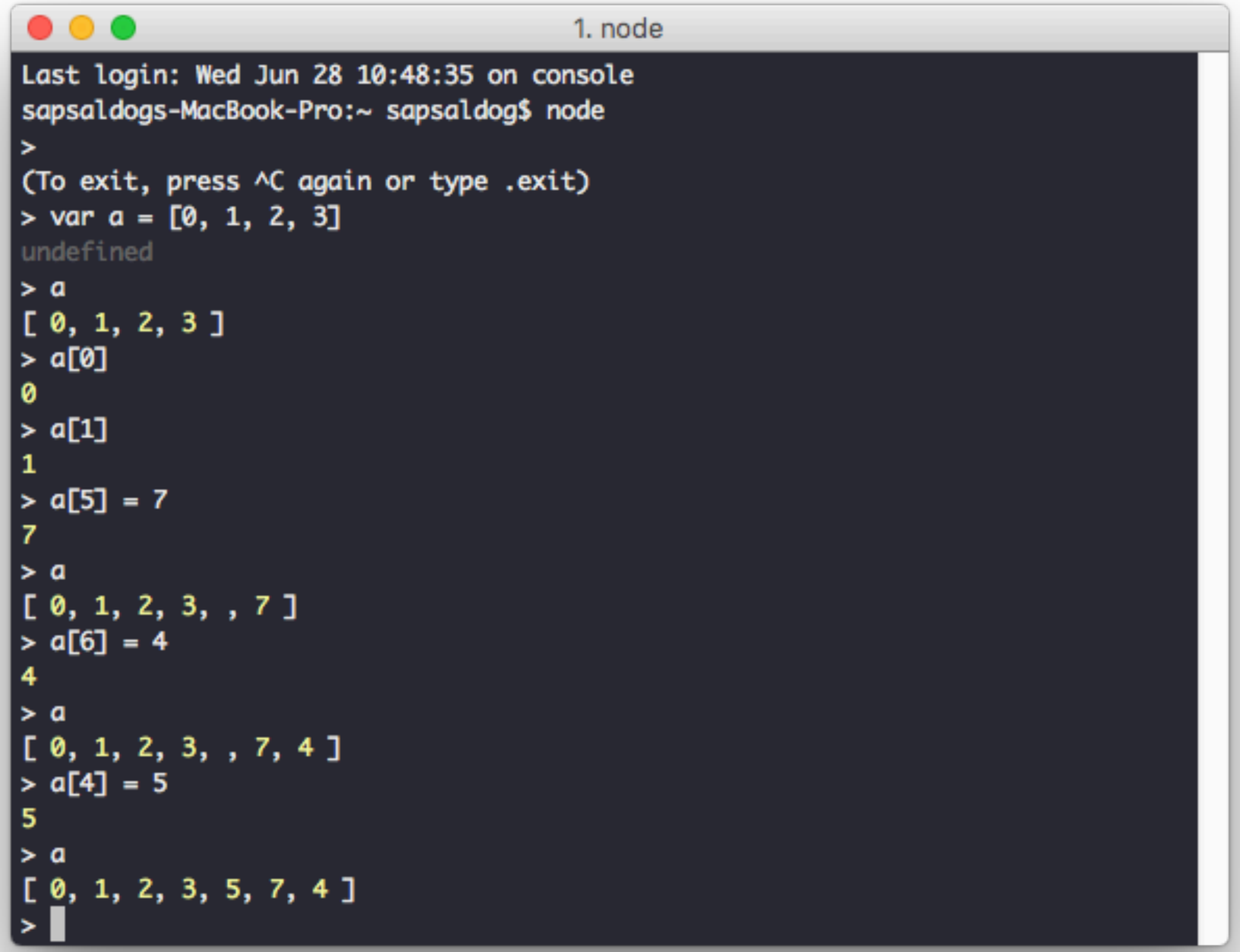
자료구조

COMPUTATIONAL THINKING

2017.07.03
최동훈

배열(Array)

2017.07.03
최동훈



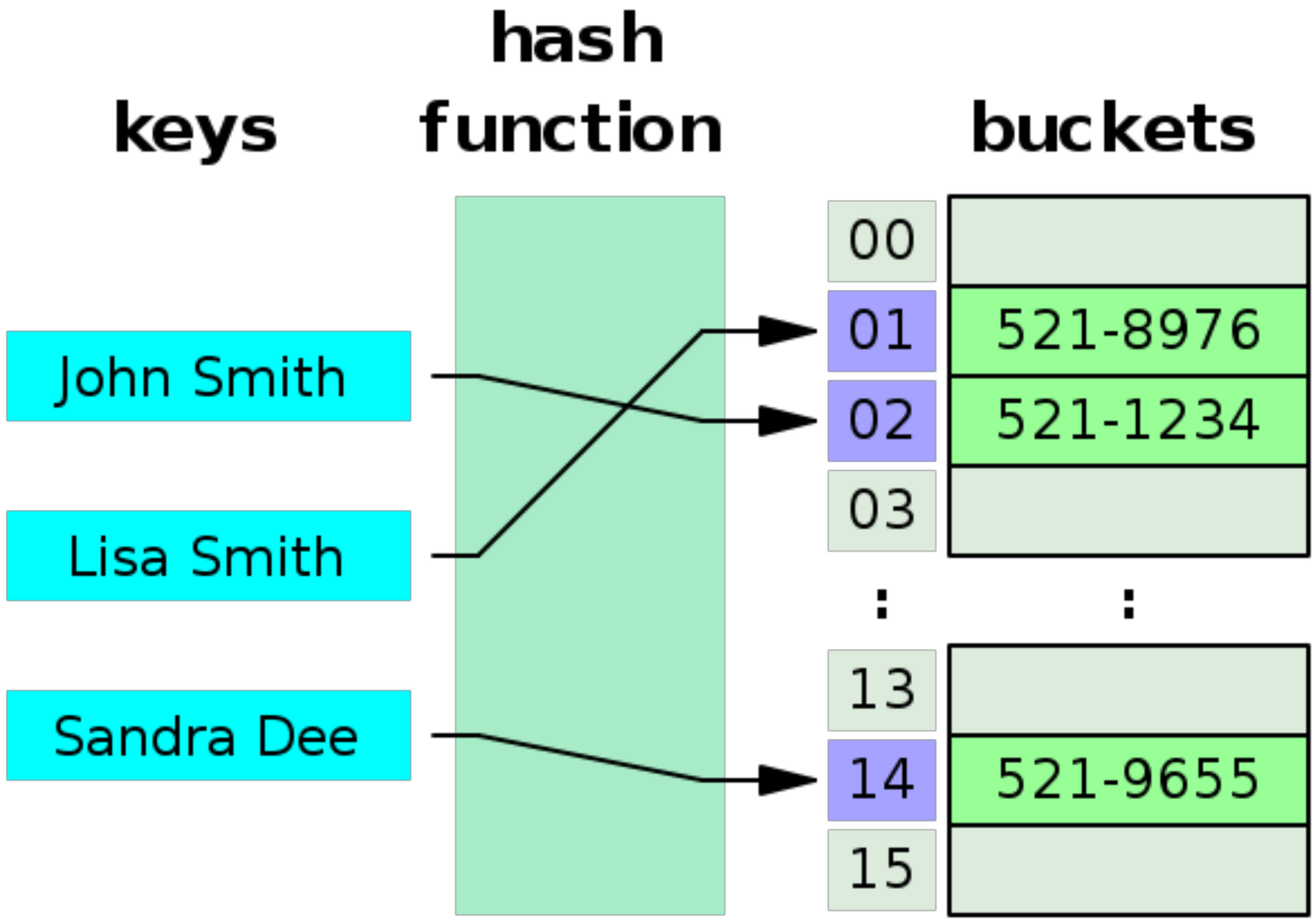
2017.07.03
최동훈

- https://www.w3schools.com/js/js_arrays.asp
- https://www.w3schools.com/js/js_array_methods.asp

