3.

It's known as $\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$

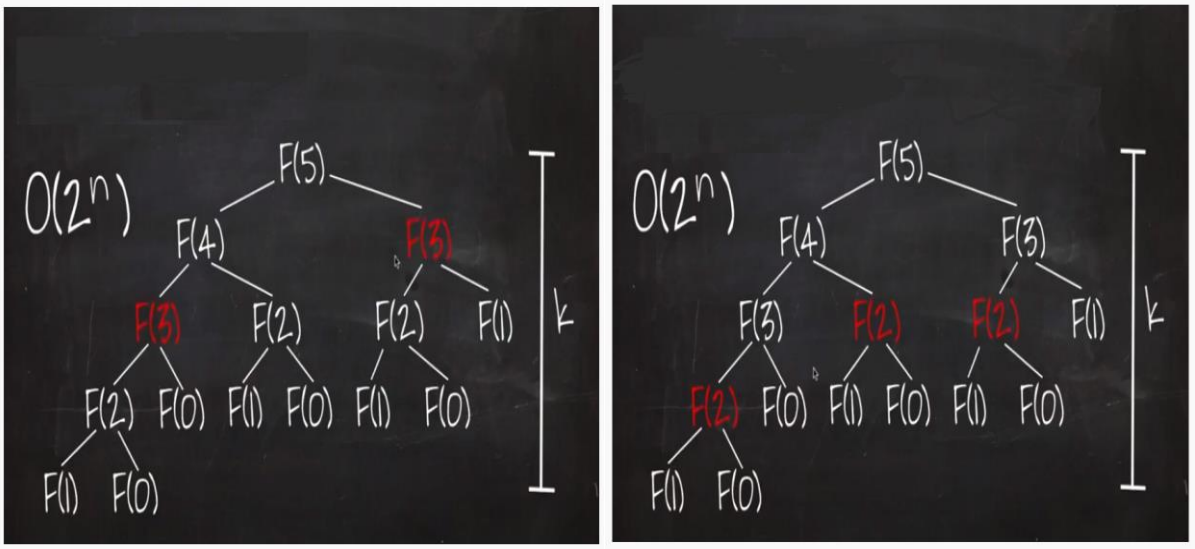
1. Fill in the blanks using recursive function

```cpp
#include <iostream>

using namespace std;

long long bin(int n, int k)
{
    if (k == 0 || n == k)
        return 1;
    else
        return bin(n - 1, k - 1) + bin(n - 1, k);
}

int main(int argc, char** argv) {

    cout << bin(45, 10);

    return 0;
}

```

2. Capture the output and process time

```
3190187286
Process returned 0 (0x0)   execution time : 22.118 s
Press any key to continue.
```

3. Explain why it takes so long



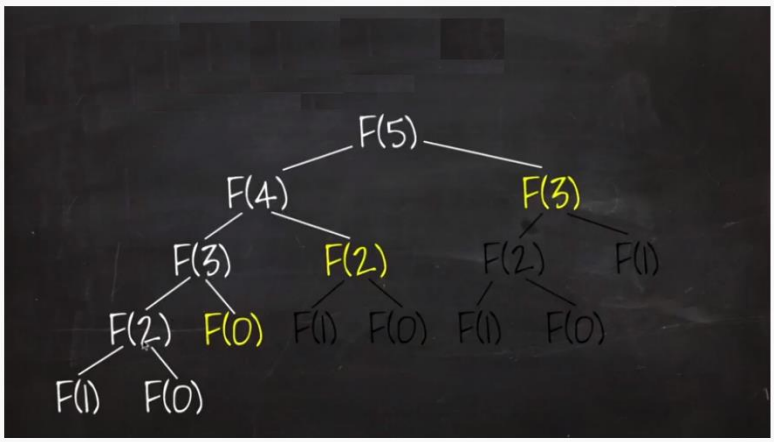
Recursive function을 하나의 트리로 대입해서 생각해보면,

매번 함수가 호출될 때 마다 두 개씩 호출이 증가하고

그것을 트리의 높이만큼 반복하는데 트리의 높이는 n 보다 작은 값을 갖게 되기에

두 개씩 n번 늘어나는 알고리즘 이기에 time complexity는 $O(2^n)$ 이다.

같은 함수를 여러 번 호출하기 때문에 이 함수는 불필요한 작업을 많이 하는 것이다.



공간을 할당하지 않는다는 점에서 Recursive function은 장점이 있지만,

Time complexity의 지나친 상승으로 위의 코드는 효율적이지 못한 코드이다.

일정부분 배열을 할당하여 해당 값을 넣고 비교하여, 위의 그림처럼 함수의 재호출을 막는다면

더 효율적인 코드가 될 수 있다.