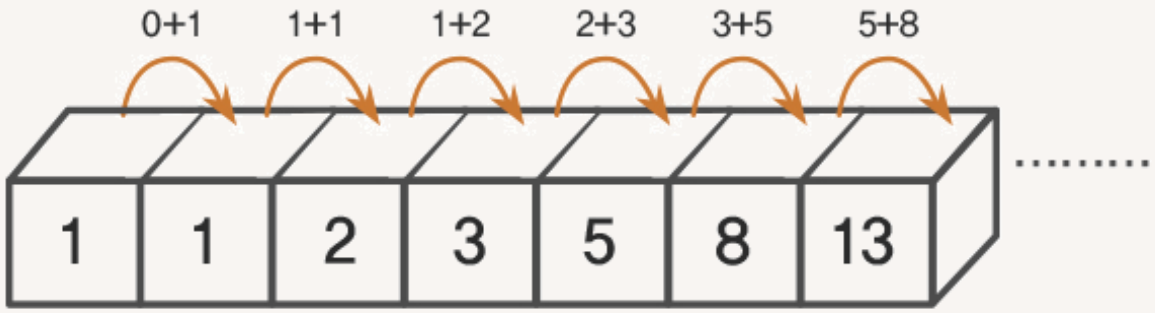
Dynamic Programming

목 차

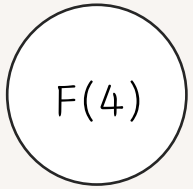
1. 피보나치 수열



$$F_n = F_{n-1} + F_{n-2}$$

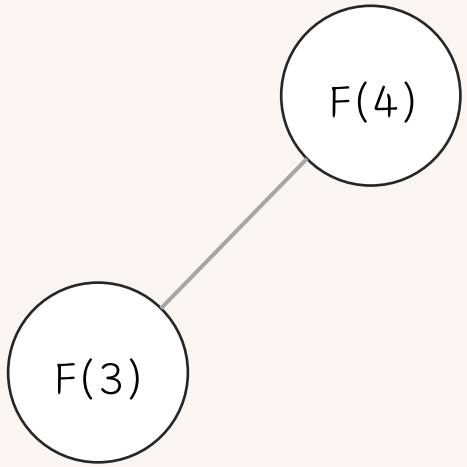
1. 피보나치 수열

```
def fibo(n):
    if n == 1:
        return 1
    elif n==2:
        return 1
    else:
        fib = fibo(n-1) + fibo(n-2)
        return fib

fibo(4)
```



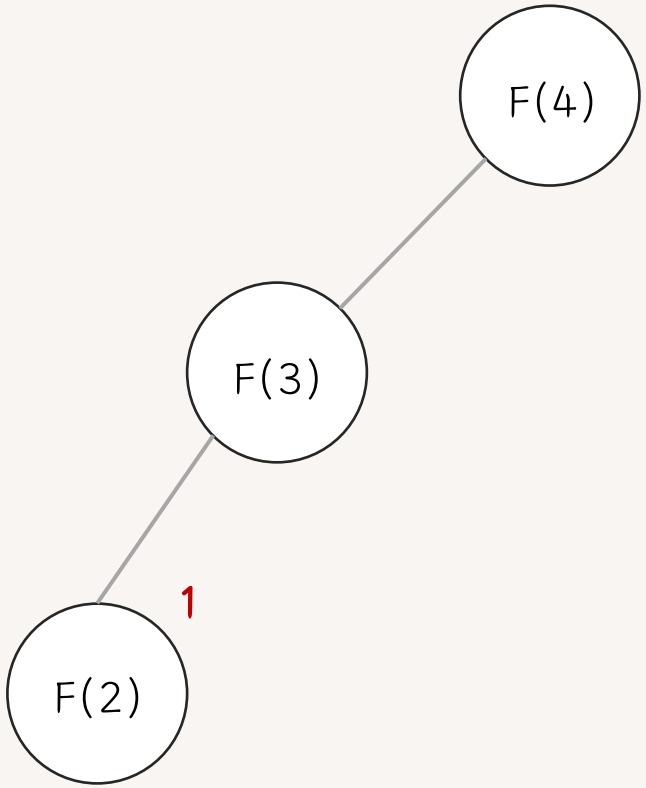
1. 피보나치 수열

```
def fibo(n):
    if n == 1:
        return 1
    elif n==2:
        return 1
    else:
        fib = fibo(n-1) + fibo(n-2)
        return fib

fibo(4)
```



