Catalan Numbers

Nth Catalan Number

1. → Unique Binary Search Tree
3. → Dyk Words
2. → Dyk Paths
4. → Intersecting chords in Circle
5. → Count Balanced Parantheses String
6. → Count No of ways of Triangulation
7. → Count No of Handshakes
8. → Count Mountain Ranges

Nth Catalan Number

$C_0 = 1$

$C_1 = 1$

$C_2 = C_0 * C_1 + C_1 * C_0 = 1*1 + 1*1 = 2$

$C_3 = C_0 * C_2 + C_1 * C_1 + C_2 * C_0 = 1*2 + 1*1 + 2*1 = 5$

$C_4 = C_0 * C_3 + C_1 * C_2 + C_2 * C_1 + C_3 * C_0$

$= 1*5 + 1*2 + 2*1 + 5*1 = 14$

$C_5 = C_0 * C_4 + C_1 * C_3 + C_2 * C_2 + C_3 * C_1 + C_4 * C_0$

$= 1*14 + 1*5 + 2*2 + 5*1 + 14*1$

$= 42$

n	C_n	n	C_n	n	C_n
1	1	11	58,786	21	24,466,267,020
2	2	12	208,012	22	91,482,563,640
3	5	13	742,900	23	343,059,613,650
4	14	14	2,674,440	24	1,289,904,147,324
5	42	15	9,694,845	25	4,861,946,401,452
6	132	16	35,357,670	26	18,367,353,072,152
7	429	17	129,644,790	27	69,533,550,916,004
8	1,430	18	477,638,700	28	263,747,951,750,360
9	4,862	19	1,767,263,190	29	1,002,242,216,651,368
10	16,796	20	6,564,120,420	30	3,814,986,502,092,304

C_N will overflow integer range when $N \geq 20$

N^{th} Catalan NO → using Dynamic Programming

$$C_N = C_0 * C_{N-1} + C_1 * N_{N-2} + C_2 * C_{N-3} + \dots C_{N-1} * C_0$$

$$\boxed{C_N = \sum_{i=0}^{N-1} C_i * C_{N-1-i}}$$

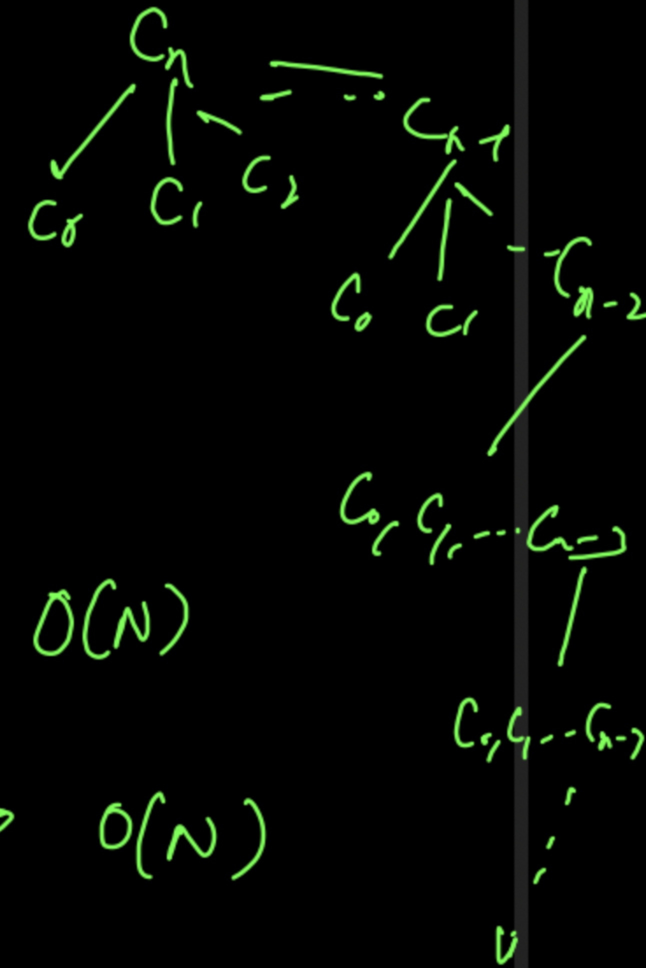
IDDP

C_0	C_1	C_2	C_3	C_4	C_5	C_6
1	1	$1+1 = 2$	$1*2 + 1*1 + 2*1 = 5$	$1*5 + 1*2 + 2*1 + 5*1 = 14$	$1*4 + 1*5 + 2*2 + 5*1 + 14*1 = 42$	132
		$C_0 * C_1 + C_1 * C_0$	$C_0 * C_2 + C_1 * C_1 + C_2 * C_0$	$C_0 * C_3 + C_1 * C_2 + C_2 * C_1 + C_3 * C_0$	$C_0 * C_4$, $C_1 * C_3$, $C_2 * C_2$, $C_3 * C_1$, $C_4 * C_0$	

Recursion

```java
public int catalan(int n){
    if(n == 0 || n == 1) return 1;

    int ans = 0;
    for(int i=0; i<n; i++){
        ans = ans + catalan(i) * catalan(n - 1 - i);
    }
    return ans;
}
```



