너희는 전혀 생각하고 있지 않아

SSG 김근호

2026/05/16

Table of contents

whoami

김근호 / rootk1m, gnosis

Security Researcher

IoT, Embedded, Automotive, Mobile, etc.
Exploit / Reversing

Vulnerability Research

18+ CVEs
40+ Bug Bounty Reports

SSG

SJU Computer Science and Engineering
Sejong Security Group

AI for Security

AI-based vulnerability analysis
SCOUT, Terminator, etc.

What I Do

Zero-Day vulnerability analysis
and discovery

What I Ask

Why does the system break?

r00t-kim.github.io

01

생각이란 무엇인가

What Does It Mean to Think?

무슨 생각하고 계신가요?

이걸 보면 무슨 생각이 드시나요?

수 정렬하기 성공

☆

시간 제한	메모리 제한	제출	정답	맞은 사람	정답 비율
1 초	128 MB	97276	55252	38224	58.320%

문제

N개의 수가 주어졌을 때, 이를 오름차순으로 정렬하는 프로그램을 작성하시오.

입력

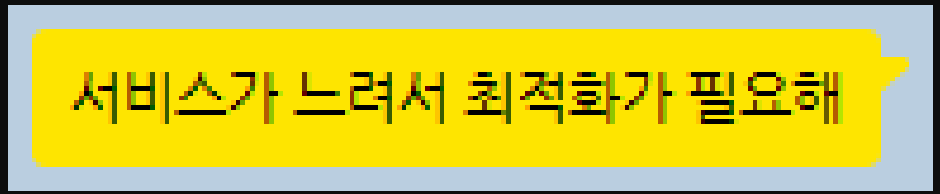
첫째 줄에 수의 개수 N(1 ≤ N ≤ 1,000)이 주어진다. 둘째 줄부터 N개의 줄에는 수 주어진다. 이 수는 절댓값이 1,000보다 작거나 같은 정수이다. 수는 중복되지 않는다.

출력

첫째 줄부터 N개의 줄에 오름차순으로 정렬한 결과를 한 줄에 하나씩 출력한다.

정렬 문제네
N <= 1000이네
sort 쓰면 되겠네
O(N log N)이면 충분하겠네

그럼 이건 어떤가요?



평균 응답 시간이 느린건가?

뭐가 느린거지?

프론트 렌더링?

서비스가 느려서 최적화가 필요해

사용자 체감이 느린건가?

네트워크 병목인가?

DB가 문제인가?

같은 “생각”인가요?

어떻게 풀지?

정렬 문제네
N <= 1000이네
sort 쓰면 되겠네
O(N log N)이면 충분하겠네

무엇이 문제지?

뭐가 느린거지?
프론트 렌더링?
DB가 문제인가?
네트워크 병목인가?

02

계산하는 사고와 묻는 사고

Calculative Thinking vs. Meditative Thinking

어떻게 풀지?

정렬 문제네
N <= 1000이네
sort 쓰면 되겠네
O(N log N)이면 충분하겠네

=

Calculative Thinking

Meditative Thinking =

무엇이 문제지?

뭐가 느린거지?
프론트 렌더링?
DB가 문제인가?
네트워크 병목인가?

Meditative Thinking

문제의 의미와 전제 자체를 다시 묻는 사고

vs.

Calculative Thinking

주어진 문제 안에서 답을 찾고, 계산하고, 최적화하는 사고

Calculative Thinking != 나쁜 사고

- 오히려 매우 필수적인 사고
- 낮은 단계가 아닌, 출발점
- 문제 해결에 효율적인 사고

너희는 전혀 생각하고 있지 않아

오히려, 이미 엄청나게 많이 계산하고, 적용하고, 풀이하고 있다.
하지만 그것으로 충분하지 않다. “진짜 문제가 무엇인지”를 보지 못한다.

03

우리는 왜 계산에 익숙해졌는가

Why We Became Good at Calculating

입시형 문제의 구조

문제는 이미 있다

Ex. “N개의 수를 오름차순으로 정렬하라”

이 문제는 어떤 유형이지?

