Q → [Number of Open Doors]

There are N doors numbered from 1 to N.

All doors are initially closed. ←

A person changes the state of the door in the following manner → (open → close, close → open)

1 2 3 4 ... N

2 4 6 8 ...

3 6 9 12 ...

4 8 12 16 ...

⋮

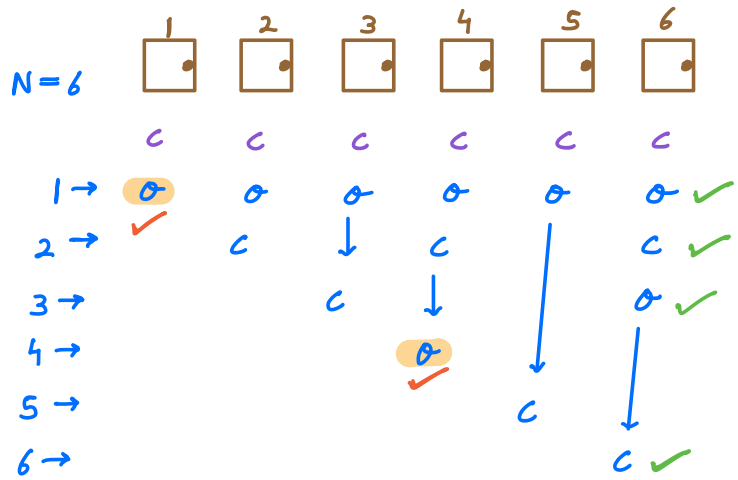
N

c → closed

o → open

$N = 10^{18}$

Find the count of open doors at the end.



Ans = 2

Observations → 1) 6 → {1, 2, 3, 6}

If y is a factor of x ⇒ x will flip at y^{th} iteration.

time a door is flipped → # factors of door number

times door number 10 will flip → 4

{1, 2, 5, 10}

2) closed → open → closed → open → closed → open ...
(1) (2) (3) (4) (5)

If # flips is odd ⇒ door is open in the end

x^{th} door is open if count of factors of x is odd.

13 → 1 * 13 → {1, 13}

14 → 1 * 14, 2 * 7 → {1, 2, 7, 14}

15 → 1 * 15, 3 * 5 → {1, 3, 5, 15}

16 → 1 * 16 ✓, 2 * 8, 4 * 4 → {1, 2, 4, 8, 16} ✓

What type of numbers have odd factors? → perfect squares

20 → 1 * 20, 2 * 10, 4 * 5

25 → 1 * 25 ✓ 5 * 5

36 → 1 * 36 ✓, 2 * 18, 3 * 12, 4 * 9, 6 * 6

30 → 1 * 30, 2 * 15, 3 * 10, 5 * 6

1, 4, 9, 16, 25 ...

Ans = # of perfect squares from 1 to N

N = 20 → $\{1^2, 2^2, 3^2, 4^2\}$ Ans = 4

N = 50 → $\{1^2, 2^2, 3^2, 4^2, 5^2, 6^2, 7^2\}$ Ans = 7

max x s.t $x^2 <= N$

$\Rightarrow x <= \sqrt{N}$ $\Rightarrow x = \text{floor}(\sqrt{N})$

largest integer <= $\sqrt{N}$ N = 100 → Ans = 10

N = 50 → $\sqrt{N} = 7.1...$ → 7 Ans = floor($\sqrt{N}$)

$N = 10^{18}$ → Ans = 10^9 $\sqrt{10^x} \rightarrow 10^{x/2}$

Q → [Josephus Problem]

paypal Altimetrik

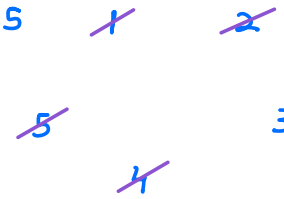
There are N people standing in a circle.
Person (1) kills the immediate clockwise neighbour & pass the knife to the next person standing in circle.
Find last person alive.

