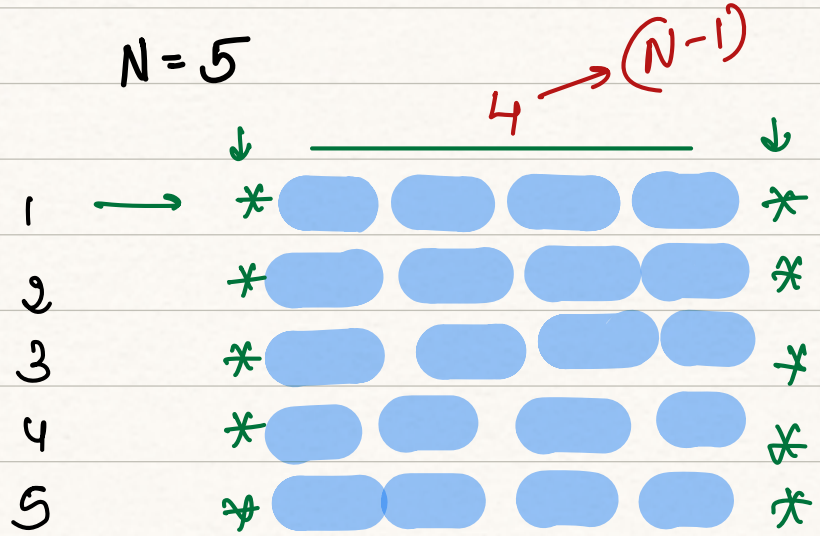
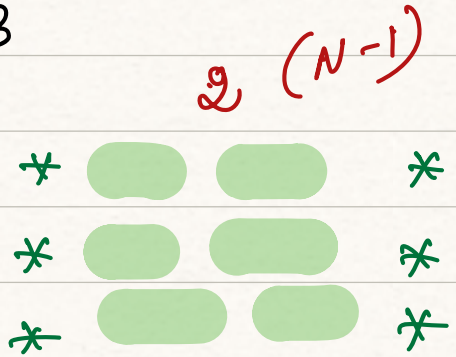
Ques 1.

N = 5



N = 3



For each row → 1 * (N-1) spaces 1 *

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

    S.O.P ("*");

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

    S.O.P ("*");

    S.O.Pln ( ) ;

}
```

N = 3

```
*  -  -  *
*  -  -  *
*  -  -  *
```

(≤3) row

1 ✓

2 ✓

3 ✓

4 ⟶ break

(≤2) space

1 ✓
2 ✓
3 [break]

1 ✓
2 ✓
3 [break]

1 ✓
2 ✓
3 [break]

Ques: Print the following pattern.

N = 3

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

N = 5

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

N = 5

row	spaces	star
1	4 [5-1]	1
2	3 [5-2]	2
3	2 [5-3]	3
4	1 [5-4]	4
5	0 [5-5]	5
	(N-row)	(row)

+1 ↓ (row)

-1 ↓ (spaces)

row + spaces = N

spaces = N - row

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

    // print (N - row) spaces

    for(int sp = 1; sp <=(N-row); sp++){
            S.O.P (" ");
    }

    // print stars = row
    for (int st = 1; st <= row; st++) {
            S.O.P ("*");
    }

    S.O.Pln ();

}
```

Ques. Print the following patter

N = 3

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

N = 5

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

N

row	Spaces	Stars
1	0 [1-1]	5 [6-1]
2	1 [2-1]	4 [6-2]
3	2 [3-1]	3 [6-3]
4	3 [4-1]	2 [6-4]
5	4 [5-1]	1 [6-5]
	(row-1)	(N+1-row)

row + star = N+1

⇒ star = N+1-row

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

    // print (row-1) spaces
    for (int sp = 1 ; sp <= (row-1) ; sp++) {
        S.O.P (" ");
    }
    // print (N+1-row) star
    for (int st = 1 ; st <= (N+1-row) ; st++) {
        S.O.P ("*");
    }

