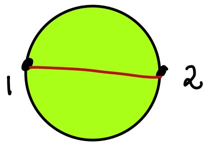
Today's Content:

- → Interesting Chords in a Circle → {Catalyn Numbers}
- → Palindrome Partitioning - II

18) Interesting Chords

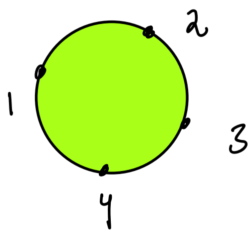
// Given 2A points, number of ways we can draw A chords in a circle with 2A points such that no 2 chords intersect

Ex1:



$F(1) = 1$

Ex2:

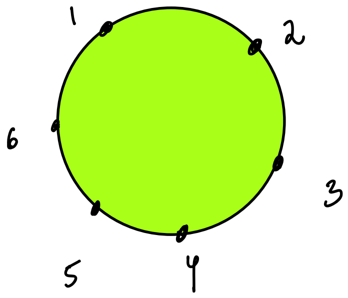


$F(2) = 2$

{ 1-2, 4-3

1-4, 2-3 }

Ex3:



$F(3) =$ {

1-2 6-3 5-4

1-2 6-5 4-3

1-4 2-3 6-5

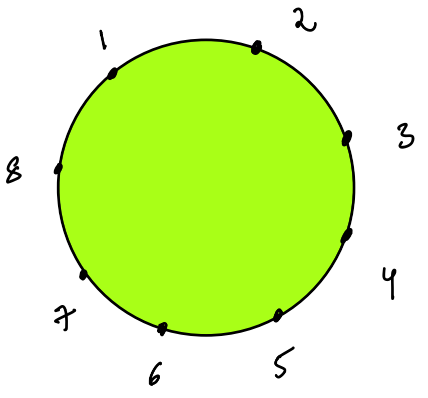
1-6 2-5 4-3

1-6 2-3 5-4 }

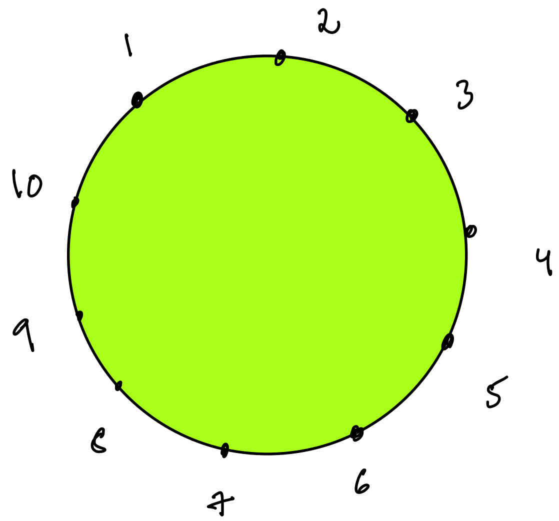
5 ways

En4:

$F(4) = F(3) + F(2)F(1) + F(2)F(1) + F(3)$



En5:



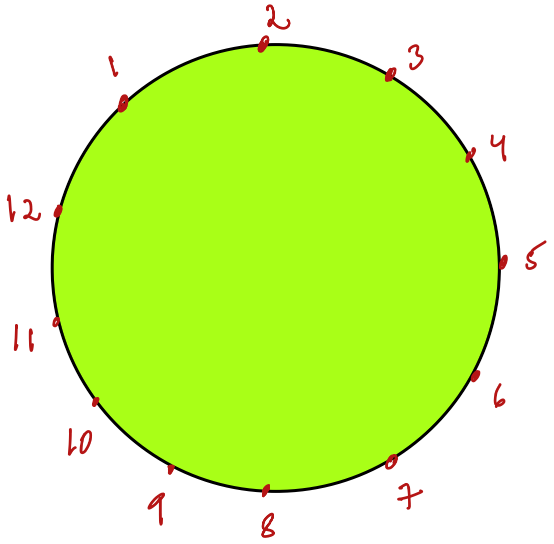
$F(5) = F(4) + F(1)*F(3) + F(2)*F(2) + F(3)*F(1) + F(4)$

$F(5) = F(4)*F(0) + F(1)*F(3) + F(2)*F(2) + F(3)*F(1) + F(4)*F(0)$

i=5

$$\left\{ \begin{array}{l} j = 4 \to 0 \\ k = 0 \to 4 \end{array} \right\}$$

Ex6:



5 * 2 = 10

{1-6} | {12-11} {10-9} {8-7} | {2-3} + {4-5}

{12-11} {10-7} {9-8} | {2-5} + {3-4}

{12-9} {11-10} {8-7}

{12-7} {11-10} {9-8}

{12-7} {11-8} {10-9}

$F(6) = f(5)^* f(0) + f(1)^* f(4) + f(2)\, f(3) + f(3)(2) + f(4)(1) + f(5)^* f(0)$