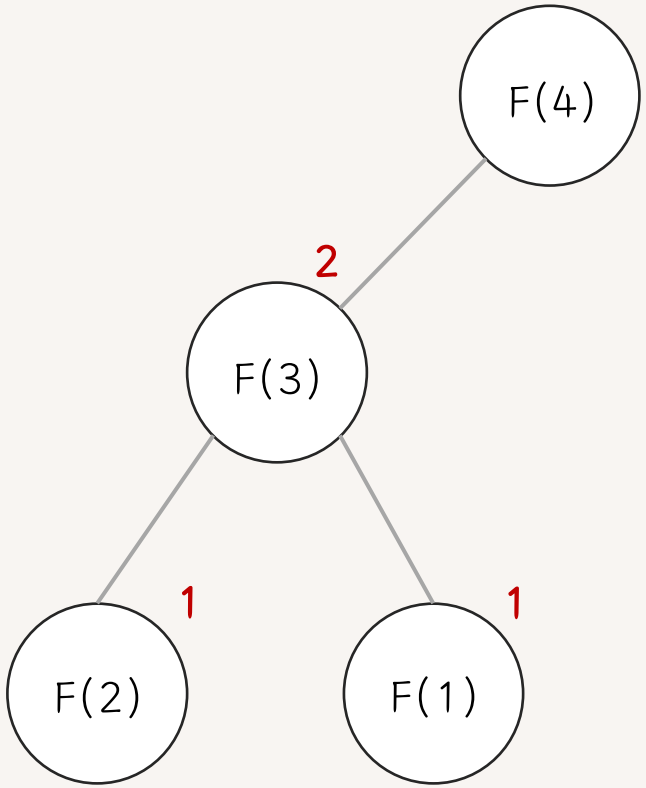
1. 피보나치 수열

```
def fibo(n):
    if n == 1:
        return 1
    elif n==2:
        return 1
    else:
        fib = fibo(n-1) + fibo(n-2)
        return fib

fibo(4)
```



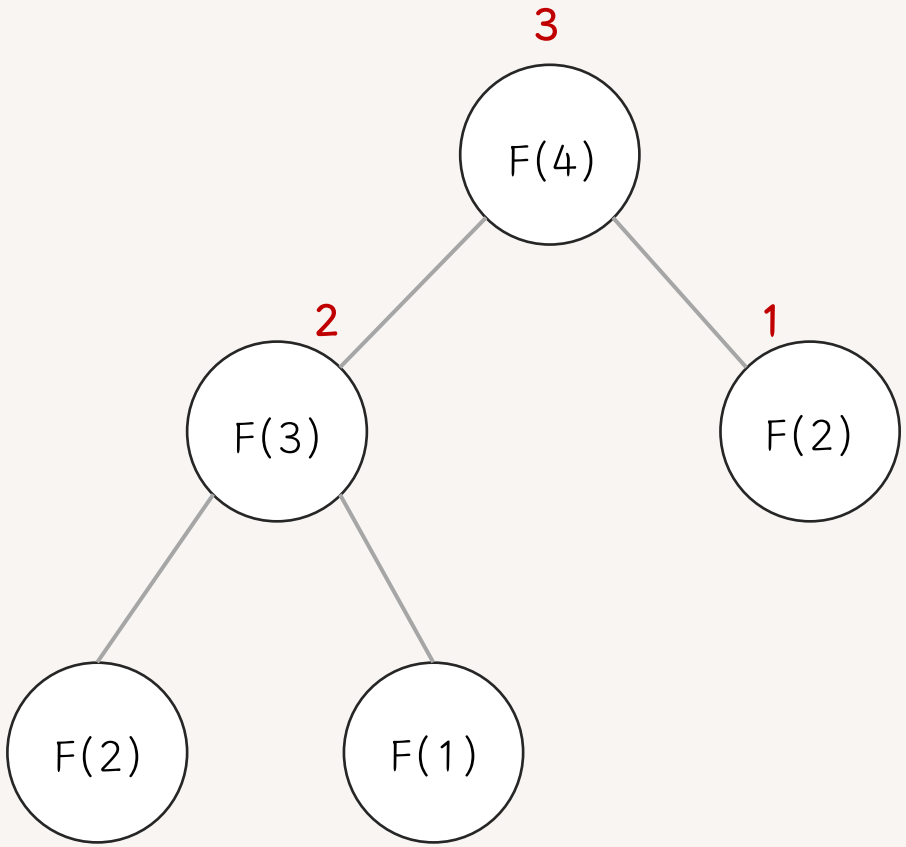
1. 피보나치 수열

```
def fibo(n):
    if n == 1:
        return 1
    elif n==2:
        return 1
    else:
        fib = fibo(n-1) + fibo(n-2)
        return fib

fibo(4)
```



