

# **Chapter 3**

## **Growth of Functions**

Algorithm Analysis

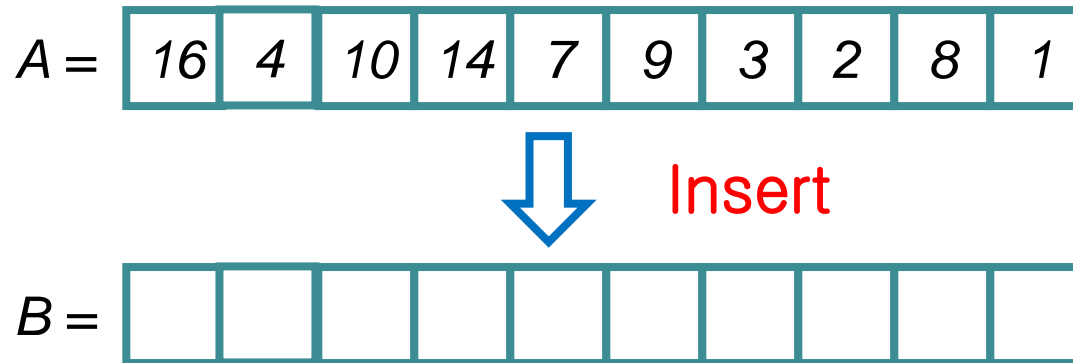
School of CSEE

# Question 1

heapsort강의에 대한 질문입니다.

1. heap을 알고리즘으로 구현하는 방법에 따라 heapify된 array의 원소들의 level과 순서가 달라질 수 있나요?

Q: How else can we build heap?



May have different order, but it is slower.

# Question 1

2. heap을 이용해서 구현한 priority queues는 들어온 순서와 상관없이 오직 우선순위만 비교하는 알고리즘인가요? 같은 우선순위를 가지는 값의 경우에도 들어온 순서는 무관한가요?

case 1) Insert [2:A], [1:B], [1:C] in this order, then Delete


case 2) Insert [2:A], [1:C], [1:B] in this order, then Delete

2에서 추가 질문)

만약 순서와 무관하다면 **priority queues** 알고리즘을 이용해 스케줄링할 때 계속해서 새로운 task가 들어온다면 일부 task가 heap 내부에서 실행이 안 되고 계속해서 대기 중인 상태로 존재하는 상황도 발생할 수 있나요?

# Question 2

ch3\_2에서 수식에 오류가 있어 보입니다. 여기 마지막에  $n^{(999/1000)} / n$  이 부분에서 결과가  $n^{(1/1000)}$ 이 아니라  $n^{(-1/1000)} = 1 / n^{(1/1000)}$ 이 되어 무한대가 아닌 0으로 수렴합니다.



한동대학교  
HANDONG GLOBAL UNIVERSITY  
School of CSEE

## Hierarchy of Orders

Given function  $f(n)$  and  $g(n)$ , we say that  $f(n)$  has **smaller order than**  $g(n)$ , if  $O(f(n))$  is strictly contained in  $O(g(n))$ .

i.e.,  $O(f(n)) \subset O(g(n))$ .

Example:

$$O(1) \subset O(\log n) \subset O(\sqrt{n}) \subset O(n) \subset O(n \log n) \subset O(n^2) \subset O(n^3) \subset O(2^n) \subset O(n2^n) \subset O(n!)$$

Where does  $O(n^{1/1000})$  belong?

$$= \lim_{n \rightarrow \infty} \frac{n^{1/1000}}{n} = \lim_{n \rightarrow \infty} \frac{1}{1000} n^{-999/1000} = 0$$

# Question 3

이번 과제를 하던 중 제가 임의적으로 처리해야하는지 확인하고 싶어 이렇게 문의를 남깁니다.

이번에 구현해야하는 함수 중 하나가 Id value를 decrease 해야하는 함수가 있습니다. 이 때 음의 정수 만큼 decrease를 하였을 때(예를 들면 원래 id 값 1000에서 -4를 decrease 한다면 1004가 되는 경우)도 decrease되었다고 봐야할 지, 아니면 재량껏 exception으로 처리하여 할 지 판단이 되지 않아 교수님께 이렇게 질문을 드립니다

# Problem 1

$O(g(n)) = \{ f(n) : \text{for any positive constant } c, \text{ there exists a constant } n_0 > 0 \text{ such that } 0 \leq f(n) \leq c \cdot g(n) \text{ for all } n \geq n_0 \}$

For  $5n^2 = O(n^3)$ , find the value of  $n_0$  and  $c$  which satisfy the condition.

