21주차 발표_민서연

복잡도

01 복잡도

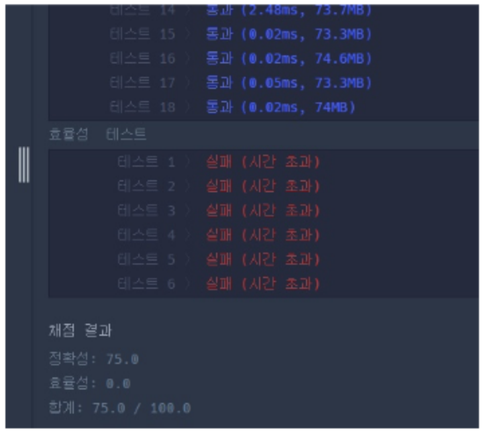
개념

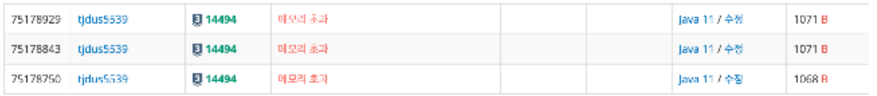
시간 복잡도(Time complexity)

프로그램이 실행되고 완료되는데 걸리는 시간

공간 복잡도(Space complexity)

프로그램이 실행되고 완료되는데 필요한 메모리





01 복잡도

개념

공간 복잡도(Space complexity)

프로그램을 실행하면 RAM 이라는 메모리에 올라가게 되는데
이때 메모리의 크기는 고정되어 있다.
그렇기 때문에! 메모리를 최대한 덜 차지하는 코드를 설계해야 한다.

01 복잡도

개념

공간 복잡도를 계산하는 방법

고정 공간 요구량
자료 구조가 사용하는 고정된 메모리 공간을 의미.
고정된 크기를 갖는 변수와 배열이 여기에 속한다.

가변 공간 요구량
필요에 따라 동적으로 할당, 해제 되는 메모리의 공간을 의미.
특정 인스턴스에 따라서 사이즈가 달라지는 변수들이 여기에 속한다.

01 복잡도

개념

공간 복잡도를 계산하는 방법

고정 공간 요구량
자료 구조가 사용하는 고정된 메모리 공간을 의미.
고정된 크기를 갖는 변수와 배열이 여기에 속한다.

가변 공간 요구량
필요에 따라 동적으로 할당, 해제 되는 메모리의 공간을 의미.
특정 인스턴스에 따라서 사이즈가 달라지는 변수들이 여기에 속한다.

고정 공간은 고정된 메모리 공간이므로
공간 복잡도는 가변 공간에 좌우된다.

01 복잡도

개념

시간 복잡도(Time complexity)

시간복잡도는 컴파일 타임과 러닝 타임으로 나뉘는데,
러닝 타임은 컴퓨터 사양이나 네트워크 환경 등에 따라서 달라질 수 있기 때문에
변수가 많아서 컴파일 시간의 복잡도만 계산한다.