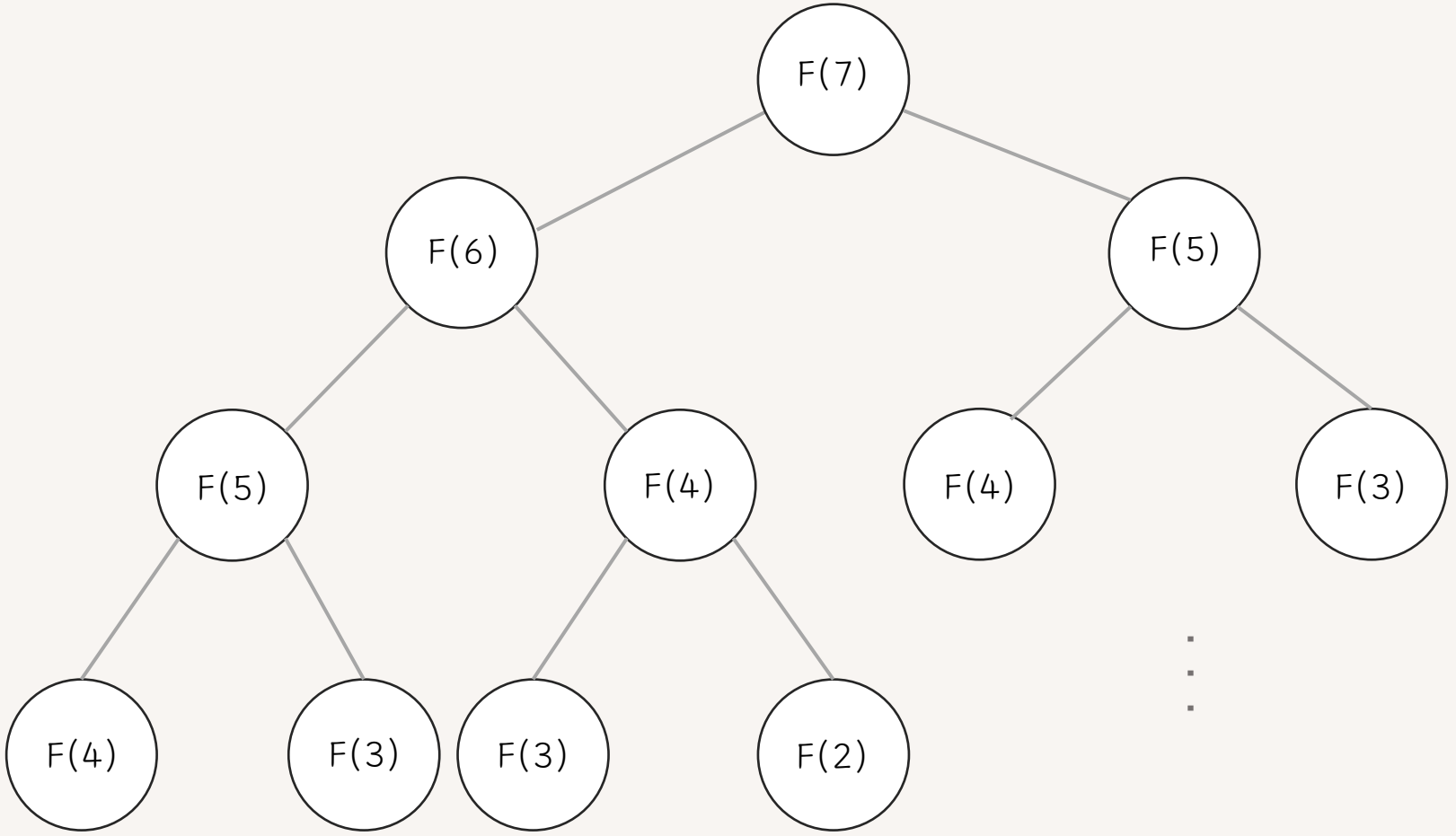
1. 피보나치 수열

```
def fibo(n):
    if n == 1:
        return 1
    elif n==2:
        return 1
    else:
        fib = fibo(n-1) + fibo(n-2)
        return fib

fibo(4)
```



