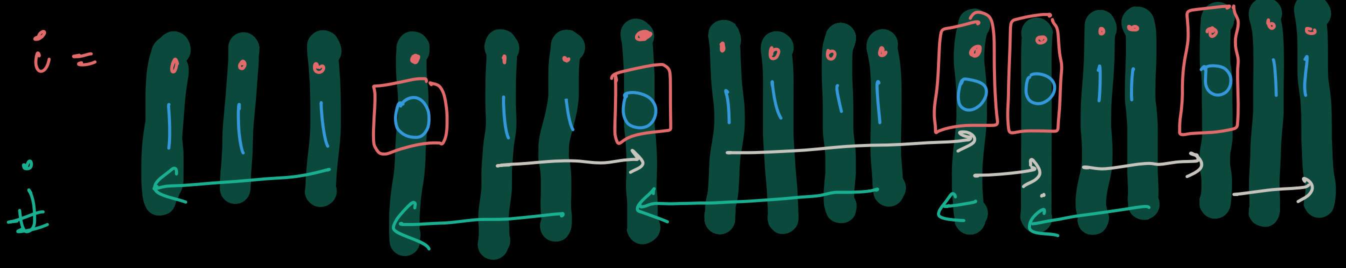
```
int replace (int a[], int n) {
        int c=0
        for( i=0; i<n; i++) { c = c+a[i] }
        if( c==n) {
            return n          // Edge case 1
        }
        if(c==0) {
            return 1          // Edge case 2
        }
        ans=0
        for(i=0; i<n; i++) {    ← O(n)
            if(a[i]==0) {
                int 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}
                }
                int cat = l+r+1
                if(cnt > ans) { ans = cat }
            }
        }
        return ans
}
```

