안녕하세요?
지식 캠퍼스의 김현철입니다.

지난 시간에 알고리즘을 통한
자동화에 대해서

우리가 얘기를 했었는데요.

그 알고리즘은 우리가 머릿속으로
생각하는 정의고요.

이것이 실제적으로
기계처럼 사용되려면

컴퓨터 안에 들어가야 하잖아요.

그 알고리즘을 컴퓨터 안에
실제로 구현돼서 들어가게 하는

이 작업을 우리가
프로그래밍이라고 얘기합니다.

그런데 프로그래밍이라는
용어도 있고

또 다른 용어로 우리가 쓰고 있는
것이 코딩이라고 얘기합니다.

아주 쉽게 얘기하면 프로그래밍하고
코딩은 같은 말이다.

이렇게 여러분들이 생각하시면
될 것 같습니다.

이 그림을 한번 살펴봐주세요.

알고리즘은 아까 순서도로
주로 표현한다고 했는데

순서도 외에도 의사코드라는 걸로
또 표현하기도 합니다.

의사코드는 말로, 우리가
일반적으로 하는 그런 말을

구조화해서 쉽게 순서를
잘 정의한 것들이거든요.

그런데 그렇게 순서도나 의사코드에서
우리가 사용하는 말을

구전은 했지만 그것을 써서 어딘가에
전달하는 것은 굉장히 많은 오해와

아니면 잘못된 개념을
전달할 수가 있어요.

예를 들어서 요리책이,
요리 순서 레시피가

좋은 알고리즘이라고 제가
이전 시간에 말씀을 드렸는데요.

순서대로 진행을 하죠.
근데 이런 말이 있어요.

예를 들어서 요리책에 보면

'냄비가 물이 한소끔 끓으면 칼칼한
맛이 날 때까지 고춧가루를 넣어라'

뭐, 이런 용어가 있다고 해보세요.

저희는 제가 아는 것을 다 썼지만

'한소끔 끓는다'는 것은 도대체
어떤 것을 얘기하는 건지

그다음에 '칼칼한 맛'이라는 게 정의가
안 돼 있으니까 사람마다 다르잖아요.

그래서 그것을 그대로
다른 사람한테 줬을 때는

그것을 사용하는 사람이
자기가 느끼는 칼칼함과

자기가 느끼는 한소끔 끓는다는 것으로
자기 나름대로 정의해서

그 문제를 해결하면
즉, 요리를 만들면

내가 만든 요리하고 다른 요리가
나올 수 있습니다.

그럼 안 되잖아요, 그렇지요?

그래서 우리는 우리가 만든
알고리즘을 누구한테 주는가 하면

컴퓨터라는 기계한테 주는 거잖아요.

기계한테 주는데 내가 사용하는
나의 상식 가지고 만든 글은

컴퓨터에 전달했을 때 어떻게
사용되는지 모르는 거예요.

그래서 이 문제를 어떻게 해결하냐면
컴퓨터 프로그래밍 언어라는 것을 만들자

이 프로그래밍 언어는 핵심이 뭐냐 하면
미리 컴퓨터가 이해할 수 있는,

컴퓨터가 전혀 오해의 소지가 없이
하도록 돼 있는 말을

미리 정해 놓는 거예요.

많죠. 10개, 20개 정도가 아니고

몇 백 개, 몇 천 개가 될 수 있습니다.

그것을 정해놓고 우리는 내가 만든
알고리즘을 나의 용어가 아닌

컴퓨터 프로그램 언어에서
미리 정의해 놓은

그 명령어들을 가지고 내가
알고리즘을 다시 표현하는 거예요.

그렇게 해서 쓴 것을 우리가 뭐라
그러냐 하면 프로그램 코드라고 그러고

프로그래밍 언어를 가지고 그렇게
프로그램 코드를 만드는 그 과정을

'프로그래밍을 한다' 이렇게 얘기하고
그렇게 만들어진 프로그램 코드가

컴퓨터라는 기계 안에 들어가면
컴퓨터는 그대로 일을 진행하면 되죠.

오해나 무슨 이런 여지가 없어요.
그것이 전체적인 그림입니다.

다시 보면 알고리즘을 프로그래밍
언어로 프로그래밍해서

프로그램 코드가
만들어지는 것이고

그렇게 되면 그것이 컴퓨터라는
기계 안에서 사용이 되는 거고요.

프로그램 코드는 하나가 아니고
굉장히 많은 것들이

서로 연결돼서 사용될 수가 있고
그 안에 데이터도 집어넣으면

데이터도 연결돼서 사용될 수가 있고
또 입력 장치, 출력 장치

그다음에 인터넷을 통해서
다른 또 프로그램 코드,

다른 컴퓨터하고 연결돼서
작동을 할 수가 있습니다.

그리고 사람이 사용되는 것은 그
컴퓨터를 우리가 사용하고 있는 거죠.