Recursion → Height → $O(N)$

Recursion → Breadth (calls) → $O(N)$

Total TC = $O(N^N)$ Exponential

Memoization

Recursive

```java
public int catalan(int n, int[] dp){
    if(n == 0 || n == 1) return 1;
    if(dp[n] != -1) return dp[n];

    int ans = 0;
    for(int i=0; i<n; i++){
        ans = ans + catalan(i, dp) * catalan(n - 1 - i, dp);
    }

    return dp[n] = ans;
}
public int numTrees(int n) {
    int[] dp = new int[n + 1];
    Arrays.fill(dp, -1);
    return catalan(n, dp);
}
```

Time → O(N * N) (Recursion, for loop)

Space → O(N) 1D DP, O(N) R.C.S.

Tabulation

Iterative

```java
int[] dp = new int[n + 1];
dp[0] = dp[1] = 1;
for(int i=2; i<=n; i++)
    for(int j=0; j<i; j++)
        dp[i] += dp[j] * dp[i - j - 1];
return dp[n];
```

Time → O(N * N)

Space → O(N) 1D DP

Nth catalan No using Binomial coeff

$$N^{th}\text{ catalan} = \frac{{}^{2n}C_n}{(n+1)} = \frac{\frac{2n!}{n!\,n!}}{n+1} = \frac{2n!}{(n+1)!\,n!}$$

Combination

$$f(0) = \frac{{}^{0}C_0}{1} = 1/1 = 1$$

$$f(1) = \frac{{}^{2}C_1}{2} = \frac{\frac{2!}{1!\,1!}}{2} = 1$$

$$f(2) = \frac{{}^{4}C_2}{3} = \frac{\cancel{4}^{2}*\cancel{3}*\cancel{2}*1}{\cancel{2}*1*\cancel{2}*1*\cancel{3}}$$

