

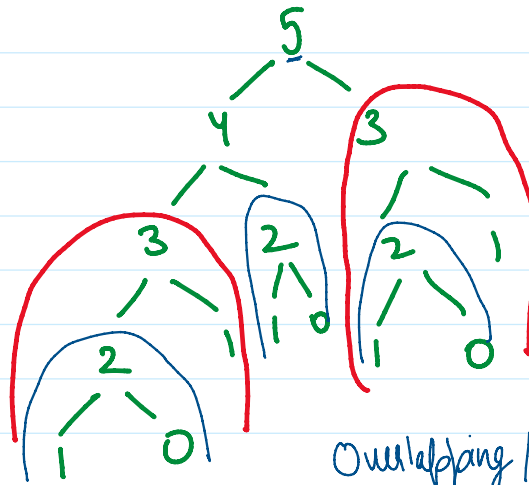
DP

09 April 2023 09:11

↳ Recursion



Fibonacci



2

2^n

$$1 + 2 + 2^2 + 2^3 + \dots = 2^n$$

Overlapping / Repeating Subproblems

Input args kitne $\rightarrow 1$ (n)

\rightarrow Range : 0 to n

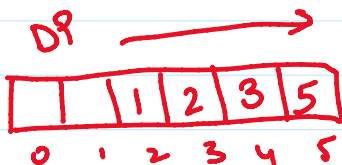
total recursive states ? $\rightarrow 2^n$

Unique subproblems ? $\rightarrow \underline{n+1}$

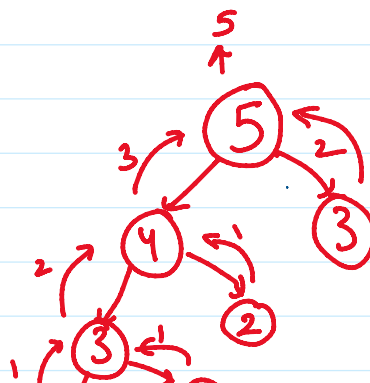
Agar kisi D.S mein kisi subproblem ka ans store kervale and usse use krke toh best krjayege?

\rightarrow Array

HashMap



```
public static int FiboTD(int n, int[] dp) {
    if (n <= 1) {
        return n;
    }
    // ...
}
```



k: V.
0 to n
n+1
O(1)

Stack Memory??
 \rightarrow

```

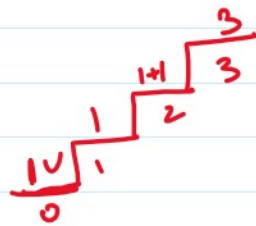
if (n <= 1) { 3
    return n;
}
if (dp[n] != 0) {
    return dp[n];
}
int sp1 = FiboTD(n - 1, dp); 3
int sp2 = FiboTD(n - 2, dp); 2
dp[n] = sp1 + sp2; 5
return sp1 + sp2;
}