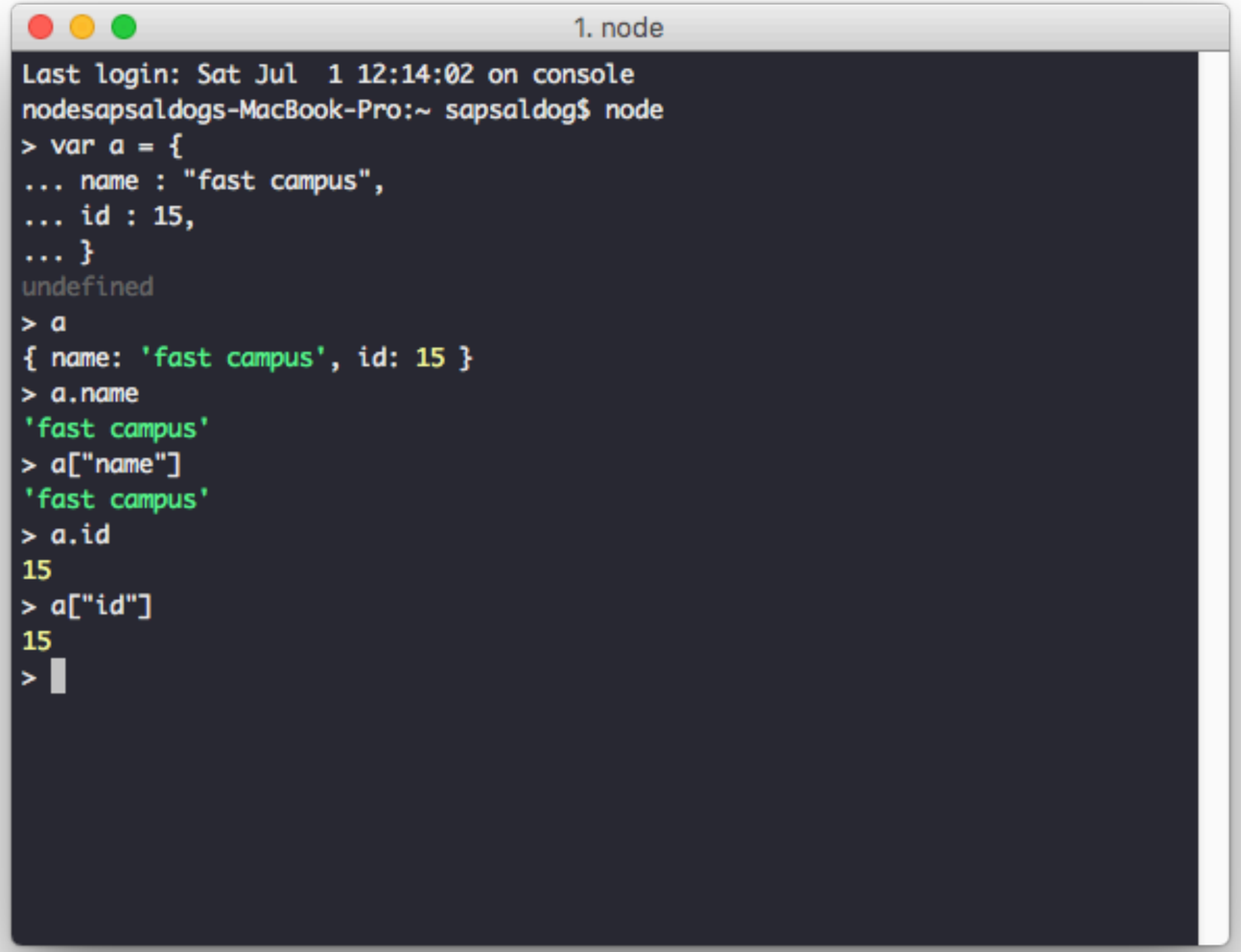
2017.07.03
최동훈

해쉬테이블(hash table)

2017.07.03
최동훈



2017.07.03
최동훈



2017.07.03
최동훈

Fast campus

당신의 커리어 전환점 패스트캠퍼스

프로토타입(prototype)

FRONTEND PROGRAMMING SCHOOL

2017.07.03
최동훈

프로토타입(prototype)

- 한국어로 “원형”의 의미를 지님
- 프로토타입을 바탕으로 객체(object)가 생성됨 -> 상속
- 생성자
- this
- 오버라이드

2017.07.03
최동훈

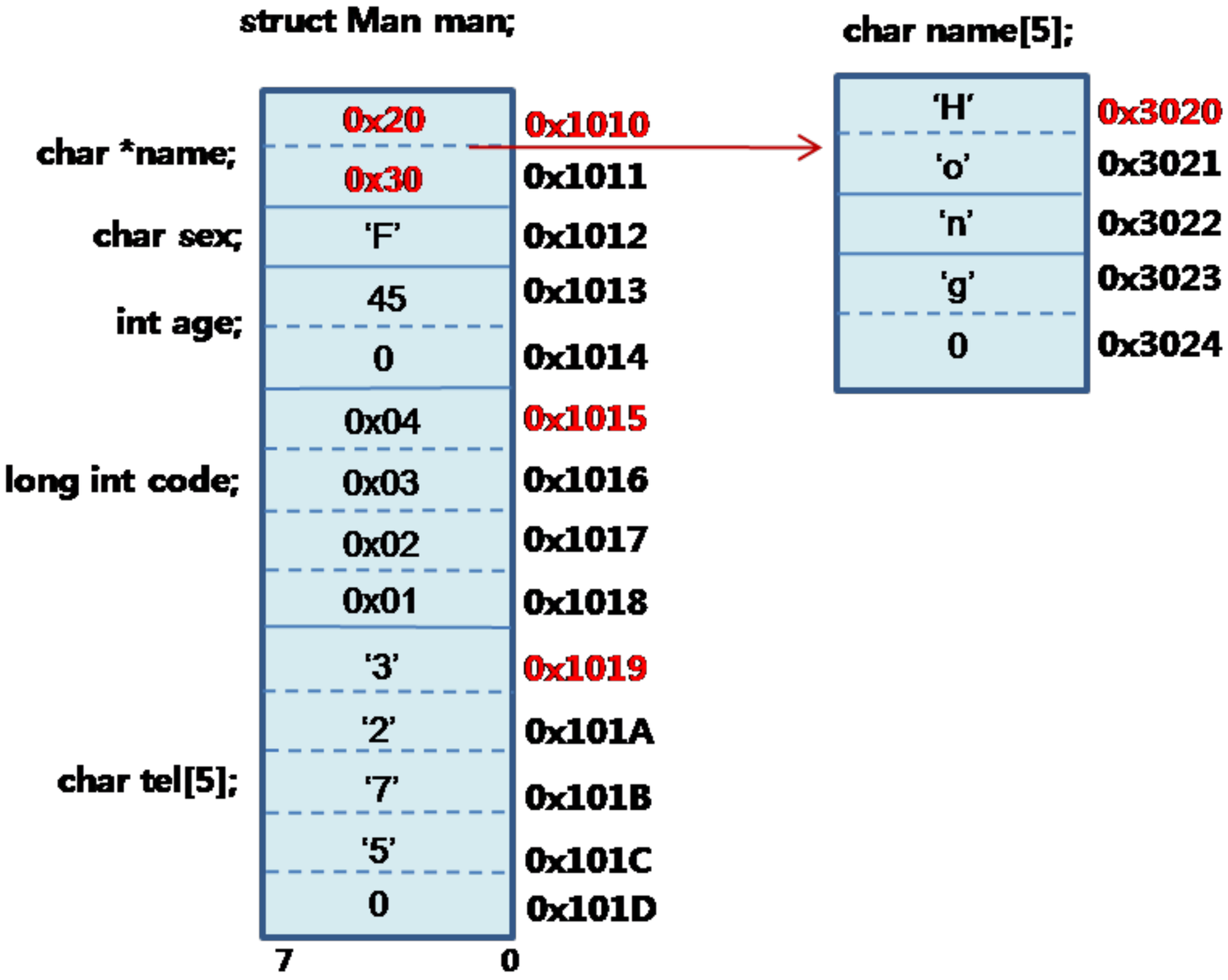
포인터(Pointer)

