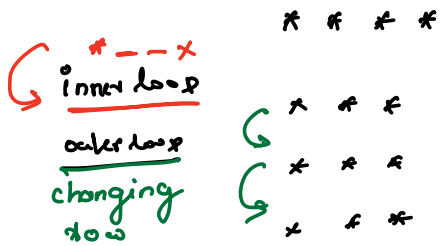
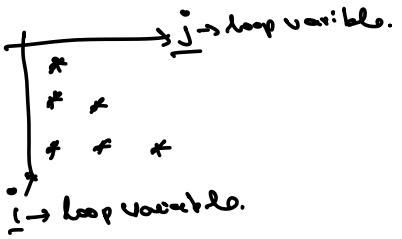




Q) Print the following pattern.

N = 3

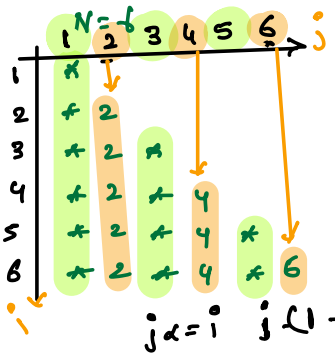
```
*
* 2
* 2 *
```

N = 5

```
*
* 2
* 2 *
* 2 * 4
* 2 * 4 *
```

N = 6

```
*
* 2
* 2 *
* 2 * 4
* 2 * 4 *
* 2 * 4 * 6
```



j = 1, 3, 5 ⇒ *
j = 2, 4, 6 ⇒ nums

if odd
    Print *
else
    Print j

1) 1 outer loop. (1 ⟶ N)
2) inner loop from (1 ——— i)
3) inside inner loop. check if j is odd/even
4) if odd →
    Print *
5) else
    Print j
6) Use println after inner loop to change row.

```
for(int i=1 ; i<=N ; i++) {
    for(int j=1; j<=i ; j++) {
        if( j%2 != 0) {
            sop("*");
        } else {
            sop(j);
        }
    }
    sopln(); → change soco.
}
```

Q> Print the following pattern.

N = 3

```
*
* 1
* 1 *
```

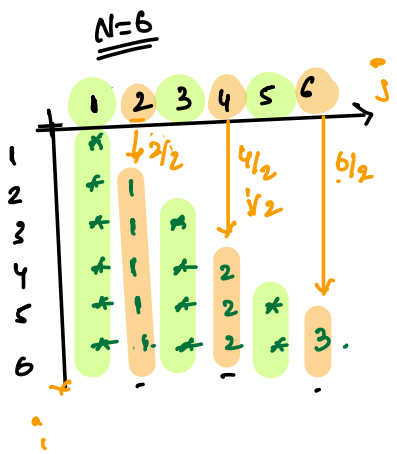
N = 5

```
*
* 1
* 1 *
* 1 * 2
* 1 * 2 *
```

