Q. There are N doors, Infront of each and every door, there is a person.

All doors are closed.

N = 1	1	2	3	4	5	...	N
N = 2	2	4	6	8	10	....	N
N = 3	3	6	9	12	15	....	N
i	i	2i	3i	4i	5i	....	

toggle the state

which all doors will be open at last?

N = 10

P1	1	2	3	4	5	6	7	8	9	10
P2	1	2	3	4	5	6	7	8	9	10
P3	1	2	3	4	5	6	7	8	9	10
P4	1	2	3	4	5	6	7	8	9	10
final ⇒	1	2	3	4	5	6	7	8	9	10

1, 4, 9

N=5 | 1 | 2 | 3 | 4 | 5

N=5 — 5 doors, 5 person

maintain the state of each door

bool arr[N+1] = 1 → open, 0 → closed

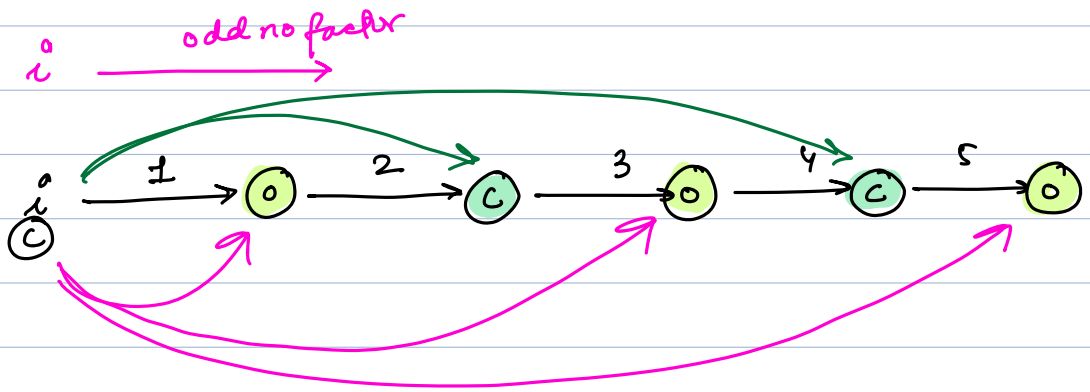
```
for( i=1; i<=N; i++)
{
    for( j = i; j<=N; j+=i)
    {
        state[j] = !state[j]
    }
}
```

T.C: $O(N^2)$

9 :- 1, 3, 9

141 :- 1, 2, 7, 14

i → factors of i



open doors → odd factors

factors occur in pairs — odd ?

24

1 * 24
2 * 12
3 * 8
4 * 6

16

1 * 16
2 * 8
4 * 4 — 1 factor

only perfect square doors will open!

1*1 → 42 $\sqrt{42} = 6.\ldots$
2*2 →
3*3 →
4*4 →
5*5 →
6*6 →
7*7 ? ✗

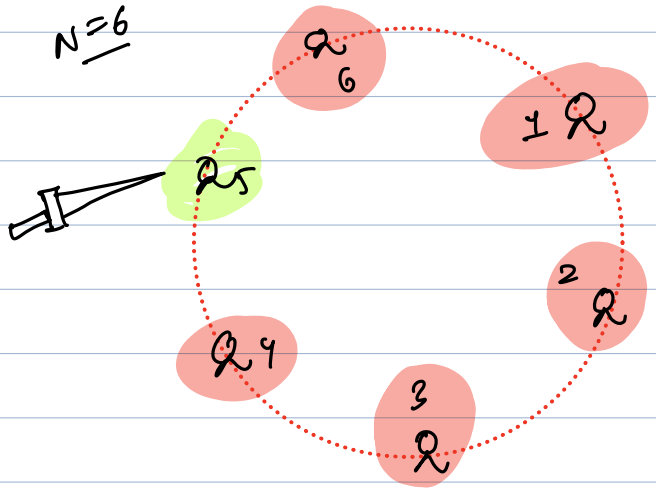
for a N,
$\sqrt{N}$ perfect square from 1→N

```
for( i=1; i*i<=N; i++)
    print ( i*i);
```

