SWIFT
Data Structure - Tree

Bill Kim(김정훈) | ibillkim@gmail.com

목차

Tree

트리(Tree)는 일종의 계층적 자료구조(Hierarchical Data Structure)로서 부모 노드 하단에 자식 노드들을 연결하여 구성되는 형태의 자료구조입니다.

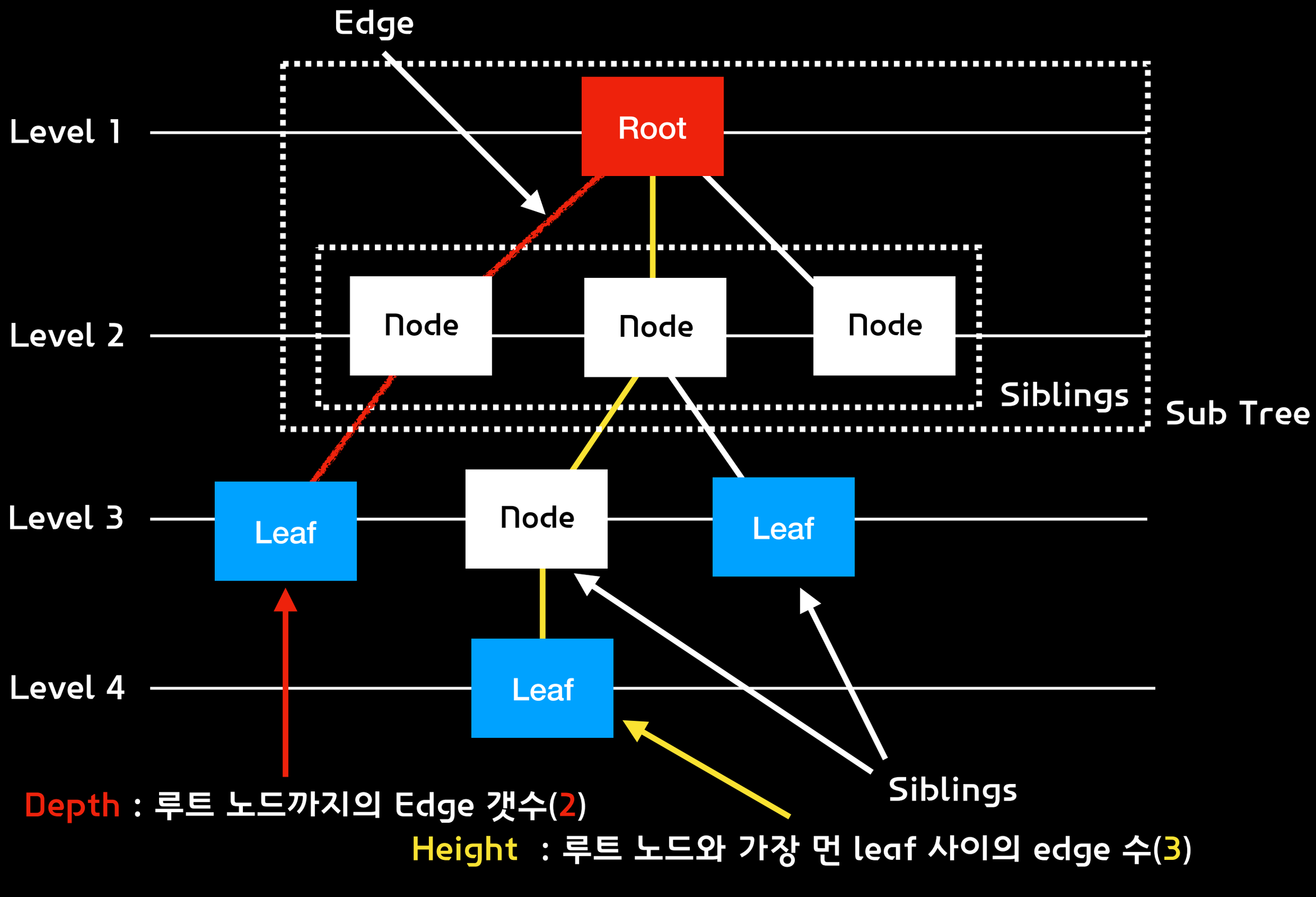
최상위(루트) 노드를 기반으로 마치 나무와 같이 아래로 자식 노드들을 가지면서 링크로 연결된 형태가 되어 트리라는 이름으로 되었습니다.

Tree

트리(Tree)에서 사용하는 기본적인 용어들은 아래와 같습니다.