$$= 2$$

Recurrence Relatn ${}^{n}C_r = {}^{n}C_{r-1} + {}^{n-1}C_{r-1}$

$\boxed{2D\ DP}$

Perks

1. Gadgets → 5 Gadgets → Headset → 13k
   - Mouse, keyboard
   - Apple multiport Adapter
   - 15k Smartwatch

2. Food → Breakfast + Lunch + Snacks + Barista

3. Uber → 20 dollars → 1500 Rs

1 Unique BST

A > B

N=0

empty BST

root==null

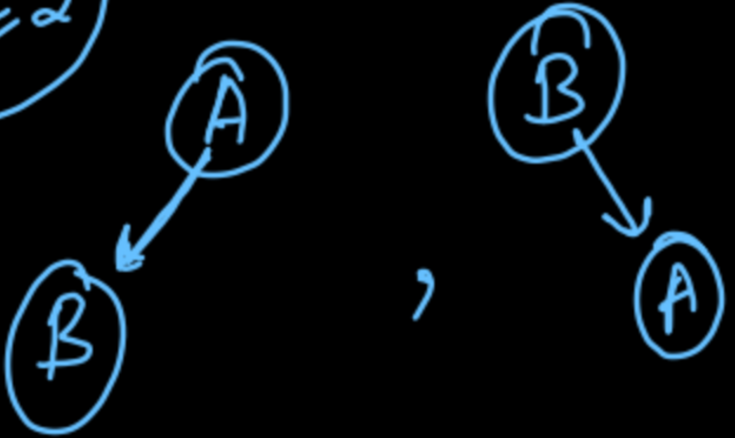
1

N=1

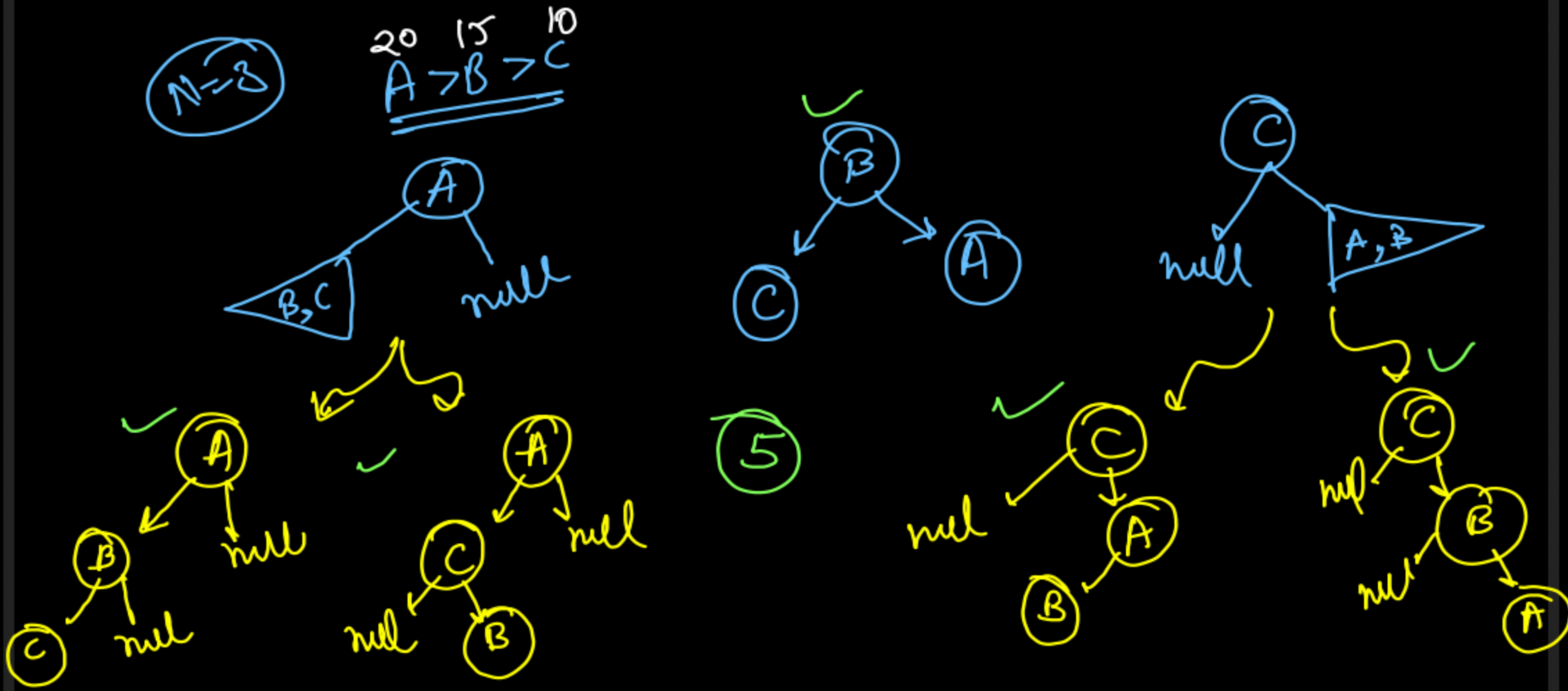
A

1

N=2



2



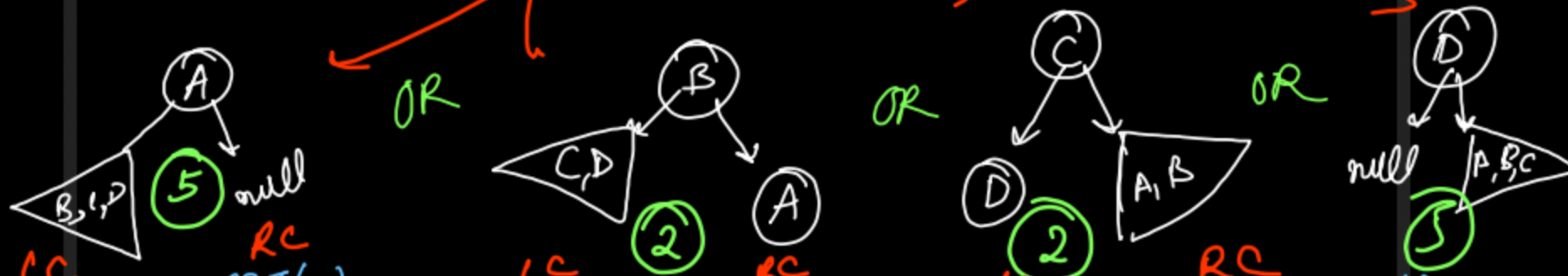
Unique BST(n) = UBST(n) with k_1 as root + k_2 as root + k_3 as root + -- k_n as root