정답도 존재한다

Ex. AC / WA로 판정된다

정답은 뭐지?

범위도 정해져 있다

Ex. N <= 1000 / 시간 제한 1초

어떤 풀이가 통과하지?

평가는 빠르고 정확한 풀이로 이루어진다

Ex. 시간 제한 / 메모리 제한 / 정답률

가장 빠른 풀이가 뭐지?

하지만 훈련되지 않은 질문

이게 진짜 문제인가?

문제는 이미 있다

Ex. “N개의 수를 오름차순으로 정렬하라”

이 문제는 어떤 유형이지?

지금 풀고 있는 건 원인인가? 증상인가?

정답도 존재한다

Ex. AC / WA로 판정된다

정답은 뭐지?

내가 당연하다고 생각한 전제는 뭐지?

범위도 정해져 있다

Ex. N <= 1000 / 시간 제한 1초

어떤 풀이가 통과하지?

평가는 빠르고 정확한 풀이로 이루어진다

Ex. 시간 제한 / 메모리 제한 / 정답률

가장 빠른 풀이가 뭐지?

왜 이 문제가 생겼지?

이게 진짜 문제인가?

04

진짜 문제는 주어지지 않는다

The Real Problem Is **NOT** Given

1. 현실은 문제를 정의해주지 않는다

서비스가 느리다.
보안 문제가 있는 것 같다.
사용자가 이 기능을 불편해한다.
이 프로젝트가 계속 밀린다.

무엇이 느린가?
정말 보안 문제인가?
사용자가 불편한 이유는 무엇인가?
프로젝트가 밀리는 이유는 무엇인가?

2. 답을 빨리 찾는 능력은
‘틀린 문제’를 빠르게 풀 수 있게 한다

Calculative Thinking은 빠르고 효율적이다.
하지만 방향이 틀리면, 아주 빠르게 이상한 곳으로 간다.

서비스가 느리다 → 쿼리 최적화부터 한다 → DB 문제가 아니었다..
strcpy를 봤다 → BOF라고 생각한다 → 실제 입력은 화이트리스트로 제한되어 있다..

3. 깊은 이해는 ‘전이’가 필요하다

공부는 반드시 필요하다.
개념, 도구, 패턴, 기초를 쌓아야 한다.

문제는 거기서 멈추면 새로운 상황으로 옮겨가기 어렵다.

단순히 문제 풀이 패턴을 외우는 것과, 그 패턴이 왜 작동하는지 이해하는 것은 다르다.

4. 보안은 ‘전제’가 공격 표면이 된다

이 입력은 여기까지 못 오겠지
이 API는 내부에서만 호출되겠지
이 앱은 신뢰해도 되겠지
이 파일은 사용자가 못 건드리겠지
이 토큰은 검증됐겠지
이 권한 경계는 문제 없겠지

...?

5. 질문 생성 능력 자체가 핵심 역량이다

Calculative Thinking은 주로 질문에 답하는 능력을 훈련한다
Meditative Thinking은 질문을 만드는 능력을 훈련한다

질문에 답하는 사람은 많다
하지만 무엇을 물어야 하는지 아는 사람은 훨씬 적다

???: 음음 그래그래 무슨 말 하는지 알겠어
근데 그래서 그걸 어떻게 하는데?

05

“왜?”의 사다리와 CS

The Ladder of “Why?” and CS

Why?

현상
↓ 왜?
직접 원인
↓ 왜?
구조
↓ 왜?
전제
↓ 왜?
진짜 문제

Why?

strcpy(buf, cmd)
↓
긴 입력이 들어오면 터진다
↓
cmd가 외부 입력일 수 있다
↓
입력 경로를 확인하지 않았다
↓
권한 경계와 호출 주체를 가정했다

왜 위험하지?

왜 긴 입력이 들어오지?

왜 외부 입력이라 생각했지?