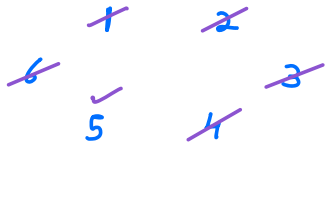
N = 1 (1) Ans = 1

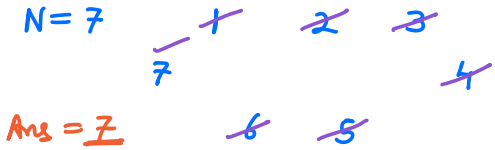
N = 2 1 ~~2~~ Ans = 1

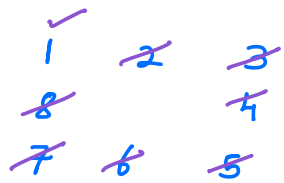
N = 3 ~~1~~ ~~2~~ 3 ✓ Ans = 3

N = 4 1 ✓ ~~2~~ ~~3~~ ~~4~~ Ans = 1

N = 5  Ans = 3

N = 6  Ans = 5

N = 7  Ans = 7

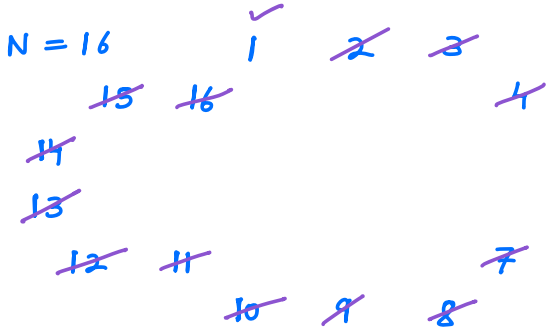
N = 8  Ans = 1

can ever position ever be the answer? → No

N →	1	2	3	4	5	6	7	8	9	10
Ans →	1	1	3	1	3	5	7	1	3	5

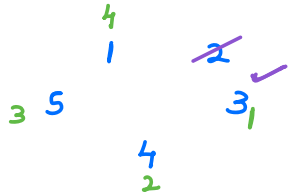
(2^0, 2^1, 2^2, 2^3)

First person is alive if N → power of 2 2^K (for any $K >= 0$)

N = 16  Ans = 1

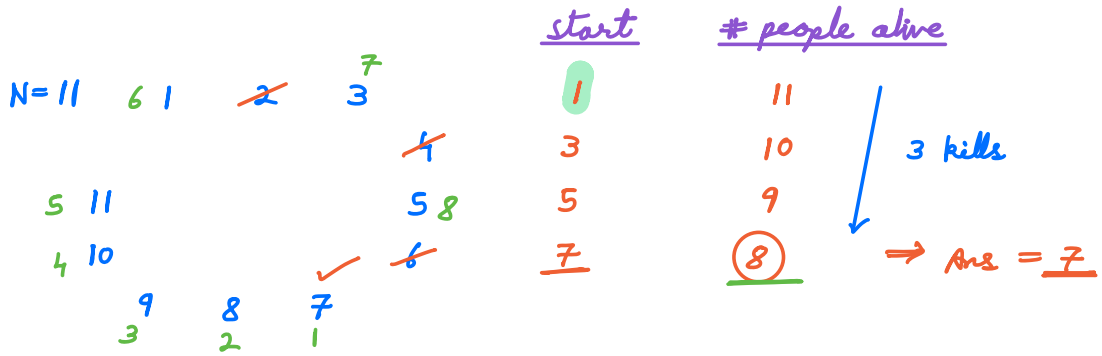
If N = 16 & the person 7 starts the game, who will win? 7

> If # alive people = 2^K (for any K)
> the person starting will be alive in the end.

