3. 점근표기법

• 점근 표기법은 어떤 함수의 증가율을 다른 함수와 비교로 표현하는 방법이다.(해석학)
• 점근 표기법은 크기 n이 커질 때, 각 함수가 어떻게 성장하는가? 이다.
• 점근 표기법은 알고리즘의 복잡도를 **단순하게 표시**하는데 사용된다.
• 점근 표기법으로 O, Ω, Θ 등이 있다.

함수	정의	주 특징
빅오 (O)	$n \geq n_0$를 만족하는 모든 n에 대하여 $f(n) \leq c \cdot g(n)$인 조건을 만족하는 두 양의 상수 c와 n_0가 존재하기만 하면 $f(n)=O(g(n))$이다.	최대시간 예측 **(상한)**
빅오메가 (Ω)	$n \geq n_0$를 만족하는 모든 n에 대하여 $f(n) \geq c \cdot g(n)$인 조건을 만족하는 두 양의 상수 c와 n_0가 존재하기만 하면 $f(n)=\Omega(g(n))$이다.	최소시간 예측 **(하한)**
빅세타 (Θ)	$n \geq n_0$를 만족하는 모든 n에 대하여 $c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$인 조건을 만족하는 세 양의 상수 $c_{1,}\ c_2$와 n_0가 존재하기만 하면 $f(n)=\Theta(g(n))$이다.	상한과 하한을 동시에 지정

① **빅오** 정의 $f(n)=O(g(n))$에 사용된 기호 '='는 좌우가 '동등하다'라는 의미가 아니다.
• 기호 '='는 '~이다'라는 의미로 해석하면 혼란을 막을 수 있다.
• $f(n)=O(g(n))$는 '$f(n)$은 $g(n)$의 빅오이다' 라는 의미이다.
• 충분히 큰 수에 대해 $f(n)=O(g(n))$은 부등식 $f(n) \leq g(n)$이 성립하는 것을 의미한다.
• $g(n)$의 값은 $f(n)$의 **상한값**이라는 의미이다. 즉, **최대시간을 예측**할 수 있게 한다.

② **빅오메가**(Ω) 함수는 **최소시간을 예측**할 수 있게 한다.

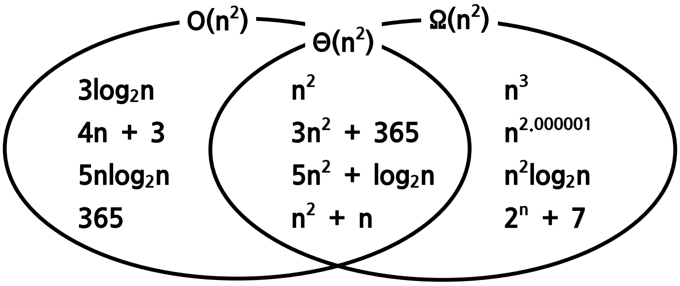
수식	
$f(x)=x^2+5x+100$	$f(x)=\Omega(1),\ f(x)=\Omega(x),\ f(x)=\Omega(x^2)$는 모두 다 성립하지만, $f(x)=\Omega(x^3),\ f(x)=\Omega(x^4)$ 등은 성립하지 않는다.

• 하한의 설정은 어떤 알고리즘이 수행되는데 필요한 최소시간을 예측할 수 있게 한다.

정렬, 검색 등의 알고리즘을 구현할 때 필요로 하는 최소시간은 과연 얼마인가? 라는 문제를 한번 생각해 볼 수 있다. 예를 들어, n개의 자료로 구성된 리스트에서 각 원소를 비교하면서 자리를 바꾸는 방법으로 정렬을 수행할 때 어떤 방법을 사용하더라도 시간복잡도가 $n\log n$ 보다 작아질 수 없음이 증명되어 있다. 이를 $f(n)=\Omega(n\log n)$로 표현할 수 있으며, **더 좋은 알고리즘 개발**을 위한 **헛수고**는 그만 두라는 의미이다.

③ **빅세타**(Θ)는 O와 Ω를 합쳐 놓은 형태이다.
• Θ는 최고차수의 상하한을 동시에 지정하므로 함수의 정확한 최고차수를 나타낼 수 있다.
• 어떤 함수를 Θ로 나타낼 때도 최고차항의 차수 이외는 모두 무시된다.