$N = 4$

20 15 10 5

$A > B > C > D$

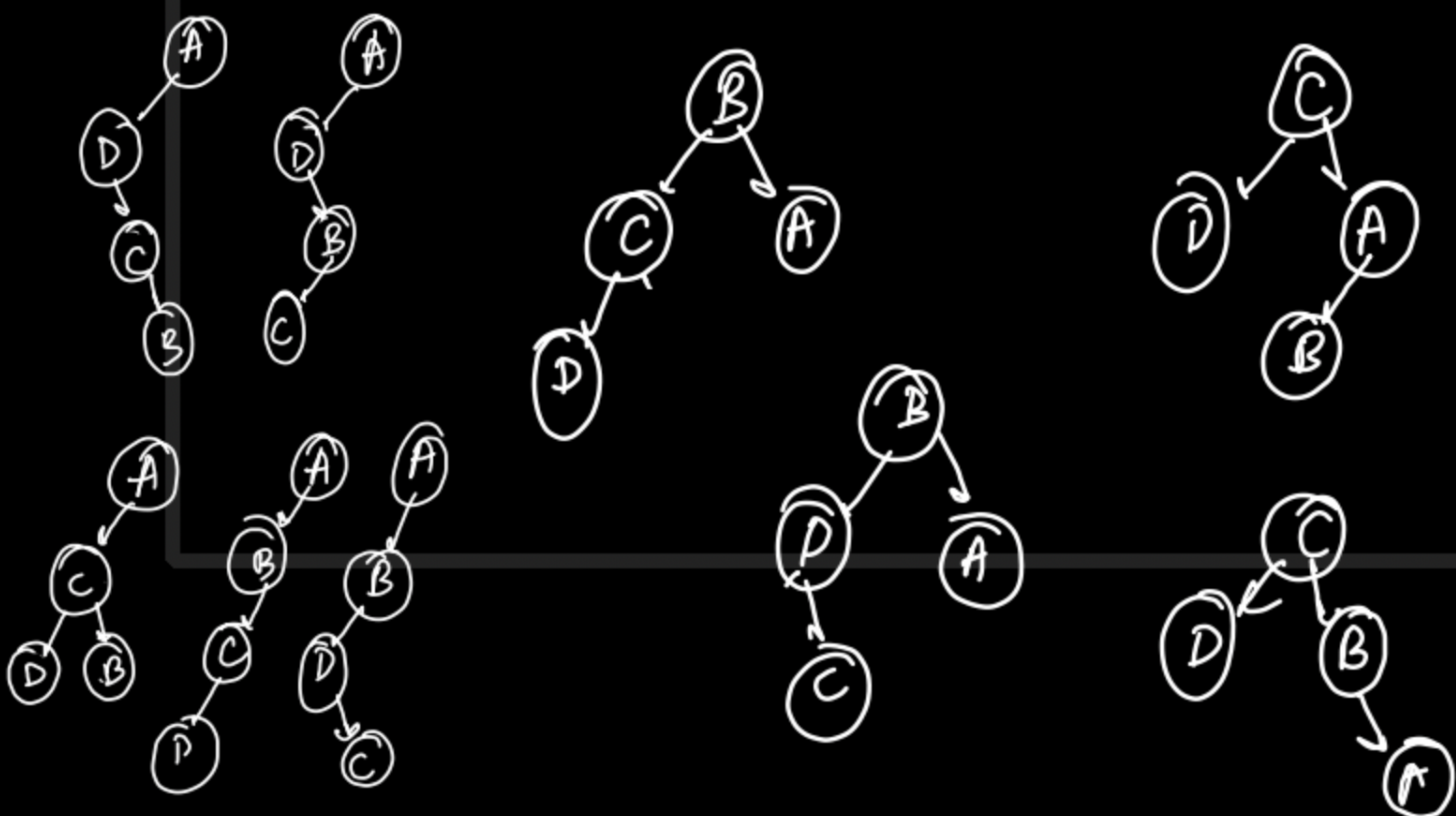
$5 + 2 + 2 + 5 = 14$



LC RC LC RC LC

$UBST(3) * UBST(0) + UBST(2) * UBST(1) + UBST(1) * UBST(2) + UBST(0) * UBST(3)$

LC * RC



② Count Balanced Paranthesess

N → opening brackets, closing brackets } pairs

N=0

" "

1

N=2

"(())"

"()()"

2

N=1

"()"

1

( )

(1),(0) → (())

(0),(1) → ()()

N=3

(())()

()(())

()()()

((()))

(()())

5

( )

(2)(0) → ((())) , (()())

(1)(1) → (())()

(0)(2) → ()()() , ()(())

N=4

$$C_4 = C_0 * C_3 + C_1 * C_2 + C_2 * C_1 + C_3 * C_0$$

( )

(3)0 + (2)1 + (1)2 + (0)3 = 14

5 | 2 | 2 | 5

(()()())
(()(()))
((())())
(((())))
((()()))

(()())()
((()))()

(())()()
(())(())

()()()()
()()(())
()(())()
()((()))
()(()())