TC: ~~$O(n^2)$~~ → $O(n)$

SC: $O(1)$ → $O(1)$

CTC: 12L   12/12 = $\boxed{12}$ pm

≈ less



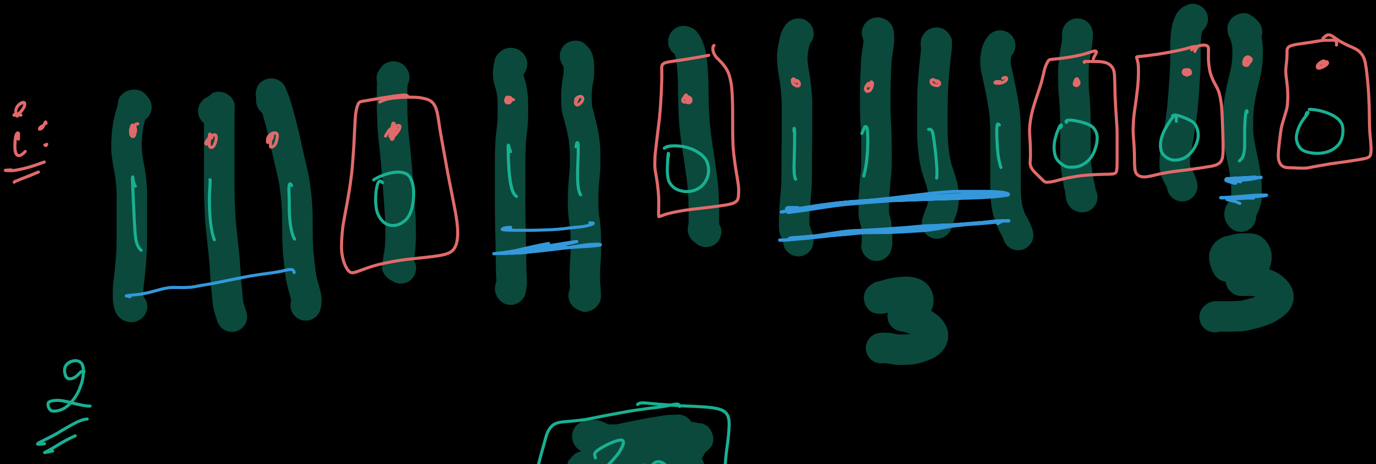
iter : 3times

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

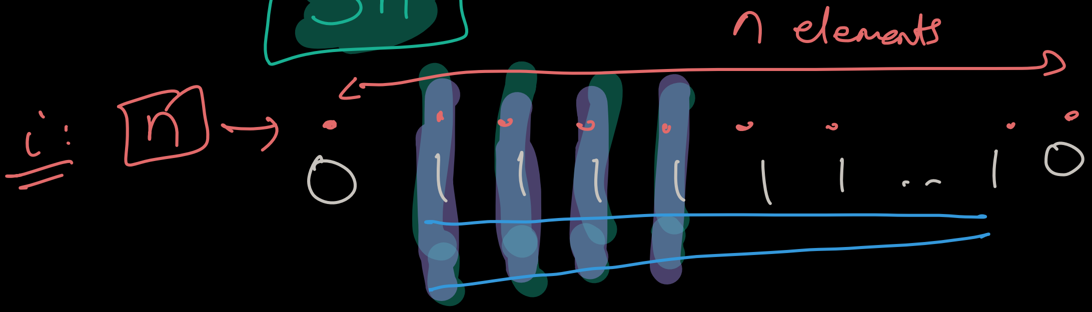
TC: 3n iter
→ O(n)

⊛ Learning: Nested loop with break condition → Pen + Paper

↳ Be careful.



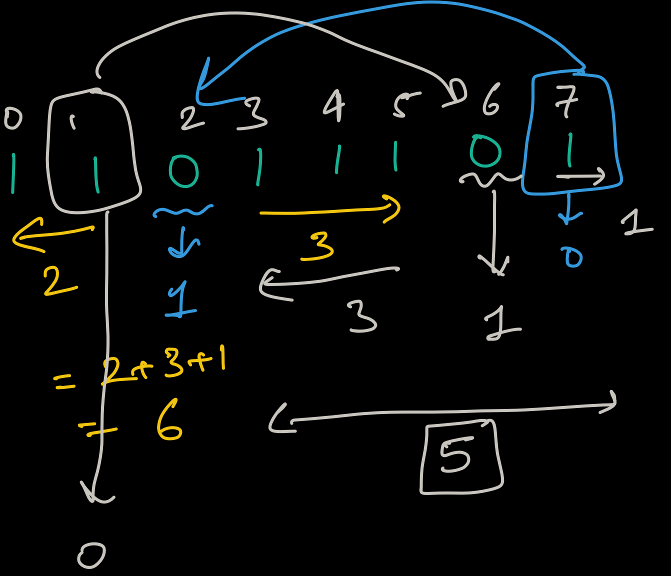
$\boxed{3n}$



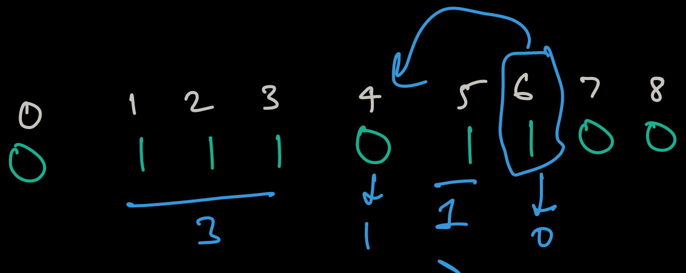
Q2: Given a binary a[ ]. Find max no. of consecutive 1s we can get by atmost 1 swap?

↳ {swap with data in array}

Eg:

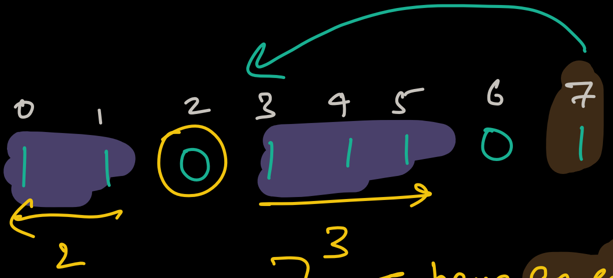


Try: Eg:



3+1+1 = 5

Eg1:

