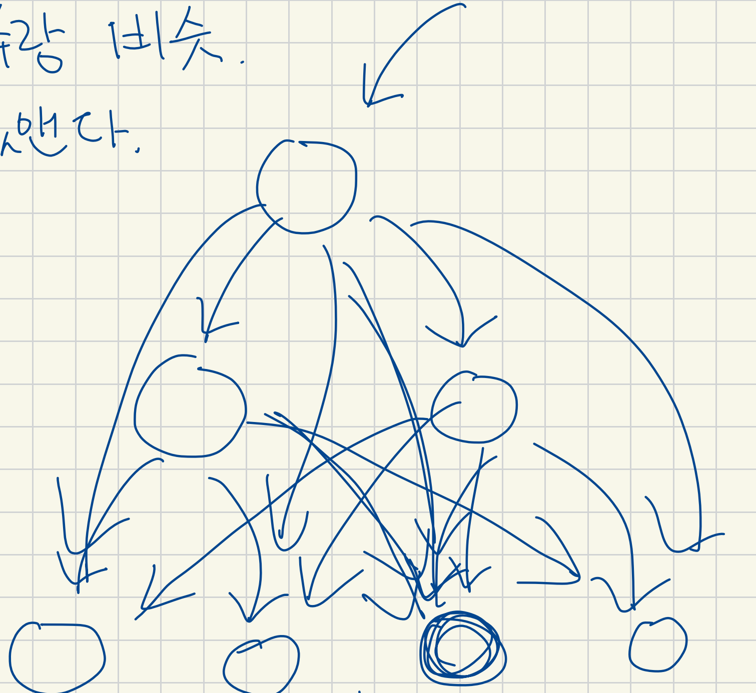
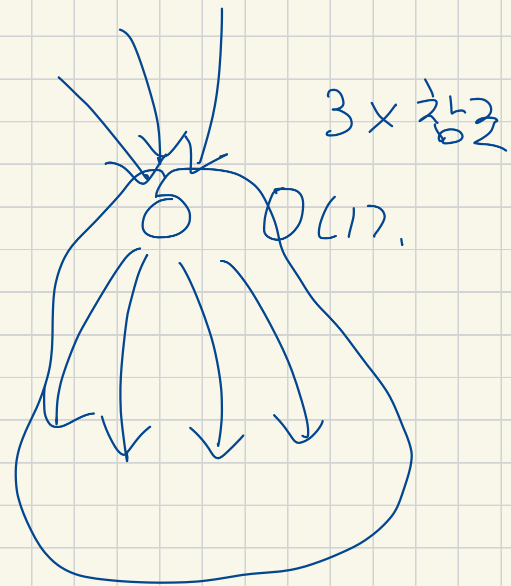
DP.

* 재귀함수랑 비슷.
* 중복을 없앤다.





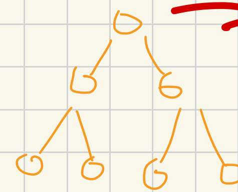
$\boxed{\pi = 3.14.}$

dp(2,3) = 5

✓ ready(2,3).