③ Count Mountain Ranges

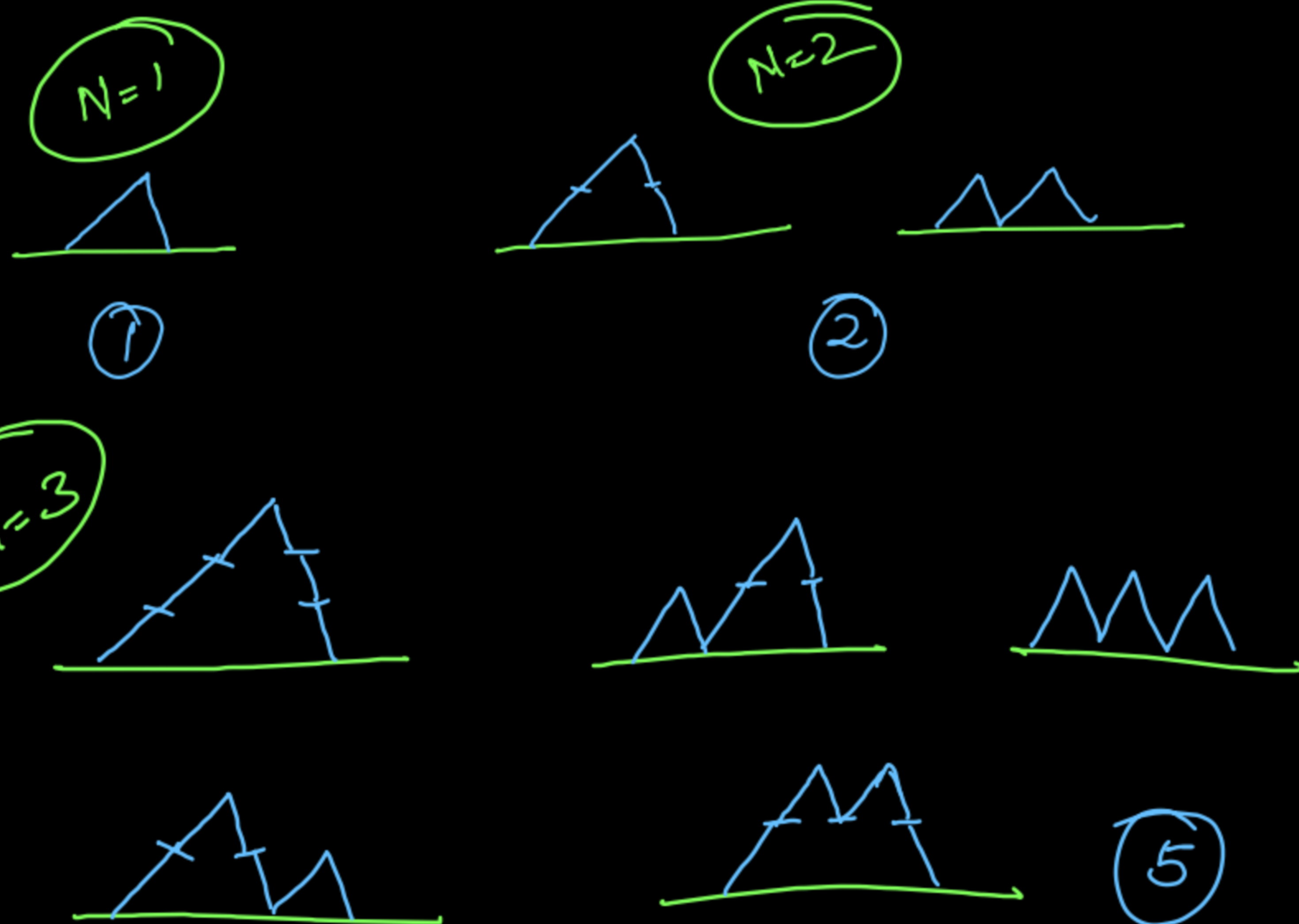
n uphills, n downhills ⇒ above or on the sea level

