JavaScript 핵심 자료구조 알아보기

1) 자료 구조(Data Structure) 개요

자료구조 개요 | 다양한 알고리즘의 기본이 되는 자료구조 이해하기

강사 나동빈

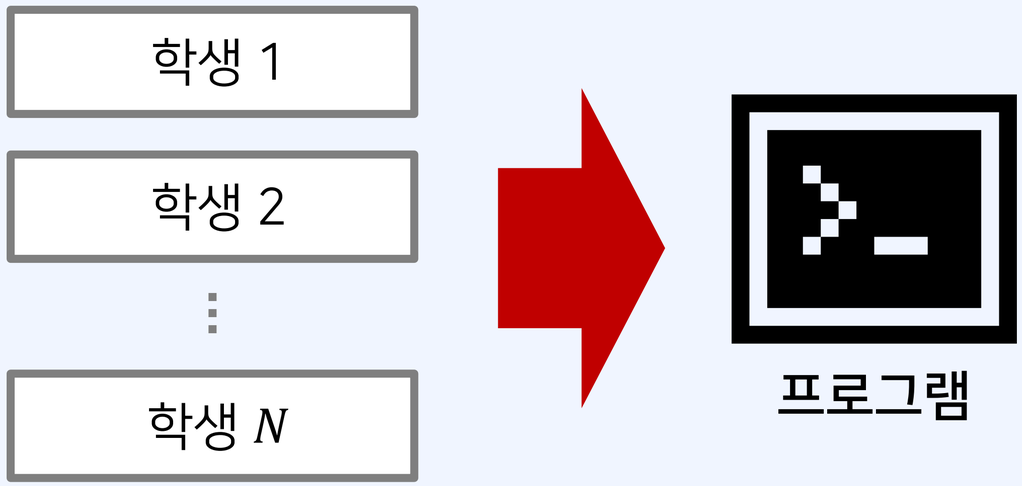
JavaScript
핵심 자료구조 알아보기

1) 자료 구조(Data Structure) 개요

자료구조(Data Structure)란?

JavaScript
자료구조
자료구조 개요

- 자료구조는 다수의 **자료(data)**를 담기 위한 구조다.
- 데이터의 수가 많아질수록 효율적인 자료구조가 필요하다.
- **예시)** 학생 수가 1,000,000명 이상인 학생 관리 프로그램

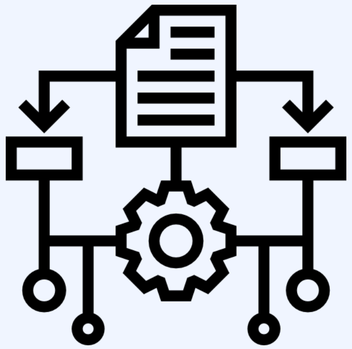


자료구조의 개요

JavaScript
자료구조
자료구조 개요

- 자료구조의 필요성에 대해서 이해할 필요가 있다.
- **성능 비교**: 자료구조/알고리즘의 **성능 측정 방법**에 대해 이해할 필요가 있다.

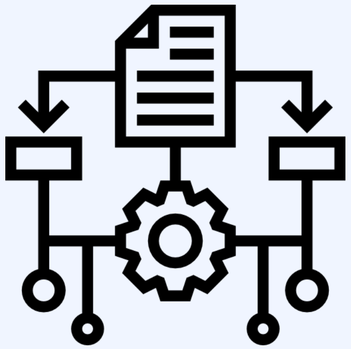
"나는 삽입과 추출이 모두 적당히 빨라!"
삽입: $O(\log N)$ / **추출**: $O(\log N)$



자료구조 1

VS.

"나는 삽입은 느리지만, 추출은 빨라!"
삽입: $O(N)$ / **추출**: $O(1)$



자료구조 2

자료구조의 필요성

JavaScript
자료구조
자료구조 개요

- 데이터를 효과적으로 저장하고, 처리하는 방법에 대해 바르게 이해할 필요가 있다.
- 자료구조를 제대로 이해하지 못하면 불필요하게 메모리와 계산을 낭비할 여지가 있다.