2017.07.03
최동훈

포인터(pointer)

- **primitive type** vs **reference type**
- 자바스크립트의 객체(object)는 pointer

2017.07.03
최동훈



2017.07.03
최동훈

재귀호출(recursive call)

2017.07.03
최동훈

재귀함수

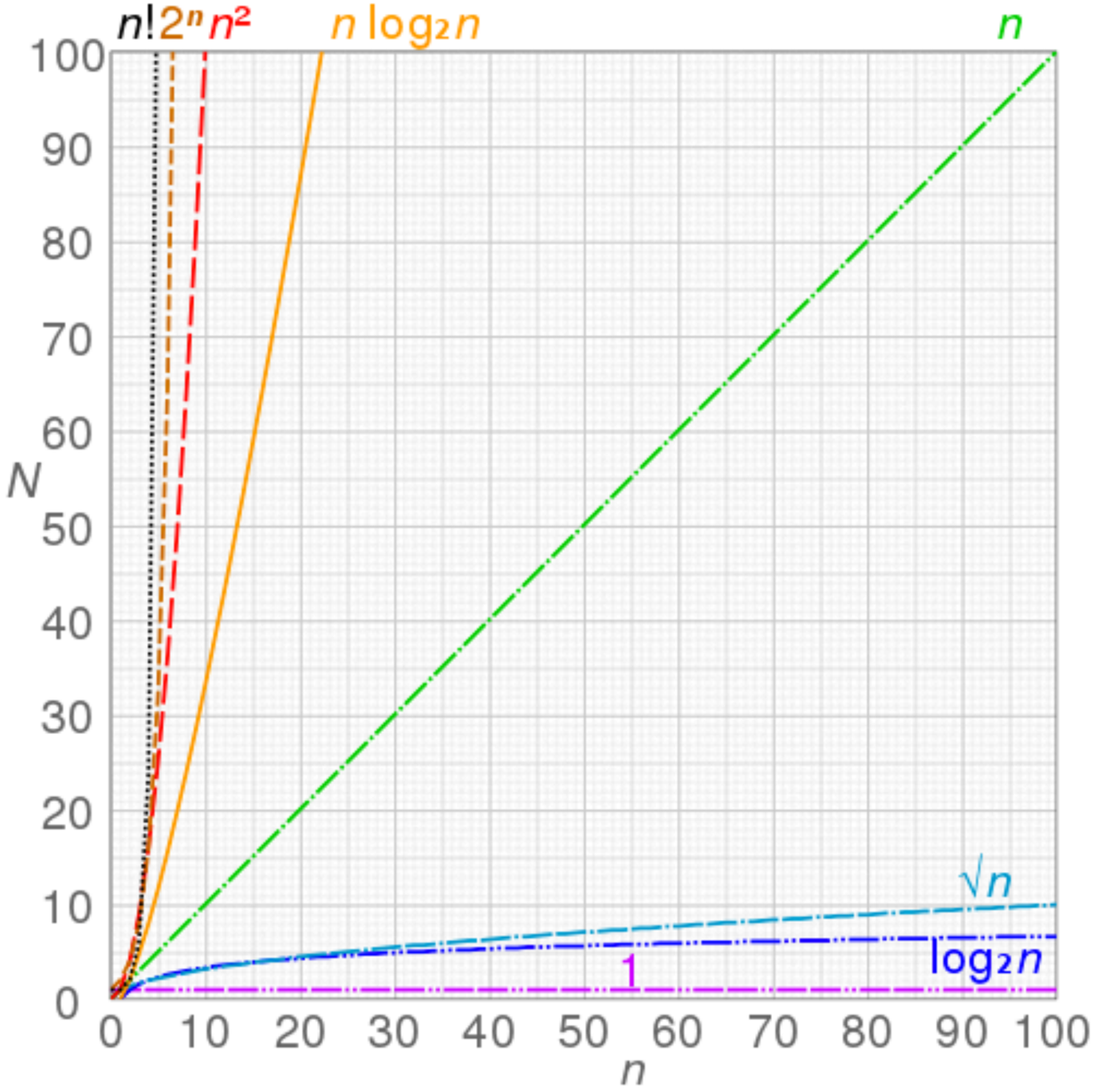
- 어떤 함수의 코드에 자기 자신을 호출하는 코드가 포함 된 함수를 재귀 함수라고 한다.
- 코드를 이해하기 쉽게 한다.
- 스택 오버플로우의 위험이 있다.
- 종료 조건을 반드시 넣어주어야 한다.

2017.07.03
최동훈

빅오 표기법(Big O notation)

- 복잡도(complexity)를 표기
- 시간 복잡도와 공간 복잡도
- 입력량과의 관계를 나타냄 - 비례의 정도

2017.07.03
최동훈



2017.07.03
최동훈

리스트

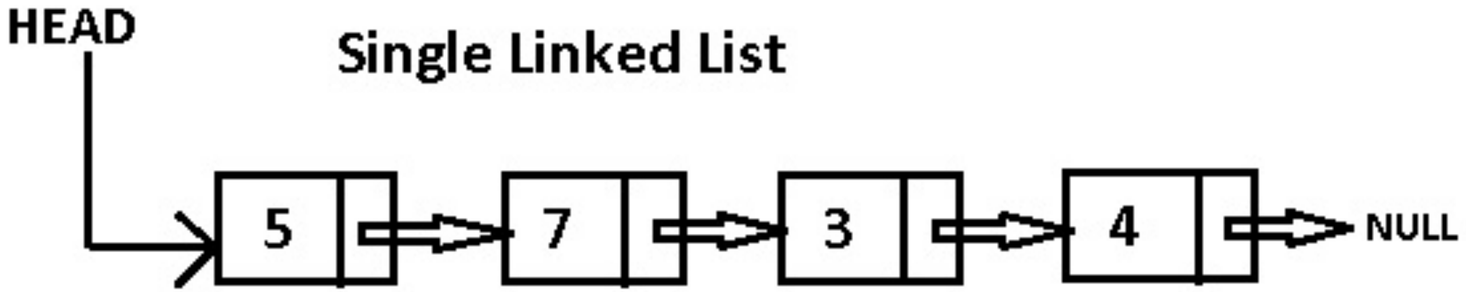
(List or Sequence)