◈ $O(n^2)$, $\Omega(n^2)$, $\Theta(n^2)$의 관계



- **$O(n^2)$**에 속하는 함수는 최고차항의 **차수가 2와 같거나 작은 함수**들이다.
- 알고리즘의 f(n)이 O(g(n))에 속하면, f(n)의 복잡도는 최악의 경우도 g(n)과 같거나 작다는 뜻이다.
- **$O(n^2)$**은 점근 상한을 제시한다.

- **$\Omega(n^2)$**에 속하는 함수는 최고차항의 **차수가 2와 같거나 큰 함수**들이다.
- 알고리즘의 f(n)이 Ω(g(n))에 속하면, f(n)의 복잡도는 최선의 경우도 g(n)과 같거나 크다는 뜻이다.
- **$\Omega(n^2)$**은 점근 하한을 제시한다.

- **$\Theta(n^2)$**에 속하는 함수는 최고차항의 **차수가 2인 함수**들이다.
- **$\Theta(n^2)$**은 점근 상한과 하한의 교집합이다.

- 이진검색의 실행시간은 항상 $O(\log_2 n)$이라고 할 수 있다.
- 이진검색의 실행시간은 항상 $\Theta(\log_2 n)$이라고 할 수 **없다.**
- 이진검색에서 검색 대상이 첫 번째 있으면, 실행시간은 $\Theta(1)$이다.
- 이진검색에서 최악의 경우 실행시간은 $\Theta(\log_2 n)$이라고 할 수 있다.
- 해서, 이진검색의 실행시간은 $O(\log_2 n)$이라고 하는 것이 가장 상세한 표현이다.

◈ $\sqrt{n}$의 점근 표기

- $\sqrt{2}$ = 1.41421...(무리수 : 순환하지 않은 무한소수)
- $\log_2 n$과 $\sqrt{n}$

n	1	2	4	8	16	32	64	128	256
$\log_2 n$	0	1	2	3	4	5	6	7	8
$\sqrt{n}$	1	$\sqrt{2}$	2	$2\sqrt{2}$	4	$4\sqrt{2}$	8	$8\sqrt{2}$	16

- $\log_2 n \leq \sqrt{n}$이다. 해서, $\sqrt{n} = \Omega(\log_2 n)$이다.

//탐구------------------------------

빅오 함수

[예제 1] $n^2+3n+2=\ O(n^2)$**이 성립하는가?**

증명 1	n≥1이면, $\lvert n^2+3n+2\rvert \le \lvert n^2+3n^2+2n^2\rvert\ =\ 6\lvert n^2\rvert$ 두 양의 상수 $c=6,\ n_0=1$가 존재(선택)하므로 **성립**한다.
증명 2	n≥2이면, $\lvert n^2+3n+2\rvert \le \lvert n^2+2n^2+n^2\rvert\ =\ 4\lvert n^2\rvert$ 두 양의 상수 $c=4,\ n_0=1$가 존재(선택)하므로 **성립**한다.

[Tip] $f(n)=O(g(n))$에서 부등식 '$f(n)\ \le\ g(n)$'을 **증명**하는 원리는 부등식이 **성립**하게 되는 **임의의 두 양의 상수** $c,\ n_0$가 **존재하는 것을 찾는 것**이다.

[예제 2] $5n^2=\ O(n^3)$**이 성립하는가?**

증명	n≥5이면 $5n^2\le n^3$이다. 따라서, 두 양의 상수 $c=1, n_0=5$가 존재함으로 $5n^2=\ O(n^3)$은 **성립**한다.

[예제 3] $n^3=\ O(5n^2)$**이 성립하는가?**

증명	$n^3\le c(5n^2)$이 성립하는 두 양의 상수 $c,\ n_0$가 존재하느냐? 이다. 부등식을 n^2으로 나누면, n≤$5c$이다. 여기서, n은 큰 수이므로 n≤$5c$를 만족하는 c는 존재하지 않는다. 따라서, $n^3=\ O(5n^2)$는 **성립하지 않는다.**

[예제 4] 1 + 2 + 3 + 4 + … + n = $O(n^2)$**이 성립하는가?**

증명	n≥1이면, 1 + 2 + 3 + 4 + … + n ≤ n^2 두 양의 상수 $c=1, n_0=1$가 존재(선택)함으로 **성립**한다.