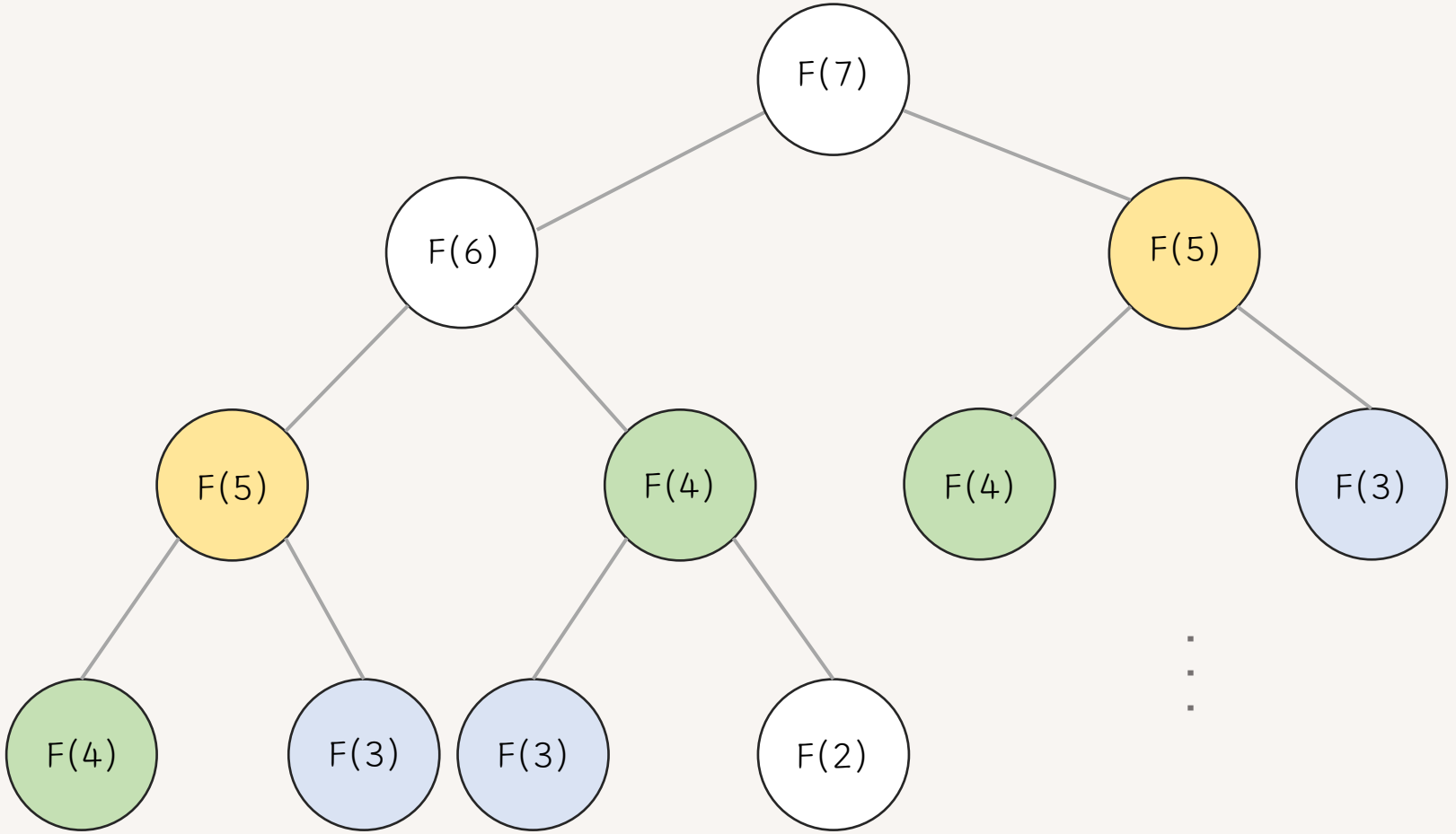
1. 피보나치 수열

Q : 7번째 피보나치 수열을 구하시오.