$j \to 5 \to 0$, $k \to 0-5$

$F(N) =$

$f(N-1)^* f(0) + f(1) f(N-2) + f(2)(N-3) + f(3)(N-4) + \dots f(N-2) f(1) + f(N-1)^* f(0)$

$f(0) = 1 \quad f(1) = 1$

// dp[N+1]

// dp[0] = 1, dp[1] = 1

TC: (N) * N → O(N²)

SC: O(N)

```
i = 2 ; i <= N ; i++) {
    dp[i] : { j : i-1 → 0 , k : 0 → i-1 }
    S=0, k=0
    j = i-1 ; j >= 0 ; j--) {
        S = S + dp[j] * dp[k]
        k++
    }
    dp[i] = s;
}
```

// return dp[N]

// N pairs of ( ) how many balanced can we get? → {Same sequence}

N=1 : ( ) } → 1

N=2 : (( )) ( ) ( ) } → 2

N=3 :

( ) * ( ) ( )

( ) (( ))

(( )) ( )

(( ) ( ))

((( )))

} → 5

(→ +1

) → −1

at any given point it will never become negative

N=4 :

(f(0)) * f(3)

(f(1)) * f(2)

(f(2)) * f(1)

(f(3)) * f(0)

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

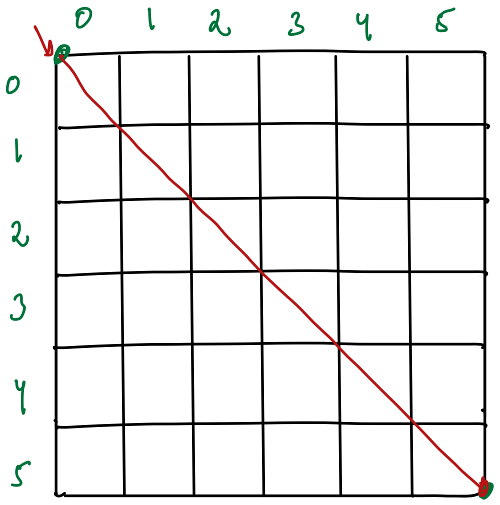
// Constraint: $N = 10^3$,

output % $(10^9 + 7)$

// Sequence : → {Catalan Sequence} → {pdf → 20/25 Catalan Numbers}

0	1	2	3	4	5	6	……	N^{th}
1	1	2	5	14	42	132		$\boxed{\frac{{}^{2N}C_N}{N+1}}$ % $\{10^9 + 7\}$

// given N×N matrix



// Top left → Bottom right

→ Bottom, right

→ Number of ways to reach TL & BR

→ Path cannot cross diagonal

→ (→ : +1

(↓ : -1

// any any point it cannot be negative

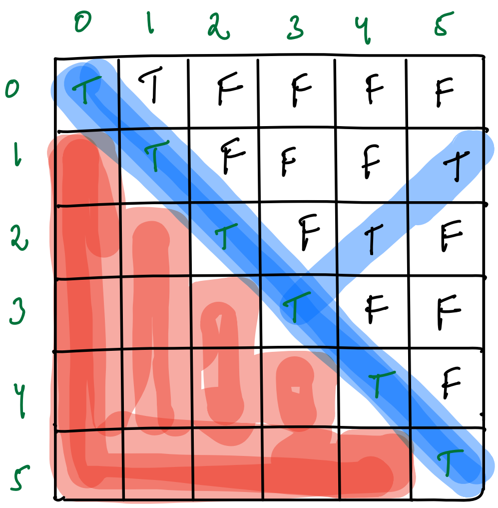
→ : (: +1

↓ :) : -1

2Q) Given a String S[N] construct a bool mat[N][N]

mat[i][j] = (True)
{ if substring [i-j] is a palindrome
i <= j

	0	1	2	3	4	5
String S =	b	b	d	a	d	b



	0	1	2	3	4	5
0	T	T	F	F	F	F
1		T	F	F	F	T
2			T	F	T	F
3				T	F	F
4					T	F
5						T

→ We need to fill all value

// dp[N][N]

dp[i][j] = { // check if substring [i j] is Palindrome }

i i+1 i+2 --- j-2 j-1 j

$$\boxed{dp[i][j] = S[i] == S[j] \ \&\& \ dp[i+1][j-1]}$$

// TC: O(N2) x O(1) → O(N2)

// SC: O(N2)

208) Palindrome Partitioning - II

// Minimum number of strings s required to make each Partion should be palindrome