N = 6

```
*
* 1
* 1 *
* 1 * 2
* 1 * 2 *
* 1 * 2 * 3.
```

1) Ist Approach



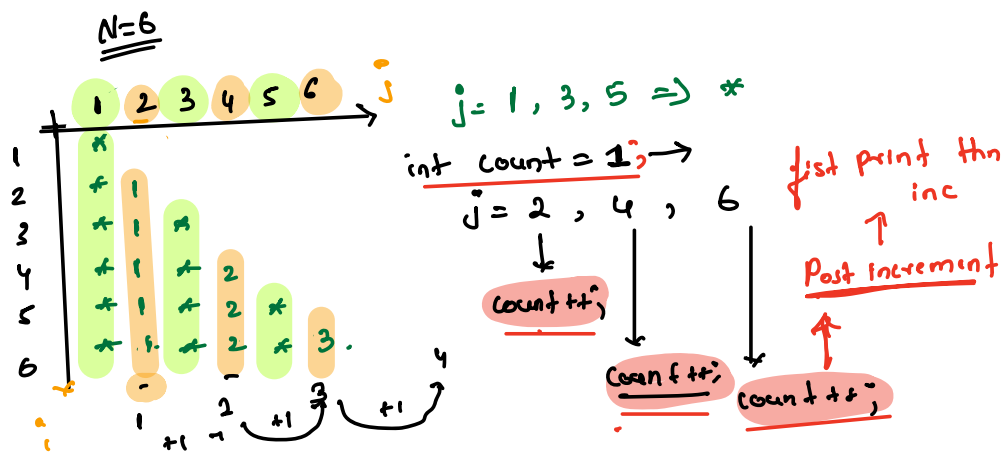
j = 1, 3, 5 => *

j = 2/2, 4/2, 6/2 => j/2

=> 1 2 3

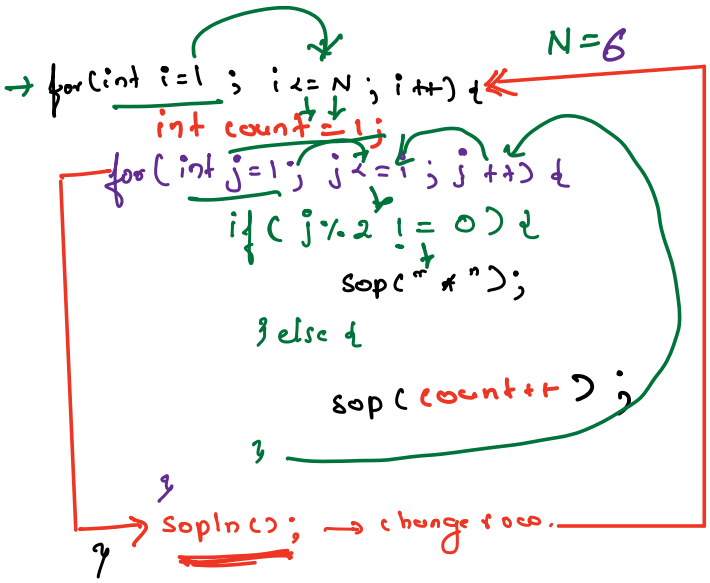
```
for(int i=1 ; i<=N ; i++) {
    for(int j=1; j<=i ; j++) {
        if( j%2 != 0) {
            sop(" * ");
        } else {
            sop( j/2); ⟶ ①.
        }
    }
    sopln(); ⟶ change row.
}
```

2nd Approach.



Solution

```
for(int i=1 ; i<=N ; i++) {
    int count=1; ⟶ ①
    for(int j=1; j<=i ; j++) {
        if( j%2 != 0) {
            sop(" * ");
        } else {
            sop( count++); ⟶ ②.
        }
    }
    sopln(); ⟶ change row.
}
```



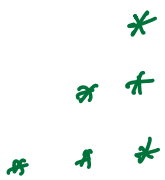

```
*
* 1
* 1 *
* 1 * 2
* 1 * 2 *
* 1 * 2 * 3.
```

i	count	j	j%2!=0	Print count++
1	1	1	T → *	
		2 → sopln();		
2	1	1	T → *	
x!		2		T → 1
	2	3 → sopln();		
3	1	1 → T → *		
		2		T → 1
	2	3 → T → *		
		4 → sopln();		count++ → 2
4	1	1 → T → *		T → 1
		2		
	2	3 → T → *		
		4		T → 2
		5 → sopln();		
5	1	1 → T → *		
		2		T - 1
	2	3 → T → *		
		4		T - 2
		5 → T - *		
		6 → sopln()		
6	1	1 → T → *		
		2		T → 1
	2	3 → T → *		
		4		T → 2
	3	5 → T → *		
		6		T → 3
		7 → sopln();		

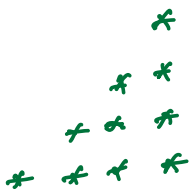
exit of loop ← 7

Q

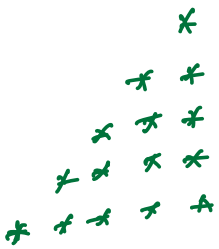
N = 3



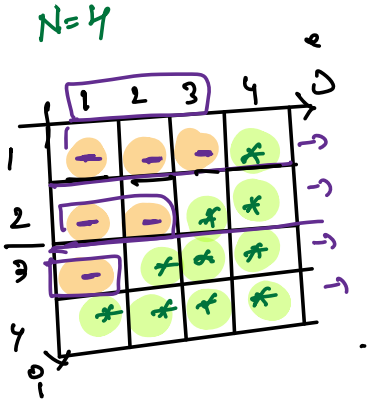
N = 4



N = 5



N = 4



i , j

N = 4

i	(N-i) spaces	stars
1	3	1
2	2	2
3	1	3
4	0	4

each row ⇒ spaces + stars

(N-i) for(1 → N-i) print("_");

(i) for(1 — i) print("*");

① innerloop

② inner loop.

