Recursion-3

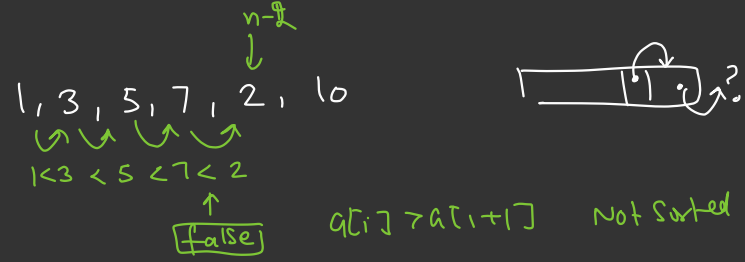
(I) Given an array of size N, check if the array is sorted

[Recursion]

arr = [1, 3, 2, 5, 7] false

arr = [1, 3, 5, 7, 10] true

Solution ⇒ Loop

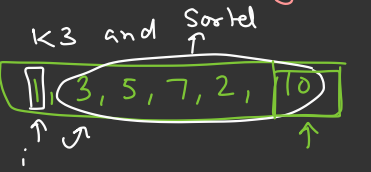



```
for(i=0, i<=n-2; i++){
    if(a[i] > a[i+1]){
        return false;
    }
}
```

```
    }
    return true
```

Rec

↳ Solve for a single element & Recursion for Rest of array



Rec Code →

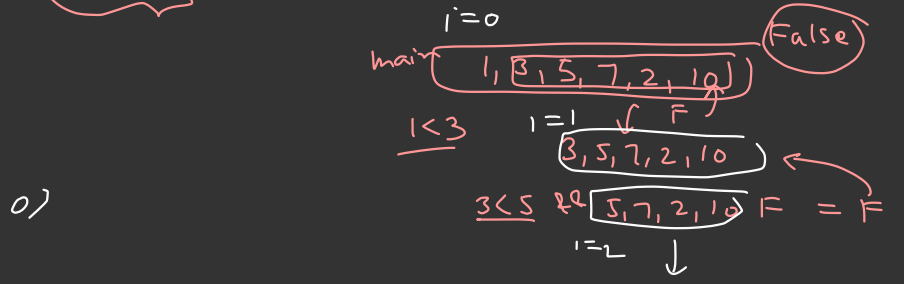
```
bool isSorted(arr, i) {
    // Base Case
    if(i == n-1){        // n -> arr.length
        return true;
    }
    // Rec case
    return a[i] < a[i+1] && isSorted(arr, i+1);
}
```

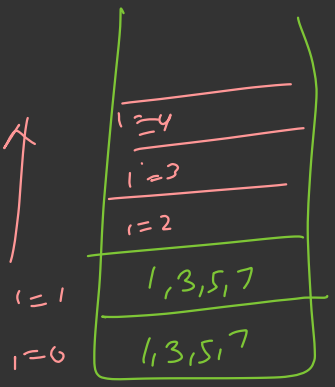
a[i] < a[i+1] && isSorted(arr, i+1)

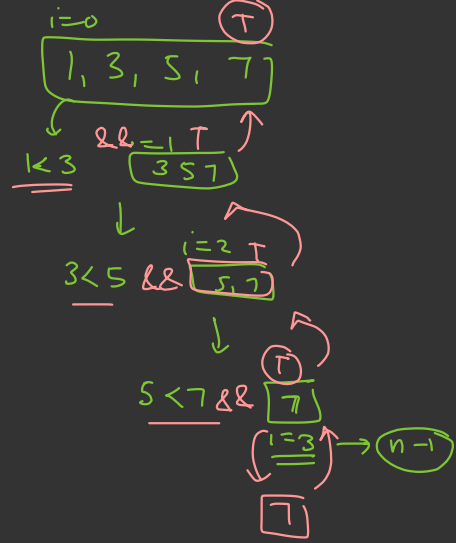
Time → O(N)

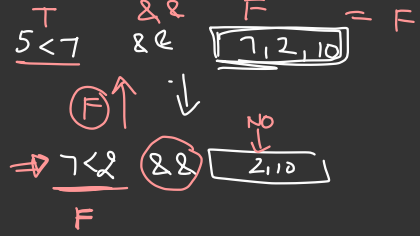
Space → O(N)

isSorted(arr, 0)