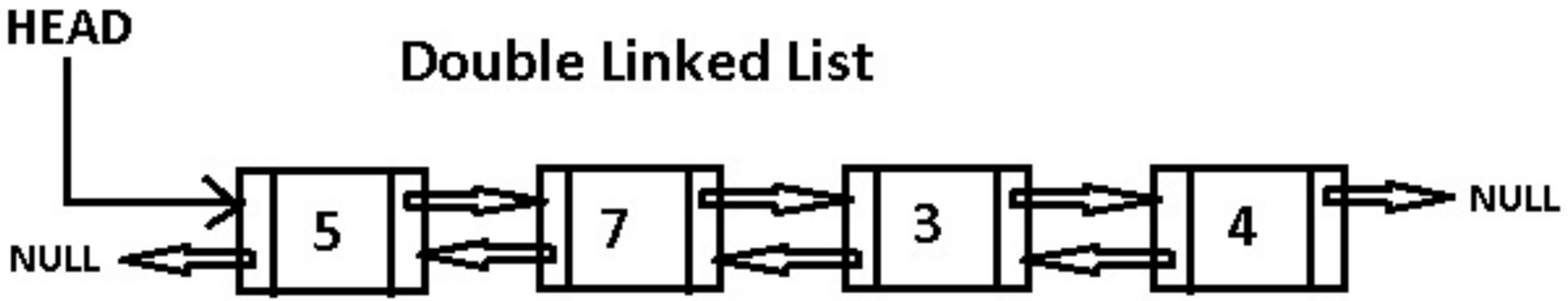
2017.07.03
최동훈

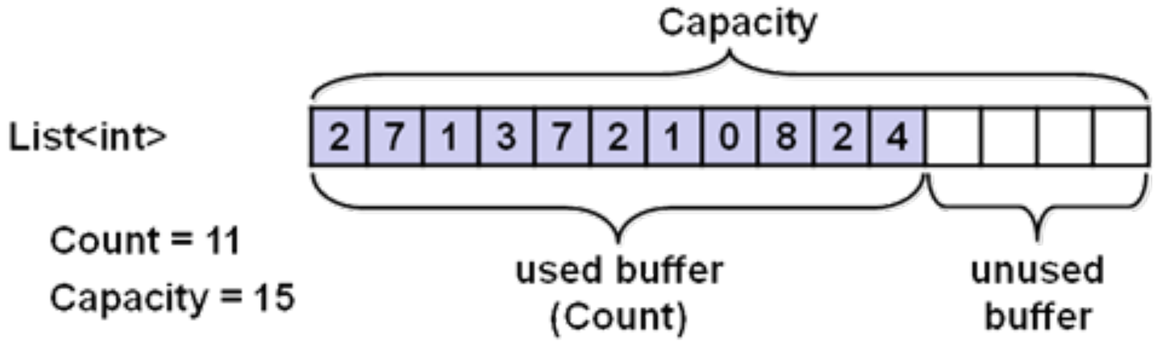
리스트(List or Sequence)

- 원소의 순서(order)가 존재
- 같은 값을 갖는 원소 허용 (중복 허용)
- 유한 수열(sequence)의 컴퓨터적인 표현
- 자바의 List Interface method 참조
  (https://docs.oracle.com/javase/8/docs/api/java/util/List.html)

2017.07.03
최동훈







2017.07.03
최동훈

Linked List vs Array List

2017.07.03
최동훈

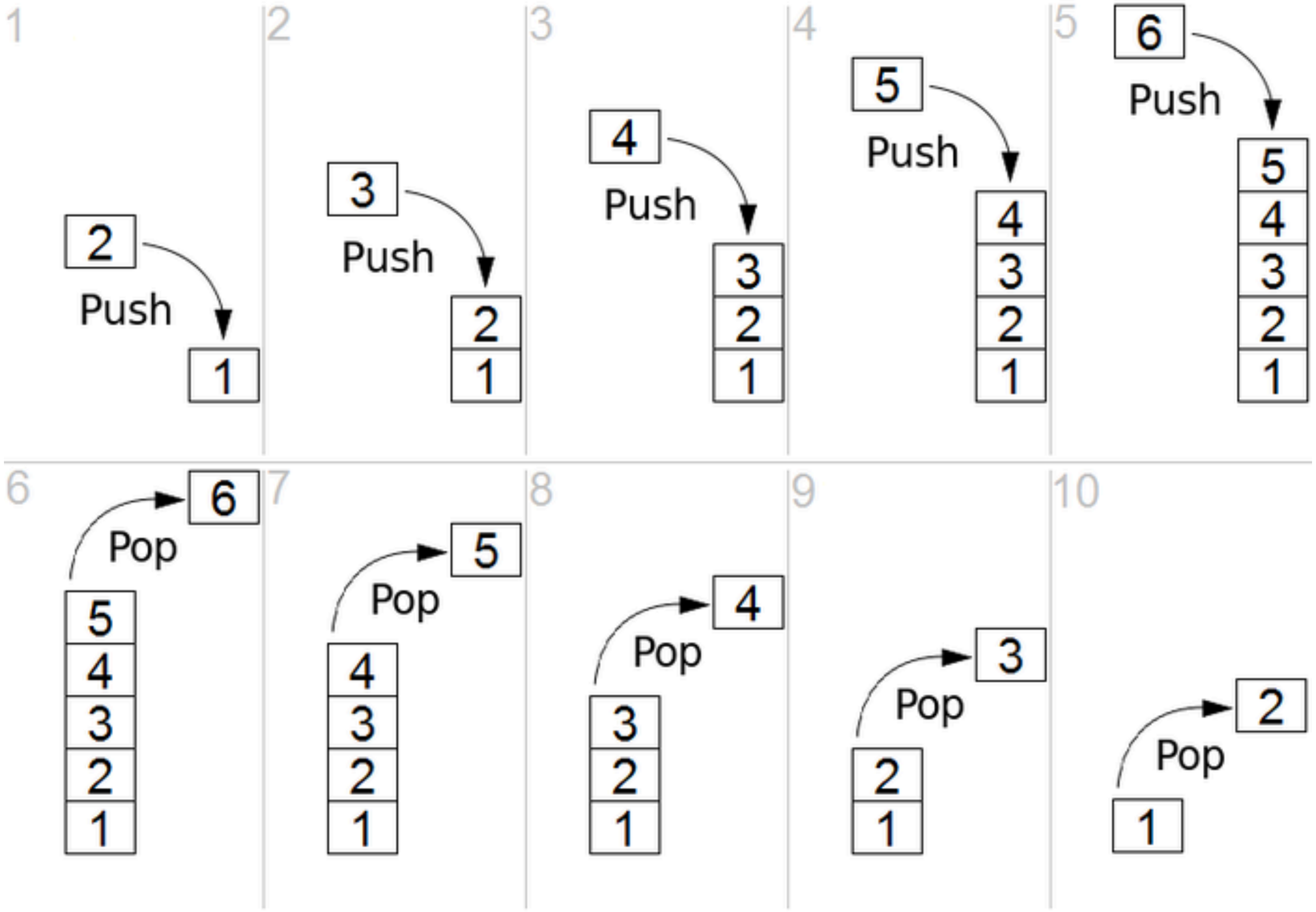
스택(stack)