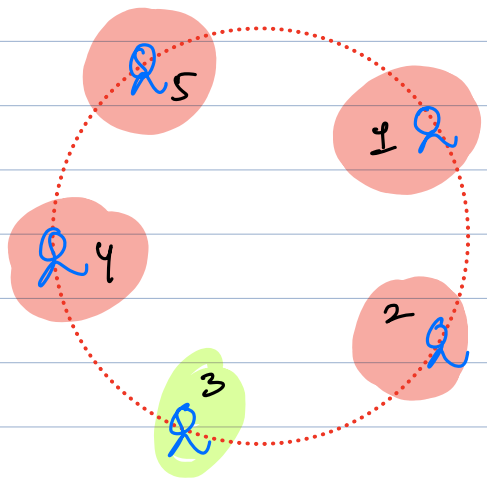
T.C: $O(\sqrt{N})$

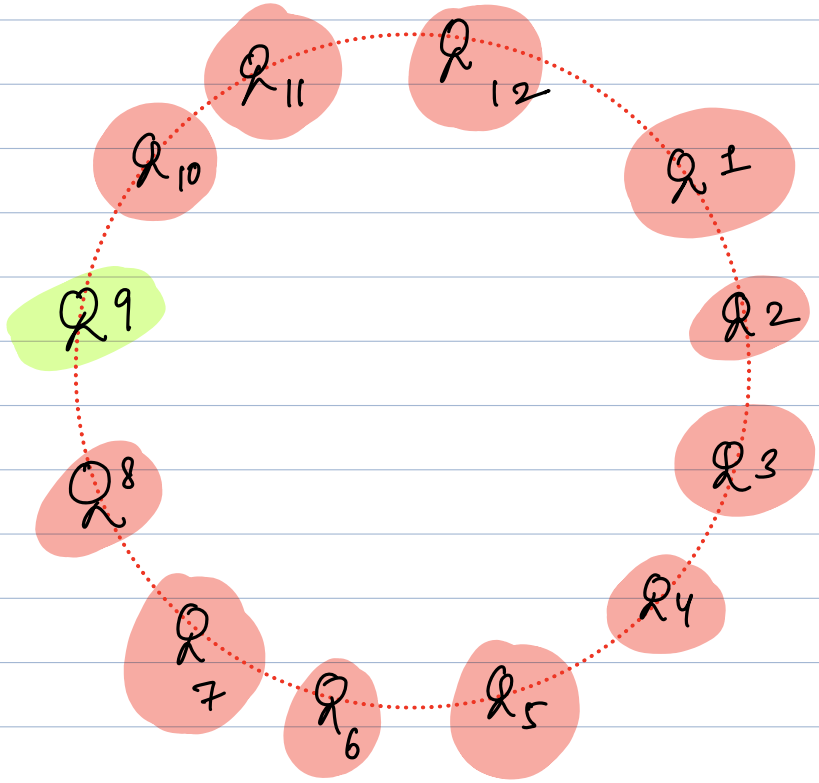
Q

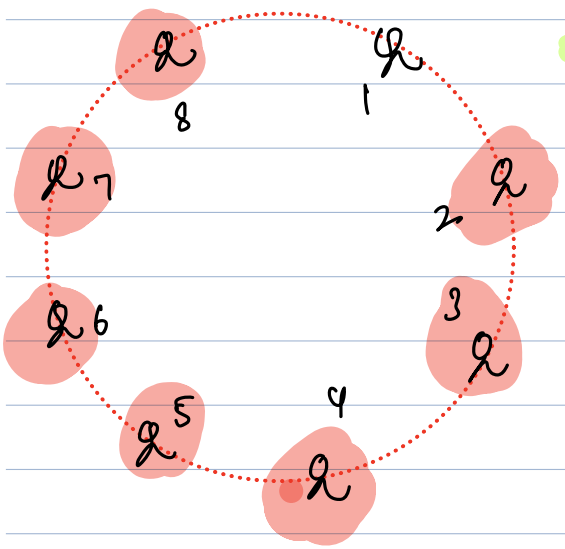
josephus → Jews vs Romans



(N) If I start with person 1, which position should josephus sit on?

