Pascal Δ

Row

0 first col = 1 1 last col = 1

1 1 sum 1

2 1 sum 2 sum 1

3 1 sum 3 sum 3 sum 1

4 1 4 6 4 1

0 1 2 3 4 5 6 7 8

There are three questions

(1) Give R and C tell the element at that place

∴ R = 5 C = 3 ans = 6

(2) Print any n^{th} row of Pascal Δ

∴ N = 4 ⇒ 1 3 3 1

(3) Draw complete Pascal Δ

① formula → $R-1 \; C \; C-1$

for ex $R=5 \quad C=3$

$5-1 \; C \; 3-1 \Rightarrow {}^{4}C_{2}$

${}^{n}C_{r} \Rightarrow \frac{n!}{r!(n-r)!}$ $\Rightarrow \frac{4!}{2!(4-2)!} \Rightarrow \frac{4!}{2! \times 2!}$

$\Rightarrow \frac{4 \times 3 \times 2 \times 1}{2 \times 1 \times 2 \times 1} = 2 \times 3 \Rightarrow 6$

using optimizing method

```
fun nCr (n-1, (-1)
```

---

① use loop for $n!$, $r!$, $(n-r)!$

② use optimizing method → TC $\Rightarrow O(r)+O(n)+O(n-r)$

optimizing ⇒ dont use for loop / bruteforce

${}^{7}C_{2} \Rightarrow \frac{7!}{2!(7-2)!} \Rightarrow \frac{7!}{2! \times 5!} \Rightarrow \frac{7 \times 6 \times 5!}{2 \times 1 \times 5!}$

${}^{10}C_{3} \Rightarrow \frac{10!}{3! \times 7!} \Rightarrow \frac{10 \times 8 \times 7!}{3 \times 2 \times 1 \times 7!}$

$^{n}C_{3} \Rightarrow \frac{10 \times 9 \times 7}{3 \times 2 \times 1}$

```
fun nCR (n, r)
{
    res = 1;
    for (i = 0 ; i < r ; i++)
    {
        res = res * (n - i) ;
        res = res / (i + 1) ;
    }
    return res;
}
```

because we know loop will run r times

Or we can say that

$\frac{10 \times 8 \times 7}{1 \times 2 \times 3}$

$\frac{10}{1} \times \frac{8}{2} \times \frac{7}{3}$

10 is basically n → n-0 → n-i

9 → n-1 → n-i

8 → n-2 → n-i

becoz for i = 0 we take 1

i = 1 → 2

i = 2 → 3

Time complexity → O(r)

Space complexity → O(1)

(2) N row → n elements

we know that

$^{n-1}C_{c-1}$

```
for( c=1 ; c <=n ;c++)
{
    print (funNcR ( n-1, c-1));
}
```

Time complexity = O(N x n)

```
num = 11;
num = num - n;
for ( row = 0; row < n; row++ )
{
    for( col = 0; col < n-row; col++)
    {
        cout << num++;
    }
    num--;
    num = num - 2(n-1-row);
    cout << endl;
}
```

n=4

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

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

n=4

count = 1 2 3 4 5

10 - 5 = 5