2017.07.03
최동훈

스택(stack)

- 후입선출 자료구조 - LIFO(Last In First Out)
- 주요 메서드
  - PUSH : 데이터를 추가
  - POP : 데이터를 제거
- javascript에서는 배열의 기본 메서드로 push와 pop이 구현되어 있음
- 새로운 context를 만들고 다시 예전 context를 되돌려야 할때 쓰는 자료 구조
- Array와 LinkedList로 구현 가능

2017.07.03
최동훈

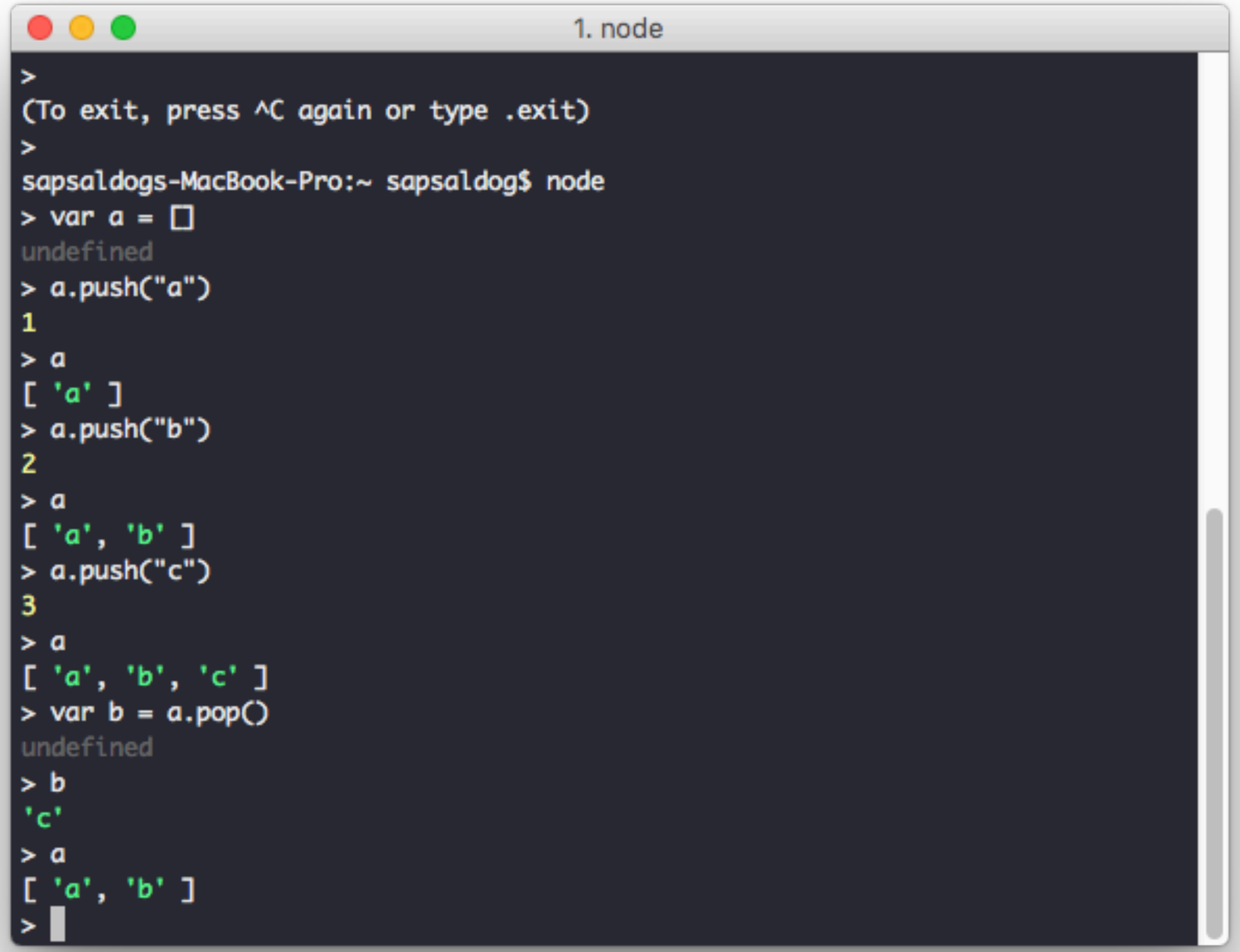


2017.07.03
최동훈

스택(stack)

- 스택오버플로우 (stack overflow)
- 스택언더플로우 (stack underflow)
- 스택 프레임 (stack frame)

2017.07.03
최동훈



2017.07.03
최동훈

Fast campus

당신의 커리어 전환점 패스트캠퍼스

큐(queue)