자료구조의 필요성

JavaScript
자료구조
자료구조 개요

- C언어를 기준으로 정수(int) 형식의 데이터가 100만 개가량이 존재한다고 가정하자.
- 해당 프로그램을 이용하면, 내부적으로 하루에 데이터 조회가 1억 번 이상 발생한다.
- 이때 원하는 데이터를 가장 빠르게 찾도록 해주는 자료구조는 무엇일까?

→ 트리(tree)와 같은 자료구조를 활용할 수 있다.

자료구조의 종류

1. 선형 구조(linear data structure)
- 배열(array)
- 연결 리스트(linked list)
- 스택(stack)
- 큐(queue)

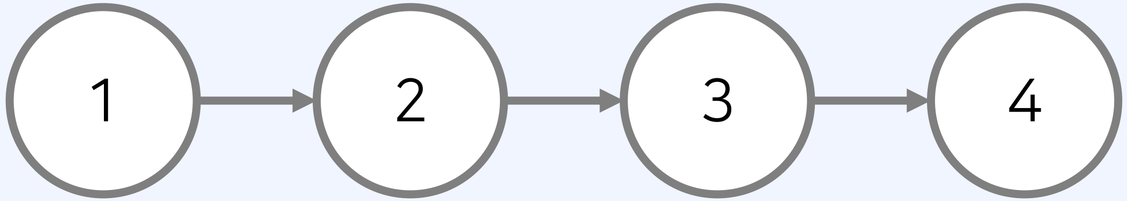
2. 비선형 구조(non-linear data structure)
- 트리(tree)
- 그래프(graph)

선형 자료구조(Linear Data Structure)

JavaScript
자료구조
자료구조 개요

- 선형 자료구조는 하나의 데이터 뒤에 다른 데이터가 하나 존재하는 자료구조다.
- 데이터가 일렬로 연속적으로(순차적으로) 연결되어 있다.

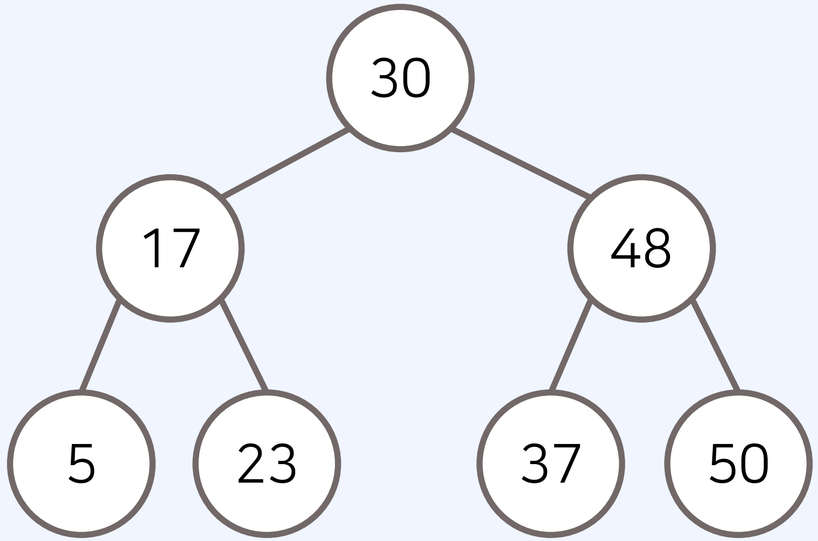
예시) 배열, 연결 리스트(linked list), 스택(stack), 큐, 덱(deque)



비선형 자료구조(Non-linear Data Structure)

- 비선형 자료구조는 하나의 데이터 뒤에 다른 데이터가 여러 개 올 수 있는 자료구조다.
- 데이터가 일직선상으로 연결되어 있지 않아도 된다.