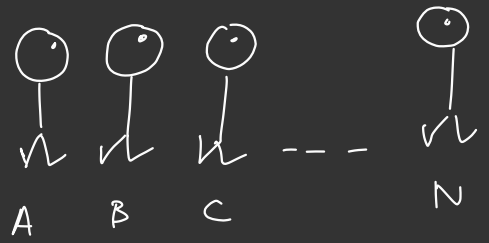



i=4
i=3
i=2
1,3,5,7
i=1
1,3,5,7
i=0


i=0
T
1, 3, 5, 7
&&=1 T
3 5 7
1<3
i=2 T
3<5 &&
5,7
T
5<7 &&
7
i=3
n-1
7


T
5<7
&&
&&
F
= F
7,2,10
F
NO
7<2
&&
2,10
F

⑨ Party Problem



→ 1 person (Solo)

→ 2 person (couple/pair)

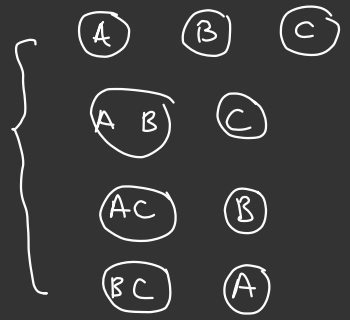
in how many ways they can go to the party

N = 3

A, B, C

output → 4

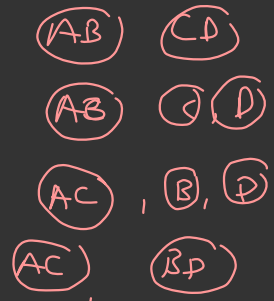
N = 1	1
N = 2	2
N = 3	4
N = 4	?



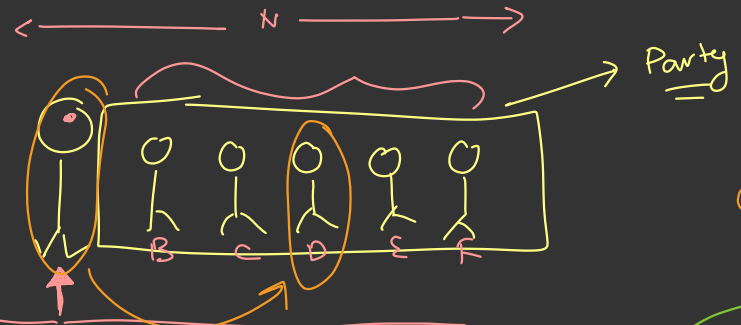
All Solo

N = 4

A, B, C, D



Many ways



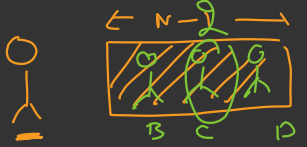
$f(N) =$

Solo

OR

Pair

$1 * f(N-1)$ $\quad + \quad$ $(N-1) * f(N-2)$



Solo $\quad 1 \cdot f(n-1)$

$+$

Pair $\quad (N-1) \cdot f(N-2)$

$$f(N) = f(N-1) + (N-1)\, f(N-2)$$

$N == 1 \rightarrow 1$

$N == 2 \rightarrow 2$

$f(2) = 2$

$f(3) = 4$

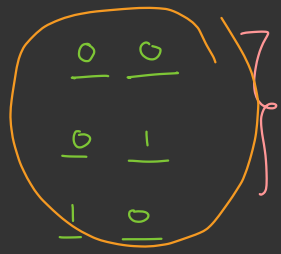
$f(4) = 10$

$$f(3) = f(2) + (3-1)f(1)$$
$$= 2 + 2.1 = 4$$

$$f(4) = f(3) + (4-1)f(2)$$
$$= 4 + 3 \times 2 = 10$$

(2) Given a int N, find No of ways to Build Boolean Strings, with No Consecutive one's of length N

N = 2



3 valid Strings

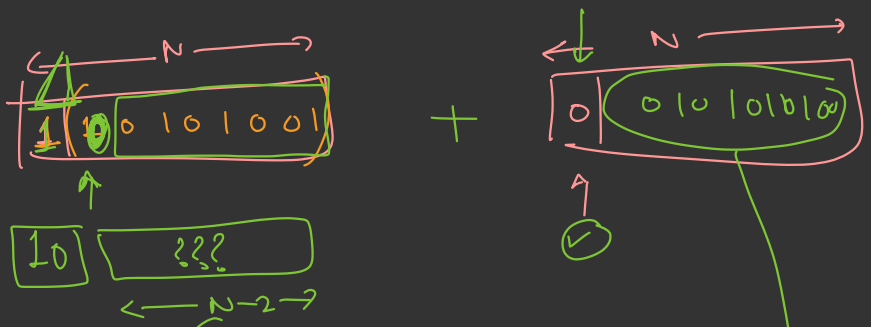
→ 1 1 ✗

N = 3

0, 0, 0
0, 0, 1
0, 1, 0
1, 0, 0
1, 0, 1

5 valid Strings.

