알고리즘 "공부" 필요할까 ?

> *수학 못해도 사는데 아무 지장 없더라. 돈계산만 잘하면 되지.*

우리가 자라면서 흔히 접하는 말이 있다. **"수학 못해도 사는데 아무 지장 없더라. 돈계산만 잘하면 되지."** 이건 거짓말이다. 사람이 먹고 살고 저축하는데 급급하면 사칙연산만으로 큰 문제가 없는게 사실이다. 수입과 지출, 신용카드 할부 관리만 하면 되니까... 그러나 일정 이상의 수입을 넘어서고 자산을 비축하여 부동산이나 주식 등을 통해 똑똑한 자산관리를 원한다면, 최소한 확률/통계와 수열(복리 계산) 등이 필요해지기 시작한다. 실제로 서로 다른 두 투자 기회를 비교할 때, 세금 문제 등이 얽히기 때문에 계산이 훨씬 복잡하다. 나는 이 즈음부터 학창시절 수학공부를 더 열심히 했어야 한다는 고민을 하기 시작했다. 수학이 삶과 괴리되어 있다는 주장은 그 정도 수입도 자산도 없기 때문에 복잡한 회계를 위한 지식이 필요 없어서 하는 소리다.

다양한 문제해결
의사결정을 진행
올바른 방법 및 절차 필요

가치, 그리고 왜 해야 하는가?

ranslate th
이 되는 쪽
산수 아닌

알고리즘 / 자료구조의 공부의 필요성

1. 필요성을 찾아보다가 "`Programmer`와 `Coder`의 차이는 자료구조/알고리즘에서 나온다"는 한 줄 짜리 주장을 보았다. 답은 교과서에 있지 않을까 봤더니, 머리말에 **알고리즘/자료구조는 특히 효율적이고 체계적인 프로그래밍 기법을 습득하는 데 기본이 된다** 고 적혀있음을 발견했다. 즉 습득력을 상승시키는 데 도움이 되는 이론이므로 공부를 해야 하는 것이다. 습득력은 개발자의 중요한 덕목이므로 당연히 공부해야겠다는 생각이 (이제는…^^) 든다.

2. 자료구조는 프로그램 구현의 난이도에 영향을 끼친다.

> *다양한 프로그램을 설계할 때, 어떠한 자료구조를 선택할지는 가장 우선적으로 고려되어야 한다. 이는* ***큰 시스템을 제작할 때 구현의 난이도나 최종 결과물의 성능이 자료구조에 크게 의존한다는 것을 많은 경험이 뒷받침하기 때문이다.*** *일단 자료구조가 선택되면 적용할 알고리즘은 상대적으로 명확해지기 마련이다. 때로는 반대 순서로 정해지기도 하는데, 이는 목표로 하는 연산이 특정한 알고리즘을 반드시 필요로 하며, 해당 알고리즘은 특정 자료구조에서 가장 나은 성능을 발휘할 때와 같은 경우이다. 어떠한 경우든, 적절한 자료구조의 선택은 필수적이다.*

3. 기본 중요하다. 그러나 실질적인 동기는 **역시 취업 때문인 듯** 싶다. 학교에서 3, 4학년들의 상담 내용을 엿들어보면 주로 취업때문에 공부를 하는 사람이 대부분이기 때문이다. SW 관련하여 면접을 보는 일들은 대부분 알고리즘 / 자료구조 능력을 필수로 본다.

어떤 문제에 대해서 "주어진 데이터 량 N"을 "일정한 시간 T" 안에 "확실히 해결" 할 수 있는 지를 대상으로 합니다.

무한의 시간, 메모리, 저장소가 있으면 뭐든 지 할 수 있겠죠. 될 때까지 해 보고 그걸 답하면 되니까요. CPU 연산능력이 넘쳐나면 알고리즘 안 써도 됩니다. (가령 1ms만에 우주의 모든 원자의 상호작용을 계산할 수 있다던가..)

즉 알고리즘은 가치는 "제한된 환경에서 성능을 논의할 때" 가치를 가집니다.