[예제 5] $n!=\ O(n^n)$**이 성립하는가?**

증명	n≥1이면, 1•2•3•4•…•n ≤ n^n = n•n•n•…•n n! ≤ n^n 두 양의 상수 $c=1, n_0=1$가 존재(선택)함으로 **성립**한다.

[예제 6] $\log 1n! = O(n\log n)$**이 성립하는가?**

증명	n≥1이면, $n! \le n^n$ ------〉 [예제 5]에서 증명함 $\log n! \le \log n^n$ ------〉 양변에 log를 취함 $\log n! \le n\log n$ $n! = O(n^n)$이 성립함으로 $\log n! = O(n\log n)$도 **성립**한다.

◈ 왜, 알고리즘의 복잡도는 주로 O로 표현하는가?

- Θ는 최고 차수를 정확하게 나타내므로 O나 Ω보다 더 정확하고 유용한 정보가 될 수 있다.
- 하지만, 알고리즘이라는 것은 최선, 평균, 최악의 경우가 있을 수 있다.
- 최악의 경우를 항상 생각해야 하므로 최고 차수가 얼마 이하인가?는 매우 중요한 점이다.
- 일반적으로, O 함수를 사용하는 이유이다.
- 그리고, 어떤 알고리즘 수행시간을 **정확하게 산출하는 것**은 매우 어렵고, 낭비를 초래할 수 있다.
  → 이유는 알고리즘 구조, 사용 언어, 입출력 크기, 수행 컴퓨터, 사용자 수 등에 영향을 받으므로
- 해서, 알고리즘 수행시간은 점근 표기법을 이용하여 개략적인 차수로 알고리즘의 복잡도를 나타낸다.

기출문제 분석

1. 다음은 알고리즘 복잡도 X를 위한 정의이다. 어떤 복잡도에 대한 정의인가? [2010년 국가 7급]

정의 : f(n) = X(g(n)) **모든 n(n≥n0)에 대해 f(n)≤cg(n)을 만족하는 두 양의 상수 c와 n0가 존재하면 f(n) = X(g(n))이다.**

① O(big oh) ② Ω(omega)
③ Γ(gamma) ④ Θ(theta)

♤ O(big oh) 함수 정의

$n \ge n_0$**를 만족하는 모든** n**에 대하여** $f(n) \le c \cdot g(n)$**인 조건을 만족하는 두 양의 상수** c**와** n_0**가 존재하기만 하면** $f(n)=O(g(n))$**이다.**

정답 : ①

2. <보기>는 알고리즘의 시간복잡도(time complexity)를 표시하는 방법 중 하나인 빅오(big-oh) 표기법에 대한 정의이다. 빈칸 ㈎와 ㈏에 각각 들어갈 수식은? [2020년 서울 7급]

----<보기>--

두 개의 함수 g(n)과 f(n)에 대해서 O(g(n))은 다음과 같이 정의한다.

O(g(n)) = {f(n) : 모든 ㈎ 에 대해서 ㈏ 인 양의 상수 c와 n0가 존재한다}

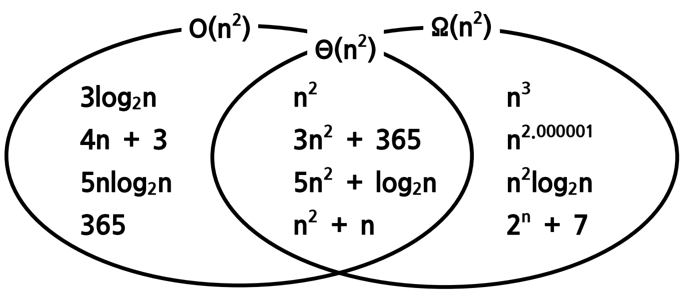
	㈎	㈏
①	n≥n0	0≤f(n)≤c×g(n)
②	n0≥n	0≤f(n)≤c×g(n)
③	n≥n0	0≤g(n)≤c×f(n)
④	n0≥n	0≤g(n)≤c×f(n)

♤ 시간복잡도를 표시하는 방법

- 점근 표기법은 크기 n이 커질 때, 각 함수가 어떻게 성장하는가? 이다.
- 점근 표기법은 알고리즘의 복잡도를 **단순하게 표시**하는데 사용된다.
- 점근 표기법으로 O, Ω, Θ 등이 있다.

