



Robot Software Education Institute

로봇 SW 교육원

# 로봇활용 소프트웨어 교육

오리엔테이션

광운대학교 로봇학부  
박광현

# 일정

CT에 관하여            13:00 ~ 14:30

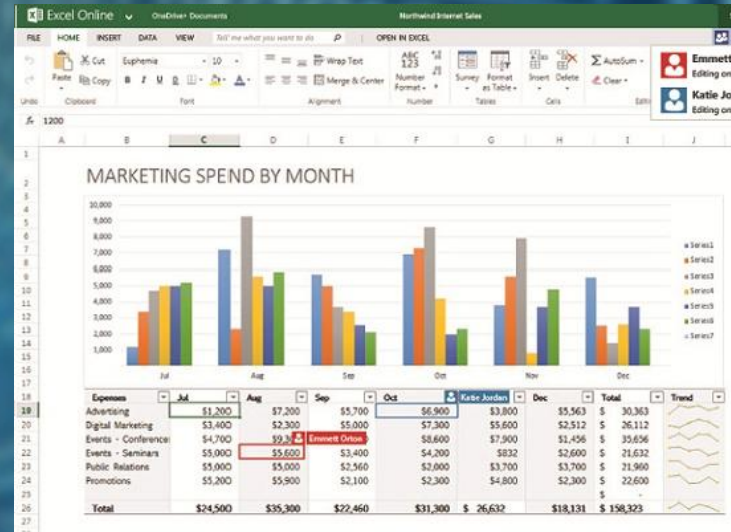
로봇 소개 및 스크래치 실습

14:30 ~ 16:00

프로세싱 실습            16:00 ~ 16:30

토론                        16:30 ~ 17:00

# 데이터를 지배하라!





# 데이터를 지배하라!

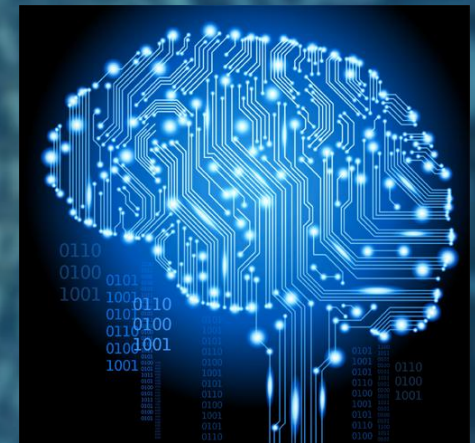
빅 데이터

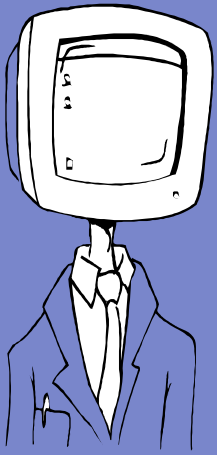


사물 인터넷



인공지능





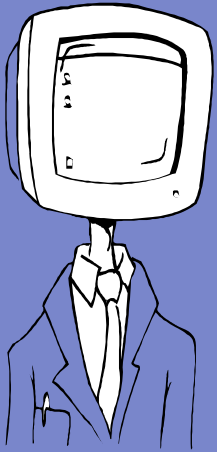
## 컴퓨팅 사고력 ?