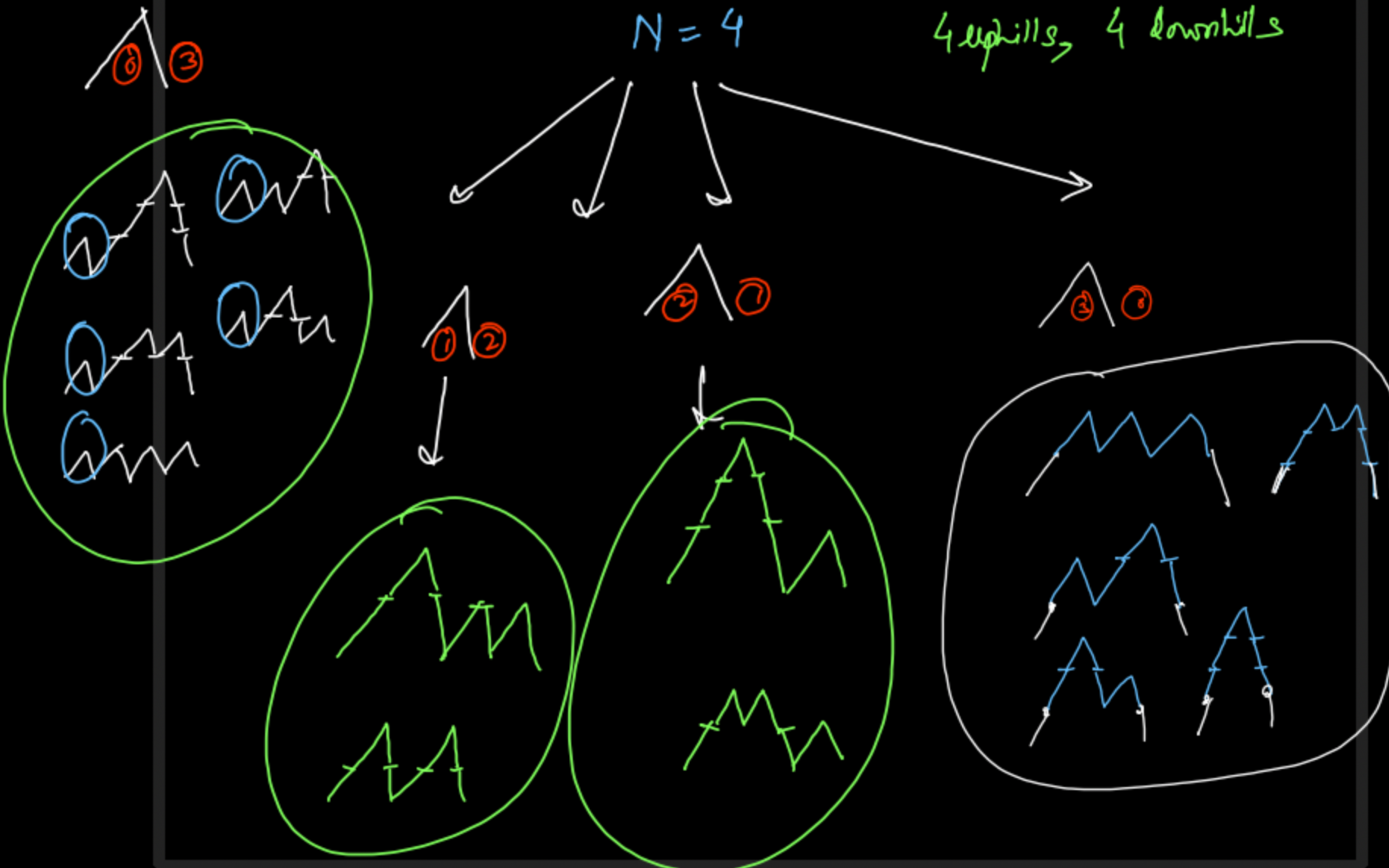
uphill → (

downhill →)

⇕

count balanced strings

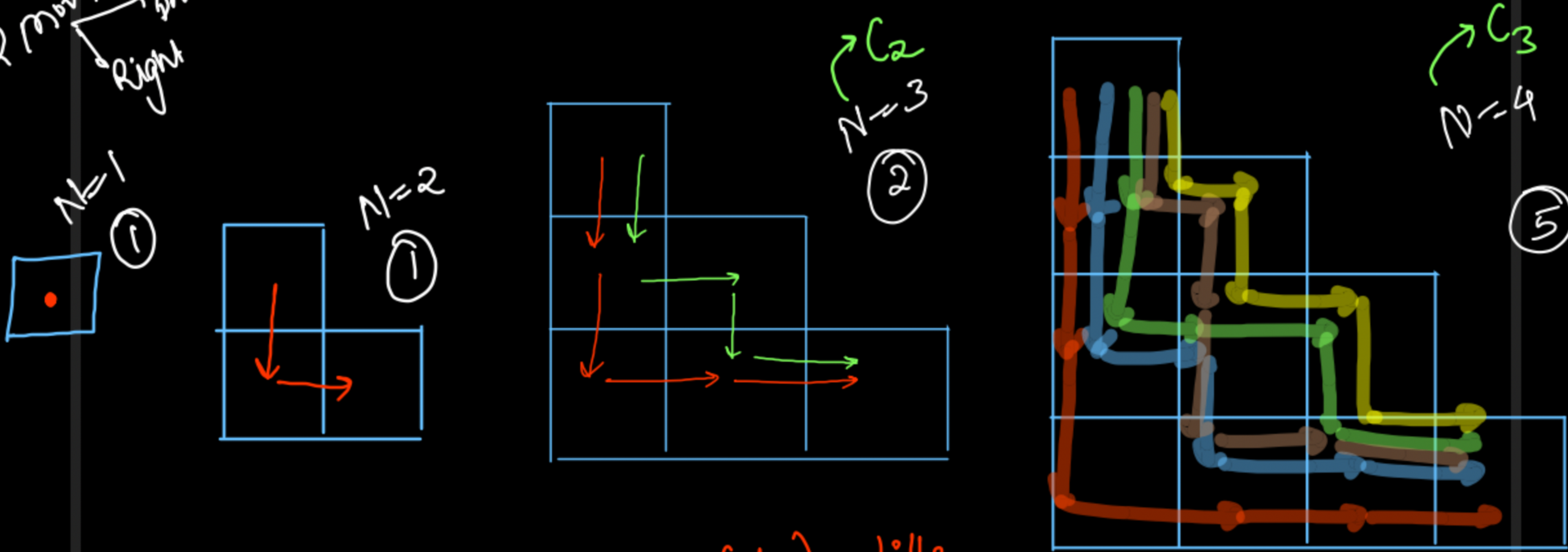



N = 4
4 uphills, 4 downhills
0 3
1 2
2 1
3 0

If entire matrix
↳ UniquePaths-1
↳ $\frac{(n+m-2)!}{(n-1)!(m-1)!}$

④ Count Paths in Upper-Right Triangle or Lower-Left Triangle

2 moves: down, Right



$(N \text{ rows}, N \text{ cols}) \Rightarrow C_{N-1} \Rightarrow (N-1)$ uphills $+ (N-1)$ downhills