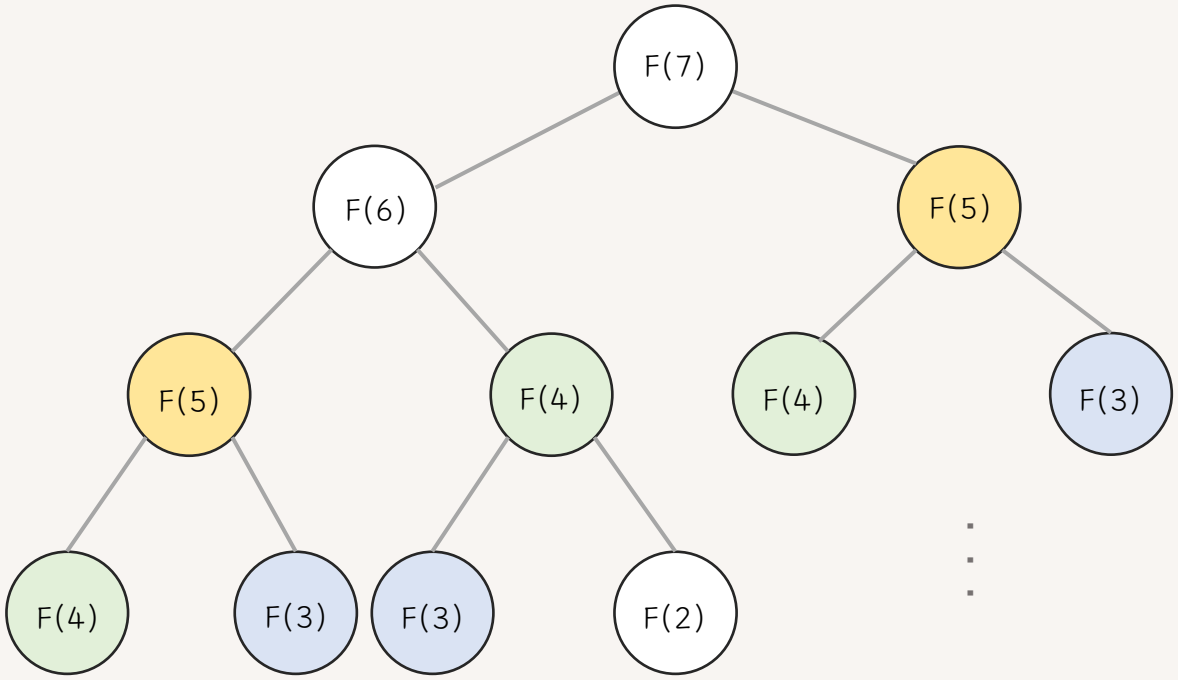


1. 피보나치 수열

Q : 7번째 피보나치 수열을 구하시오.



