시간복잡도

🧩 1단계: 기본 반복문 예제

예제 1. 중첩 반복문

```
for i in range(n):
    for j in range(n):
        print(i, j)
```

$n \times n = n^2$

$\Rightarrow O(n^2)$

⚙️ 2단계: 선형 반복 + 조건문

예제 2. 2중 반복이지만 j가 i 이후로만 순회

```
for i in range(n):
    for j in range(i, n):
        print(i, j)
```

$n+(n-1)+(n-2)+\cdots+1=\frac{n(n-1)}{2}$

$\Rightarrow O(n^2)$

🧮 3단계: 로그가 포함된 반복문

예제 3. 반복마다 절반씩 줄어드는 경우

```
i = n
while i > 1:
    i //= 2
```

$\log_2 n$

$\Rightarrow O(\log n)$

⚡ 4단계: 선형 + 로그 반복문

예제 4. 반복마다 절반씩 줄어드는 내부 루프

```
for i in range(n):
    j = n
```

$n \times \log_2 n$

$\Rightarrow O(n \log n)$

* n을 $\underbrace{1/2 \text{ 하는 걸}}_{\log_2 n}$ $\underbrace{n\text{번}}_{n}$ 반복

생각..

```
    while j > 0:
        j //= 2
```

🧠 5단계: 분할정복 기본형

예제 5. 이진 분할 정복

```
def f(n):
    if n <= 1:
        return 1
    return f(n // 2) + f(n // 2)
```

$$
\begin{aligned}
T(n) &= 2T(\tfrac{n}{2}) + 1 \\
&= 2\{2T(\tfrac{n}{4}) + 1\} + 1 = 2^2T(\tfrac{n}{4}) + (2+1) \\
&= 2^2\{2T(\tfrac{n}{2^3}) + 1\} + (2+1) = 2^3T(\tfrac{n}{2^3}) + (2^2+2+1) \\
&\vdots \\
&= 2^kT(1) + \frac{2(2^k-1)}{2-1} = 2^kT(1) + (2^{k+1}-1) \qquad (n=2^k\text{라 가정}) \\
&= nT(1) + (2n-1)
\end{aligned}
$$

$\Rightarrow O(n)$

🧩 6단계: Merge Sort 유형

예제 6. 합병정렬

```
def merge_sort(arr):
    if len(arr) <= 1:
        return arr
    mid = len(arr)//2
    left = merge_sort(arr[:mid])
    right = merge_sort(arr[mid:])
    return merge(left, right)  # O(n)
```

$$
\begin{aligned}
T(n) &= 2T(\tfrac{n}{2}) + cn \\
&= 2\{2T(\tfrac{n}{2^2}) + \tfrac{cn}{2}\} + cn = 2^2T(\tfrac{n}{2^2}) + 2cn \\
&\vdots \\
&= 2^kT(1) + kcn \qquad (n=2^k\text{라 가정},\ k=\log_2 n) \\
&= nT(1) + c \cdot n\log_2 n
\end{aligned}
$$

$\Rightarrow O(n+n\log n) = O(n\log n)$

🧩 7단계: 분할은 여러 개지만 정복은 가벼운 경우

예제 7. 삼등분 분할 정복

```
def tri(n):
    if n <= 1:
        return 1
    return tri(n//3) + tri(n//3) + tri(n//3)
```

$$
\begin{aligned}
T(n) &= 3T(\tfrac{n}{3}) + 1 \\
&\vdots \\
&= 3^kT(1) + \frac{3(3^{k-1}-1)}{3-1} = 3^kT(1) + \frac{3^k}{2} - \frac{3}{2} \\
&= nT(1) + \frac{1}{2}n - \frac{3}{2} \qquad (n=3^k\text{라 가정})
\end{aligned}
$$

$\Rightarrow O(n)$

🧩 8단계: Karatsuba 곱셈형 (고급)

예제 8. 분할정복 곱셈

```
def karatsuba(x, y):
    if x < 10 or y < 10:
        return x * y
    n = max(len(str(x)), len(str(y)))
    m = n // 2
    high1, low1 = divmod(x, 10**m)
    high2, low2 = divmod(y, 10**m)
    z0 = karatsuba(low1, low2)
    z1 = karatsuba((low1 + high1), (low2 + high2))
    z2 = karatsuba(high1, high2)
    return (z2 * 10**(2*m)) + ((z1 - z2 - z0) * 10**m) + z0
```

$O(1) \rightarrow$ if x < 10 or y < 10:

$O(n)$: n = max(...) ~ high2, low2 = divmod(y, 10**m)

윗쪽 (high1), 아래쪽 (low1)

$T(\frac{n}{2})$: z0, z1, z2

$O(n) \rightarrow$ return

$$T(n) = 3T\left(\frac{n}{2}\right) + cn$$
$$= 3\left(3T\left(\frac{n}{2^2}\right) + \frac{cn}{2}\right) + cn = 3^2T\left(\frac{n}{2^2}\right) + \left(\frac{3}{2}+1\right)cn$$
$$\vdots$$
$$= 3^kT\left(\frac{n}{2^k}\right) + \frac{\frac{3}{2}\left(\frac{3^k}{2^k}-1\right)}{\frac{3}{2}-1}cn = 3^kT\left(\frac{n}{2^k}\right) + \left(\frac{3^{k+1}}{2^k} - \frac{3}{2}\right)cn$$
$$= \underbrace{n^{\log_2 3}T(1)}_{O(n^{\log_2 3})} + \underbrace{\left(\frac{(2n)^{\log_2 3}}{n} - \frac{3}{2}\right)}_{n^{\log_2 3 - 1}} \underbrace{cn}_{n} \qquad (n=2^k \text{라고 가정}, \ k=\log_2 n)$$

$n^{\log_2 3 - 1} \times n = O(n^{\log_2 3})$

$\Rightarrow O(n^{\log_2 3})$

🧠 9단계: 복합형 예제

예제 9. 분할정복 + 로그 루프 결합

```
def g(n):
    if n <= 1:
        return 1
    for i in range(n):
        k = n
        while k > 1:
            k //= 2
    return g(n//2) + g(n//2)
```

$O(1) \rightarrow$ return 1

$O(n\log n) \rightarrow$ for i in range(n): ~ k //= 2

$2T(\frac{n}{2}) \rightarrow$ return g(n//2) + g(n//2)

$$T(n) = 2T\left(\frac{n}{2}\right) + n\log_2 n$$
$$= 2\left\{2T\left(\frac{n}{4}\right) + \frac{n}{2}\log_2\frac{n}{2}\right\} + n\log_2 n = 2^2T\left(\frac{n}{2^2}\right) + 2n\log_2 n - n$$
$$\vdots$$
$$= 2^kT(1) + kn\log_2 n - n\times\frac{(k-1)(k-2)}{2} \qquad (n=2^k\text{라 가정}, \ k=\log_2 n)$$
$$= nT(1) + n\underbrace{\left(k^2 - \frac{k^2-3k+2}{2}\right)}_{O(\log^2 n)}$$

$\Rightarrow O(n\log^2 n)$