

[회수]

Merge Sort Time Complexity explain.

Merge Sort

$T(1) = 1$
 $2^{n-1} T(2) = 2^{n-1} T(1) + m \cdot 2^{n-1}$
 $2^{n-2} T(4) = 2^{n-2} T(2) + m \cdot 2^{n-2}$
 $2^{n-3} T(8) = 2^{n-3} T(4) + m \cdot 2^{n-3}$
 \vdots
 $2^2 T(2^{k-1}) = 2^2 T(2^{k-2}) + m \cdot 2^2$
 $2^1 T(2^{k-1}) = 2^1 T(2^{k-2}) + m \cdot 2^1$
 $2^0 T(2^k) = 2^0 T(2^{k-1}) + m \cdot 2^0$

 $T(2^k) = 2^k + m \cdot 2^k \cdot k$
 $k = \log_2 n$
 $T(n) = n + n \cdot \log n$

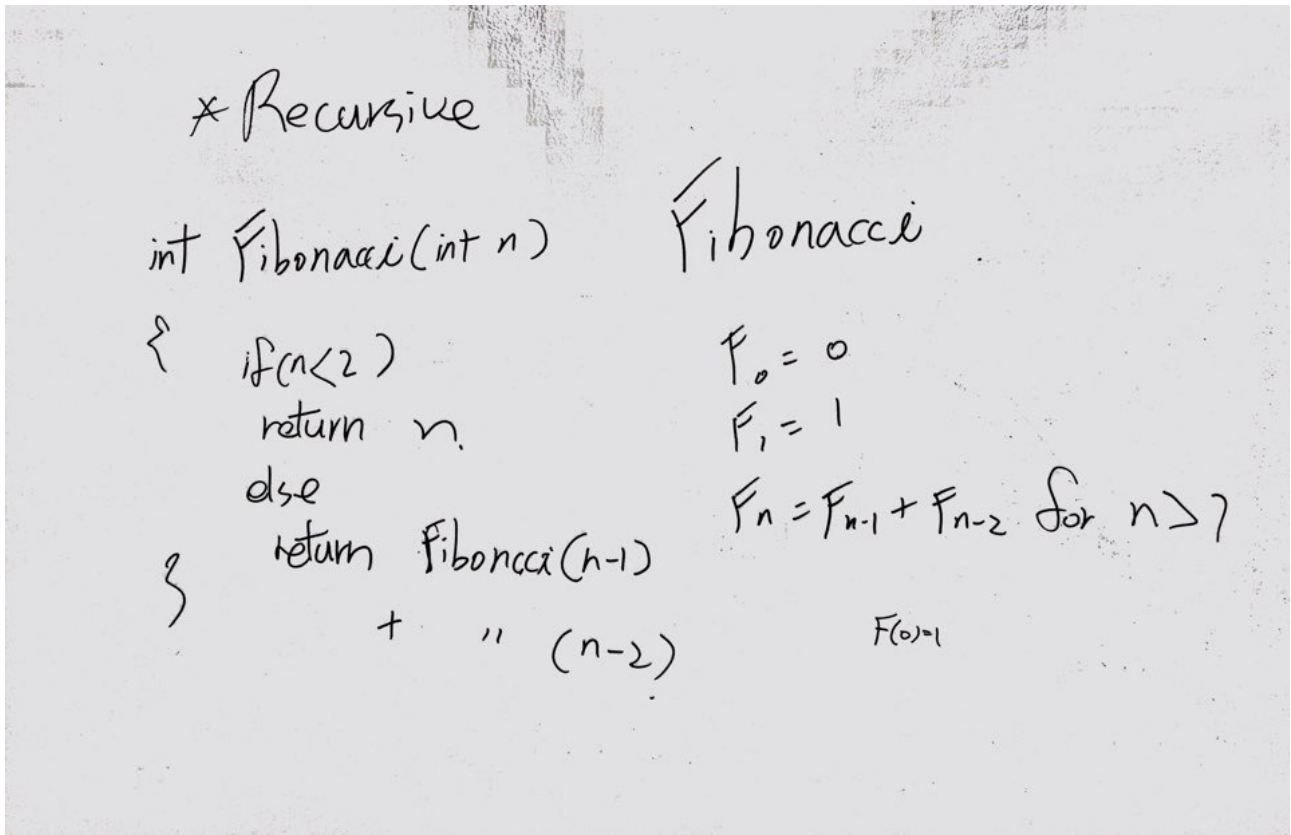
$T(n)$ if $n=1$ k
 $n > 1 \rightarrow T(n) = 2T(\frac{n}{2}) + mn$

Divide C
 Conquer $2 \times T(\frac{n}{2})$
 Combine $O(n)$

$cn + n \log n \rightarrow \Theta(n \log n)$
 $cn + k \rightarrow \Theta(n)$

[동준]