함수	정의	주 특징
빅오 (O)	$n \geq n_0$를 만족하는 모든 n에 대하여 $0 \leq f(n) \leq c \cdot g(n)$인 조건을 만족하는 두 양의 상수 c와 n_0가 존재하기만 하면 $f(n)=O(g(n))$이다.	최대시간 예측 **(상한)**
빅오메가 (Ω)	$n \geq n_0$를 만족하는 모든 n에 대하여 $f(n) \geq c \cdot g(n) \geq 0$인 조건을 만족하는 두 양의 상수 c와 n_0가 존재하기만 하면 $f(n)=\Omega(g(n))$이다.	최소시간 예측 **(하한)**
빅세타 (Θ)	$n \geq n_0$를 만족하는 모든 n에 대하여 $0 \leq c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$인 조건을 만족하는 세 양의 상수 c_1, c_2와 n_0가 존재하기만 하면 $f(n)=\Theta(g(n))$이다.	상한과 하한을 동시에 지정

↓ $O(n^2)$, $\Omega(n^2)$, $\Theta(n^2)$의 관계



정답 : ①

3. 함수 또는 수식들의 복잡도에 대한 설명 중 가장 적절하지 않은 것은? [2021년 군무원 7급]

① $\sum_{k=1}^{n} k$는 $O(n^2)$이다.

② 함수 $2n^2+3n+4$는 $\Omega(n^2)$이다.

③ 함수 $3n\log n+n^2+2$는 $\Theta(n\log n)$이다.

④ $\sum_{k=1}^{n} k^3$는 $\Theta(n^4)$이다.

♤ **복잡도**

• 함수 $3n\log n + n^2 + 2$는 $\Theta(n\log n)$이다.(×) → $\Theta(n^2)$이다.

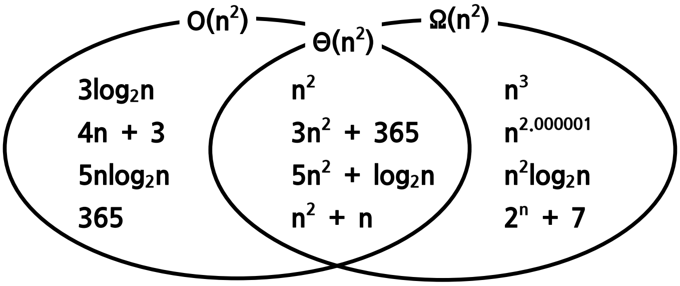
• $\sum_{k=1}^{n-1} k = 1 + 2 + 3 + \cdots + (n-1) = \dfrac{n(n-1)}{2} = O(n^2)$

• $\sum_{k=1}^{n} k = 1 + 2 + 3 + \cdots + n = \dfrac{n(n+1)}{2} = O(n^2)$

• $\sum_{k=1}^{n} k^2 = 1^2 + 2^2 + 3^2 + \cdots + n^2 = \dfrac{n(n+1)(2n+1)}{6} = O(n^3)$

• $\sum_{k=1}^{n} k^3 = 1^3 + 2^3 + 3^3 + \cdots + n^3 = \dfrac{n^2(n+1)^2}{4} = \Theta(n^4)$

// $O(n^2)$, $\Omega(n^2)$, $\Theta(n^2)$의 관계



• **$O(n^2)$**에 속하는 함수는 최고차항의 **차수가 2와 같거나 작은 함수**들이다.
• **$\Theta(n^2)$**에 속하는 함수는 최고차항의 **차수가 2인 함수**들이다.
• **$\Omega(n^2)$**에 속하는 함수는 최고차항의 **차수가 2와 같거나 큰 함수**들이다.

정답 : ③

4. 시간복잡도 함수 f(n)의 점근표기법에 대한 설명으로 가장 옳지 않은 것은? [2021년 서울 7급]

① f(n)=$\Theta(n^2)$이면 f(n)=$O(n^2)$이다.