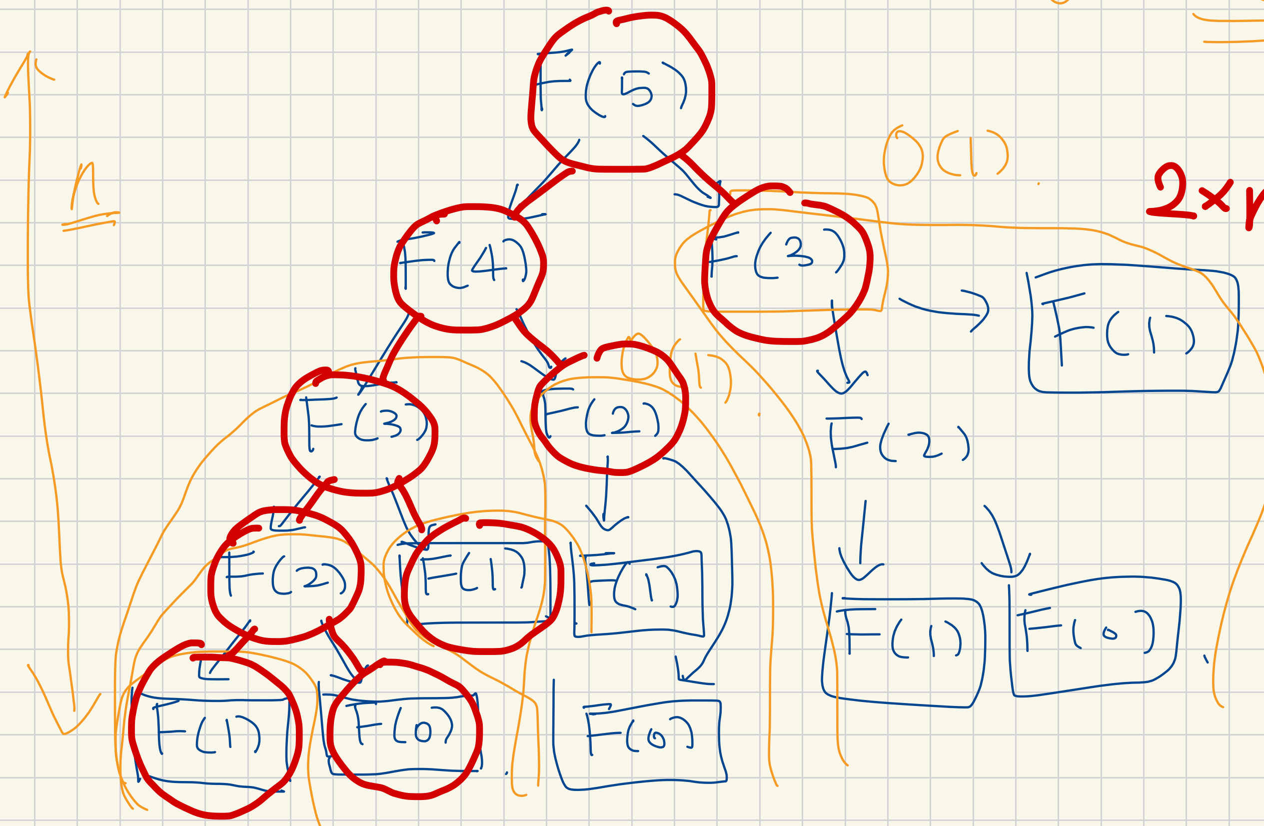
$F(0) = F(1) = 1.$

$F(n) = F(n-1) + F(n-2).$



$\rightarrow O(2^n).$

$O(2^n)$



$2 \times n.$

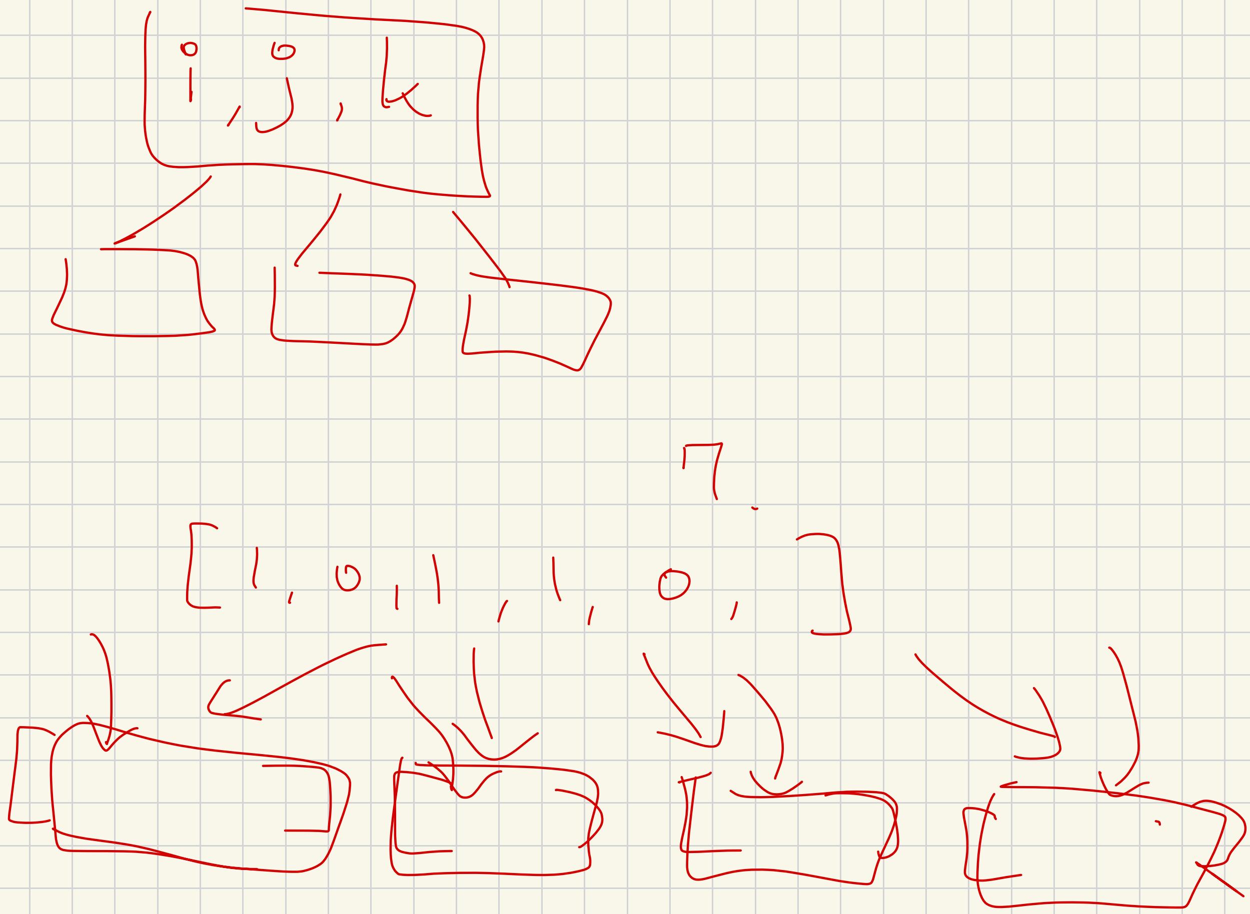
$O(n).$

dp[i] := i에서 가질 값. ready[i] := dp[i]가 계산되어 있는가?

dp[i][j]   ready[i][j]   dp[i][j].

```
void solve(int i, int j) {
    if (ready[i][j]) return;
    ready[i][j] = true;
    solve(i-1, j);
    solve(i-1, j-1);
    dp[i][j] = dp[i-1][j] + dp[i-1][j-1];
}
```

✓ $dp(i, j) = dp(i-1, j) + dp(i-1, j-1)$.



i, j, k
7.
[1, 0, 1, 1, 0]

9095. 1, 2, 3 더하기.

1+1+1+1
1+1+2
1+2+1
2+1+1
2+2
1+3
3+1

dp(i) = n=i 일때 문제의 답.

= i를 1, 2, 3 의 합으로 표현하는 가짓수.

i-1 ~ 1

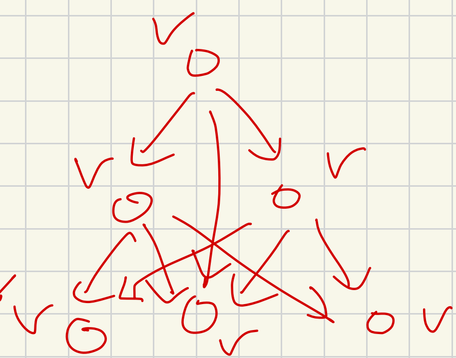