Root : 트리의 맨 위에 있는 노드, 유일한 부모가 없는 노드
Node : 값을 가지고 있는 기본 단위 객체
Edge : 상위 노드와 하위 노드 간의 연결하는 선(link)
Parent : 해당 노드 바로 위에 있는 부모 노드
Child : 해당 노드 바로 아래에 있는 자식 노드
Sibling : 같은 부모를 같는 노드들
Leaf : 더 이상 자식 노드가 없는 최하위 노드
Sub Tree : 주어진 노드의 모든 하위 목록
Depth : 노드에서 루트까지의 edge 수
Height : 루트 노드와 가장 먼 leaf 사이의 edge 수
Level : depth + 1
Tree traversal : 트리의 모든 노드를 한번씩 방문하는 과정

Tree


Edge
Level 1
Root
Level 2
Node
Node
Node
Siblings
Sub Tree
Level 3
Leaf
Node
Leaf
Level 4
Leaf
Siblings
Depth : 루트 노드까지의 Edge 갯수(2)
Height : 루트 노드와 가장 먼 leaf 사이의 edge 수(3)

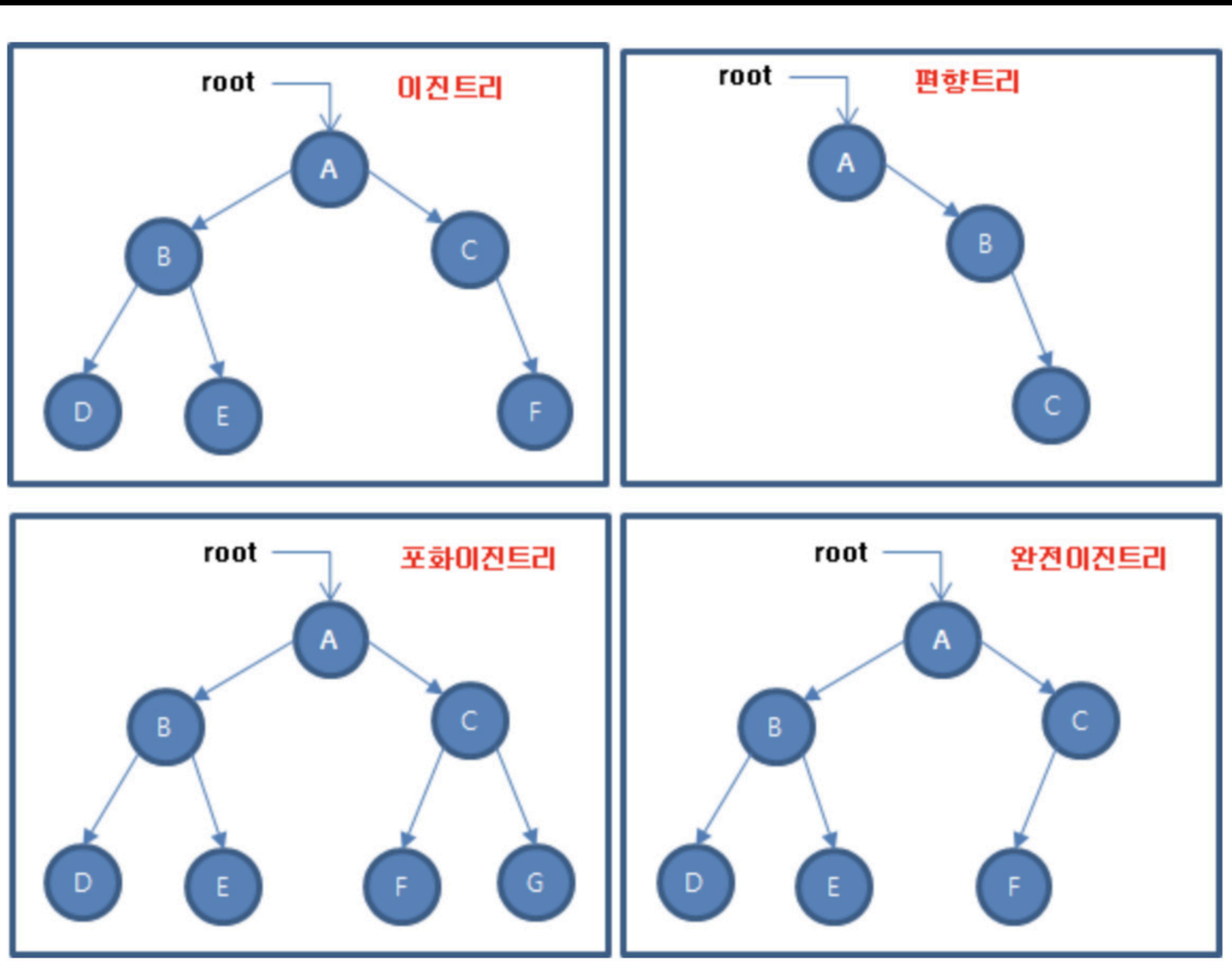
Concept

트리(Tree)는 Binary Tree(이진 트리)와 Non-Binary Tree로 크게 구분할 수 있습니다.

이진 트리란 자식 노드가 최대 2개만 있는 트리로서 가장 많이 사용되는 트리의 형태로서 모양에 따라서 아래와 같이 분류될 수 있습니다.