② f(n)=$O(n^2)$이면 f(n)=$\Theta(n^2)$이다.

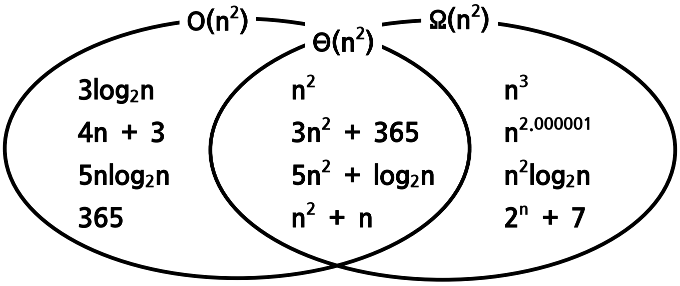
③ f(n)=$O(n^2)$이면 f(n)=$O(n^3)$이다.

④ f(n)=$\Omega(n^2)$이면 f(n)=Ω(n)이다.

♤ **시간복잡도**

- f(n)=$O(n^2)$이면 f(n)=$\Theta(n^2)$이다.(×)
  → **$O(n^2)$**에 속하는 함수는 최고차항의 **차수가 2와 같거나 작은 함수**들이다.
  → **$\Theta(n^2)$**에 속하는 함수는 최고차항의 **차수가 2인 함수**들이다.
- 해서, f(n)=$O(n^2)$이면 f(n)=$\Theta(n^2)$은 **성립하지 않는다.**

// **$O(n^2)$, $\Omega(n^2)$, $\Theta(n^2)$의 관계**



- **$O(n^2)$**에 속하는 함수는 최고차항의 **차수가 2와 같거나 작은 함수**들이다.
- 알고리즘의 f(n)이 O(g(n))에 속하면, f(n)의 복잡도는 최악의 경우도 g(n)과 같거나 작다는 뜻이다.
- **$O(n^2)$**은 점근 상한을 제시한다.

- **$\Omega(n^2)$**에 속하는 함수는 최고차항의 **차수가 2와 같거나 큰 함수**들이다.
- 알고리즘의 f(n)이 Ω(g(n))에 속하면, f(n)의 복잡도는 최선의 경우도 g(n)과 같거나 크다는 뜻이다.
- **$\Omega(n^2)$**은 점근 하한을 제시한다.

- **$\Theta(n^2)$**에 속하는 함수는 최고차항의 **차수가 2인 함수**들이다.
- **$\Theta(n^2)$**은 점근 상한과 하한의 교집합이다.

정답 : ②

5. 점근적 표기법(asymptotic notation)에 관한 사칙연산으로 옳지 않은 것은? [2009년 국가 7급]

① $O(n) + O(n) = O(n)$

② $O(n^2) + O(nlogn) = O(n^2)$

③ $O(n^2) + O(n) = O(n^2)$

④ $O(\log n) \cdot O(n^2) = O(n^2)$

♤ 빅오(O) 표현

$O(\log n) \cdot O(n^2) = O(n^2 \log n)$ → 곱셈은 덧셈과 다르다.

정답 : ④

6. 알고리즘 시간복잡도 O(1)이 의미하는 것으로 옳은 것은? [2002년 기술고시]

① 모든 O(1) 알고리즘은 어떤 컴퓨터에서나 똑같은 수행시간을 갖는다는 뜻이다.

② 모든 O(1) 알고리즘은 한 컴퓨터에서나 동일한 수행시간을 갖는다는 의미이다.

③ 알고리즘의 입력 데이터 수가 한 개라는 의미이다.

④ 알고리즘의 수행시간이 입력 데이터 수와 관계없이 일정하다는 의미이다.

⑤ 알고리즘의 길이가 입력 데이터보다 작다는 의미이다.

♤ 시간복잡도 O(1)

- O(1)은 수행시간이 **상수**시간이라는 뜻이다.
- 상수시간은 어떤 상황과 관계없이 일정한 속도로 빠르게 자료를 처리할 수 있다는 뜻이다.
- 점근 표기에서 상수시간은 O(1)로 표현한다.

정답 : ④