1+1+1
1+2
2+1
3

+1 ⇒ 4.

1+1
2

+2 ⇒ 4.

$\boxed{1}$ +3 ⇒ 4.



← 방향성 그래프.
(사이클 X).

DP.
⇕
Directed
Acyclic
Graph

→ DAG.

- 피보나치 점화식 비슷하게..
- i-1, i-2, ... , 1, 0에서 전부 계산되었다고 가정!

보통 DP점화식은 파라미터가 감소하는 방향이 많이 나온다

$$dp(i) = dp(i-1) + dp(i-2) + dp(i-3).$$

$dp(1) = 1$ $\quad$ $dp(0) = 1.$

$\quad$ [ ]

$dp(2) = 2.$

[2], [1, 1].

$1+1+1$

$1+2.$

$2+1.$

$3.$

$dp(3) = 2+1+1.$

1463. 1로 만들기.

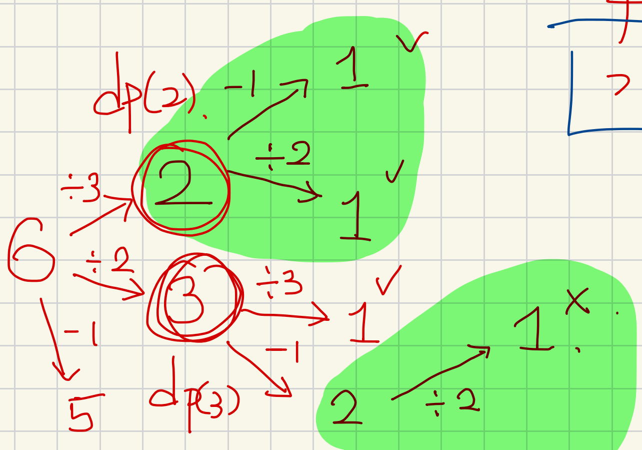
$dp(i)$ = i를 1로 만들기 위한 연산의 최솟값.

$i \le 10^6$.

$dp[i] = 1e9;$ ← 10^9.

$$dp(i) = \min\{dp(i/3)+1,\ dp(i/2)+1,\ dp(i-1)+1\}.$$

$\boxed{\text{가능할 경우에만 고려.}}$



"2에서 1까지 줄이는 비용".

$$dp(6) = \min\{dp(2)+1,\ dp(3)+1,\ dp(5)+1\}$$