0, 1, 1 ✗
1, 1, 0 ✗



N=1

0 / 1 → 2 ways

f(N)

No of ways to build a binary NO consecutive one

⇒ f(N) = f(n-2) + f(N-1) ways

Base

N=0 → 1

N=1 → 2

N=2 → 3

N=3 → 5

N=4 ⇒ 8 // TODO

f(2) = f(1) + f(0)

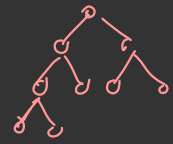
⇒ 2 = 1 + f(0)

⇒ f(0) = 1

Same as fibo

Time $O(2^N)$

Space $O(N)$



$f(2) = f(0) + f(1)$

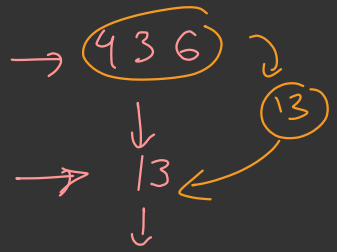
$= 1 + 2$

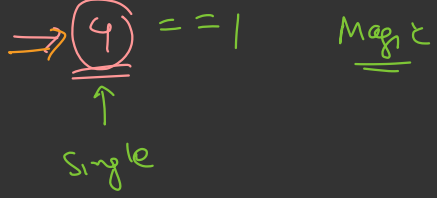
$= 3$

Is Magic ?

(9) Given a number A, check if it is a magic number or not.

A number is said to be a magic number if the sum of its digits is calculated till a single digit recursively by adding the sum of the digits after every addition. If the single digit comes out to be 1, then the number is a magic number.






```
bool isMagic(int N) {
    if (N < 10) {
        N == 1 → return true
               → return false
    }
    int sum = sumOfDigits(N);
    return isMagic(sum);
}
```

sumofDigits(436) {

4+3+6

= 13

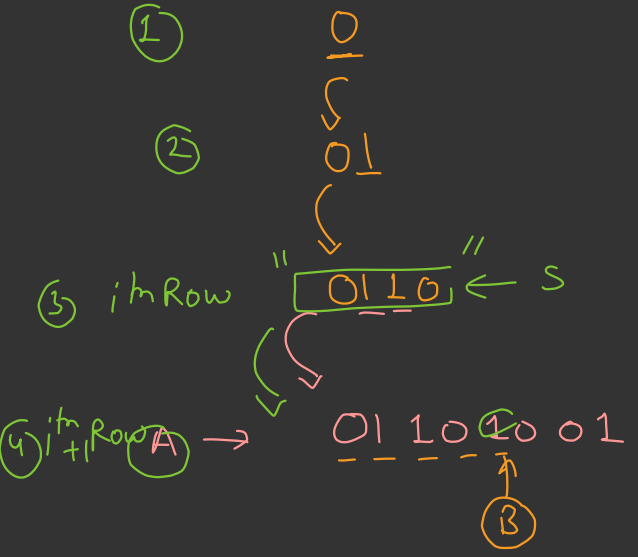
3

3

K^{th} Symbol (TRICKY)

On the first row, we write a 0. Now in every subsequent row, we look at the previous row and replace each occurrence of 0 with 01, and each occurrence of 1 with 10.

Given row number A and index B, return the Bth indexed symbol in row A. (The values of B are 1-indexed.).

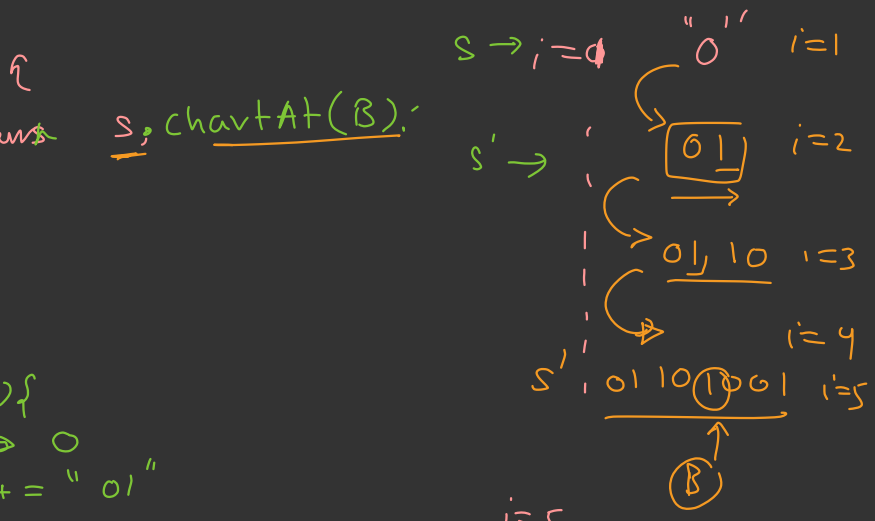



```
s' = "";
for(i=0, i< s.length(), i++){
    if (s.charAt(i) == '0'){
        s' += "01"
    else{
        s' += "10"
    }
}
```