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

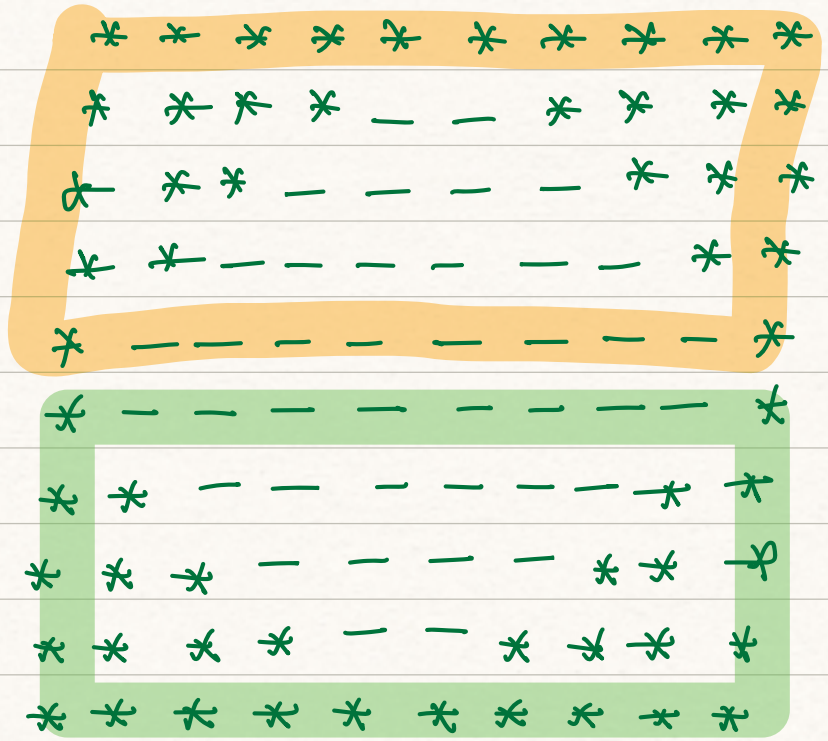
N = 3

```
* * *
- * *
- - *
```

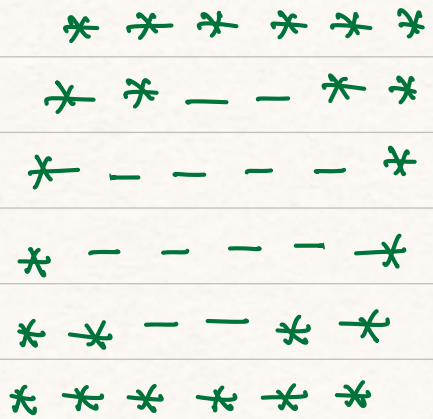
row (<=3)	sp (<= row-1)	st (<= N+1-row)
1 //	1 [1<=0]	1 [1<=3]
	[break]	2 [2<=3]
		3 [3<=3]
		4 [4<=3]
		[break]
2 //	1 [1<=1] ✓	1 [1<=2]
	2 [2<=1]	2 [2<=2]
	[break]	3 [3<=2]
		[break]
3 //	1 [1<=2]	1 [1<=1]
	2 [2<=2]	2 [2<=1]
	3 [3<=2]	(break)
	(break)	
4 →		break

Ques.

N=5



N=3



Upper half

N = 5 ⇒ N+1 = 6

FH → print star first then spaces

SH → print spaces first then stars

row + star = N+1

star = (N+1 − row)

row	FH: star	FH: space	SH: spaces	SH: star
1	5	0	0	5
2	4	1	1	4
3	3	2	2	3
4	2	3	3	2
5	1	4	4	1
	(N+1−row)	(row−1)	(row−1)	(N+1−row)

(N+1−row) * — 1st loop

(row−1) spaces

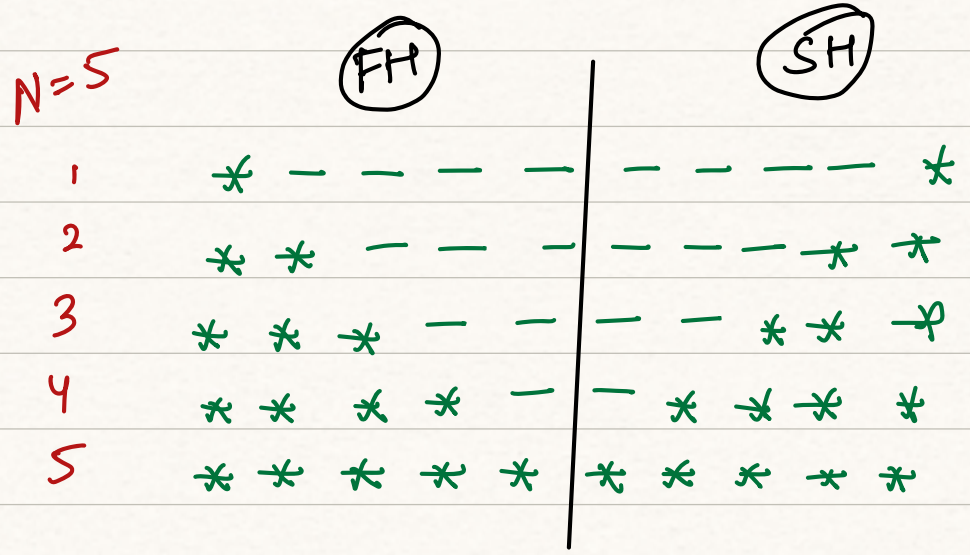
(row−1) spaces

(N+1−row) * — 3rd loop

row - 1 + row - 1 = (2 row - 2) spaces ⇒ 2*(row-1)

