점근적 분석 및 표기 예시

러닝타임 n분 영화의 총 렌더링 시간을 $T(n)$이라 가정하자. 렌더링 작업은 세 가지 작업의 합으로 구성된다.

- **VFX 렌더링 (이차 시간):** n개의 클립이 상호 참조하는 복잡한 AI 색 보정 작업. 시간은 n의 제곱에 비례 (약 $2n^2$ 초).
- **오디오 인코딩 (선형 시간):** n분짜리 오디오 트랙을 압축하는 작업. 시간은 n에 정비례 (약 $30n$ 초).
- **파일 헤더 저장 (상수 시간):** n과 무관하게 5초가 걸리는 메타데이터 저장 작업 (약 5 초).

점근적 표기(Asymptotic Notation)

$$T(n) = 2n^2 + 30n + 5$$

점근적 분석은 n이 "충분히 클 때"를 본다. 장편 영화처럼 n이 매우 커지면, $2n^2$ (VFX) 항이 $30n$ (오디오)이나 5 (헤더) 항을 압도하며 전체 수행 시간을 지배한다.
이 함수의 **점근적 증가율**은 n^2에 의해 결정된다. $g(n) = n^2$을 기준으로 상황을 분석해보자.

1. O (Big-O): 점근적 상한 (Upper Bound)

*"아무리 오래 걸려도 **기껏해야** $g(n)$의 증가율을 넘지 않는다."*

- O 표기는 알고리즘 수행 시간의 상한을 나타낸다.
- $T(n) = 2n^2 + 30n + 5$의 증가율은 n^2의 증가율보다 빠르지 않다. n이 충분히 크면 $T(n)$은 항상 $3n^2$ (즉, $c = 3$인 cn^2)보다 작다.
- 이 렌더링은 n^3의 증가율보다는 확실히 빠르며, 최악의 경우 증가율이 n^2에 묶여있음을 보장한다.

$$T(n) = O(n^2)$$

2. Ω (Omega): 점근적 하한 (Lower Bound)

*"아무리 빨라도 **적어도** $g(n)$의 증가율만큼은 걸린다."*

- Ω 표기는 알고리즘 수행 시간의 하한을 나타낸다.
- $2n^2$ 항이 존재하므로, $T(n)$은 n이나 $\log n$의 증가율보다는 항상 느릴 수밖에 없다. n이 충분히 크면 $T(n)$은 항상 $1n^2$ (즉, $c = 1$인 cn^2)보다 크다.
- 오디오 인코딩($30n$)을 아무리 최적화해도, n^2 항 때문에 전체 작업의 증가율은 n^2보다 낮아질 수 없다.

$$T(n) = \Omega(n^2)$$

3. Θ (Theta): 점근적 동일 (Tight Bound)

*"이 작업의 증가율은 **정확히** $g(n)$ 등급이다."라는 가장 정밀한 표현*

- 위에서 $T(n) = O(n^2)$ (상한)임과 $T(n) = \Omega(n^2)$ (하한)임을 모두 확인했다. 상, 하한이 n^2으로 일치한다.
- 렌더링 시간의 병목 현상은 n^2 항이고, n이 커질 때 다른 항들은 무시할 수 있다. 따라서 이 작업의 점근적 효율성은 n^2과 정확히 같다.

$$T(n) = \Theta(n^2)$$