예시) 트리(tree), 그래프(graph)



자료구조와 알고리즘

JavaScript
자료구조
자료구조 개요

1. 효율적인 자료구조 설계를 위해 알고리즘 지식이 필요하다.
2. 효율적인 알고리즘을 작성하기 위해서 문제 상황에 맞는 적절한 자료구조가 사용되어야 한다.
3. 프로그램을 작성할 때 자료구조와 알고리즘 모두 고려해야 한다.

프로그램의 성능 측정 방법

JavaScript
자료구조
자료구조 개요

- **시간 복잡도(time complexity)**: 알고리즘에 사용되는 연산 횟수를 측정한다.
- **공간 복잡도(space complexity)**: 알고리즘에 사용되는 메모리의 양을 측정한다.
- 공간을 많이 사용하는 대신 시간을 단축하는 방법이 흔히 사용된다.

프로그램의 성능 측정 방법: Big-O 표기법

JavaScript
자료구조
자료구조 개요

- 복잡도를 표현할 때는 **Big-O 표기법**을 사용한다.

① 특정한 알고리즘이 얼마나 효율적인지 수치적으로 표현할 수 있다.

② 가장 빠르게 증가하는 항만을 고려하는 표기법이다.

- 다음 알고리즘은 $O(n)$의 시간 복잡도를 가진다.

```
let n = 10;
let summary = 0;

for (let i = 0; i < n; i++) {
  summary += i;
}
console.log(summary);
```

[실행 결과]

```
45
```

프로그램의 성능 측정 방법: Big-O 표기법

- 다음 알고리즘은 $O(n^2)$의 시간 복잡도를 가진다.

```
let n = 9;

for (let i = 1; i <= n; i++) {
  for (let j = 1; j <= n; j++) {
    console.log(`${i} X ${j} = ${i * j}`);
  }
}
```

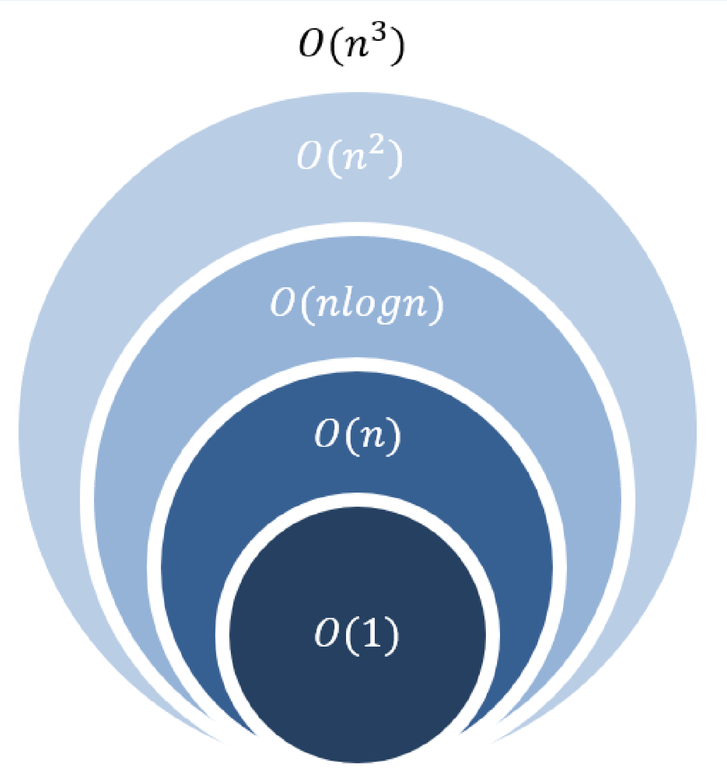
[실행 결과]

```
1 X 1 = 1
1 X 2 = 2
…
9 X 8 = 72
9 X 9 = 81
```

프로그램의 성능 측정 방법

JavaScript
자료구조
자료구조 개요

- 일반적으로 연산 횟수가 **10억**을 넘어가면 1초 이상의 시간이 소요된다.