그래서 알고리즘은 자동처리를 위한
알고리즘 아이디어이고

그다음에 그것이 프로그래밍 코드로
만들어지면 만들어진 그것을

우리가 소프트웨어라고
궁극적으로는 얘기하는데

'자동처리 기계' 이렇게 우리가
얘기를 할 수가 있습니다.

몇 가지, 이전 시간에 제가
잠깐 보여드린 예인데요.

프로그래밍 과정이
어떻게 만들어지는지를

간단하게 이것을 가지고
소개해 볼게요.

이것은 제가 가지고 있는
스마트폰에서

제가 심심할 때마다 하는
게임 중에 하나인데요.

저 로봇이 각 칸마다 밟아서
초록색 칸이 있잖아요.

그쪽으로 가게 하는 것을
내가 정의를 하는 겁니다.

그런데 사용될 수 있는 명령어는
밑에 있는 다섯 가지가 다예요.

한 발 전진, 그다음에
깜빡 불을 켜는 것

왼쪽으로 방향 트는 것, 오른쪽으로
방향 트는 것, 점프하는 것

자, 이 명령어만 사용해서
내가 갖다가 어떤 순서대로 뒀을 때

저 로봇이 파란 칸으로
갈 수 있을까요?

언뜻 이 그림만 보고 생각해 보면

한 발, 한 발, 한 발 가서 왼쪽으로 틀고
점프해서 올라가고 점프해서 내려가고

그다음에 불 켜고 그림 될 것 같죠.

잠깐 생각해 보세요, 여러분들
그래서 제가 그렇게 썼어요.

전진, 전진, 전진, 좌회전,
점프, 점프, 불 켜기 그렇게 했어요.

그랬더니 성공적으로 거기에 갔습니다.

이게 컴퓨터 프로그래밍의
가장 기본적인 본질입니다.

있는 명령어들을 내가 갖다가
순서대로 순서도에서 만들고

순서대로 뭔가를
정의해서 집어넣는 거죠.

컴퓨터는 그대로 합니다, 그대로요.

자, 또 다른 예를
제가 가지고 왔는데

이것은 언플러그드 활동에서
제가 가지고 왔습니다.

언플러그드 활동에서 가지고 왔고

'소프트웨어와 놀자'라는
사이트가 있는데요.

거기에는 굉장히 다양한 언플러그드
활동들을 보여주고 있는데

거기의 예를 가지고 왔습니다.

이 예는 뭐냐 하면 우리가 집에서
사용하고 있는 자동청소기

자동청소기를 프로그래밍 해 보자
하는 거예요.

자동청소기가 어떻게 되겠습니까?

저기 본 것처럼
사각형을 만들어놓고

그 안에 청소해야 될
종이 조각들을 많이 놔둡니다.

그리고 우리가 어떤 로봇을 만들어서
저기 있는 종이들을

다 청소할 수 있는 것들을
어떻게 명령을 내려 볼까?

어떻게 명령어들을
어떤 순서대로 해 볼까?

이런 활동이었고요.

학생들이 모여서
'어떤 명령어를 쓸까?'부터

정의를 하는 작업 활동을
하게 됩니다.

프로그래밍 언어가 아니기 때문에
미리 명령어가 정해져 있지는 않고요.

우리가 생각할 때 이 명령어는
정말 잘 쓸 거라는

명령어를 만들어 보는
작업부터 했고요.

그것을 통해서 아이들이 자기들이 만든 명령어 가지고 프로그램을 만들었어요.

이 순서대로 하면 될 거예요.

그렇게 했더니 학생이
그러면 그 순서대로

내가 로봇인 것처럼, 컴퓨터인 것처럼
듣고 거기서 시키는 대로

'한 발 앞으로 가세요.' 하면
한발 앞으로 가고

'좌로 45도 트세요.' 그러면
좌로 45도 틀고

그대로 했을 때 과연 저기 있는
쓰레기들이 다 청소가 됐을까?

먼지들이 청소가 됐을까?

그거를 계속 반복적으로 해 보는 거죠.

그렇게 해서 첫 번째에 될 리는 없죠,
그렇죠?

될 리가 없으면 학생들이 모여서
다시 새로운 명령어가 필요한지

아니면 이 순서를 다르게
어떻게 바꿔야 되는지

그래야지 저 문제가 해결될 수
있는지를 다시 고민하고

그것을 컴퓨터 프로그램에서는
'디버깅 한다'고도 하고, 뭐 이렇게합니다.

그래서 컴퓨터 프로그래밍을 만든다,
알고리즘을 만든다는

한 번에 되는 일은 없습니다.

없고, 한번 해보고 다시 개선을 하고
다시 조정을 하고 다시 해보고

다시 또 어떤 예외적인 상황이
발생하는 것을 고려해서

다시 또 개선을 해 보고 그 과정을
거쳐서 최종적인 알고리즘,

최종적인 프로그램 코드, 최종적인
소프트웨어가 나오게 되어 있습니다.