1. 피보나치 수열

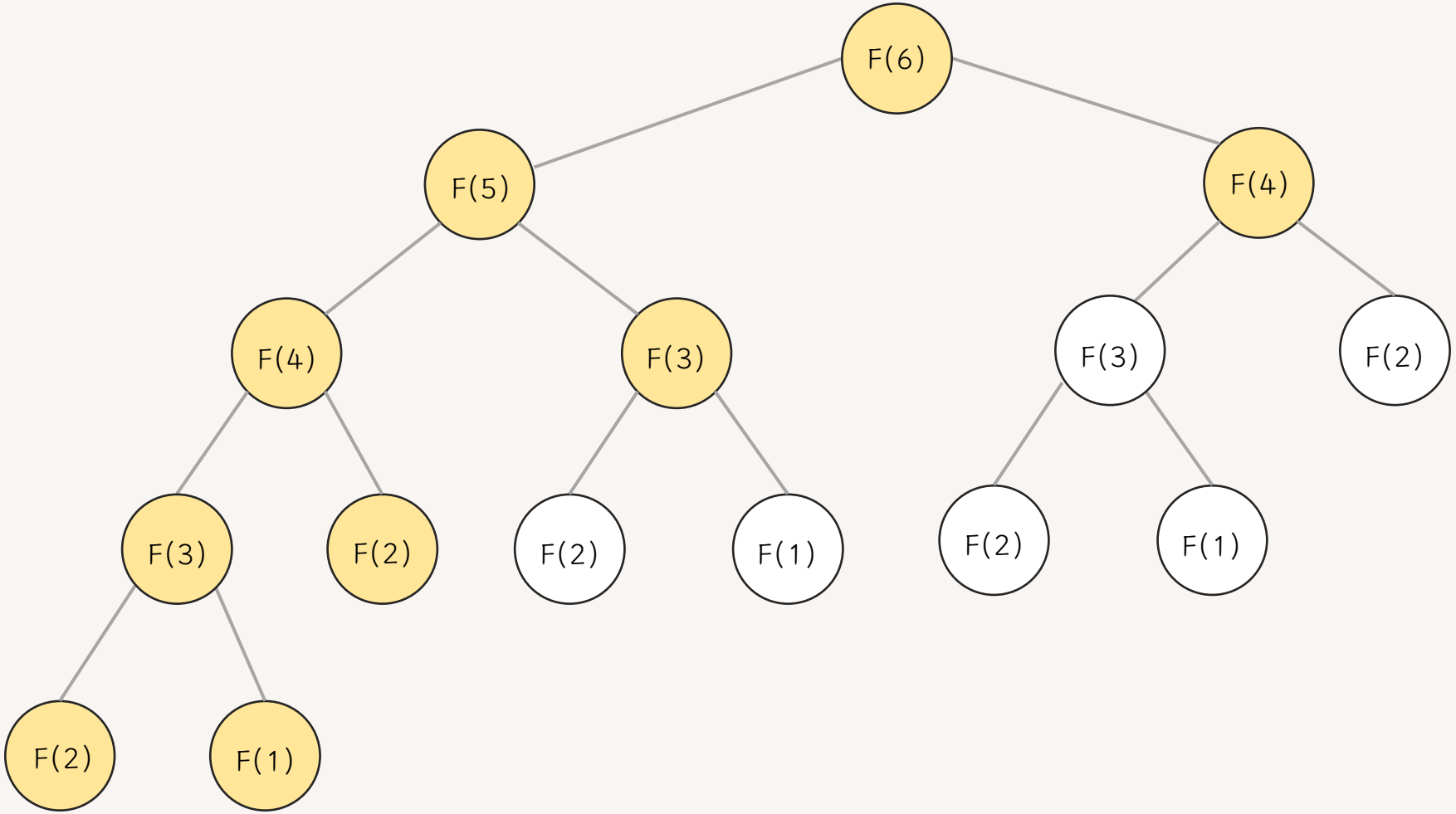



```
def fibo(n):
    if n == 1:
        return 1
    elif n==2:
        return 1
    else:
        fib = fibo(n-1) + fibo(n-2)
        return fib

print(fibo(40))
```

18초

3-1. 메모이제이션(Memoization)