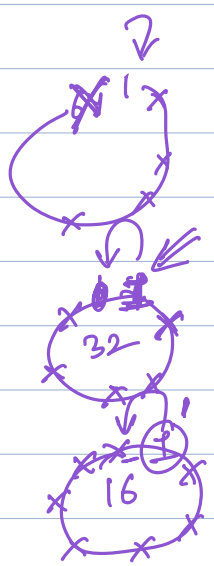


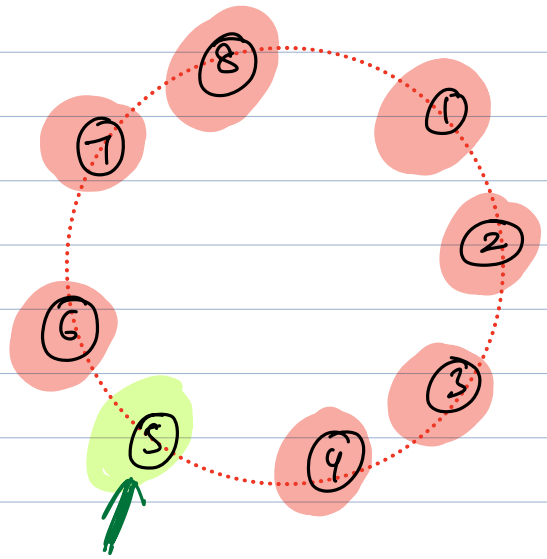




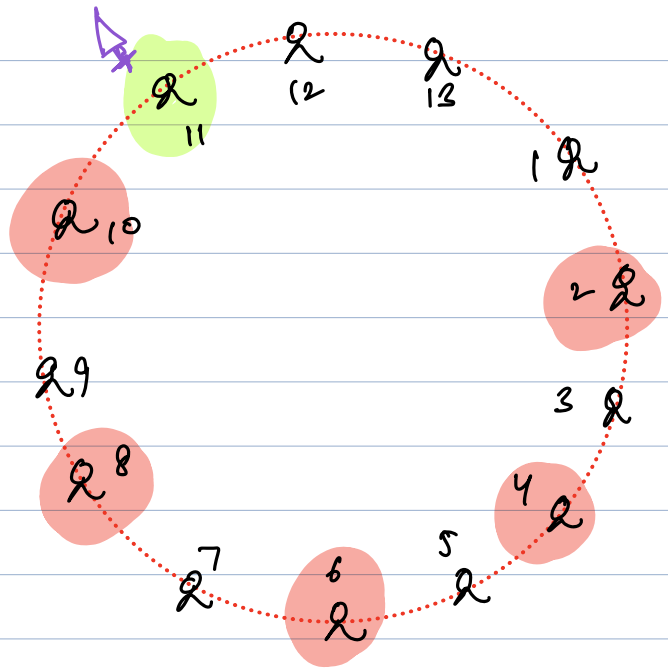
N	
1	1
2	1
3	3
4	1
5	3
6	5
7	7
8	1
9	3
10	5
11	7
12	9
13	11
14	13
15	15
16	1

power of 2
↓
ans = 1



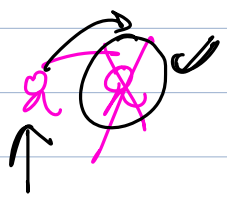


power of 2
↓
start pt will survive



N = 13 — kill 5 people

8



kill x people

2x

2x + 1 ≡ sword
≡ survivor

$x = N - (\text{nearest power of } 2)$

ans = 2x + 1;

$2^a \equiv 1 << a$

$\log_2 N$ — int part

2

$1 << \log_2 N$

15 → 8

66 → 64

$\log_2 15 = 3...$

10:25

Q ⇒ Find the majority element given the array of N elements.