Ans)  $(n_0, c): (1, 5) \quad (5, 1), \dots$

$(1, 5): \text{ when } n \geq 1, 0 \leq 5n^2 \leq c \cdot n^3, \text{ where } c = 5.$

# Problem 2

Let  $f(n)$  and  $g(n)$  be asymptotically nonnegative functions.  
Select all the statements that are true.

- 1)  $5n^2 = O(n^3)$
- 2)  $f(n) = O(g(n))$  and  $O(g(n)) = f(n)$  have the same meaning.
- 3)  $n^2 + O(n) = O(n^2)$
- 4)  $\frac{1}{2}n^2 = \Theta(n^2)$
- 5)  $\frac{1}{2}n^2 = o(n^2)$

# Problem 4

For the functions,  $\lg n$  and  $\log_8 n$ , what is the asymptotic relationship between these functions?

( $\lg n = \log_2 n$  :  $\lg n$  is a notation for a binary logarithm)

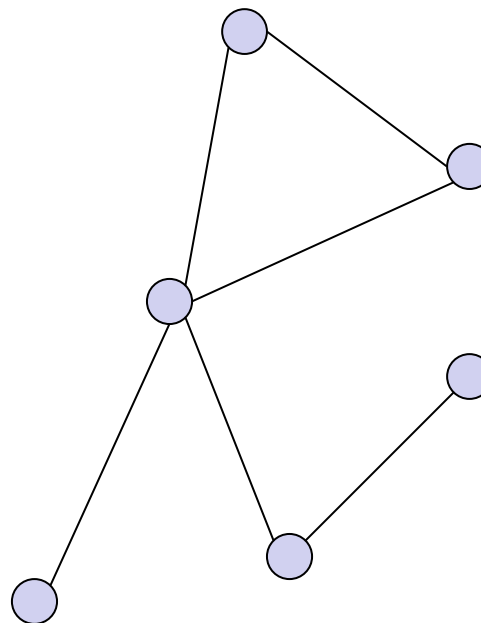
Select all that apply.

- 1)  $\lg n = O(\log_8 n)$
- 2)  $\lg n = \Omega(\log_8 n)$
- 3)  $\lg n = \Theta(\log_8 n)$



# Problem 5

- Clique problem:
  - Input: undirected graph  
 $G = (V, E)$
  - Output: largest subset  $C$  of  $V$  such that every pair of vertices in  $C$  has an edge between them
- Best known algorithm:  
 $O(n 2^n)$  time.



# Problem 5

- Goto ChatGPT, and ask it to solve clique problem with C++ language. Also measure execution time.

Then try it when  $n=15, 20, 23$ , and  $25$ , etc.

# Chapter 6

## Heap Sort

Algorithm Analysis

School of CSEE

# Problem 1

Is the following a max-heap or not?

(1)

100	90		89	67	25	80	
-----	----	--	----	----	----	----	--

(2)

100	90	52	89	67	25	55	80
-----	----	----	----	----	----	----	----

(3)

100	40	82	35	25	1		
-----	----	----	----	----	---	--	--

# Problem 2

Select all the statements that are true.

- 1) An array that is sorted in ascending order is a min-heap.
- 2) The smallest element in the max-heap is the last element in the array.

# Problem 3

For following array, apply 'Build-Max-Heap( $A$ )'.

1	2	3	4	5	6	7	8	9	10
<b>20</b>	<b>1</b>	<b>16</b>	<b>9</b>	<b>10</b>	<b>6</b>	<b>8</b>	<b>9</b>	<b>3</b>	<b>2</b>

# Problem 4

Do the following operations in sequence to a max priority queue. (Integer number is key.) What is the final result?

- |                 |                 |                 |
|-----------------|-----------------|-----------------|
| 1. Insert(A, 4) | 2. Insert(B, 3) | 3. Insert(C, 5) |
| 4. Insert(D, 4) | 5. Delete       | 6. Delete       |
| 7. Insert(E, 4) | 8. Insert(F, 5) | 9. Delete       |

# Problem 4

1. Insert(A, 4)	(A,4)				
2. Insert(B, 3)	(A,4)	(B,3)			
3. Insert(C, 5)	(C,5)	(B,3)	(A,4)		
4. Insert(D, 4)	(C,5)	(D,4)	(A,4)	(B,3)	
5. Delete	(D,4)	(B,3)	(A,4)		
6. Delete	(A,4)	(B,3)			
7. Insert(E, 4)	(A,4)	(B,3)	(E,4)		
8. Insert(F, 5)	(F,5)	(A,4)	(E,4)	(B,3)	
9. Delete	(A,4)	(B,3)	(E,4)		