2nd loop

N=5



row + spaces = N

spaces = N - row

row	FH: star	FH: spaces	SH: spaces	SH: stars
1	1	4	4	1
2	2	3	3	2
3	3	2	2	3
4	4	1	1	4
5	5	0	0	5
	(row)	(N - row)	(N - row)	(row)

+1 ↓ (star)

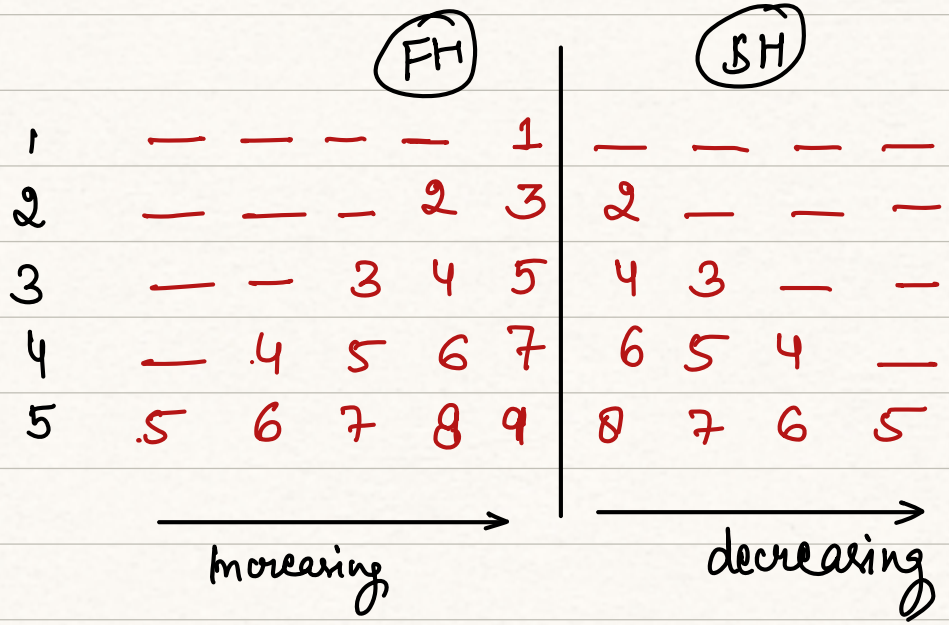
-1 ↓ (spaces)

(row) * (N-row) spaces (N-row) spaces (row) *

2*(N-row) spaces

Full Numeric Pyramid

N=5



Print in Range

A = 5 B = 10

⇒ 5 6 7 8 9 10

```
for (int i = A ; i <= B ; i++ ){
    S.O.P (i) ;
}
```

i = ~~5~~ ~~6~~ ~~7~~ ~~8~~ ~~9~~ ~~10~~ (11) → break

5 6 7 8 9 10

```
for (int i = B ; i >= A ; i--) {
    S.OP (i);
}
```

10 9 8 7 6 5

i = ~~10~~ ~~9~~ ~~8~~ ~~7~~ ~~6~~ ~~5~~ (4) → break

10 9 8 7 6 5

N=5

	FH					SH			
1	—	—	—	—	1	—	—	—	—
2	—	—	—	2	3	2	—	—	—
3	—	—	3	4	5	4	3	—	—
4	—	4	5	6	7	6	5	4	—
5	5	6	7	8	9	8	7	6	5

increasing → decreasing

row + space = N

space = N - row

	FH				SH			
row	space	start 1	end 1		start 2	end 2	spaces	2*row
1	4	1	1		—	—	4	2
2	3	2	3		2	2	3	4
3	2	3	5		4	3	2	6
4	1	4	7		6	4	1	8
5	0	5	9		8	5	0	10
	(N - row)	(row)	(2 row - 1)		(2 row - 2)	(row)	(N - row)	

FH: space -1 ↓, start 1 +1 ↓, end 1 +2 ↓

SH: start 2 = 2(row - 1)

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

    // print (n - row) spaces

    int start1 = row;
    int end1 = 2*row - 1;
    for(int i = start1; i <= end1; i++){
        S.O.P (i);
    }

    int start2 = 2*row - 2;
    int end2 = row;

    for(int i = start2; i >= end2; i--){
        S.O.P (i);
    }

    // print (n - row) spaces).

    S.O.P ln ();
}
```

First half

Second half

N = 5

row	start1	end1	i
1	1	1	✓✓
2	2	3	✓✓
3	3	5	✓✓

start2	end2	i
2*1 - 2 = 0	1	✓✓
2*2 - 2 = 2	2	✓✓
2*3 - 2 = 4	3	✓✓

```
- - - - 1 - - - -
- - - 2 3 2 - - -
- - 3 4 5 4 3 - -
```