N = 5 

start	# people alive
1	5
3	(4)

⇒ Ans = 3



For any N, what is the # kill to reach alive people as power of 2 → $N - 2^x$

2^x → largest power of 2 <= N

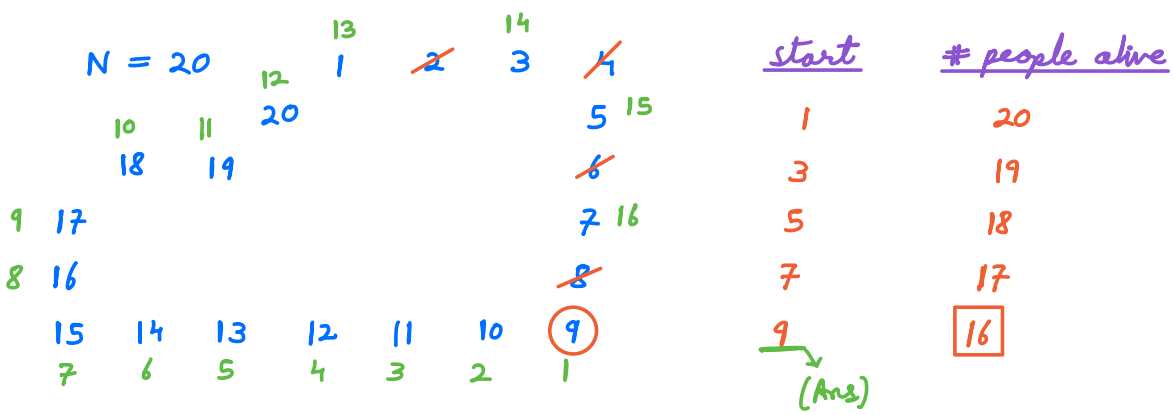
N = 10 $\log_2(10) = 3.14$ → $2^3 <= 10$

N → largest power of 2 <= N → floor($\log_2$(N)) = 2

N = 50 floor($\log_2$(N)) = floor(5.64) = 5 → $2^5 = 32$

Ans = 1 + 2 * (# kill) = 1 + 2 * ($N - 2^x$) (start)

x = floor($\log_2$(N)) ✓



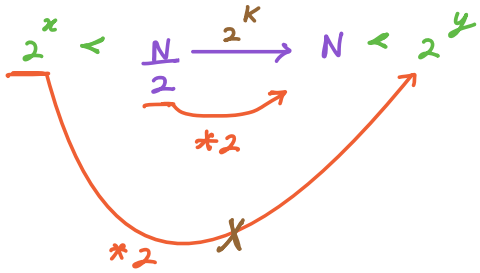
$\log_2 x = y \Rightarrow 2^y = x$

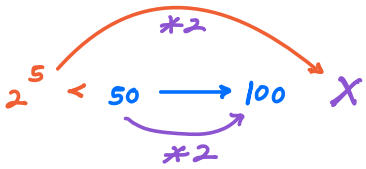
10:45 PM

N → 10000 # kills in 1 iteration = 5000

N # kills in 1 iteration = N/2

100 ——— 50





$x < y$

$2x \neq 2y$

Q → Given an integer array A, return if there exists a majority element i.e element with frequency > $\frac{N}{2}$

if there is no such element, return -1. SC = O(1)