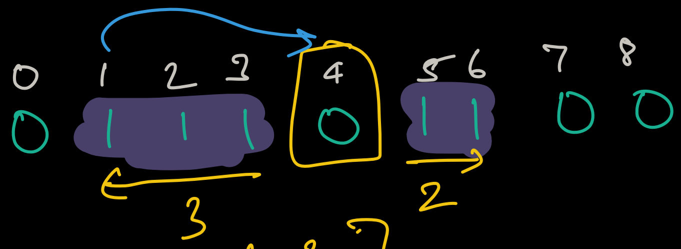


l=2
r=3

I have an extra 1

= l+r+1

Eg2:



l=3
r=2

ans = l+r

C

l+r < C

idea:

```
if extra 1:
    cnt = l+r+1
else
    cnt = l+r
```

Pseudocode:

```
int replace (int a[], int n) {
        int c=0
        for( i=0; i<n; i++) { c= c+a[i] }
        if( c==n) {
                return n    //Edge case 1
        }
        if(c==0) {
                return 0    // Edge case 2
        }
        ans=0
        for(i=0; i<n; i++) {
                if(a[i]==0) {
                        int 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}
                        }
                        cnt= l+r
                        if( l+r < c) {   // Extra 1
                                cnt++
                        }
                        if(ans < cnt) { ans= cnt }
                }
        }
        return ans
}
```

TC: O(n)

SC: O(1)

$\boxed{\text{Break: 8:25 am}}$

Q3: No. of triplets.

Given a[N]. Count no. of triplets i, j, k st. $i < j < k$ && $a[i] < a[j] < a[k]$

indices

Eg: $a[5]$: {2 6 9 4 10}

0	1	2	3	4
2	6	9	4	10

i <	j <	k	a[i] <	a[j] <	a[k]
0	1	2	2	6	9
0	1	4	2	6	10
0	2	4	2	9	10
1	2	4	6	9	10
0	3	4	2	4	10

$\boxed{\text{Ans} = 5}$

Eg: $a[6]$ = {4 1 2 6 9 7}

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

i <	j <	k	a[i] <	a[j] <	a[k]
0	3	4	4	6	9
0	3	5	4	6	7
1	2	3	1	2	6
1	2	4	1	2	9
1	2	5	1	2	7
1	3	4	1	6	9
1	3	5	1	6	7
2	3	4	2	6	9
2	3	5	2	6	7

$\boxed{9}$

Bruteforce soln:

↳ Check all triplets

For every triplet i, j, k, check if it satisfies the cond^n $a[i] < a[j] < a[k]$

↓

ans++

```
int countTriplets(int a[], int n) {
        c = 0
        for(i = 0; i < n; i++) {  // i: 0 → n
            for(j = i+1; j < n; j++) {  j: i+1 → n  [j > i]
                for(k = j+1; k < n; k++) {  k: j+1 → n  [k > j]
                    if(a[i] < a[j] &&
                       a[j] < a[k]) {
                        c++
                    }
                }
            }
        }
        return c;
}
```

TC: $O(N^3)$

SC: $O(1)$

Optimise:

$$a[6] = \{ \overset{0}{4}\ \overset{1}{1}\ \overset{2}{2}\ \boxed{\overset{3}{6}}\ \overset{4}{9}\ \overset{5}{7} \}$$

Hint: In how many triplets is idx 3, mid element.

(left) i	j = 3	k (right)
4	6	9
1		7
2		
3		2

$3 \times 2 = \boxed{6}$

(left) → elements lesser than me $a_i < a_j$

(right) → elements greater than me $a_k > a_j$

	0	1	2	3	4	5	
Eg: a[6]:	{4	1	2	6	9	7}	
l =	0	0	1	3	4	4	
r =	3	4	3	2	0	0	
cnt(l*r)	0	0	3	6	0	0	= 9 Sum

#idea: for every element a[i]:

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

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

→ cnt = l*r

→ ans = ans + cnt

```
int countTriplets(int a[], int n) {
        ans = 0
        for(i=0; i<n; i++) {
            l=0, r=0
            //Check left for elements less than a[i]
            for(j=i-1; j>=0; j--) {
                    if(a[j] < a[i]) { l++ }
            }
            // Check on right for elements more than a[i]
            for(j=i+1; j<n; j++) {
                    if(a[i] < a[j]) { r++ }
            }
            cnt = l*r
            ans = ans+cnt
        }
        return ans
}
```

TC: O(N^2)

SC: O(1)

$O(N^3) \to O(N^2)$ → $O(N \log N)$: Balanced Binary Search Tree

→ $O(N \log N)$: Segment Trees

Qn: Josephus Problem {Squid Game in Prog.}

N people standing in a circle (clockwise).