3 → outer loop

for(i → N){

for(1 — N-i){

sop("_"); → spaces

for(1 → i){

sop("*"); → stars

}

sop(nc);

}

Solution

```
for(int i=1 ; i<=N ; i++){
    for(int j=1 ; j<=N-i ; j++){        // jscope.
        sop(" ");
    }
    for(int j=1 ; j<=i ; j++){          // jscope
        sop("* ");
    }
    soplnC);
}
```

Break → 15 mins ⇒ (Ravijeet · Chaudhary, Shathapathi, Onkar)

10:40

Q) N = 3

```
* * *
  * *
    *
```

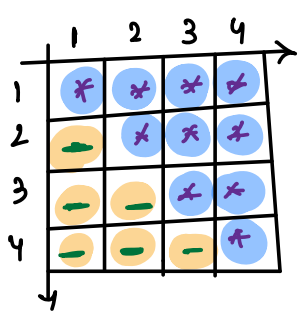
N = 4

```
* * * *
  * * *
    * *
      *
```

N = 5

```
* * * * *
  * * * *
    * * *
      * *
        *
```

N=4



i	(i-1) spaces	N-i+1 stars
1	0	4
2	1	3
3	2	2
4	3	1

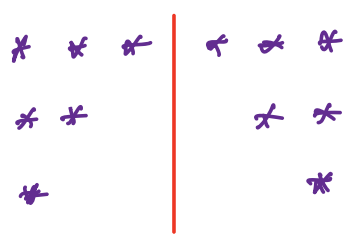
Spaces + stars

for(1 ⟶ i-1) + for(1 ⟶ N-i+1).

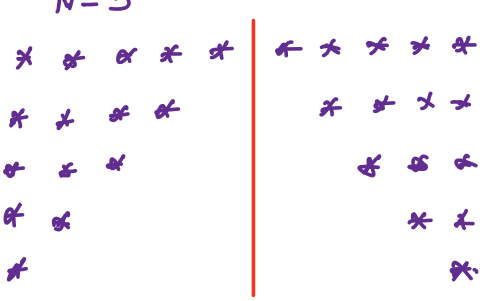
```
for(int i=1; i<=N; i++){
    for(int j=1; j<=i-1; j++){        // j scope.
        sop(" ");
    }
    for(int j=1; j<=N-i+1; j++){      // j scope.
        sop("*");
    }
    soplnC);
}
```

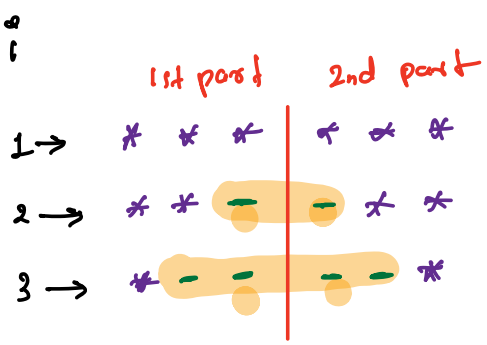
Q:-

N=3



N=5





N=3

i	1st part: st	1st part: sp	2nd part: sp	2nd part: st
1	3	0	0	3
2	2	1	1	2
3	1	2	2	1
	N-i+1	i-1	i-1	N-i+1

st + sp + sp + st

st + 2*(sp) + st

each row → stars + spaces + spaces + stars.

stars → 1 → n-i+1

spaces → 1 → (i-1) + 1 → (i-1)

Combined

+ N-i+1

(1 → N-i+1) + (1 → 2x(i-1)) + (1 → N-i+1)

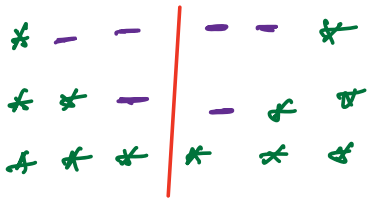
① ② ③

1 → outer loop

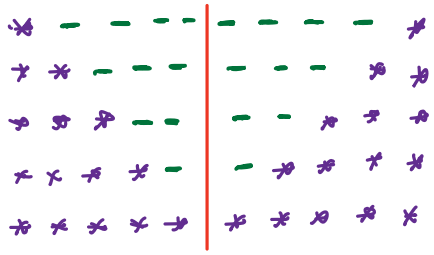
3 → inner loops.

```
for(int i=1; i<=N; i++){
    for(int j=1; j<=N-i+1; j++){
        sop("*");  // star
    }
    for(int j=1; j<=2*(i-1); j++){
        sop("");  // space
    }
    for(int j=1; j<=N-i+1; j++){
        sop("*");  // star
    }
    sopln();
}
```

