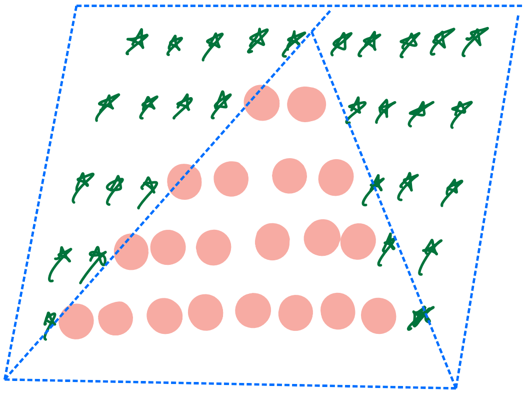
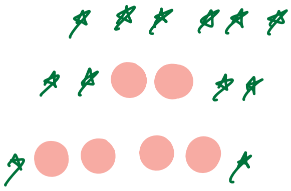
Q: N = 5



N = 3



N = 5

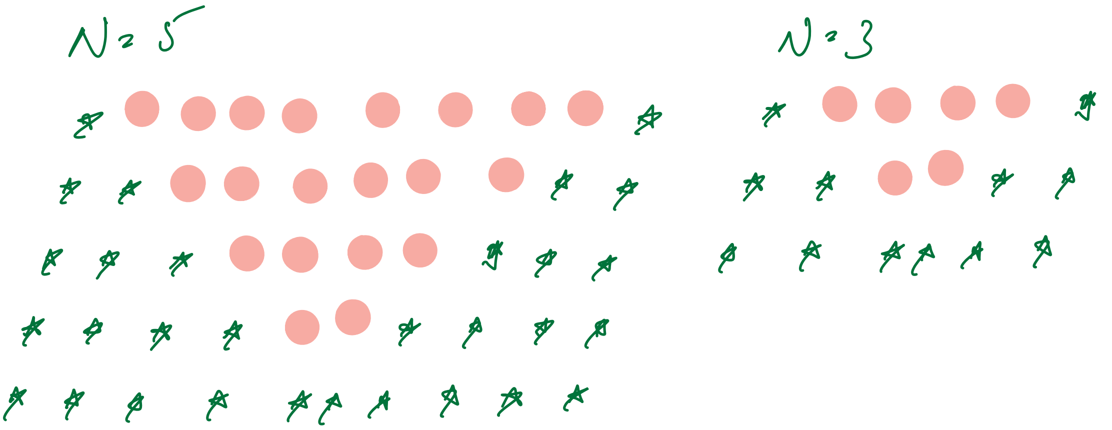
2xRow	Row	$*_1$	Spaces	$*_2$
2	1	5	0	5
4	2	4	2	4
6	3	3	4	3
8	4	2	6	2
10	5	1	8	1
		⇓ n+1-row	⇓ 2xrow-2	↓ n+1-row

$Row + *_1 = N+1$

$*_1 = N+1 - Row$

```
for(int row=1; row ≤ N; row++){
    for(int st=1; st ≤ N+1-row; st++){
        SOP("*");
    }
    for(int sp=1; sp≤ 2xrow-2; sp++){
        SOP("  ");
    }
    for(int st=1: st ≤ N+1-row; st++){
            SOP("*");
    }
    SOPln();
}
```

Q:



[N=5]

2×Row	Row	*1	spaces	*2
2	1	1	8	1
4	2	2	6	2
6	3	3	4	3
8	4	4	2	4
10	5	5	0	5
		⇓ row	⇓ 2×N−2×row	⇓ row

$$2 \times row + spaces = 2 \times N$$

$$spaces = 2 \times N - 2 \times row$$

```
for(int row=1; row ≤N; row++){
        for(int st=1; st ≤ row ; st++){
                SOP("*");
        }
      for(int sp=1; sp≤ 2*N-2*row; sp++){
              SOP("  ");
      }
     for(int st=1; st ≤ row ; st++){
             SOP("*");
     }
     SOPln();
}
```

Q: N=5

```
● ● ● ● * ● ● ● ●
● ● ● * * * ● ● ●
● ● * * * * * ● ●
● * * * * * * * ●
* * * * * * * * *
```

N=3

```
● ● * ● ●
● * * * ●
* * * * *
```

2*Row	Row	Spaces	*
2	1	4	1
4	2	3	3
6	3	2	5
8	4	1	7
10	5	0	9

[N=5]

⇓ n-row ⇓ 2xrow-1

Q:

```
★ ★ ★ ★ ★ ★ ★ ★ ★
● ★ ★ ★ ★ ★ ★ ★
● ● ★ ★ ★ ★ ★
● ● ● ★ ★ ★
● ● ● ● ★
```

[N=5]

2xRow	Row	spaces	★
2	1	0	9
4	2	1	7
6	3	2	5
8	4	3	3
10	5	4	1

⇓ row-1 ⇓ 2xn-2xrow+1

2xrow + ★ = 2xN+1

★ = 2xN-2xrow+1

Break - 10pm

Q: N=5

```
0 0 0 0 1 0 0 0 0
0 0 0 2 3 2 0 0 0
0 0 3 4 5 4 3 0 0
0 4 5 6 7 6 5 4 0
5 6 7 8 9 8 7 6 5
```

Row	0s	[s_1	e_1]	[s_2	e_2]	0s
1	4	[1	1]	—		4
2	3	[2	3]	[2	2]	3
3	2	[3	5]	[4	3]	2
4	1	[4	7]	[6	4]	1
5	0	[5	9]	[8	5]	0
	⇓	⇓	⇓	⇓	⇓	⇓
	n-row	row	2*row-1	2*row-2	row	n-row

```
for( int row=1; row<=N; row++){
    for(int zero=1; zero<= N-row;zero++){
            SOP(0);
    }
    for( int num = row; num <= 2*row-1; num++){
            SOP( num);
    }
    for (int num = 2*row-2; num >= row; num--){
            SOP(num);
    }
    for(int zero=1; zero<= N-row;zero++){
            SOP(0);
    }
    SOPln();
}
```

Q: N=5

```
0 0 0 0 5  0  0  0  0
0 0 0 4 8  12 0  0  P
0 0 3 6 9  12 15 0  0
0 2 4 6 8  10 12 14 0
1 2 3 4 5  6  7  8  9
```

Row	0s	mult	cnt	0s
1	4	5	1	4
2	3	4	3	3
3	2	3	5	2
4	1	2	7	1
5	0	1	9	0
	⇓ n-row	⇓ n-row+1	⇓ 2×row-1	⇓ n-row

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

    for(int  zero=1;  zero ≤ n-row;  zero++){
                 SOP( 0);
    }

    for( int  cnt=1;  cnt ≤ 2×row-1 ;  cnt++){

             SOP( cnt × ( n-row+1));
    }
    for(int  zero=1;  zero ≤ n-row;  zero++){
                 SOP( 0);
    }
    SOPln();
}
```

Row = 3   Mult => n-row+1 = 3

Tot => 2×row-1 = 5

```
for(int cnt=1; cnt ≤ tot; cnt++)
        SOP( mult × cnt);
}
```

[1   2   3   4   5] × 3

3   6   9   12   15