01 복잡도

개념

시간 복잡도 표현 방법

최상의 경우: 오메가 표기법

평균의 경우: 세타 표기법

최악의 경우: 빅오 표기법

01 복잡도

개념

시간 복잡도 표현 방법

최상의 경우: 오메가 표기법

평균의 경우: 세타 표기법

최악의 경우: 빅오 표기법 시간 복잡도는 최악을 기준으로 빅오 표기법으로 판단해서 성능을 예측

01 복잡도

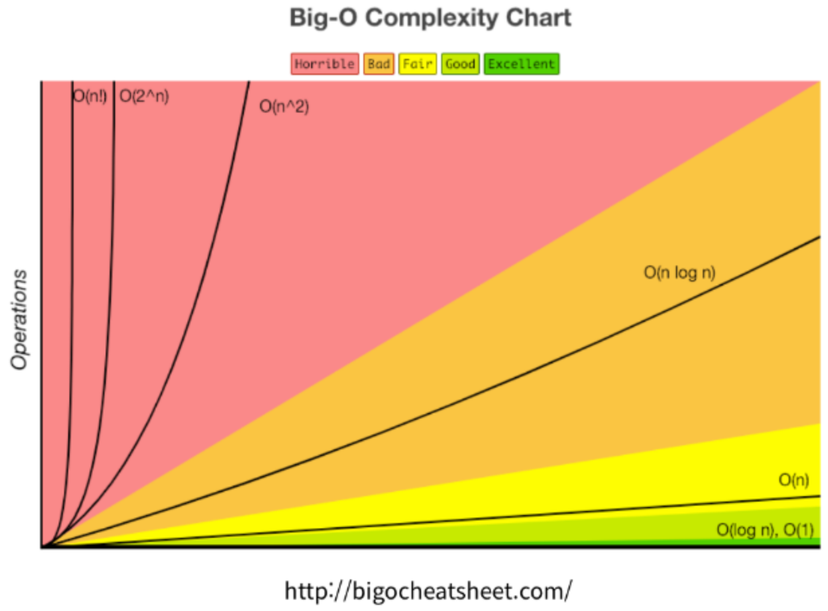
빅오 표기법

빅오 표기법은 불필요한 연산을 제거해서 알고리즘 분석을 쉽게 할 목적으로 사용된다.

빅오로 측정되는 복잡성에는 시간과 공간 복잡도가 있다.
시간 복잡도는 입력된 N의 크기에 따라 실행되는 조작의 수를 나타낸다.
공간 복잡도는 알고리즘이 실행될 때 사용하는 메모리의 양을 나타낸다.

01 복잡도

빅오 표기법



http://bigocheatsheet.com/

알고리즘을 수행하기 위해 프로세스가 수행 해야하는 연산을 수치화 한 것

실행시간이 아닌 연산 횟수로 판별하는 이유!
하드웨어 또는 사용하는 프로그래밍 언어에 따라 실행시간의 편차가
달라지기 때문에 명령어의 연산 횟수로 판별한다.

01 복잡도

시간 복잡도 단계

- O(1) : 입력자료의 수에 관계 없이 일정한 시간을 갖는다.

  -> 입력 자료와 연관된 연산 없이 탐색, 출력, 반환을 하는 경우
- O(logn): 입력자료의 수에 따라 시간이 흐를수록 시간이 조금씩 증가한다.

  -> 이진 검색, 퀵 정렬, 병합 정렬, 힙 정렬
- O(n): 입력 자료의 수에 따라 선형적인 실행 시간이 걸리는 경우 입력 자료마다 일정 시간이 할당 된다.

  -> 반복문 루프를 도는 경우
- O(nlogn): 큰 문제를 일정 크기 갖는 문제로 쪼개고 다시 모으는 경우
- O(n^2): 이중 루프 내에서 입력 자료를 처리할 때

  -> 피보나치 수열
- O(n^3): 삼중 루프 내에서 입력 자료를 처리 할 때

01 복잡도

시간 복잡도 단계

시간 복잡도가 빠른 순서

O(1) > O(logn) > O(n) > O(nlogn) > O(n^2) > O(n^3) > O(2^n) > O(n!)

자료구조에서의 시간복잡도

자료구조	평균			최악		
	Search	Insert	Delete	Search	Insert	Delete
Array	O(n)	N/A	N/A	O(n)	N/A	N/A
Sorted Array	O(log n)	O(n)	O(n)	O(log n)	O(n)	O(n)
Linked List	O(n)	O(1)	O(1)	O(n)	O(1)	O(1)
Stack	O(n)	O(1)	O(1)	O(n)	O(1)	O(1)
Hash table	O(1)	O(1)	O(1)	O(n)	O(n)	O(n)
Binary Search Tree	O(log n)	O(log n)	O(log n)	O(n)	O(n)	O(n)

02 출처

https://coduking.com/entry/자료구조란-feat-시간복잡도-공간복잡도
https://joyhong-91.tistory.com/12
https://junghyun100.github.io/Time-Complexity/
https://ontheway.tistory.com/47
https://insight-bgh.tistory.com/506