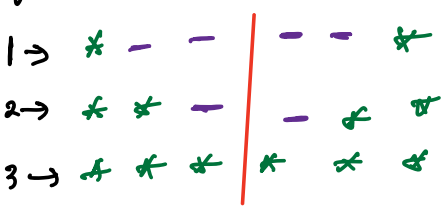
Q ⇒ N = 3

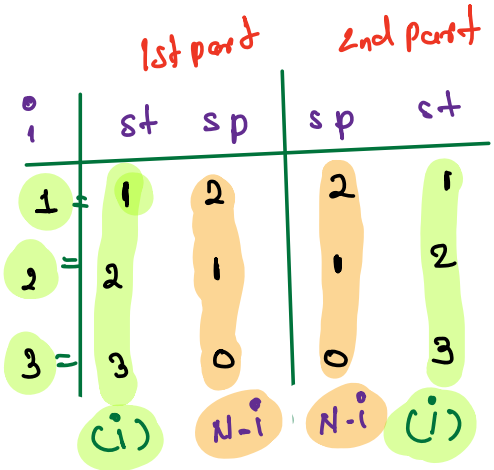


N = 5



N = 3



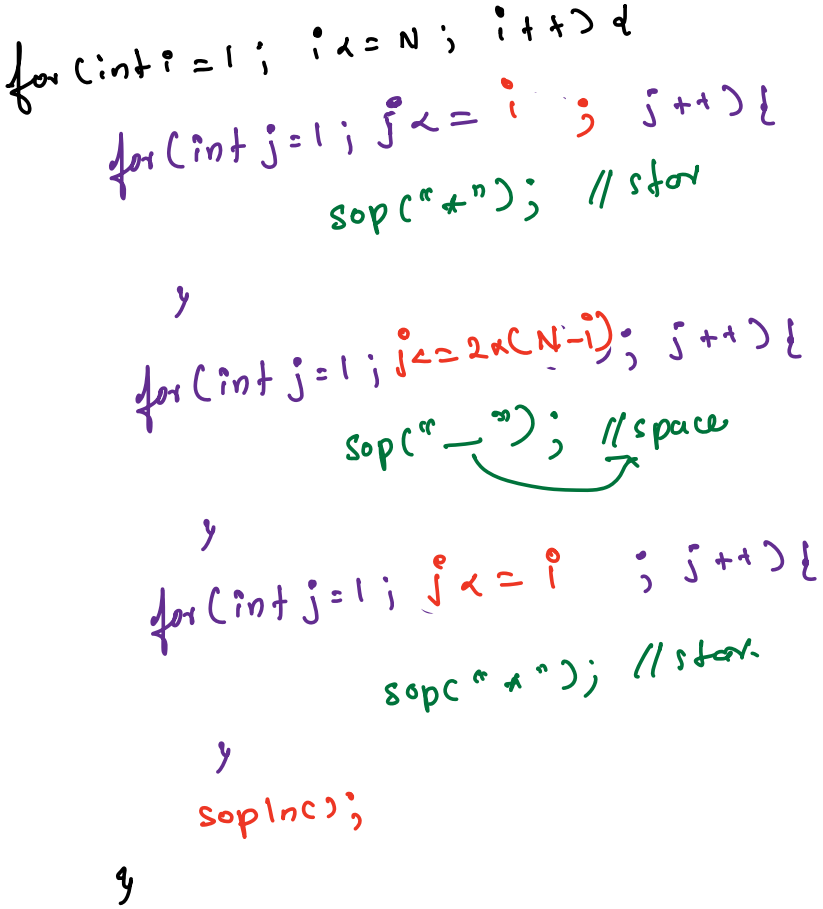


each row → stars + space + space + stars.

$i + N-i + N-i + i$

$i + 2 \times (N-i) + i$

for(1 — i) + for(1 → 2×(N-i)) + for(1 — i):



https://www.interviewbit.com/snippet/01d54300e26559b42d22/