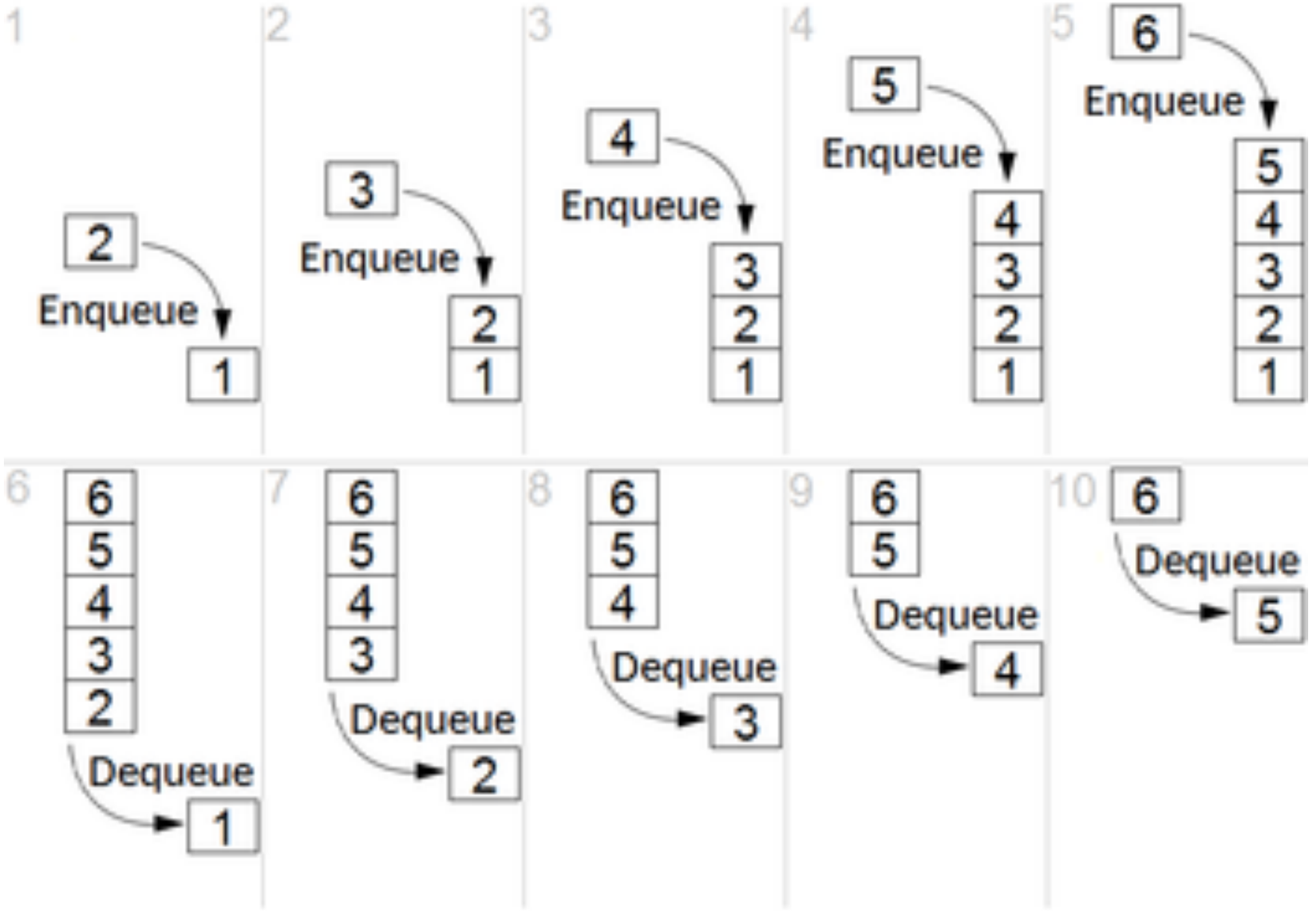
FRONTEND PROGRAMMING SCHOOL

2017.07.03
최동훈

큐(queue)

- 선입선출 자료구조 - FIFO(First In First Out)
- 주요 메서드
  - enqueue : 큐에 데이터를 넣음
  - dequeue : 큐에 데이터를 뺌
- 대기표 시스템 같은곳에 유용

2017.07.03
최동훈



2017.07.03
최동훈

큐(queue)

- queue overflow / underflow
- 환형큐 (circular queue) - 구현
- 우선순위큐 (priority queue)

2017.07.03
최동훈

2017.07.03
최동훈

Fast campus

당신의 커리어 전환점 패스트캠퍼스

트리(tree)

FRONTEND PROGRAMMING SCHOOL

2017.07.03
최동훈

트리(tree)

- 계층 구조를 표현하는 자료구조
- 루트와 서브트리로 되어 있음

2017.07.03
최동훈

트리(tree) 용어

- 루트(root)
- 부모(parent) - 자식(child)
- 형제(siblings)
- 잎(leaf) 노드 - 자식이 없는 노드
- 깊이(depth)
- 트리의 높이(height of tree)
- 완전 이진 트리 (complete binary tree)

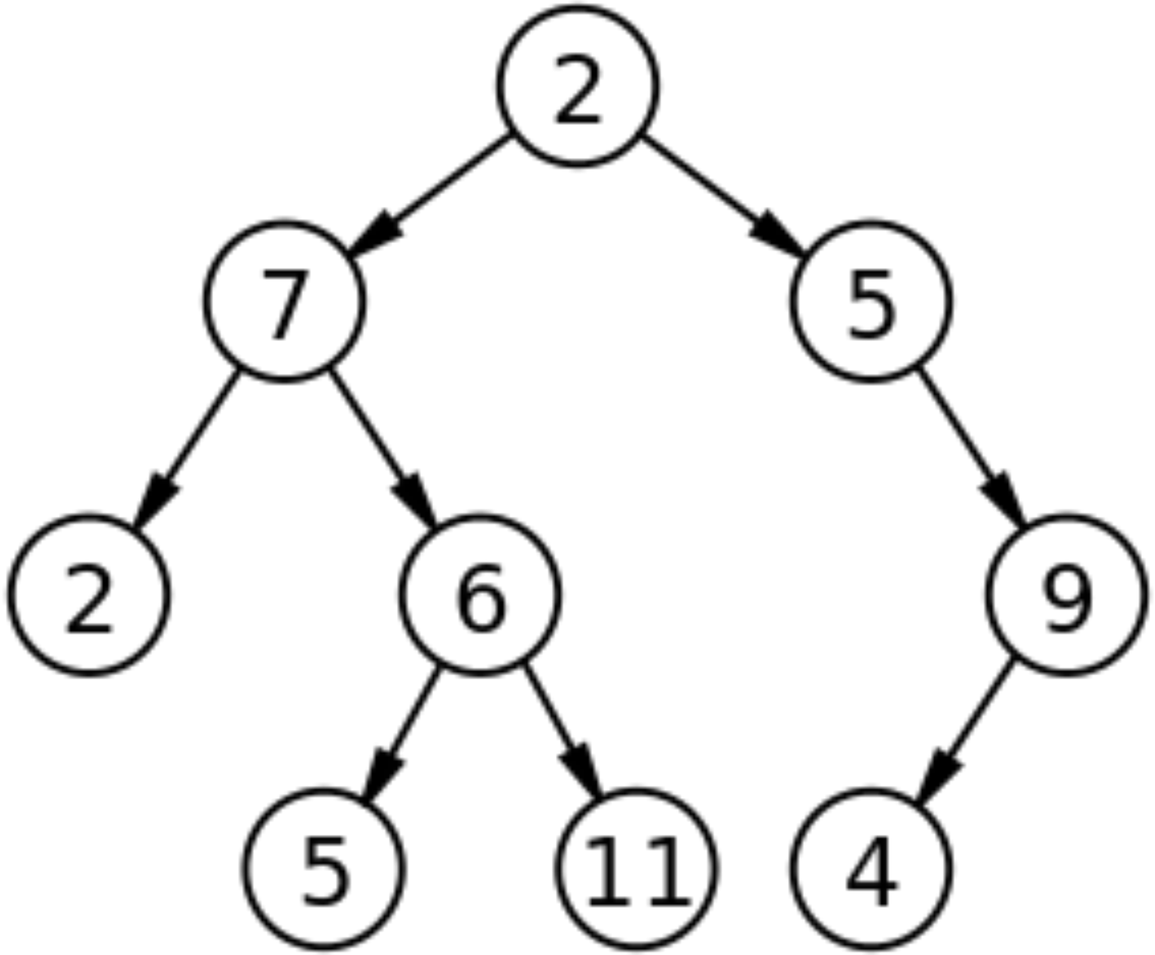
2017.07.03
최동훈

트리 종류

- 이진 트리
  - 이진 탐색 트리
    - AVL-tree
    - Red-black tree
  - 쓰레드 이진 트리
  - 힙(heap)
- B트리
- 트라이

