파인만 알고리즘 1p

1. 칠판에 문제를 적는다
2. 골똘히 생각한다
3. 칠판에 답안을 적는다

→ 문제 해결과정을 단계로 나누자!!

프로그래밍 대회를 위한 6단계 문제해결 알고리즘

1. 문제를 읽고 이해한다.
2. 문제를 익숙한 언어로 재정의 한다.
3. 어떻게 해결할지 계획을 세운다.
4. 계획을 검증한다.
5. 프로그램으로 구현한다.
6. 어떻게 풀었는지 돌아보고, 개선할 방법을 찾는다.

(상황이해) (단순화) 복기 및 개선

문제이해 → 문제 정의 ↑

→ 계획 수립 → 계획 검증 → 구현

(사용 알고리즘 선택) (올바른알고리즘인지 판단) (구현)

1단계: 문제를 읽고 이해하기 2P

그림 & 입출력 예제만을 보고 문제의 요구사항을 유추하곤 한다.

⇓

큰 다리가를 치를 성급한 방법

Why? 궁극적 목적을 알아도 사소한 제약조건이

잘못된 이해 ⇒ 못푸는 문제

2단계: 재정의와 추상화.

내가 다루기 쉬운 개념을 이용해 자신의언어로 풀어 쓰는 것. 요구사항이 복잡할수록 중요※

(어려운 문제)

문제의 추상화.

현실 세계의 개념을 수학적 / 잔상학적 개념으로 옮겨 표현하는 과정

⇓

프로그래밍이 나아갈 방향을 결정한다.

추상화 부분의 선택, 문제의 재정의 → 좋은 프로그래머가 되는 필수적인 방법

3단계: 계획 수립

문제해결 방식의 선정, 알고리즘, 자료구조 선택

문제 해결에 있어서 가장 중요한 부분 ★★★

알고 있는 알고리즘, 자료구조 적용가능한 단순한 문제

⇒ 간단한 과정

But 바로 떠오르지 않는다면 가장 복잡한 2번을 하게된다.

4단계: 계획 검증

구현에 앞서 세운 계획을 검증

선정한 알고리즘이 모든 경우 요구조건을 정확히 수행하는가?

시간과 메모리가 문제의 제한에 들어가는가?

4장, 5장에서 다뤄질 것이다

5단계: 계획 수행하기(구현하기)

프로그램 작성단계.

부정확, 비효율적인 구현 ⇒ 제대로 동작하지 않음

"정확한 구현의 중요성"

6단계: 회고하기 (피드백)

4P

장기적으로 가장 큰 영향

해결한 문제를 돌이켜보고 개선하는 과정

더 효율적인 알고리즘

더 간결한 코드

같은 알고리즘을 유도하는 직관적인 방법

외울의사결정코드도, 업밀한증명도 없는 추상적인기술

효과적인 회고방법

문제를 풀 때마다 자신의 경험을 기록으로 남기는 것

자신이 어떤식으로 접근했는지

결정적 아이디어는 아니더라도 득엇는지.

한번에 solve 하지 못했다면

오답노트도 꼭적기 → 자신이 자주 틀리는 부분을알게된다.

다른사람의 코드를 보는것도 좋은 방법!!

IF solve 하지 못했다면

일정시간이 지나도 도무지 알수없을경우 "타인의코드 참조"

But 반드시 복기를 할것!

나는왜 못풀었는가?