Ex1: a n a | c | o n o | a a } ans = 4

0 1 2 3 4 5 6 7 8 9 10 11

Ex2: a | b c b | a b a a a b a | a → 4 → {0-0, 1-3, 4-10, 11-11}

a b c b a | b a a a b | a a → 3

a | b c b | a b a | a a b a a → 4

→ ans = 3

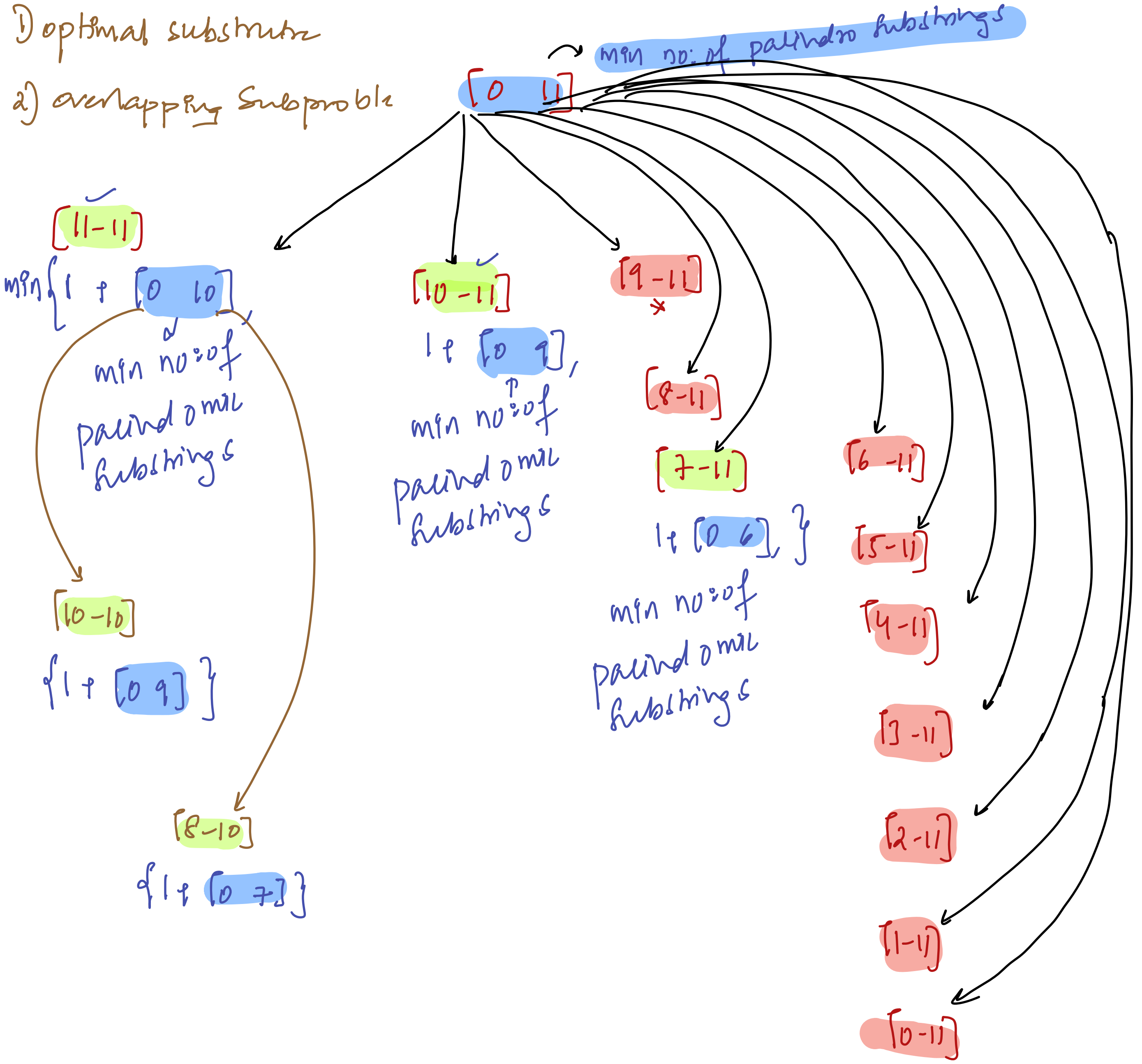
Ex3: | a | b | c | d | e → ans = 5

Ex4: a b c b a → ans = 1

	0	1	2	3	4	5	6	7	8	9	10	11
Eg2:	a	b	c	b	a	b	a	a	a	b	a	a

1) optimal substructure

2) overlapping Subproblem



// dp[i] = { for substring [0 i], calculate min number of palindrom substrings }

```
s: 0 1 2 3 . . — . i-4 i-3 i-2 i-1 i

dp[i] = {
    ans = N
    j = i; j>=0; j--){                      0 ... j-1 j ... i
        ispali(s, j, i){ → // [j i] check palindrom or not?
            ans = min(ans, dp[j-1]+1)
                           if j=0 : dp[-1]
        }
    }
    dp[i] = ans
}

return dp[N-1]
```

Note: To solve this we can simply create a bool mat[N][N] where mat[i][j] contains whether substring [i j] is palindrome or not

TC: Construct mat[][] + N * {N}

↓

$O(N^2)$

TC: $O(N^2)$

SC: $O(N^2) + O(N) \approx \underline{\underline{O(N^2)}}$