Ques1. Print n stars "*"

N = 5 * * * * *

N = 3 * * *

```
for ( int i=1 ; i <= N ; i++ ) {
        S.O.P ("*");
}
```

Quiz

$\boxed{N=5}$

✓

```
for (int i=1 ; i<=N; i++){
        S.O.P ("*");
}
```

i = 1 2 3 4 5

⇒ 5 times

N=5

X

```
for (int i=1; i<N; i++) {
        S.O.P ("*");
}
```

i = 1 2 3 4

⇒ 4 times

N=5

X

```
for (int i=0; i<=N; i++) {
        S.O.P ("*");
}
```

i= 0 1 2 3 4 5

⇒ 6 times

Ques. Print a square

3 x 3

N = 3

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

N=5

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

Hello → N times
[loop]

```
for ( int i=1 ; i <= N ; i++) {

        // Print N stars
        for (int j=1; j<=N; j ++) {
                S.O.P ("*");
        }
        S.O.Pln(); // → go to the next line
}
```

N=3

Output →

```
* * *
* * * ↵
* * * ↵
↵
```

i	[i<=3]	j	[j<=3]
1	✓	1	✓
		2	✓
		3	✓
		4	x [break]
2	✓	1	✓
		2	✓
		3	✓
		4	x [break]
3	✓	1	✓
		2	✓
		3	✓
		4	x [break]
4	x →		[break]

Ques. Print the following stair case pattern

N=3

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

N=5

```
1 *
2 * *
3 * * *
4 * * * *
5 * * * * *
```

row	stars
1	1
2	2
3	3
4	4
5	5

count of stars = row

```
for ( int row = 1 ; row <= n ;  row++) {

      // Print stars = row

   for ( int star = i ; star <= row; star ++) {
           S.O.P ("*");
   }
   S.O.Pln ( );

}
```

N = 4

```
1      *
2      *  *
3      *  *  *
4      *  *  *  *
```

row [row <= 4]	star [star <= row]
1 ✓	1 [1 <= 1] ✓ 2 [2 <= 1] × break
2 ✓	1 [1 <= 2] ✓ 2 [2 <= 2] ✓ 3 [3 <= 2] × break

3 ✓

1 [1<=3] ✓

2 [2<=3] ✓

3 [3<=3] ✓

4 [4<=3] x break

4 ✓

1 [1<=4] ✓

2 [2<=4] ✓

3 [3<=4] ✓

4 [4<=4] ✓

5 [5<=4] x break

5 → break

Ques. Print a rectangle (rows) N = 3 M = 5 (count of stars in each row)

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

N = 2 M = 4

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

```
int N = scn.nextInt();
int M = scn.nextInt();

// N rows
for (int row = 1 ; row <= N ; row++) {
```

[row<=2] row	[star<=4] star
1 ✓	1 ✓
	2 ✓

```
// Print M stars
   for(int star=1; star<=M; star++){
               S.O.P ("*");

   }
   S.O.Pln();
}
```

N = 2   M = 4

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

3 ✓
4 ✓
5 ✗ break

2 ✓

1 ✓
2 ✓
3 ✓
4 ✓
5 ✗ break

3 ✗ → break

Ques. reversed staircase pattern

N=3

		star	
1	* * *	3	=> 4
2	* *	2	=> 4
3	*	1	=> 4
			(N+1)

N = 5

* * * * *
* * * *
* * *
* *
*

row + stars = N+1

=> stars = (N+1 - row)

N = 4

	row		stars		
+1 ↓	1	* * * *	-1 ↓ 4	[5-1]	=> 5
	2	* * *	3	[5-2]	=> 5
	3	* *	2	[5-3]	=> 5
	4	*	1	[5-4]	=> 5
			[N+1 - row]		(N+1)

```
for(int row=1; row<= N; row++) {

    // Print star = (N+1 - row)
      int  x = N+1-row; // total stars
  for (int star=1 ; star <= x ;star++){
              S.O.P ("*");
  }
  S.O.Pln ();
}
```

$\boxed{N=3}$

```
* * *  ↲
* *    ↲
*      ↲
```

(row<=3) row	x (4-row)	star (star<=x)
1 ✓	3	1 ✓ 2 ✓ 3 ✓ 4 (false)
2 ✓	2	1 ✓ 2 ✓ 3 (false)
3 ✓	1	1 ✓ 2 (false)

4 ✗ ⟶ break

N= 4

row		star
4 →	* * * *	4
3	* * *	3
2	* *	2
1	*	1

count of stars = row

④ → 1

```
for(int row=N; row >= 1; row--) {

        for(int star=1; star <= row; star++) {

                S.O.P ("*");

        }

    S.O.P ln( );

}
```

N = 4

row + star = N

star = N - row

Row		star			
0	* * * *	4	[4-0]	⇒	4
1	* * *	3	[4-1]	⇒	4
2	* *	2	[4-2]	⇒	4
3	*	1	[4-3]	⇒	4
					[N]

Ques. Print the following pattern

N=3

```
* _ _ *
* _ *
* *
```

N=5

```
1  * _ _ _ _ *
2  * _ _ _ *
3  * _ _ *
4  * _ *
5  * *
```

	row		Spaces		
↓	1	↓	4 [5-1]	⇒	5
	2		3 [5-2]	⇒	5
+1	3	-1	2 [5-3]	⇒	5
	4		1 [5-4]	⇒	5
	5		0 [5-5]	⇒	5
			[N-row]		N

row + spaces = N

⇒ spaces = N - row

In one row → 1 * spaces [N-row] 1 *
↳ Loop

```
for (int row = 1 ; row <= N ; row++) {

    S.O.P ("*");

    // Print (N-row) spaces (-)
    for(int space = 1; space <= (N-row); space++) {
            S.O.P ("-");
    }

    S.O.P ("*");

    S.O.Pln ();

}
```

N = 3

```
* _ _ *   ↲
* _ *     ↲
* *       ↲
```

row [row<=3]	space (space <= 3-row)
1 ✓	1 [1<=2] ✓ 2 [2<=2] ✓ 3 [3<=2] (break)
2 ✓	1 [1<=1] ✓ 2 [2<=1] (break)

3 ✓

4 [break]

1 [1<=0]
[break)

Ques. Print the following pattern

N = 4

	1	2	3	4
1	*			
2	*	2		
3	*	2	*	
4	*	2	*	4

N = 7

	1	2	3	4	5	6	7
1	*						
2	*	2					
3	*	2	*				
4	*	2	*	4			
5	*	2	*	4	*		
6	*	2	*	4	*	6	
7	*	2	*	4	*	6	*

row	star
1	1
2	2
3	3
4	4

stars = row

```
for (int row =1; row <=n; row++) {

for(int star = 1; star <= row; star++) {
        if (star %2 ==0) {
                S.O.P (star);
        } else {
                S.O.P ("*");
        }
}
```

row	star
1 ✓✓	1 [1<=1] ✓
	2 [2<=1] [break]
2 ✓✓	1 [1<=2] ✓✓
	2 [2<=2] ✓✓
	3 [3<=2] (break)

S.O.P ln();

}

N = 5

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

3 ✓	1 [1<=3] ✓
	2 [2<=3] ✓
	3 [3<=3] ✓
	4 [4<=3] (break)
4 ✓	1 [1<=4] ✓
	2 [2<=4] ✓
	3 [3<=4] ✓
	4 [4<=4] ✓
	5 [5<=4] (break)
5 ✓	1 [1<=5] ✓
	2 [2<=5] ✓
	3 [3<=5] ✓
	4 [4<=5] ✓
	5 [5<=5] ✓
	6 [6<=5] (break)

6 → break