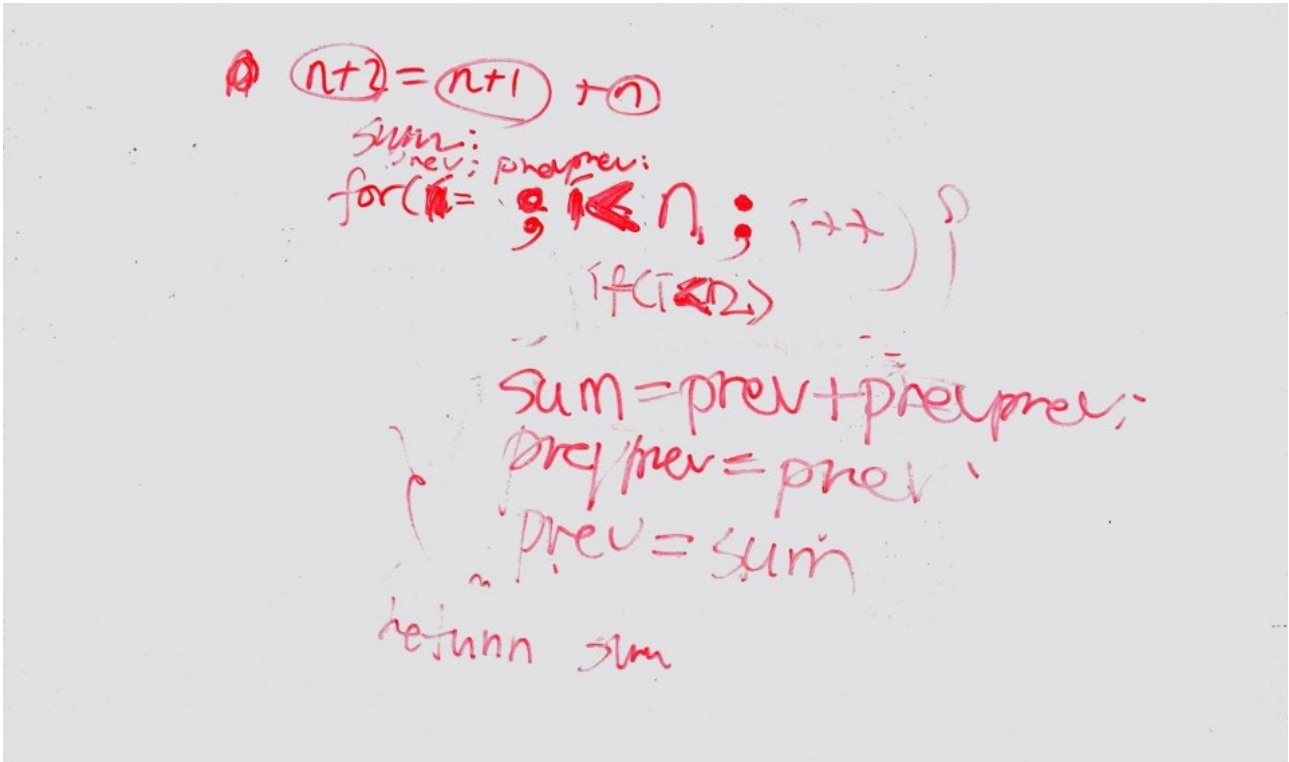
Fibonacci Explain but It was't enough
So I'm preparing again



<AS by DongJun>

```
int Fibonacci(int n){  
  
    int result;  
    int i;  
    int * FiboArr=(int*)malloc(sizeof(int)*n);  
    //malloc된 값을 해제할 free는 main에서 해제합니다!  
  
    FiboArr[0]=0;  
    FiboArr[1]=1;  
  
    for(i=2; i<=n; i++)  
    {  
        result=FiboArr[i-1]+FiboArr[i-2];  
        FiboArr[i]=result;  
    }  
    return FiboArr[i];  
}
```

[기법]



<AS by DongJun>

```
int fibonacci(int n)
{
    if (n == 0)
        return 0;
    if (n == 1)
        return 1;

    int result=0;
    int prevPrev=0;
    int prev=1;

    for(int i=2; i<=n; i++)
    {
        result = prev + prevPrev;
        prevPrev = prev;
        prev = result;
    }

    return result;
}
```

[회수]
Divide & Conquer

Handwritten notes on a piece of paper showing a diagram of a tree structure and recursive calculations for a function $F(n)$.

Diagram: A tree structure with root n and children $n/2$ and $n/2$. The root is labeled n , and the children are labeled $n/2$. The root is also labeled $n/2$ and $n/2$.

Recursive calculations:

$$F(8) = F(7) + F(6)$$

$$= F(6) + F(5)$$

$$= 2(F(4) + F(3))$$

$$= 2(F(3) + F(2))$$

$$= 2(F(2) + F(1))$$

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

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

Divide
Conquer
Combine

Matrix multiplication example:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

$$= \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} a^2 + bc & ab + bd \\ ac + cd & d^2 + bc \end{pmatrix}$$

$$= \begin{pmatrix} a^3 + 2abc + bcd & a^2b + b^2d + bd^2 + b^2c \\ a^2c + bc^2 + acd + cd^2 & d^3 + 2bcd + abc \end{pmatrix}$$

[규원]
Divide & Conquer

Handwritten notes on a piece of paper showing a diagram of a tree structure and recursive calculations for a function $F(n)$.

Diagram: A tree structure with root n and children $n/2$ and $n/2$. The root is labeled n , and the children are labeled $n/2$. The root is also labeled $n/2$ and $n/2$.

Recursive calculations:

$$F(0) = 0 = a$$

$$F(1) = 1 = b$$

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

$$F(3) = F(2) + F(1)$$

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

$$F(4) = F(3) + F(2)$$

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

$$= F(1) + F(0) + F(1) + F(1) + F(0) = 3F(1) + 2F(0)$$

$$F(5) = F(4) + F(3)$$

$$= 3F(1) + 2F(0) + 2F(1) + F(0)$$

$$= 5F(1) + 3F(0)$$

Divide
conquer;
merge

Matrix multiplication example:

$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 2 & 3 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 2 & 3 \\ 3 & 5 \end{pmatrix} = \begin{pmatrix} 2 & 3 \\ 3 & 5 \end{pmatrix}$$