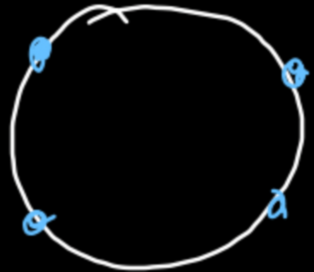
5 Handshakes / Non-Intersecting chords

4

Every person should do a handshake, and no person can do 2 handshakes

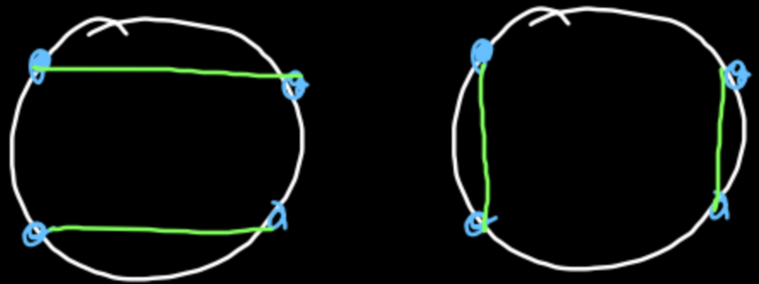
$N=2$

1



$N=4$

2

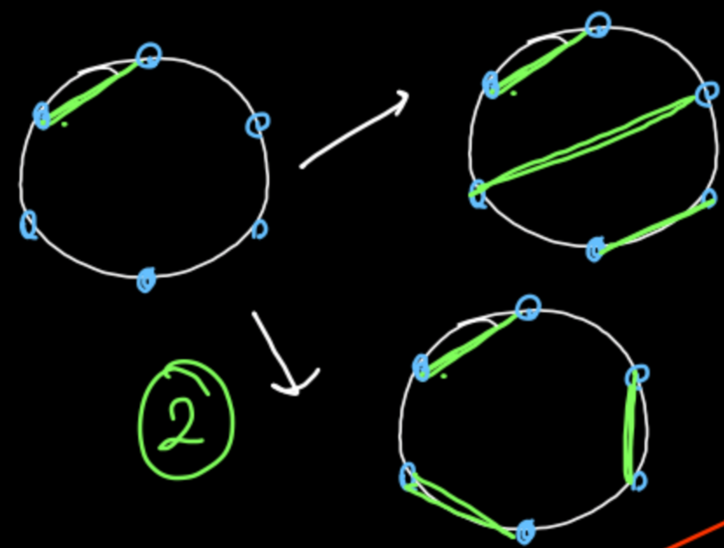


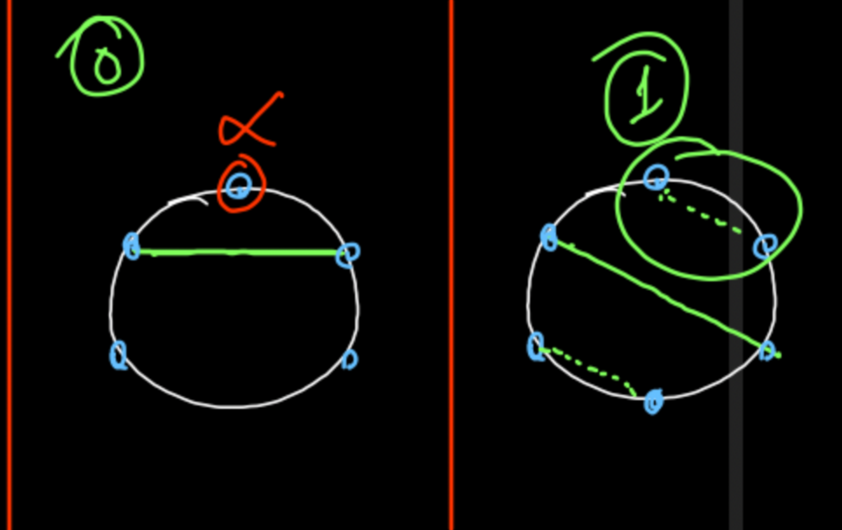
C_0, C_1, C_2, C_3, C_4

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

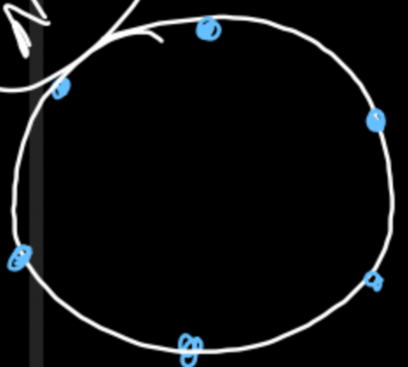
1 1 2 5 14

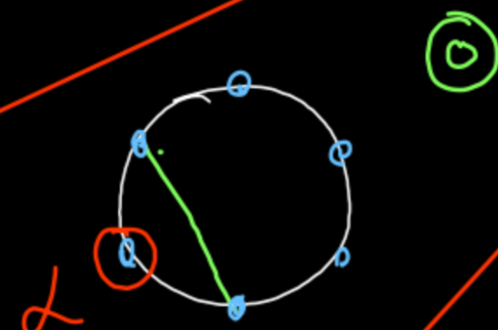
$N=0$ 1 C_0
$N=2$ 1 C_1
$N=4$ 2 C_2 $\rightarrow C_{N/2}$
$N=6$ 5 C_3
$N=8$ 14 C_4