Find the element which

freq > N/2 more than half

1 6 1 2 1 2 1

N = 7

need? > 7/2

> 3

majority = 4

3 4 3 6 1 3 2 5 3 3 3

N = 11

majority

need? > 11/2

> 5

after 6

4 6 5 3 4 5 6 4 4 4

N = 10

need > 10/2

= > 5

Nobody in majority

Y = 6

count freq of each number

T.C: $O(N^2)$

$O(N \log N)$ sort

Hashmap → $O(N)$

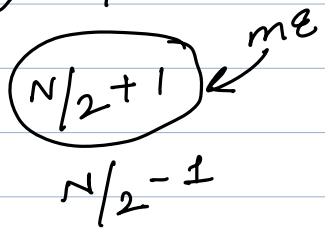
```
for( i=0; i<N; i++)
{
    int cnt=0;
    for( j=0; j<N; j++)
        if( arr[j]==arr[i])
            cnt++;
    if(cnt > N/2) return arr[i];
}
```

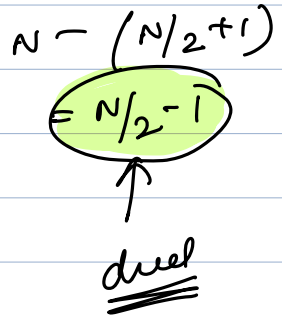
T.C: O(N)	S.C: O(1)

Can there be more than one majority element?