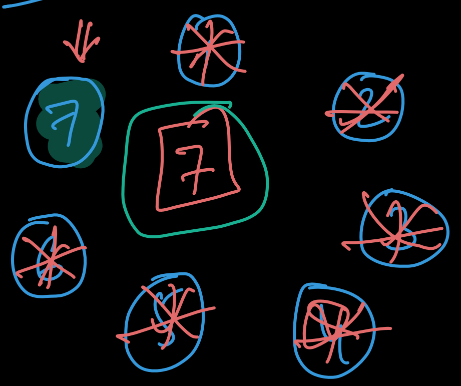
Person 1 has a knife. He kills the next person in clockwise dir.

he passes the knife to surviving person in clockwise dir.

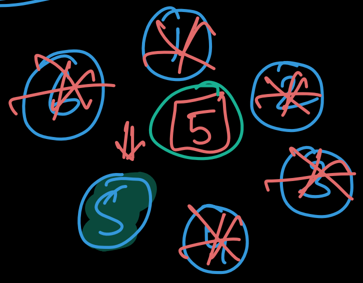
Repeat this process until 1 survives.

Find last man standing.

$N=7 \xrightarrow[1+(3)(2) = \boxed{7}]{3 \text{kills}} 4$



$N=6 \xrightarrow[1+(2)(2) = 5]{2 \text{kills}} 4$

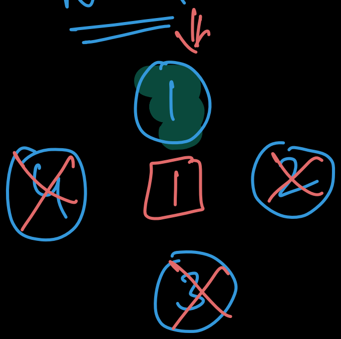


Obs1: ~~if n=even, N-1 survives~~
~~if n=odd, N survives~~

or

Obs2: ~~largest prime <=N ?~~

N= 4



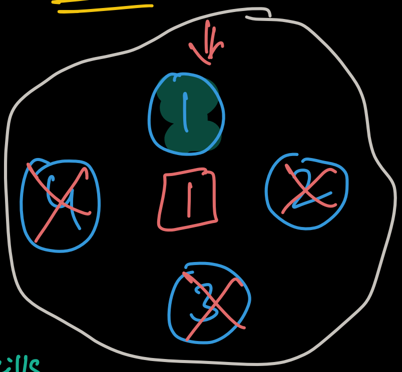
N=1

1 Survives

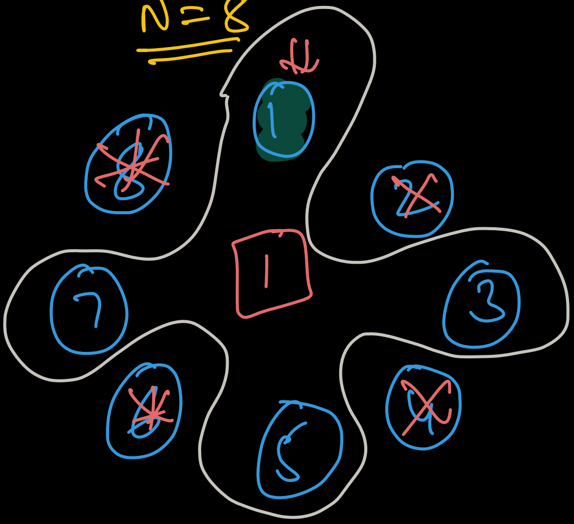
N= 2

1 Survives

N= 4

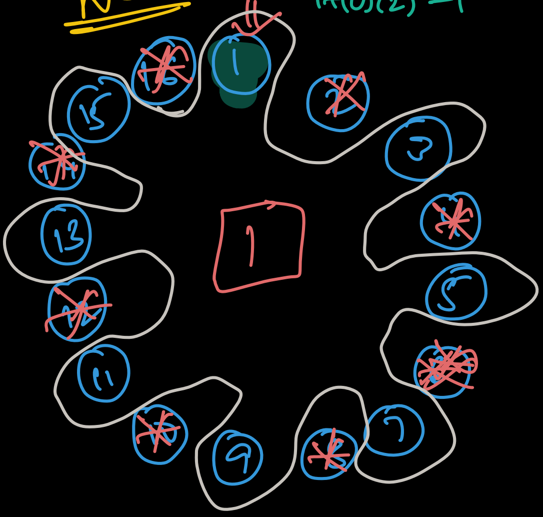


N=8



N=16 $\xrightarrow{0 \text{ kills}}$ 16

$1+(0)(2)=1$

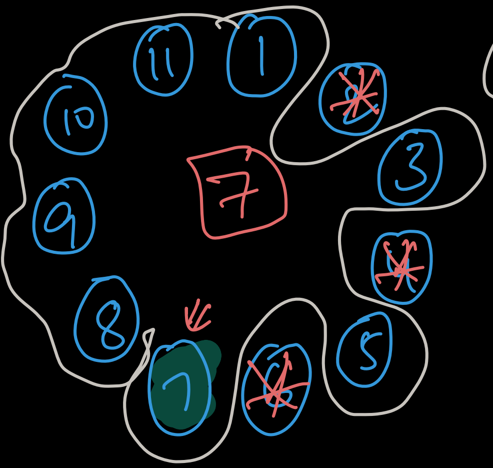


8 surviving

