Big-O

작성 일시	@June 6, 2023 8:57 AM
자료	https://www.youtube.com/watch?v=BEVnxbxBqi8 https://window6kim.tistory.com/30
# 주차	2

알고리즘 스피드를 표현법을 배워볼겁니다.

알고리즘의 스피드는 “빠르다”, “느리다” 는 시간으로 표현하지 않습니다.

왜냐하면 같은 알고리즘이라도 컴퓨터의 성능(하드웨어)에 따라 속도는 다를 수 있기 때문이다.

따라서 알고리즘의 속도는 “완료 까지 걸리는 절차의 수”로 표현한다.

우리는 시간복잡도를 BigO를 사용해 설명한다. 이것을 이용하면 읽기 쉽고 빠르게 설명이 가능하다.

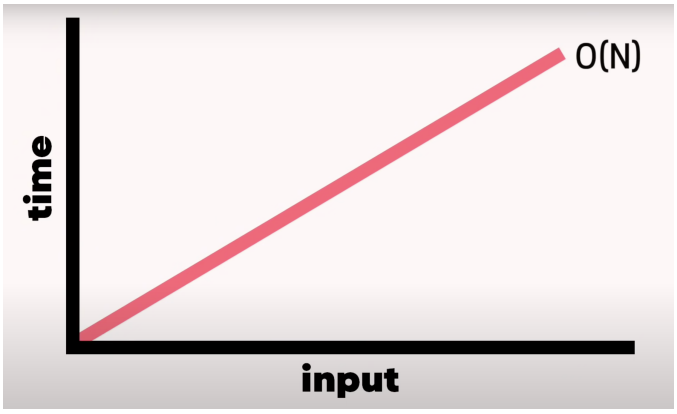
O(n)

입력 데이터의 크기에 비례해서 처리 시간이 걸리는 알고리즘(Linear Time)

예로 들어 linear search(선형 탐색)을 생각해보자

선형 탐색은 한 개씩 한 개씩 검색을 한다. 따라서 20개 데이터가 있다면 원하는 데이터를 찾기 위해 20개의 스텝이 필요하다.

→ input size = N → N steps → linear search 알고리즘의 시간 복잡도는 O(n)



O(1)

입력 데이터의 크기와 상관 없이 언제나 스텝이 정해진 알고리즘 (Constant Time)

```
const arr = [1,2,3,4,..., 100];
function printFirst(arr){
  console.log(arr[0]);
}
```

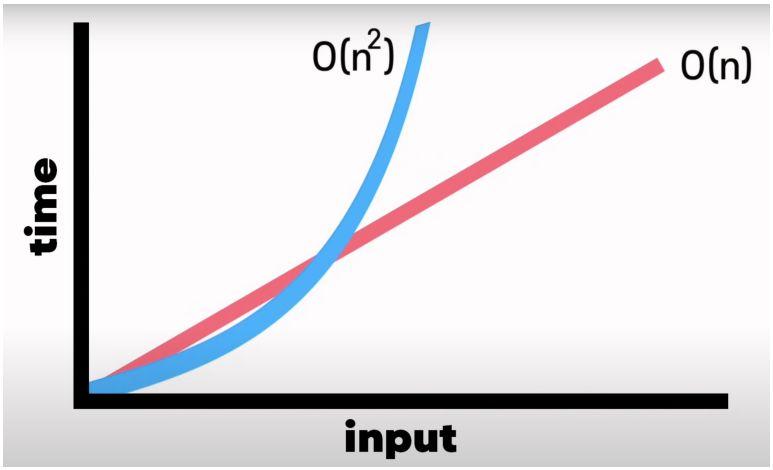
```
function printFirstTwice(arr){
  console.log(arr[0]);
  console.log(arr[0]);
}
```

만약 첫 번째 요소를 두번 프린트한다고 하면 이 작업을 하기 위해 2개의 스텝이 필요할 것이다. 그렇다면 시간 복잡도는 O(2)일까? 아니다. 여전히 O(1)이다.

BigO는 큰 원리에만 관심이 있다. 여기서 함수는 **인풋사이즈가 커져도 관계 없이 미리 정해진 숫자(constant)에 따라 작동한다는 것**이다.

O(n^2)

입력 데이터의 크기에 제곱에 비례하여 처리 시간이 걸리는 알고리즘(Quadratic Time)



해당 유형은 중첩 for 문내에서 입력 데이터를 처리하는 경우에 나타난다. 문제 해결을 위한 단계의 수는 입력 데이터 값 n의 제곱이 된다.

O(log n)

이진 검색 알고리즘을 설명할 때 쓰는 것(logarithmic complexity)

이진 탐색(Binary Search)

데이터가 **정렬되어 있는** 배열에서 특정한 값을 찾아내는 알고리즘이다.

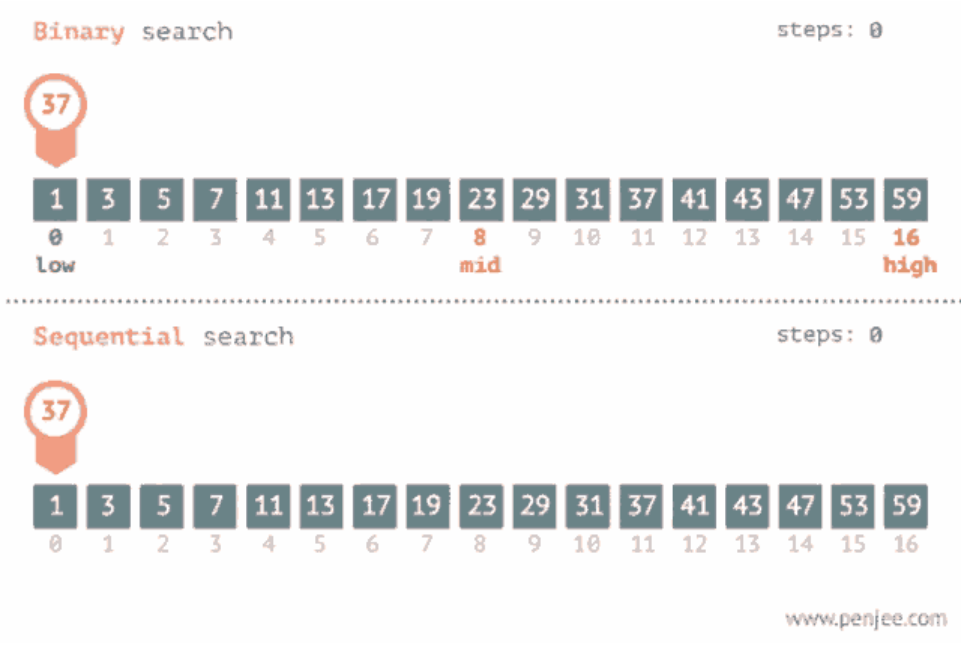
예로 들어 37의 값을 아래의 배열에서 찾는 다고 할 때,

먼저 배열을 절반으로 나눈 가운데 값(arr[mid])와 탐색하고자 하는 37값을 비교한다.

만약 37 < arr[mid] → mid 기준으로 배열의 좌측 절반을 다음 탐색함

만약 37> arr[mid] → mid 기준으로 배열우 우측 절반을 다름 탐색함

이 과정을 원하는 값을 찾을 때까지 반복함



만약 인풋의 크기가 32라고 하면 이진탐색으로 값을 찾는 절차를 거치면 log232 = 5 (2^5 = 32) 번의 절차가 나온다.

이는 인풋의 크기 32만큼 전부 탐색하는 것에 비해 시간 복잡도가 감소한다.