total = N

freq > N/2

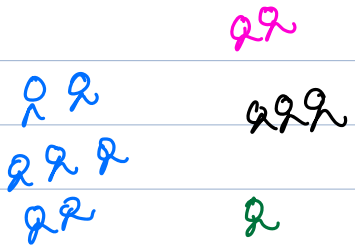




2 people fights & they both die

NME * NME

NME * ME



N = 13

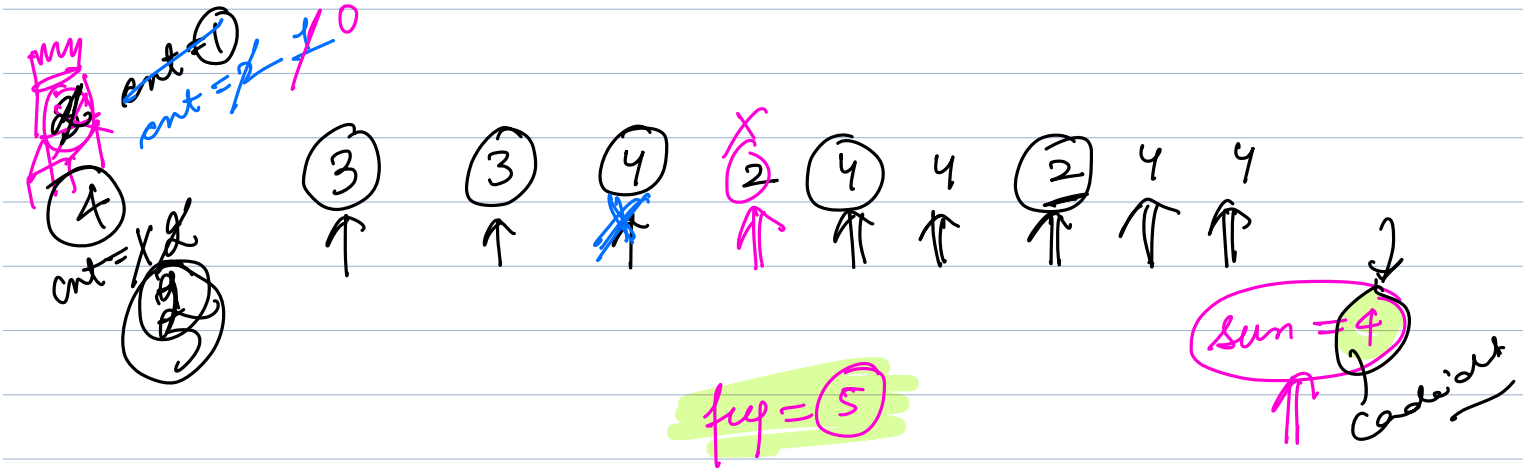
	ME	NME	
N	7	6	13
N-2	6	5	11

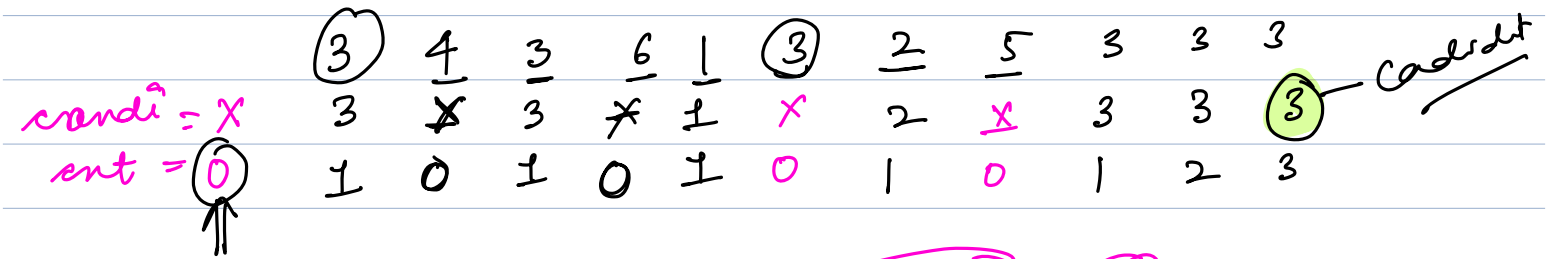
N/2+1

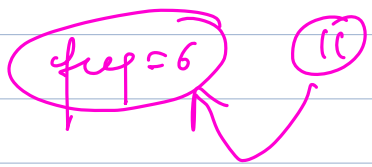
N-2/2 + 1 = N/2 - 1 + 1

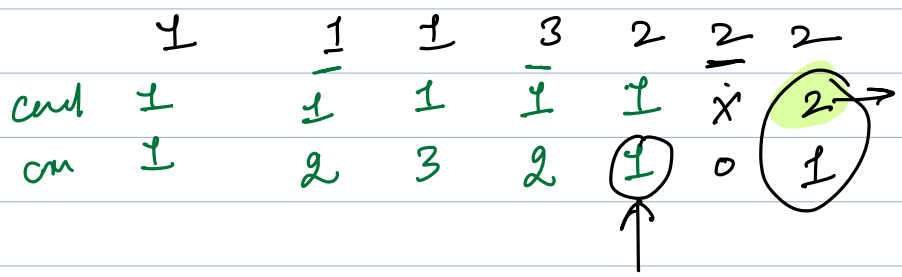
If 2 distinct elements are deleted, the elemet which was in majority remain in majority

survive - majority









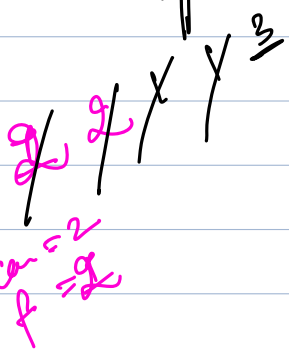
freq = 3 need? 4

If there exists a majority, majority will survive

Moore's voting algo

N/k majority

#N/3 major elmt



```
int me = arr[0]
int cnt = 1;

for(int i=1; i<n ; i++)
{
    if ( arr[i] == me)        → if same group
    {
        cnt++;
    }
    else if ( count ==0)      → if nobody to claim to be maj
    {
        me = arr[i];
        cnt =1;
    }
    else{ count--; }
}

int freq=0;
for ( i=0; i<N; i++)
    if (arr[i] == me)
        freq++;

if ( freq > N/2) return me;
return -1;
```