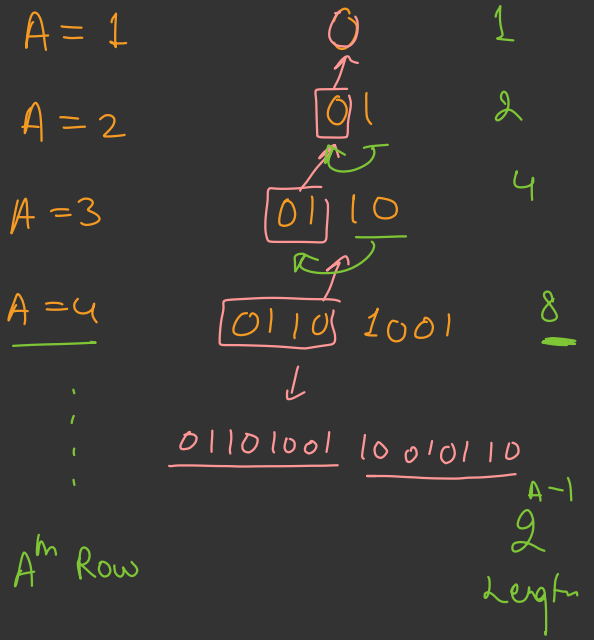
```
String generateString( string s, i, A ){
    // Base Case
    if (i == A){
        return s.charAt(B);
    }

    // Rec Case
    s' = ""
    for( — s){
        letter → 0
            s' += "01"
        else s' += "10"
    }
    generateStrig (s', i+1, A).
}

generateString( "0", 1, 5)
```



Another way

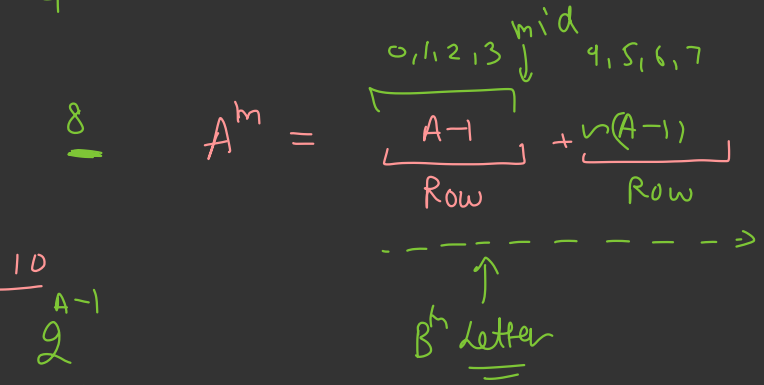


2^3

0 → 01
1 → 10

l = 8

A^m = (A−1) Row + ∩(A−1) Row



B < mid ⇒ first

B ≥ mid ⇒ second part

A → A−1 → A−2 → ...

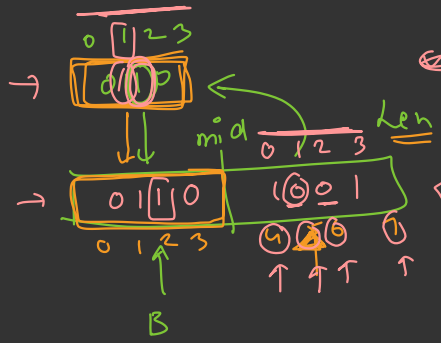
$f(A) = f(A-1),\ \cap f(A-1)$

B

A == 0 → 0

```
int getDigit (int A, int B) {
    if (A == 0)
        return 0;
    // R
    int len = 1<<(A-1)        // 2^(A-1)
```

O(1) ← 1<<(A-1) ← O(A) or O(log A)



0 = n(L)

5 digit → (1st digit of R)

5 - 8/2

B = 5

```
    if (B is in first half) {
        return getDigit(A-1, B);
    }
    else {
        return getDigit(A-1, B - len/2);
    }
}
```

3

3

Grey Code

The gray code is a binary numeral system where two successive values differ in only one bit.

Given a non-negative integer A representing the total number of bits in the code, print the sequence of gray code.

A gray code sequence must begin with 0.