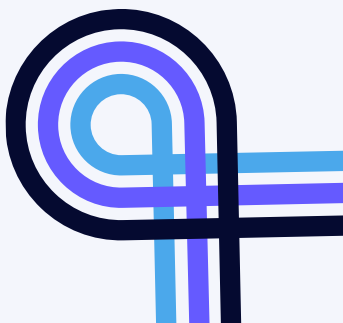


# Tutorial 2

## Analysis and Design of Algorithms

Asymptotic Analysis





# Asymptotic Analysis

The **exact running time** of an algorithm is a complex expression, therefore, we **estimate it**.

We consider only the **highest order** term of expression and we **suppress** any constant factors

**Example:**

$3n^2+2n+4$  is  
 $n^2$  asymptotically



**Exercise 2-1** From CLRS (©MIT Press 2001)

Asymptotically rank the following functions:

$n, n^{1/2}, \log(n), \log(\log(n)), \log^2(n), (\frac{1}{3})^n, 4, (\frac{3}{2})^n, n!$



# Asymptotic Notations

- Asymptotic notations are mathematical tools to represent the time complexity of algorithms for asymptotic analysis
- There are 2 methods to find the complexity class of an algorithm:

1

**Definition**

2

**Limit Test**

# Asymptotic Notations:

## Definiton

**Big-Oh Notation:**

$$T(n) = O(f(n))$$

There exists positive constants  $c$  and  $n_0$  where:

$$0 \leq T(n) \leq cf(n) \\ \text{for all } n \geq n_0$$

**Big-Omega Notation:**

$$T(n) = \Omega(f(n))$$

There exists positive constants  $c$  and  $n_0$  where:

$$T(n) \geq cf(n) \geq 0 \\ \text{for all } n \geq n_0$$

**Big-Theta Notation:**

$$T(n) = \Theta(f(n))$$

There exists  $c_1$ ,  $c_2$  and  $n_0$  where:

$$0 \leq c_1f(n) \leq T(n) \leq c_2f(n) \\ \text{for all } n \geq n_0$$



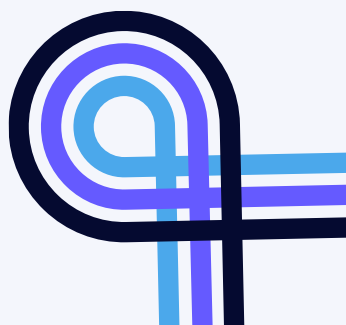

**Exercise 2-4** From CLRS (©MIT Press 2001)

For every given  $f(n)$  and  $g(n)$  prove that  $f(n) = \Theta(g(n))$

a)  $g(n) = n^3, f(n) = 3n^3 + n^2 + n$

b)  $g(n) = 2^n, f(n) = 2^{n+1}$

c)  $g(n) = \ln(n), f(n) = \log_{10}(n) + \log_{10}(\log_{10} n)$



# Asymptotic Notations:

## Definiton II

Small-Oh Notation:  
 $T(n) = o(f(n))$

For any positive constant  $c$ ,  
there is a positive constant  
 $n_0$  where:

$$0 \leq T(n) < cf(n) \\ \text{for all } n \geq n_0$$

Small-Omega Notation:  
 $T(n) = \omega(f(n))$

For any positive constant  $c$ ,  
there is a positive constant  
 $n_0$  where:

$$0 \leq T(n) > cf(n) \\ \text{for all } n \geq n_0$$

# Asymptotic Notations: Limit Test

The limit test is used to determine the dominance class of a function

## Asymptotic Notations and the Limit Test

If  $\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)}$

- ①  $= 0$ , then  $f(n) = o(g(n))$ .
- ②  $= \infty$ , then  $f(n) = w(g(n))$ .
- ③  $= c \in \mathbb{R}^+$ , then  $f(n) = \Theta(g(n))$ .
- ④  $\neq \infty$ , then  $f(n) = O(g(n))$ .
- ⑤  $\neq 0$ , then  $f(n) = \Omega(g(n))$ .



**Exercise 2-4** From CLRS (©MIT Press 2001)

For every given  $f(n)$  and  $g(n)$  prove that  $f(n) = \Theta(g(n))$

a)  $g(n) = n^3, f(n) = 3n^3 + n^2 + n$

b)  $g(n) = 2^n, f(n) = 2^{n+1}$

c)  $g(n) = \ln(n), f(n) = \log_{10}(n) + \log_{10}(\log_{10} n)$





### Exercise 2-5

For every given  $f(n)$  and  $g(n)$  prove that  $f(n) = o(g(n))$  or  $f(n) = \omega(g(n))$

a)  $f(n) = n^3, g(n) = n^2$

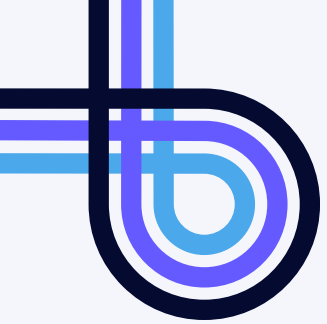
b)  $f(n) = \log(n), g(n) = \log^2(n)$





**Exercise 2-6** From CLRS (©MIT Press 2001)

Let  $f(n)$  and  $g(n)$  be asymptotically non-negative functions. Using the basic definition of  $\Theta$ -notation, prove that  $\max(f(n), g(n)) = \Theta(f(n) + g(n))$ .



**All done!**