if N is pow of 2 , whosoever has the knife wins the game

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

$1+(3)(2) = \boxed{7}$

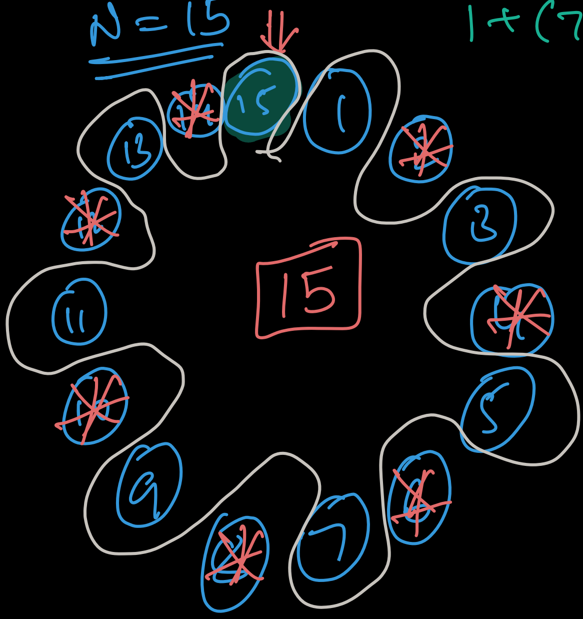


8 alive

→ pow of 2

$$N=15 \xrightarrow{7\text{ kills}} 8$$

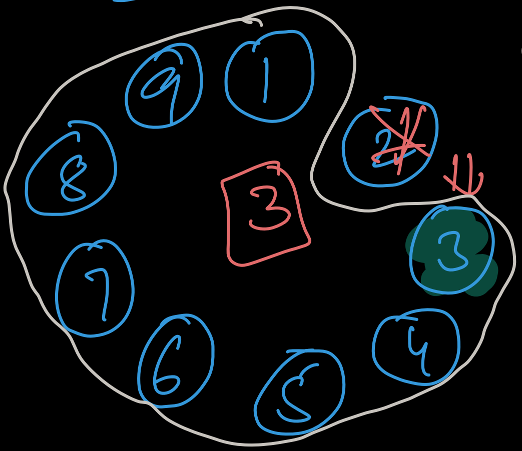
$$1+(7)(2) = \boxed{15}$$



8 alive

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

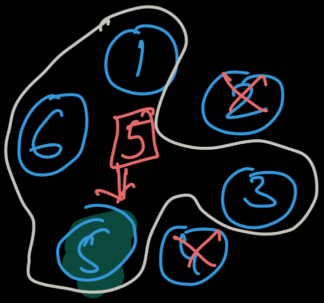
$$1+(1)(2) = \boxed{3}$$



8 alive

$$N=6 \xrightarrow{2\text{ kills}} 4$$

$$1+(2)(2) = \underline{5}$$



$$\boxed{N=100}$$

$$100 \xrightarrow{36\text{ kills}} 64$$

$$1+(36)(2)$$

$$= \boxed{73}$$

$$\boxed{N=1000}$$

$$\underline{1000} \xrightarrow{488\text{ kills}} 512$$

$$1+(488)(2)$$

$$\Rightarrow 977$$

Adv. module

Josephus Problem - 2

→ can start from anywhere

→ can skip K people at a time