왜 그 경로를 믿고 있지?

왜를 끝까지 묻다 보면 구조가 보인다

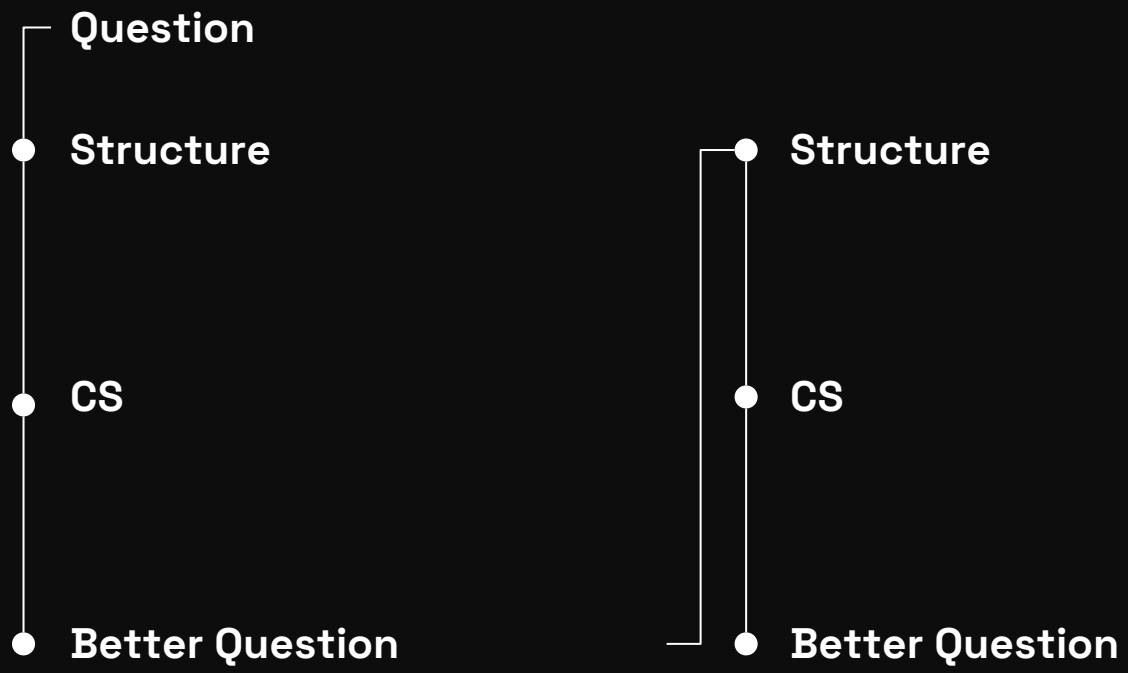
느리다
→ 병목
→ 캐시 / DB / 네트워크 / 복잡도

취약하다
→ 신뢰 경계
→ 입력 경로 / 권한 / 프로세스 / 메모리

그리고 그 구조를 설명하는 언어가 CS다

Memory
Process
Network
OS
Permission
State
Abstraction
Complexity
Interface
Trust Boundary
...

The Question Loop (Some call it Top-Down)



06

질문을 만드는 사람

Becoming a Question-Maker

동아리의 의미

어떻게 풀어요?

어디가 취약한 거예요?

뭘 공부해야 해요?

왜 이게 풀려요?

취약점을 외우는 데서 멈추지 말고 원리와 조건을 이해하자

왜 이게 터지는 거예요?

코드 한 줄이 아니라 구조와 원리를 보기 시작하자

뭘 설명하지 못하죠?

내가 모르는 것을 인식하고 다음 질문을 스스로 만들기 시작하자

원리

구조

인지

문제를 푸는 힘

+

문제를 다시 묻는 힘

Calculative Thinking + Meditative Thinking

Thank you

Do you have any questions?

awqs1221@naver.com
010-4399-8177
https://r00t-kim.github.io/

문제를 풀고 있습니까?
문제를 묻고 있습니까?