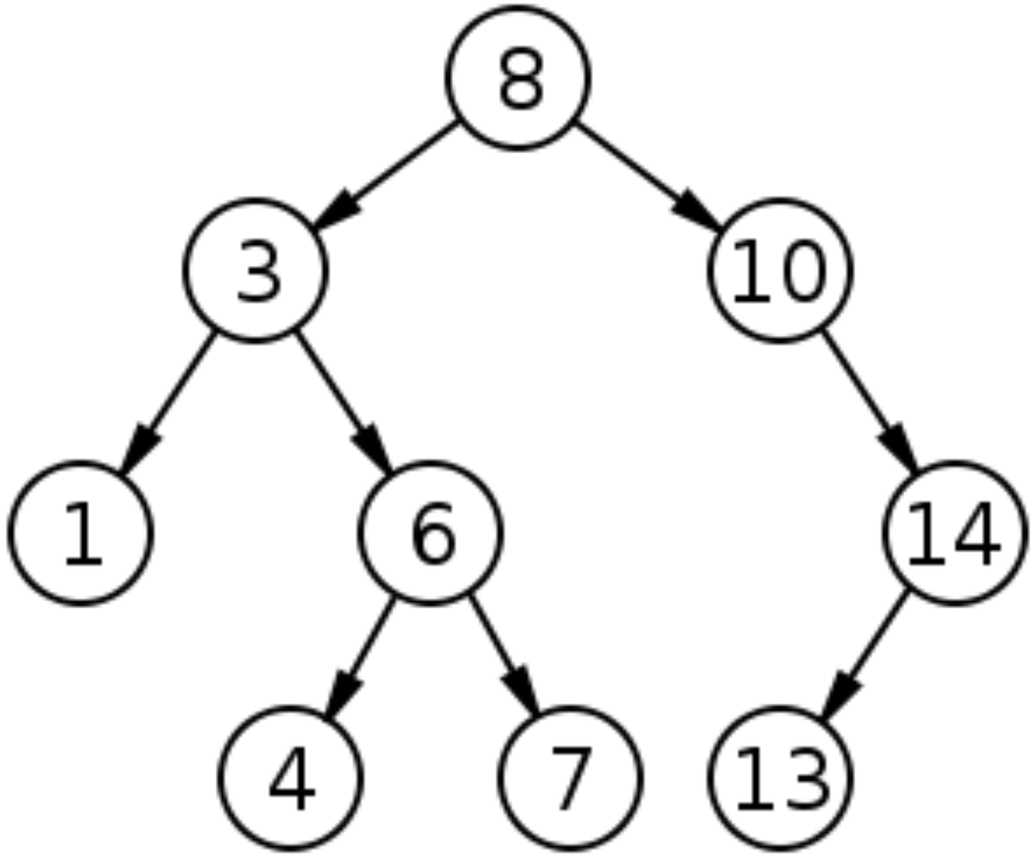
2017.07.03
최동훈

이진트리(binary tree)



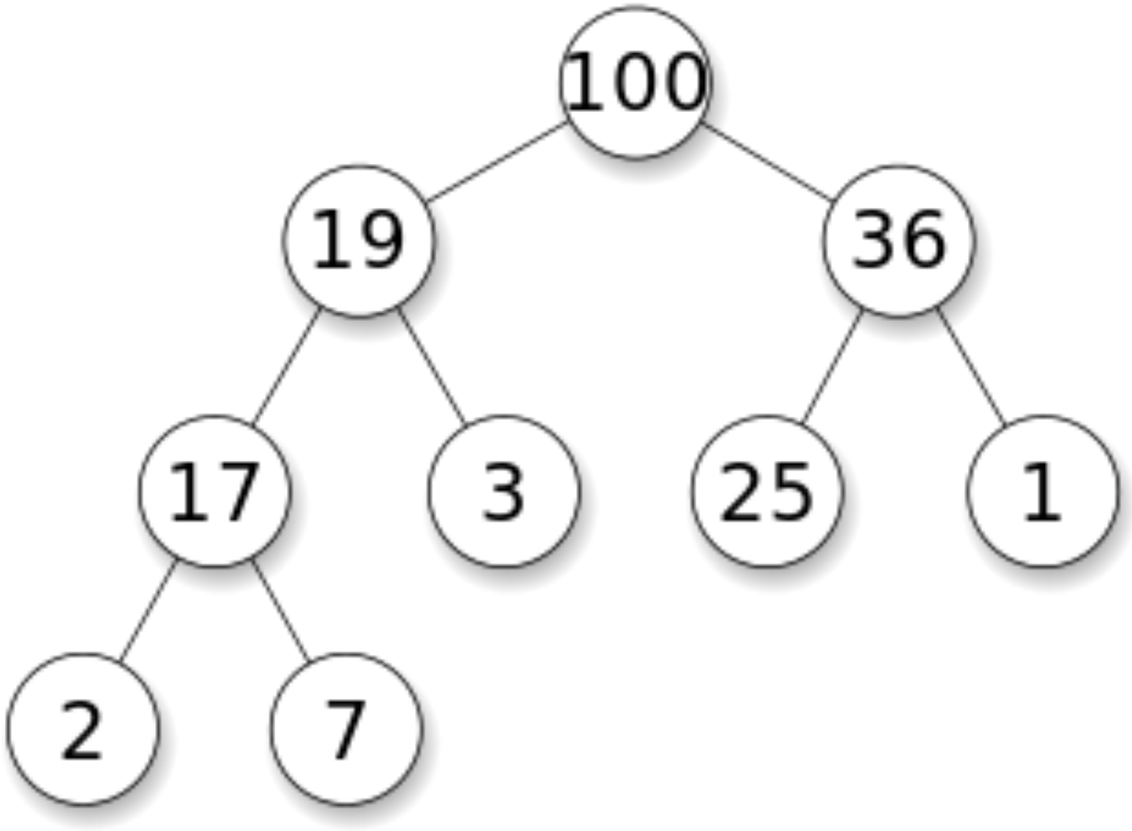
2017.07.03
최동훈

이진 탐색 트리
(binary search tree)



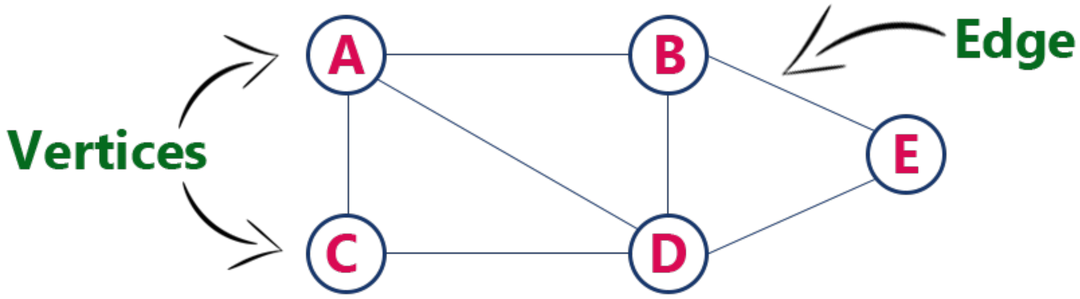
2017.07.03
최동훈

힙 트리(heap tree)



2017.07.03
최동훈

그래프(graph)



2017.07.03
최동훈

그래프 용어

- 방향 그래프(directed graph), 무방향 그래프(undirected graph)
- 정점(vertex)
- 변(edge)
- 차수 (degree)