```

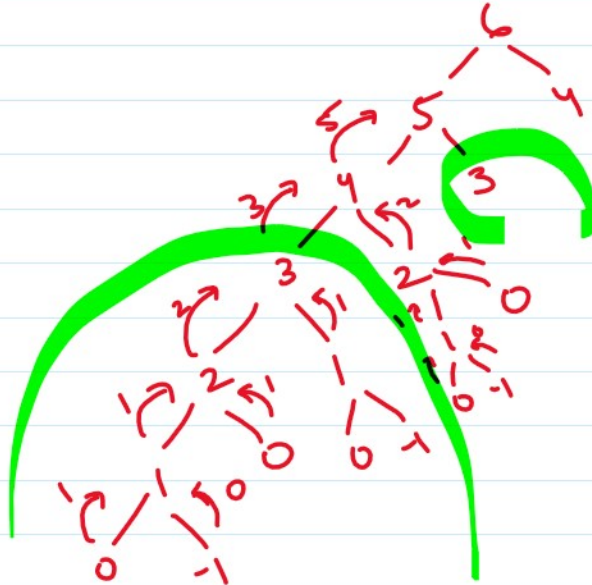


Stack memory!
 4
 10000

Climbing Stairs

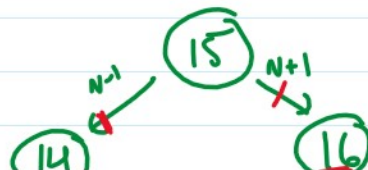


1 1
 2 1
 1 2



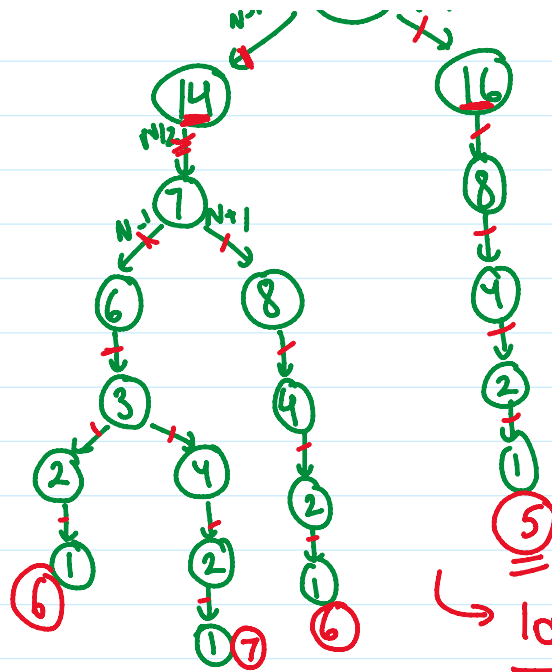
2^n
 $\frac{n+1}{2}$
 Go to n

20001
 1 20



1) Odd
 $N-1$
 $2^{N/2} \times \dots \rightarrow N/2$

2000
 1 + 10000
 2 10000
 5000
 2500
 1250
 625
 312
 156



$$2^{\frac{n}{2}} \times \frac{n}{2}$$

Odd $\rightarrow 2$
 Even $\rightarrow 1$

$$\frac{n}{2}$$

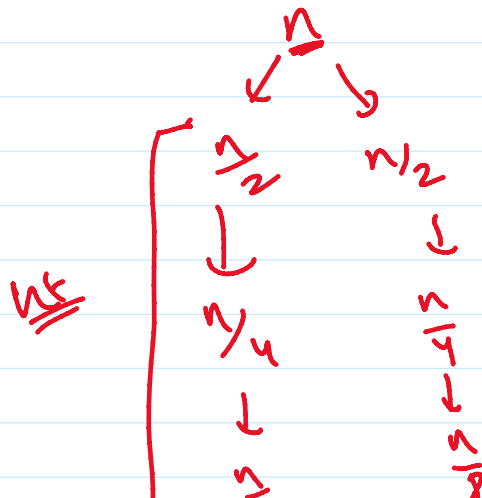
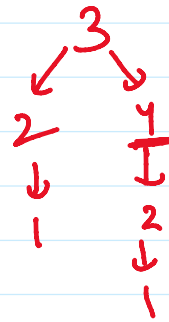
$$1 + \frac{n}{2} \approx \frac{n}{2}$$

$\log N \rightarrow$

9

```
public static int opsBU(int n) {
    int[] dp = new int[n+1];
    for(int i=2; i<=n; i++) {
        if(i%2==0) {
            dp[i]=1+dp[i/2];
        } else {
            int sp1 = 1+dp[i-1];
            int sp2 = 1+dp[i+1];
            int res= Math.min(sp1, sp2);
            dp[i]=res;
        }
    }
    return dp[n];
}
```

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

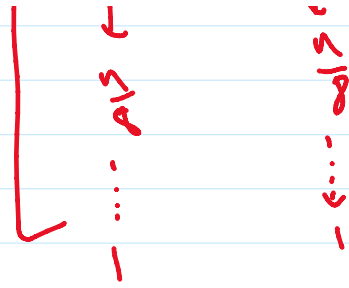


$$2 \cdot \left(\frac{n}{2} + \frac{n}{4} + \frac{n}{8} + \dots + 1 \right)$$

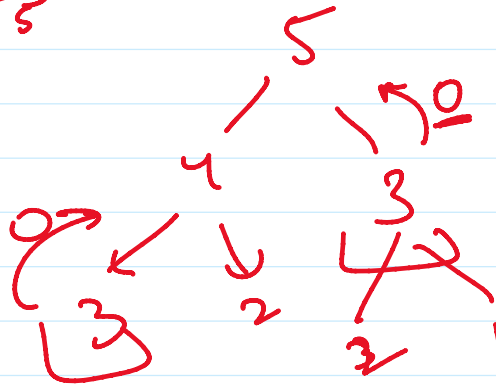
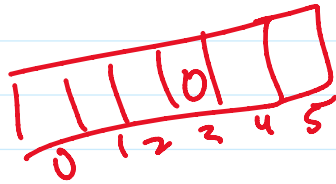
$0 < a < 1$

$$2 \cdot \left(\frac{a}{1-a} \right)$$

$$2 \cdot \left(\frac{a}{1-\frac{1}{2}} \right)$$



$$2 \left(\frac{a}{1 - \frac{1}{2}} \right) = \boxed{\frac{2a}{1}} = \boxed{2a}$$