한 번 계산한 결과를 저장해 다시 사용



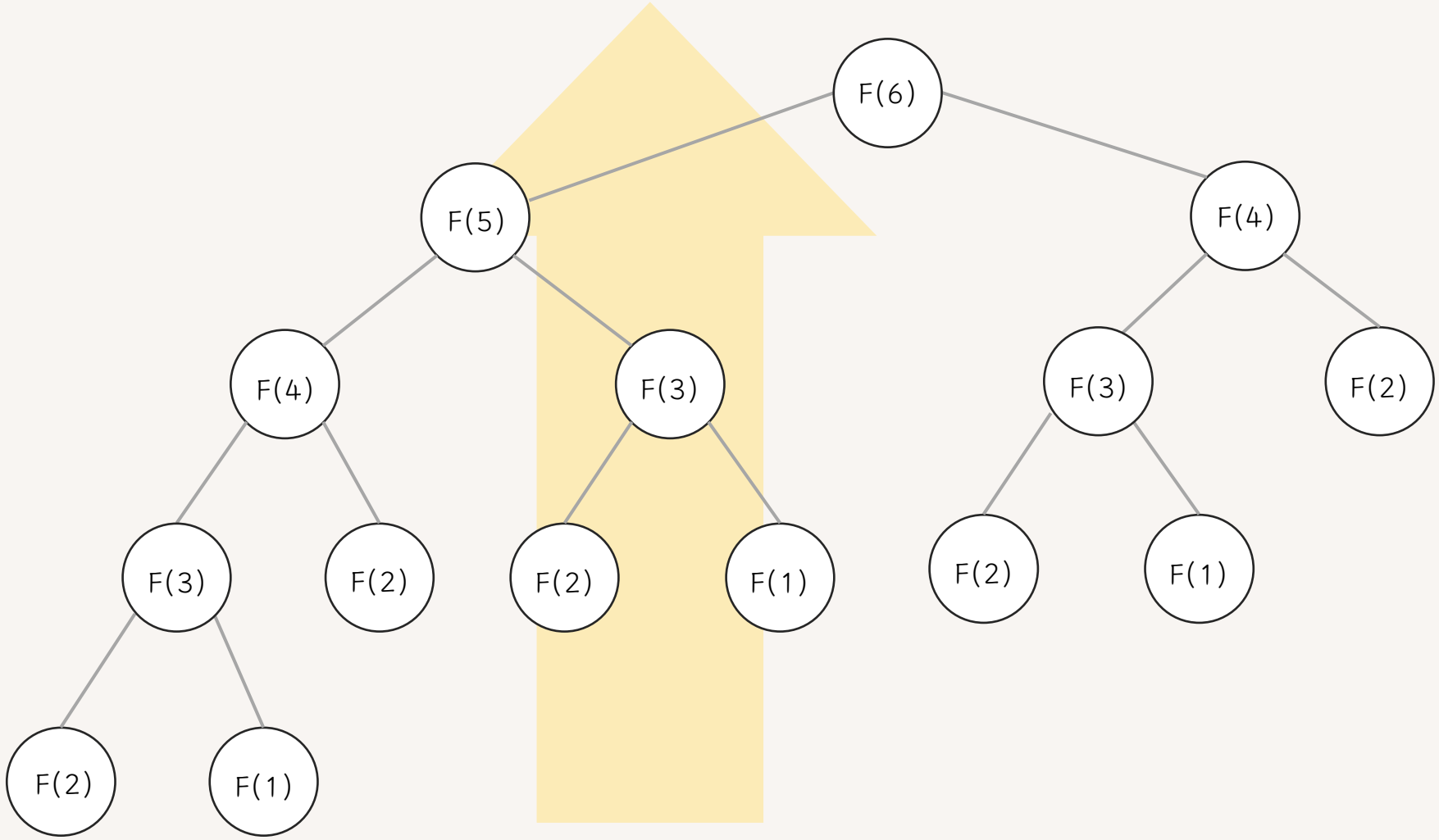
3-1. 메모이제이션(Memoization)

한 번 계산한 결과를 저장해 다시 사용

```
fib = [0]*100

def fibo(n):
    if n == 1 or n == 2:
        return 1

    if fib[n] !=0:
        return fib[n]

    fib[n] = fibo(n-1) + fibo(n-2)
    return fib[n]

print(fibo(40))
```

0.01초

3-2. Bottom-up



3-2. Bottom-up

```
fib = [0] * 100

fib[1] = 1
fib[2] = 1
n = 40

for i in range(3, n + 1):
    fib[i] = fib[i - 1] + fib[i - 2]


print(fib[n])
```

끝.