이것을 위해서 많은 프로그래밍 언어가
이 세상에 나와 있는데요.

프로그래밍 언어마다 마치 인간의
언어처럼 명령어들이 또 다 다르구요.

또 각각 프로그래밍 언어마다
잘하고 있는 부분이 있습니다.

'우리는 이것을 더 잘할 수 있는
명령어를 더 많이 가지고 있어요.'

이런 것들이 있어서 많은
명령 프로그래밍 언어들을

여러분들이 볼 수가 있는데

우리가 지금 이 수업에서
얘기하고 있는

컴퓨팅 사고력을 얘기하기 위해서는

일반적으로 어른들이 상업용으로 쓰는
컴퓨터 프로그래밍 언어보다는

교육용으로, 컴퓨팅 사고력을
향상하기 위한 목적으로 만든

교육용 프로그래밍 언어들이 있습니다.

그것을 사용하게 되는데
그중에 가장 대표적인 것이

'스쿠레치'라고 하는
프로그래밍 언어고요.

교육용 프로그래밍 언어입니다.

스쿠레치는 MIT 미디어 랩에서
2003년에 만들었고

최근에 버전3까지 나와서

전 세계에서 가장 많은 아이들에게
사용되고 있는 그런 언어입니다.

왼쪽 그림에서 보시면 명령어들이
마치 레고 블록처럼 모양을 가지고

레고 블록 하나가
하나의 명령어입니다.

이렇게 있고 아이들은 그것을
마우스로 끌어다가

착착 레고 쌓듯이
붙여나가는 거죠.

붙여나가면 가운데 있는 것이
그 만든 프로그램이고

그것을 클릭하면 내가 명령한 대로
오른쪽 화면에서 고양이가

어떤 활동을 하고, 움직이고, 작업을
하고, 그런 것들을 즉각 즉각 내가

고쳐가면서 레고 블록 쌓듯이
할 수 있게 만들었습니다.

이것은 굉장히 또 다른 철학을
밑에 배경에 깔고 있는데요.

스쿠레치를 만든 첫 번째 책을 보면
가운데, 제가 그림으로 그려놓은 거

그 책에서 제가 가져온 그림인데
이런 철학을 가지고 있습니다.

유치원에서 우리가 배우는 것처럼

반복적인 순서를 통해서
우리는 배운다.

뭐냐 하면 먼저 상상을 하고
그다음에 그것을 만들어 보고

그다음에 그것을 다른 사람들하고
공유를 해 보고

그다음에 그것을 가지고 다시
또 개선을 하고 다시 만들고

이것을 계속 반복해 나가는 거죠.

예를 들면 우리가 다 같이, 나는 뭔가
비행기를 만들었어, 유치원에서요.

또 옆에 아이는 다리를 만들었어,

"어, 너 다리 만들었니? 내 비행기가
너 다리를 지나갈 수 있는 걸 볼까?"

'윙' 하다가 다리에 걸리는 거예요.

"그러면 다리를 좀 크게 만들어 보자."

그래서 비행기가 지나갈 수 있는
다리를 크게 만들었더니

또 다른 애가 옆에서 보고

"아, 나는 구름을 만들었는데
그럼 이걸 어떻게 할까?"

굉장히 거기서 서로 교류하고 협업하고
개선하고 이게 반복적으로 일어나는

이 작업이 수월하게 일어나도록
저 프로그래밍 언어가 설계됐습니다.

그리고 저 프로그래밍 언어는
전 세계, 글로벌적으로

모든 아이들이 들어와서
자기가 만든 작품을 올려놓고

남이 만든 작품을 갖다가
내가 고쳐도 보고

고쳤을 때는 내가 내 작품을
누가 어떻게 고쳤는지도

바로 알아볼 수 있게
만들어놨어요.

그러니까 우리가 첫 시간에
얘기했었던 컴퓨팅 사고력

즉, 컴퓨팅적으로 사고를 하고
글로벌적으로 사고를 하고

정보적으로 사고를 하고

그다음에 협업 능력, 공유 능력,
창조 능력이 한꺼번에 다 들어가서

활동할 수 있게
그렇게 설계가 된 것이고

제가 좀 극단적으로 얘기하면

이 프로그래밍 언어만
잘 사용해서 활동을 하면

컴퓨팅 사고력의 기본적인 것들이
다 습득이 될 수 있다.

이렇게 얘기를 할 수 있습니다.

그래서 컴퓨팅 사고력을 습득하는 데
가장 큰 연습과 방법과 도구는

교육용 프로그래밍 언어를 가지고

작품 활동을, 창업, 창조활동을
내가 직접 해보는 거다.

그리고 그것을 다른 친구들과
공유해 보고, 나눠서 얘기해 보고

이것이 다른 사회적으로 어떤 영향을
미칠지에 대해서 토론해 보고

이 과정이 컴퓨팅 사고력을
우리가 다 경험해 볼 수 있는

그런 거라고 우리가
얘기할 수 있습니다.