A = [1 6 1 1 2 1] freq(1) > $\frac{6}{2}$ Ans = 1

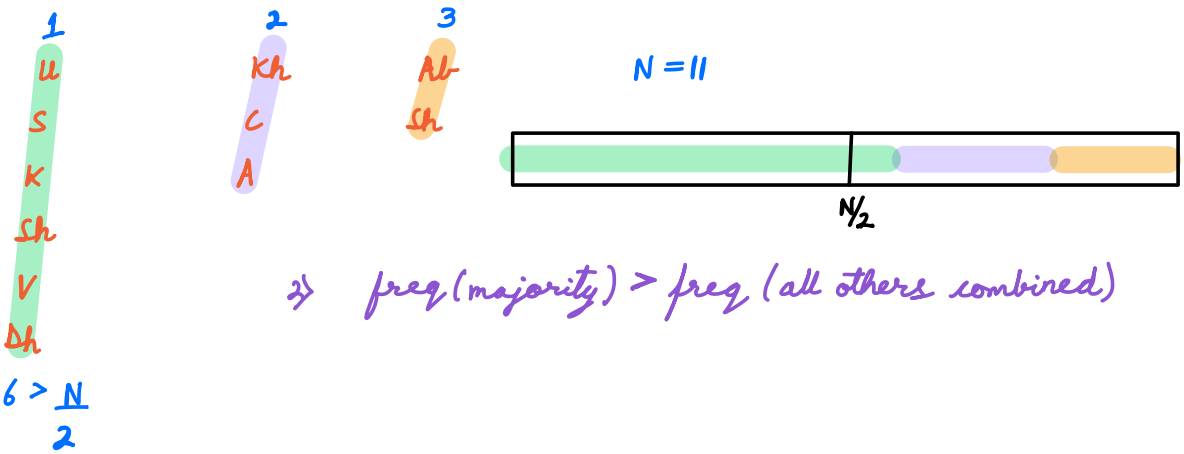
A = [1 6 2 1 6 2] Ans = -1

A = [3 4 3 6 1 3 2 3 3 5 3] freq(3) > $\frac{11}{2}$ Ans = 3

Bruteforce → ∀ i check if freq(A[i]) > N/2 by travelling array.

TC = $O(N^2)$ SC = O(1)

observation → 1) At max there can be one majority element.



$6 > \frac{N}{2}$

2) freq(majority) > freq(all others combined)

3) 2 people leave at once & 2 people are of different group.

1	2	3
~~U~~	~~Kh~~	~~Ab~~
S	C	Sh
K	A	
Sh		
V		
Dh		

$5 > \frac{N}{2}$

1	2	3
~~U~~	Kh	~~Ab~~
S	C	Sh
K	A	
Sh		
V		
Dh		

$5 > \frac{N}{2}$

1	2	3
U	~~Kh~~	~~Ab~~
S	C	Sh
K	A	
Sh		
V		
Dh		

$6 > \frac{N}{2}$

N

11

$11 - 2 = 9$

freq(majority) > freq(all others combined)

−1 −1

OR

freq(majority) > freq(all others combined)

−2

↓ all same

↓ can have different groups

✓

Moore's Voting Algorithm → Removing two distinct elements does not change the majority element.

	0	1	2	3	4	5	6	7	8	9	10
A = [	3	4	3	6	1	3	2	3	3	5	3]

N = 11

→

ans = ~~3~~ ~~3~~ ~~4~~ ~~2~~ ~~3~~ (3) ✓

freq = ~~1~~ ~~1~~ ~~1~~ ~~1~~ ~~1~~ 1

	0	1	2	3	4	5	6	7	8	9	10	11
A = [	3	3	4	4	4	4	3	3	3	5	3	3]

N = 12

ans = ~~3~~ ~~4~~ ~~3~~ (3) ✓

freq = ~~1~~ ~~2~~ ~~1~~ ~~1~~ ~~2~~ ~~1~~ ~~1~~ ~~1~~ 2

A = [4 6 5 3 4 5 6 4 4] (0 1 2 3 4 5 6 7 8) N = 9

ans → ~~4~~ ~~5~~ ~~4~~ ~~6~~ (4) x Ans = -1 freq(4) = 4 ≠ N/2

freq → ~~1~~ ~~1~~ ~~1~~ ~~1~~ 1

A = [1 2 3 4 1 1] (0 1 2 3 4 5)

(1) x freq(1) ≠ N/2

f = 2

```
ans = A[0]
freq = 1
for i → 1 to N-1
    if(freq == 0)
        ans = A[i]
        freq = 1
    else if( A[i] == ans)
        freq += 1
    else
        freq -= 1
if(freq == 0)
    return -1
cnt = 0
for i → 0 to N-1
    if( A[i] == ans)
        cnt += 1
if( cnt > N/2)
    return ans
else
    return -1
```

TC = O(N)

SC = O(1)