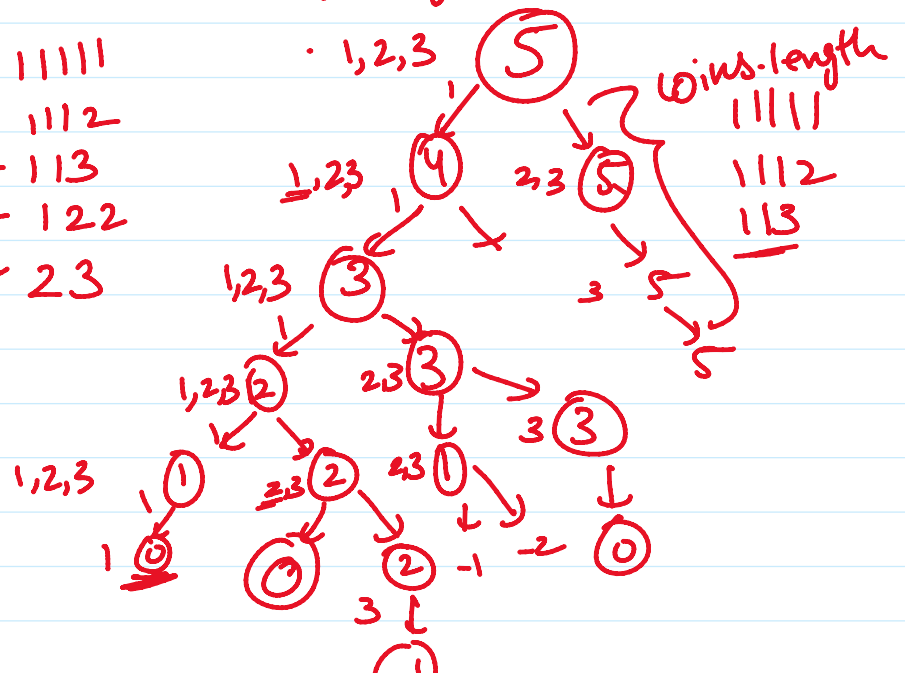
Coin change combinations

Amount = 5

coins = {1, 2, 3}

How many ways to change??

- 11111
- 1112
- 5 - 113
- = - 122
- 23



Recursive calls $\rightarrow 2^{nt}$

$\hookrightarrow 2^{A+cl}$

Complexity

Exclude \Rightarrow coins.length

Include All \Rightarrow A

Mix \Rightarrow $A + \text{coins.length}$

$\begin{pmatrix} A & 0 \\ A, 1 \\ A, 2 \\ A, 3 \end{pmatrix}$
 $\begin{pmatrix} A-1, 0 \\ A-1, 1 \\ A-1, 2 \\ A-1, 3 \end{pmatrix}$
 $\begin{pmatrix} A-5, 0 \\ A-5, 1 \\ A-5, 2 \\ A-5, 3 \end{pmatrix}$

$\underbrace{\hspace{10em}}_{A. \text{ coins.length}}$

$[A, 0]$

	0	1	2	3
0	1	1	1	0
1	1	0	0	0
2				
3				
4				
5				

$[0, 2, 3]$

```
private static int changeTD(int[] coins, int A, int idx, Integer[][] dp) {
    if (A == 0) {
        return 1;
    }
    if (A < 0 || idx == coins.length) {
        return 0;
    }
    if (dp[A][idx] != null) {
        return dp[A][idx];
    }
    // include
    int sp1 = changeTD(coins, A - coins[idx], idx, dp);

    // exclude
    int sp2 = changeTD(coins, A, idx + 1, dp);
    dp[A][idx] = sp1 + sp2;
    return sp1 + sp2;
}
```