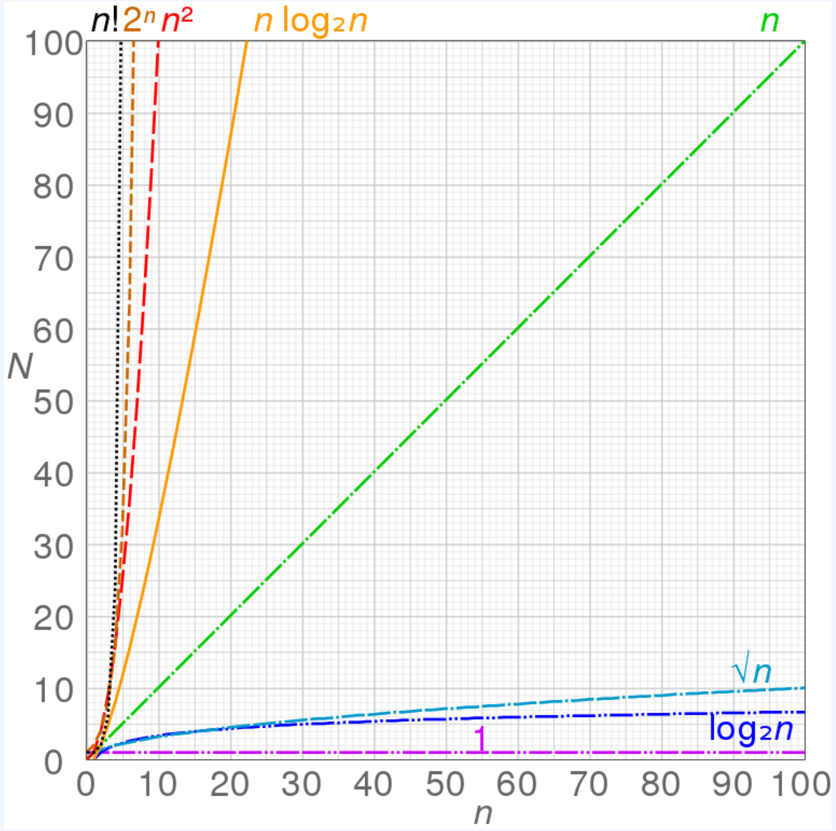
[예시] n이 1,000일 때를 고려해 보자.

- $O(n)$: 약 1,000번의 연산
- $O(nlogn)$: 약 10,000번의 연산
- $O(n^2)$: 약 1,000,000번의 연산
- $O(n^3)$: 약 1,000,000,000번의 연산

프로그램의 성능 측정 방법

JavaScript 자료구조
자료구조 개요

- 각 시간 복잡도를 비교해보자.



* https://en.wikipedia.org/wiki/Big_O_notation

프로그램의 성능 측정 방법

JavaScript
자료구조
자료구조 개요

- **Big-O 표기법**으로 시간 복잡도를 표기할 때는 가장 큰 항만을 표시한다.
- 가장 큰 항에 붙어 있는 **계수는 제거**한다.

$$O(3n^2 + n) = O(n^2)$$

- 현실 세계에서는 **동작 시간이 1초 이내**인 알고리즘을 설계할 필요가 있다.

프로그램의 성능 측정 방법

JavaScript
자료구조
자료구조 개요

- 코딩 테스트에서 메모리의 크기를 나타낼 때는 **일반적으로 MB 단위로 표기**한다.

```
int a[1000]: 4KB
int a[1000000]: 4MB
int a[2000][2000]: 16MB
```

자료구조를 적절히 활용하기

JavaScript
자료구조
자료구조 개요

- 자료구조의 종류로는 스택, 큐, 트리 등이 있다.
- 프로그램을 작성할 때는 자료구조를 적절히 활용하여 시간 복잡도를 최소화하여야 한다.