1. 전(정) 이진 트리(Full Binary Tree) : 모든 레벨에서 노드들이 모두 채워져 있는 트리
2. 완전 이진 트리(Complete Binary Tree) : 마지막 레벨을 제외하고 노드가 모두 채워져 있는 트리
3. 포화 이진 트리(Perfect Binary Tree) : 모드 레벨의 노드가 모두 차있는 트리
4. 편향 트리(Skewed Binary Tree) : 트리의 노드가 왼쪽이나 오른쪽으로 한쪽으로만 채워진 트리

Concept



이진 트리 종류

Concept

대표적인 많이 사용하는 트리(Tree)의 종류는 아래와 같은 것들이 있습니다.

1. Binary Tree : 가장 기본적인 이진 트리, 자식 노드가 최대 2개
2. Binary Search Tree : 이진 탐색(binary search)과 연결리스트(linked list)를 결합한 자료구조의 일종으로서 효율적인 탐색 및 자료 빈번한 입력과 삭제가 가능하게 해주는 트리
3. Balanced Binary Tree : 좌우 균형이 맞는 이진트리, 좌우 모양이 같음
4. B-tree : Balanced Binary Tree와 비슷하지만 노드당 2개의 자식을 가질 수 있는 트리
5. Splay tree : Binary Search Tree의 유형으로서 최근에 접근한 노드가 트리의 위쪽으로 이동하는 트리
6. Red-black Tree : 자가균형이진탐색 트리(self-balancing binary search tree)로써 각 노드에 대해서 색깔을 지정하여 트리를 구성하여 빠른 검색을 도와주는 트리

Features

Tree의 특징을 살펴보면 아래와 같습니다.

- 한 세트의 노드들로 구성
- 루트 노드를 기준으로 하위에 자식 노드들을 연결하여 구성
- 계층 구조 형태를 가지는 자료구조
- 한 노드는 두 번 이상 참조 불가능
- 루트를 가르키는 노드는 없음
- 트리는 Cycle이 없는 형태의 그래프
- 파일 시스템, 검색 엔진, 자료 저장, 컴파일러 등에서 활용

Implementation

Swift를 활용하여 가장 기본적인 Tree를 구현해보겠습니다.
본 문서에서는 단순한 형태의 Non-Binary Tree를 구현해볼 예정입니다.

우선 필요한 메소드는 아래와 같습니다.

- init : 트리를 초기화하는 함수
- addChild : 자식 트리를 추가
- search : 특정 트리를 검색

Implementation

```
public class Tree<T> {
    public var value: T

    public weak var parent: Tree?
    public var children = [Tree<T>]()

    public init(_ value: T) {
        self.value = value
    }

    public func addChild(_ node: Tree<T>) {
        children.append(node)
        node.parent = self
    }
}
```

Implementation

```
extension Tree where T: Equatable {
    public func search(_ value: T) -> Tree? {
        if value == self.value {
            return self
        }
        for child in children {
            if let found = child.search(value) {
                return found
            }
            return nil
        }
    }
```

Implementation

```
extension Tree : CustomStringConvertible {
    public var description: String {
        var s = "\(value)"
        if !children.isEmpty {
            s += " {" + children.map { $0.description }.joined(separator: ", ") +
"}"
        }
        return s
    }
}
```

Implementation

```
let tree = Tree<String>("car")

let sedan = Tree<String>("sedan")
let suv = Tree<String>("suv")

sedan.addChild(Tree<String>("sonata"))
sedan.addChild(Tree<String>("g70"))

suv.addChild(Tree<String>("sorento"))
suv.addChild(Tree<String>("gv80"))

tree.addChild(sedan)
tree.addChild(suv)

print(tree.description) // car {sedan {sonata, g70}, suv {sorento, gv80}}
```

References

[1] 스위프트: 트리: Tree: #자료구조: #깊이우선탐색: #레벨정렬탐색: #검색알고리즘: Swift4 : https://the-brain-of-sic2.tistory.com/24

[2] [Swift 자료구조 ch12] Tree 의 기본정의 : https://kor45cw.tistory.com/257

[3] [Swift 자료구조 ch13] Tree 의 구현 : https://kor45cw.tistory.com/258

[4] Swift, Data Structure, Binary Search Trees : https://devmjun.github.io/archive/BinarySearchTree

[5] Swift, Data Structure, trees : https://devmjun.github.io/archive/tree

References

[6] 트리의 종류와 이해 : http://www.secmem.org/blog/2019/05/09/트리의-종류와-이해/

[7] [Swift] 이진 탐색 트리 : https://milyo-codingstories.tistory.com/60

[8] 개념 정리 - (4) 자료 구조 편 : https://brunch.co.kr/@toughrogrammer/12

[9] 자료구조 트리(Tree) : https://yeolco.tistory.com/52

[10] 자료구조 - 트리 (Tree) : https://www.slideshare.net/choong83/tree-61521983

Thank you!