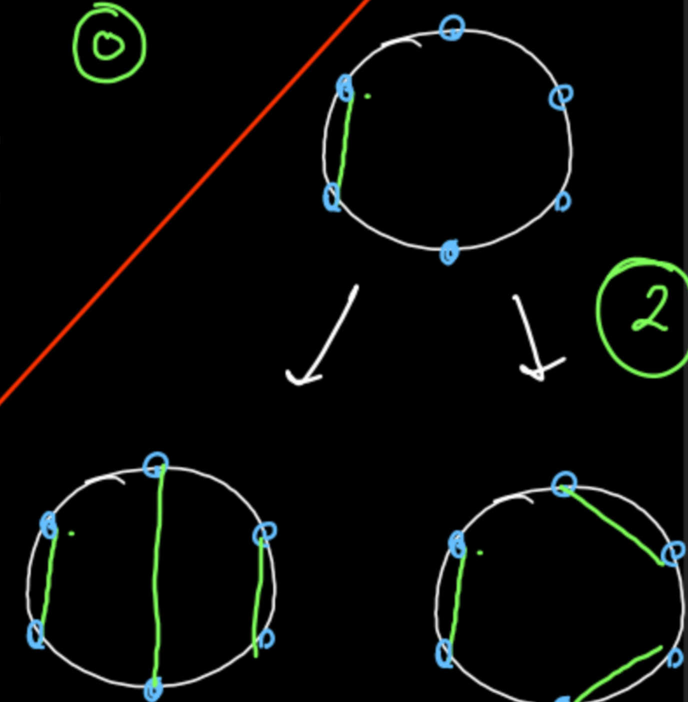




$N=6$ $= 5$







$(N=6) \rightarrow C_3 = C_0 * C_2 + C_1 * C_1 + C_2 * C_0$

Dyck words

X Y C_N

substring starting with 0th index
↑
in all prefix
↳ count of x ≥ count of y

x = '('   y = ')'

count of balanced paranthesis strings

N = 2

XYXY
XXYY

2

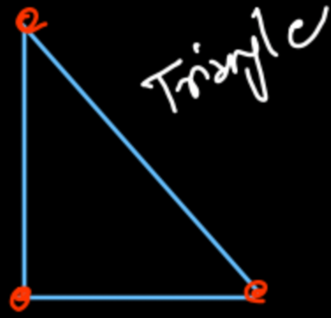
N = 3

xyxyxy
xxyyxy
xyxxyy
xxxyyy
xxyxyy

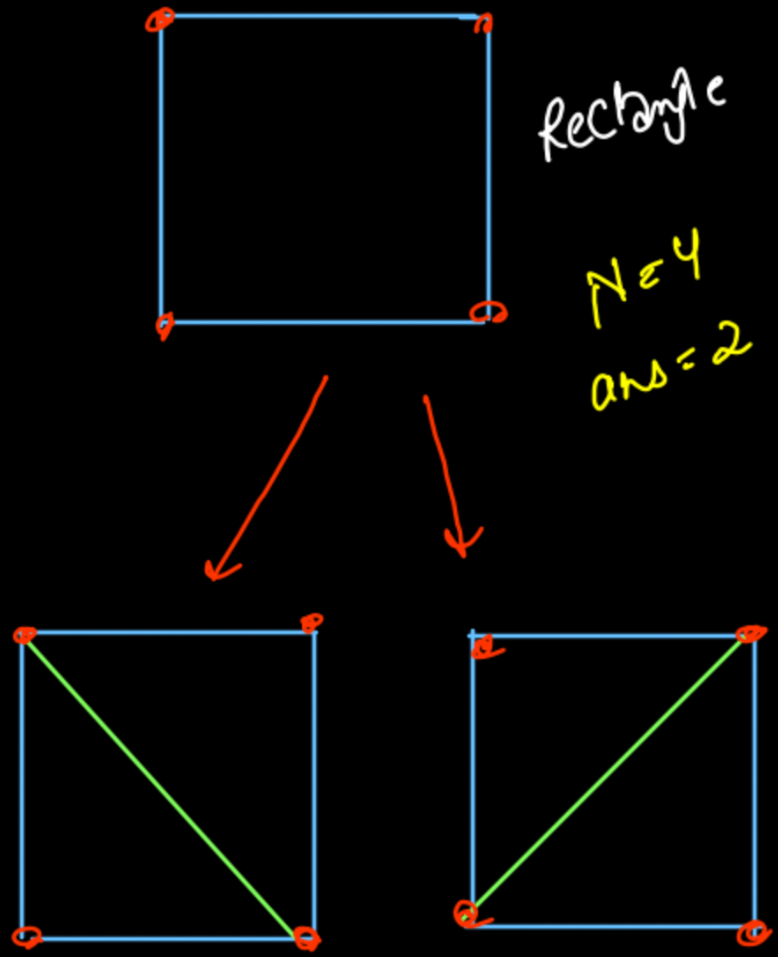
5

8 Count ways of Triangulation

N = 2
ans = 1



N = 3
ans = 1



Rectangle

N = 4
ans = 2

$N=2 \Rightarrow 1 \longleftrightarrow C_0$

$N=3 \Rightarrow 1 \longleftrightarrow C_1$

$N=4 \Rightarrow 2 \longleftrightarrow C_2$

$N=5 \Rightarrow 5 \longleftrightarrow C_3$

$N=6 \Rightarrow 14 \longleftrightarrow C_4$

$$ans(N) = C_{N-2}$$

$$\text{Polygon}(n) = \sum \text{Left}(i) + \text{Right}(j)$$