그리고 현실은 항상 CPU능력과 메모리는 제한되어 있고, 역사적으로 CPU/RAM이 우리가 가진 데이터를 다 다루기에 충분했던 적은 없습니다.

다만, 저렴한 100만원짜리 서버도 동접 5000도 안되는 일상적인 업무(쇼핑몰, 게시판)등등을 처리하는 데는 CPU/RAM차고 넘친다는 거죠.

그래서 몰라도 게시판 만들 수 있습니다.

하지만 같은 게시판이라도, 사이즈가 커져서 제한된 서버머신(32CPU, 1TRam이라고 하죠)에서 정해진 시간 T(3초라 합시다)안에 무식한 방법(전부 다 되는지 안 되는지 직접해 보는 방법)으로 처리가 불가능하면 "알고리즘에 생사"가 걸립니다.

이게 "알고리즘 능력의 가치"입니다. 구멍가게는 존재도 몰라도 되고, 큰 사업을 하려면 목숨이 걸립니다.

빅데이터 처리

알고리즘이란 결국 "최적화"입니다.

먼저, 권위에 기대어 보자면 현재 실리콘밸리의 회사들과 국내에서 나름 개발자들 처우가 좋다는 회사들 중 알고리즘 테스트를 하지 않는 회사는 없다. 내가 아는 선에서 국내에서는 삼성전자도 LG전자도 쿠팡도 카카오도 네이버와 라인도 알고리즘 테스트를 하거나 도입 중이다. 구글에서 주니어 인터뷰 시에 시스템설계 인터뷰는 잘 못해도 되지만, 알고리즘 인터뷰에서 좋은 인상을 남기지 못하면 탈락한다. 주니어 개발자들이 꿈꾸는 직장과 환경은 아마 이름도 소속도 모르는 익명의 누군가가 다니는 회사가 아니라 구글, 아마존, 네이버, 라인, 카카오 일 가능성이 높다. 그리고 그곳의 개발자들은 장담하는데 저런 주장을 하는 사람들보다는 한참은 높은 수준에 있다. 그들은 구현 능력보다 자료구조와 알고리즘과 관련된 문제를 해결할 수 있는지를 더 중요하게 생각한다. 기본기가 탄탄하다면 구현 능력은 당연히 따라온다는 것을 10여년이 넘는 채용 경험과 연구로 알고 있기 때문이다.

알고리즘은 필요한가 ?

- 프로그래밍은 단순히 소프트웨어 아키텍처와 객체지향 설계 만이 아님
- 프로그래밍은 주어진 환경에서 체계적이고 효율적으로 문제해결을 위한 과정임
- (질문) 컴퓨터 빨라지고 메모리 가격도 저렴해지고 있다면, 알고리즘은 덜 중요하게 될까 ?

경험적인 성능평가

```
$ cat reverse.py
def reverseA():
    count = 10**5
    nums = []
    for i in range(count):
        nums.append(i)
    nums.reverse()



def reverseB():
    count = 10**5
    nums = []
    for i in range(count):
        nums.insert(0,i)
```

