소프트웨어를 어떤 관점에서 바라보고 어떻게 잘 가르쳐야 할지 고민을 해왔어요. 새로운 교육기관을 만드는데 그 주체가 정부라면 성공적이 아닐 가능성이 높고, 아름답지 않다는 생각이 있었어요. 하지만 새로운 시스템을 만든다는 사실 자체는 매우 중요하고 의미있는 일이잖아요.

개발자들은 모두 같은 고민을 해요. '내가 소프트웨어를 조금 더 제대로 배웠으면 지금보다 훨씬 더 빨리 훌륭하게 해냈을텐데'라고 생각하죠. 배움에 대한 아쉬움이 있는거에요. 우리 멘토님들이 완전 만렙 끝판왕인 분들이거든요. 그 분들이 가지고 있는 배우는 방법에 대한 의견을 여기서 실현시켜 볼 기회가 생기는거에요.

좋은 자료들이 온라인에 많아요. 동영상 사이트에서 검색해서 조회수 많은 걸 클릭해서 재생하면 훌륭하거든요. 무조건 가르치려 들 필요는 없고 학생들이 갈 수 있는 방향을 잘 가이드해주면 되죠.

훌륭한 선생님이 있으면 훌륭한 학생이 나와요. 하지만 많은 수의 육성은 쉽지 않죠. 교육에 희생이라는 단어를 덧붙이곤 하잖아요? 희생을 통한 교육은 큰 규모로 이어질 방법이 없어요. 그래서 이런걸 시스템화하는게 중요한 문제에요. 그 시스템화를 이노베이션 아카데미에서 하는거죠.

본과정 이너 서클에서 c언어를 배우기 때문에 ''네카라쿠배''에서 원하는 표면적인 역량을 향상시키기엔 부족하다고 생각하잖아요? 하지만 그런 곳에 가서 면접을 보면 나에게 어떤 문제가 있고 뭘 놓치고 있는지 알게되요. 언어가 문제가 아니라 그 이면에 있는 내용들이 중요하고 그것들을 배우기 위한 게 본과정 코스에요.

아우터 서클 문제 수준의 현실적인 문제를 만나거나 만들 수 있다면, 그 현실의 문제를 해결하는 것이 훨씬 중요하다고 보고 있어요. 그런 기회를 만들기 위해서 학생들과 회사들을 지속적으로 만나게 하려고 하는거죠.

사회적인 친구들이 동료 학습을 더 잘 할 가능성이 높잖아요? 사회적이라는건 타고난 성향일수도 있고 훈련받은 결과일 수도 있어요.

혼자 프로젝트하고 온라인 강의나 잘 정리된 자료를 보면서 공부해도 시간을 들이면 할 수 있어요. 근데 그건 빨리 가는 방법이 전혀 아니죠. 그리고 동료 학습을 해서 얻을 수 있는 것들에 비하면 많은 것을 놓칠 가능성이 높아요. 옛날에 제가 배울 때는 혼자 학습하는게 비교적 용서가 됐어요. 책에 있는거 진도 나가면 되는거고 언어도 많지 않았구요. 하지만 지금은 전혀 그렇지 않거든요. 배워야 할 게 정말 많아요. 그래서 같이 배우는게 훨씬 효율적인거죠.

설계 패턴이라는 게 있어요. 문제마다 가장 효율적이고 수월한 접근 방식을 패턴화해서 전형적으로 해결하는거죠. 우리가 충분히 학습하고 나면 내가 고민했던 방식과 다른 사람이 고민했던 방식이 결국 비슷했구나라는걸 알게 되는 포인트가 있어요. 그걸 알기 전까지는 경험이 필요한거에요. 패턴을 먼저 학습한다는건 답을 먼저 보는거잖아요? 방법을 가르쳐주고 문제를 풀면 다 비슷한 답이 나오지만, 과정을 알려주지 않은 상태에서 각자 해결한 답을 보면 전혀 다른 방법을 이야기하기도 해요. 이게 동료학습의 큰 장점이죠. 서로 만나서 이야기를 하지 않으면 누릴 수 없는거에요.