2017.07.03
최동훈

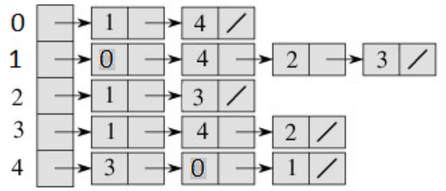
그래프 연산 종류

- G는 Graph 자료 구조를 의미
- adjacent(G, x, y) : x에서 y로 가는 경로가 존재 하는지 질의
- neighbors(G, x) : x와 이웃관계인 정점들을 나열
- add_vertex(G, x) : 정점 x를 추가
- remove_vertex(G, x) : 정점 x를 제거
- add_edge(G, x, y) : 변 xy를 추가
- remove_edge(G, x, y) : 변 xy를 제거
- get_vertex_value(G, x) : x와 연관된 값을 반환
- set_vertex_value(G, x, v):

2017.07.03
최동훈

그래프 표현 방법

- 인접 리스트(adjacency list)
- 인접 행렬(adjacency matrix)



	0	1	2	3	4
0	0	1	0	0	1
1	1	0	1	1	1
2	0	1	0	1	0
3	0	1	1	0	1
4	1	1	0	1	0

2017.07.03
최동훈

검색(search)

2017.07.03
최동훈

깊이 우선 탐색 (depth-first search)

- Stack 사용
- 주로 재귀호출을 이용
- BFS보다 코드가 상대적으로 단순

2017.07.03
최동훈

너비 우선 탐색 (breadth-first search)

- Queue 사용
- 주로 while 루프를 사용 -> stack overflow 위험성 없음
- 상대적으로 DFS보다 코드가 복잡함

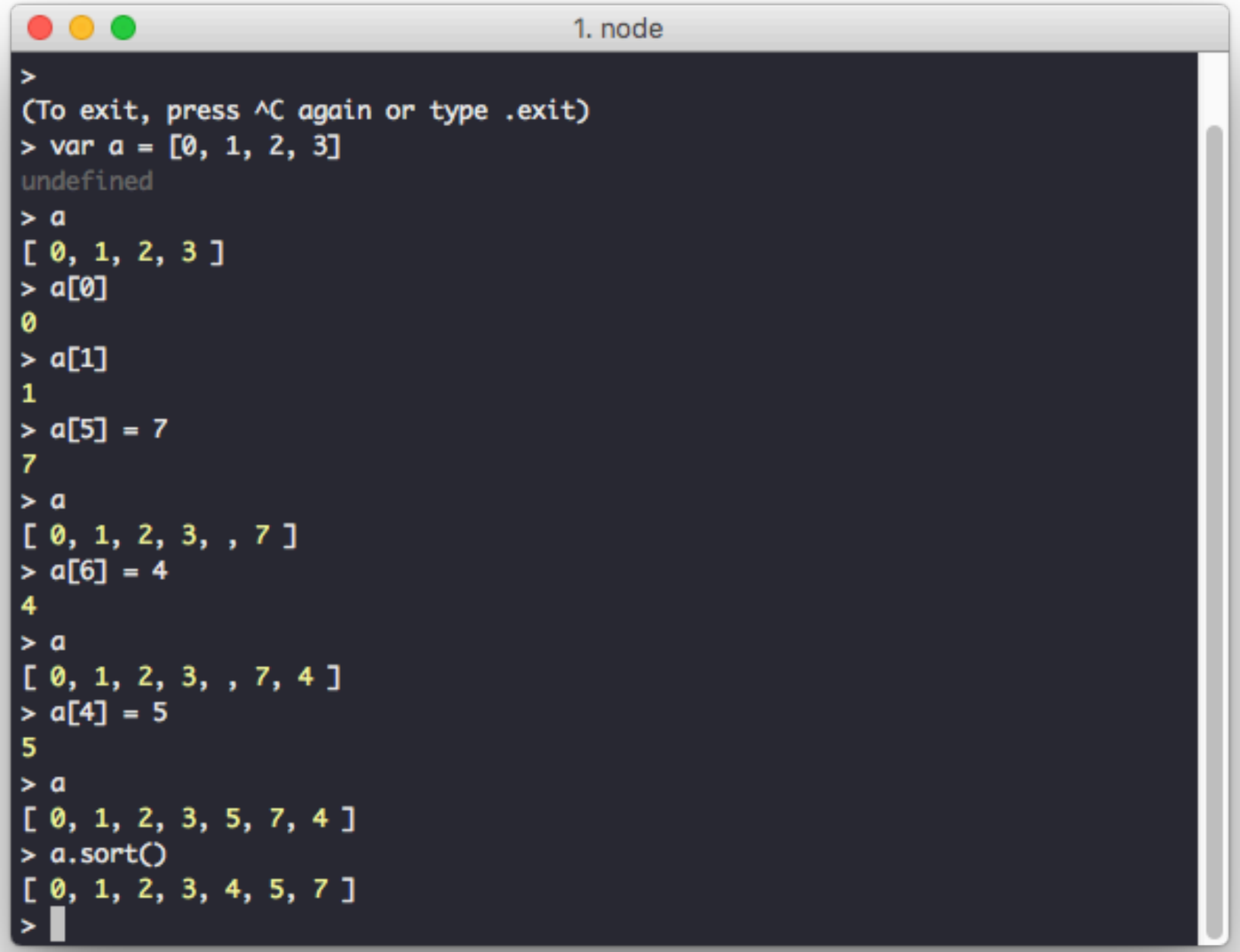
2017.07.03
최동훈

	너비 우선 탐색 (BFS)	깊이 우선 탐색 (DFS)	다익스트라 (Dijkstra)	플로이드 와샬 (Floyd Washall)
탐색 방식	자신과 연결된 주변 정점 부터 탐색해 나감	자신과 연결된 정점을 선택해, 그 정점에서 연결된 모든 정점을 파고들어가며 끝날 때 까지 들어가 탐색	간선에 **음의 가중치가 없는** 그래프에서, 어느 한 정점에서 각 정점까지 **최소 가중치를 갖는** 루트 탐색 (BFS + 최단경로 찾기)	간선에 **양,음의 가중치**가 있는 그래프에서, **모든 정점에 대해 각 정점까지 최소 가중치를 갖는 루트 탐색** (모든 정점+최단경로 찾기)
특 징	깊이가 깊은 그래프에 대해 높은 성능	넓이가 넓은 그래프에 대해 높은 성능	비교적 빠르게 최단경로 탐색 가능	모든 정점에서의 최단 경로와, 양, 음수 값의 모든 가중치에 대해 탐색 가능 코드가 단순함
제약 조건	너비가 넓은 그래프에 대해 낮은 성능	깊이가 깊은 그래프에 대해 낮은 성능	한 정점에 대해서만 가능하며, 음수 가중치는 불가	속도가 느림
이용되는 자료구조	큐(Queue)	스택(Stack)	BFS 응용(+ 최소 힙)	3중 for문
시간 복잡도	인접행렬: $O(V^2)$ 인접리스트: O(V+E) (V: 정점의 개수, E: 간선의 개수)		일반: $O(V^2)$ 최소 힙 사용: $O(E+V\log V)$	$O(V^3)$

2017.07.03
최동훈

정렬(sort)

2017.07.03
최동훈



2017.07.03
최동훈

Fast campus

당신의 커리어 전환점 패스트캠퍼스

알고리즘

FRONTEND PROGRAMMING SCHOOL

2017.07.03
최동훈

당신의 커리어 전환점 패스트캠퍼스

THANK YOU :-)

2017.07.03
최동훈