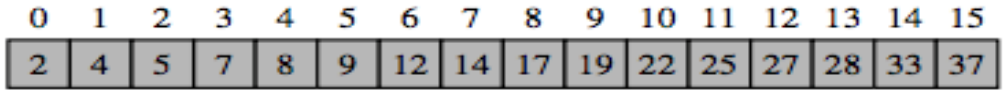
경험적인 성능평가

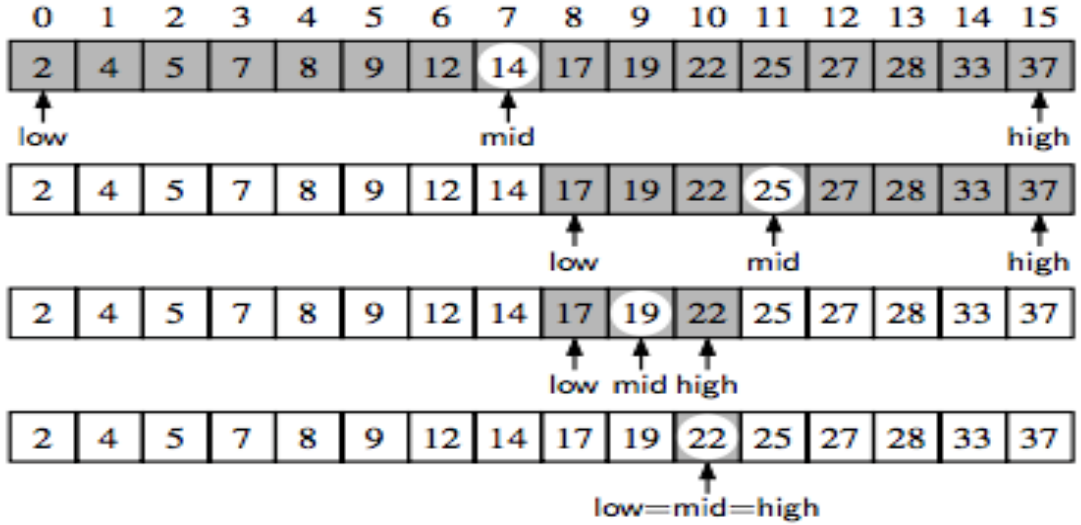
```
$ python3 -m timeit -s "import reverse as r" "r.reverseA()"
2 loops, best of 5: 127 msec per loop

$ python3 -m timeit -s "import reverse as r" "r.reverseB()"
1 loop, best of 5: 345 sec per loop
```

이진 검색

- 정렬된 자료에서 값(키) 22 을 효율적으로 찾기 위한 방법





- 만약 정렬이 안되어 있는 경우 다른 비교 방법과 비교 횟수는 ?
- 만약 자료찾기를 100번 실행한다면 수행시간은 ?

결론

- 낮은 하드웨어 및 클라우드 사용 비용에도 좋은 프로그래밍을 위해 알고리즘(자료구조 포함) 이해가 필요함
- 문제 해결 방법(알고리즘)은 다양하며 우리는 효율적인 방법을 찾는 것이 필요함
- 효율적인 알고리즘을 다루는 데이터의 특징에 따라 달라질 수 있음

경험적인 성능평가

```
$ python3 -m timeit -s "import reverse as r" "r.reverseA()"
20 loops, best of 5: 12.5 msec per loop
20 * 12.5 = 240.5 msec
$ python3 -m timeit -s "import reverse as r" "r.reverseB()"
1 loop, best of 5: 3.15 sec per loop
3.15 * 1000 = 3150 msec
```

* 알고리즘은 중요합니다.

단, 평생 하드웨어의 연산 능력이 문제가 될 만큼 큰 작업을 안하면 쓸 일이 없습니다.

하지만 과연 그럴까요?

* 알고리즘을 모르면, 개발자라고 불러주기 아깝습니다.

알고리즘은 이해하기 좋게 표현하면 "최적화"입니다. 아이디어만 제시할 거라면 최적화를 필요 없겠지만, 실제 사용할 물건을 만들 꺼라면 극한의 최적화는안 해도 적어도 쓸 만큼의 "최적화"는 해야 합니다.

그런데 A급 최적화 전문가는 아니라고 해도, 최적화에 대해서 F학점, 낙제를 한다? 과연 개발자라고 할 수 있을지 모르겠습니다.

물론 현실에서는 최적화 못해도 개발자라고 합니다. 머신파워로 해결하니까요.

그렇다고 해도, "코더"죠. 절대로 "프로그래머"가 될 수는 없다 봅니다.

현실적으로 개발자 = 코더 + 프로그래머 집단으로 본다면,

알고리즘을 몰라도 개발자(=코더)라고 칭할 수 있겠습니다.

하지만 연봉을 많이 주는 개발 회사는 개발자(=프로그래머)를 뽑으려고 하지, 개발자(=코더)를 뽑으려고 하지 않을 겁니다.

알고리즘을 모르면, 아무리 잘해 봐야 C급 개발자(보통은 D급 개발자)입니다.

(어느 분의 말처럼, 저라면 안 뽑아요.)