문제를 수립하고 해결책을 만들어  
컴퓨팅 시스템을 통해  
효과적으로 수행되도록 표현하게 하는  
사고 과정

초·중·등 단계 Computational Thinking 도입을 위한  
기초 연구, 한국과학창의재단, 2014.06

컴퓨팅의 기본적인 개념과 원리를 기반으로  
문제를 효율적으로 해결할 수 있는 사고 능력

소프트웨어 교육 운영 지침, 교육부, 2015.02



## 컴퓨팅 사고력의 구성 요소

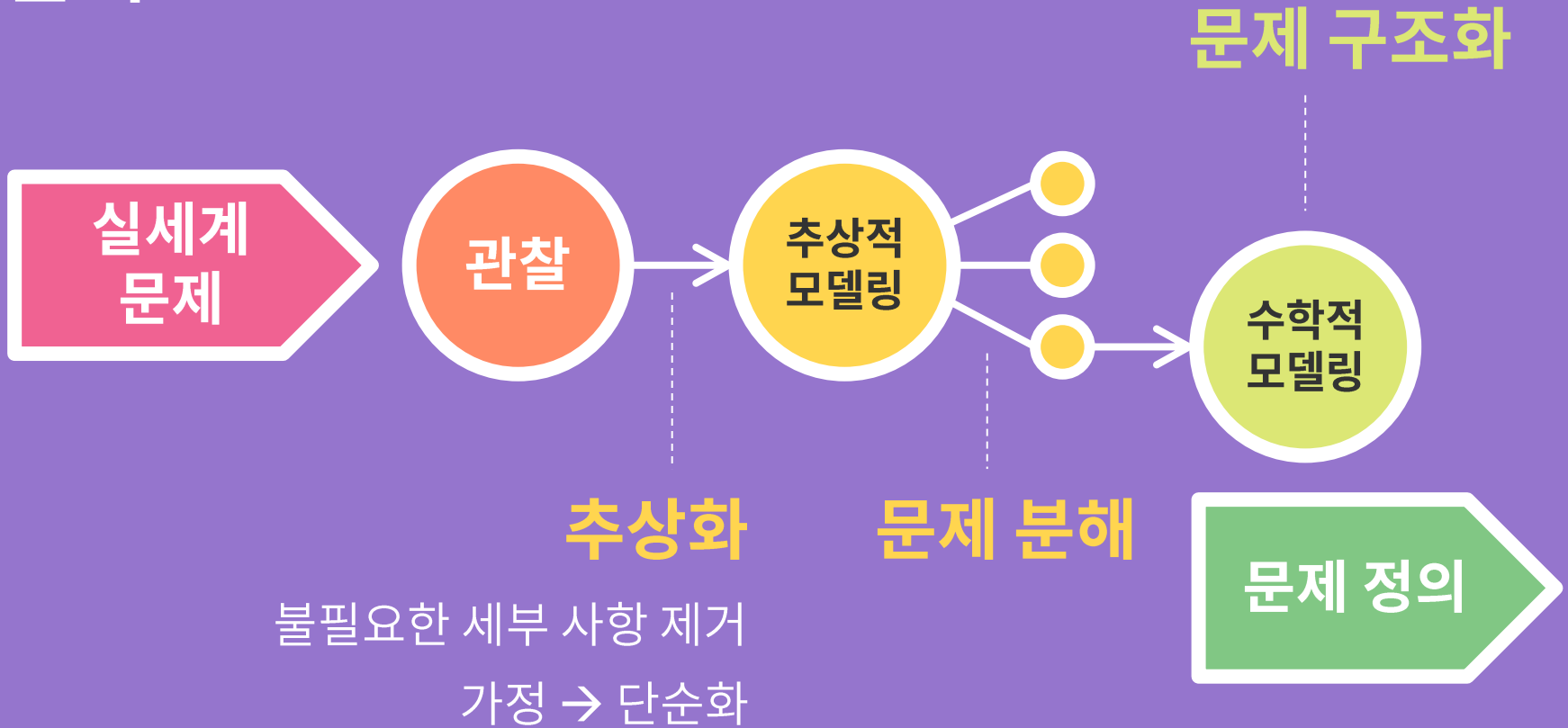
- 문제를 컴퓨터로 해결할 수 있는 형태로 구조화하기
- 자료를 분석하고 논리적으로 조직하기
- 모델링이나 시뮬레이션 등의 추상화를 통해 자료를 표현하기
- 알고리즘적 사고를 통하여 해결 방법을 자동화하기
- 효율적인 해결 방법을 수행하고 검증하기
- 문제 해결 과정을 다른 문제에 적용하고 일반화하기

소프트웨어 교육 운영 지침, 교육부, 2015.02

컴퓨터의 계산 능력을 활용한

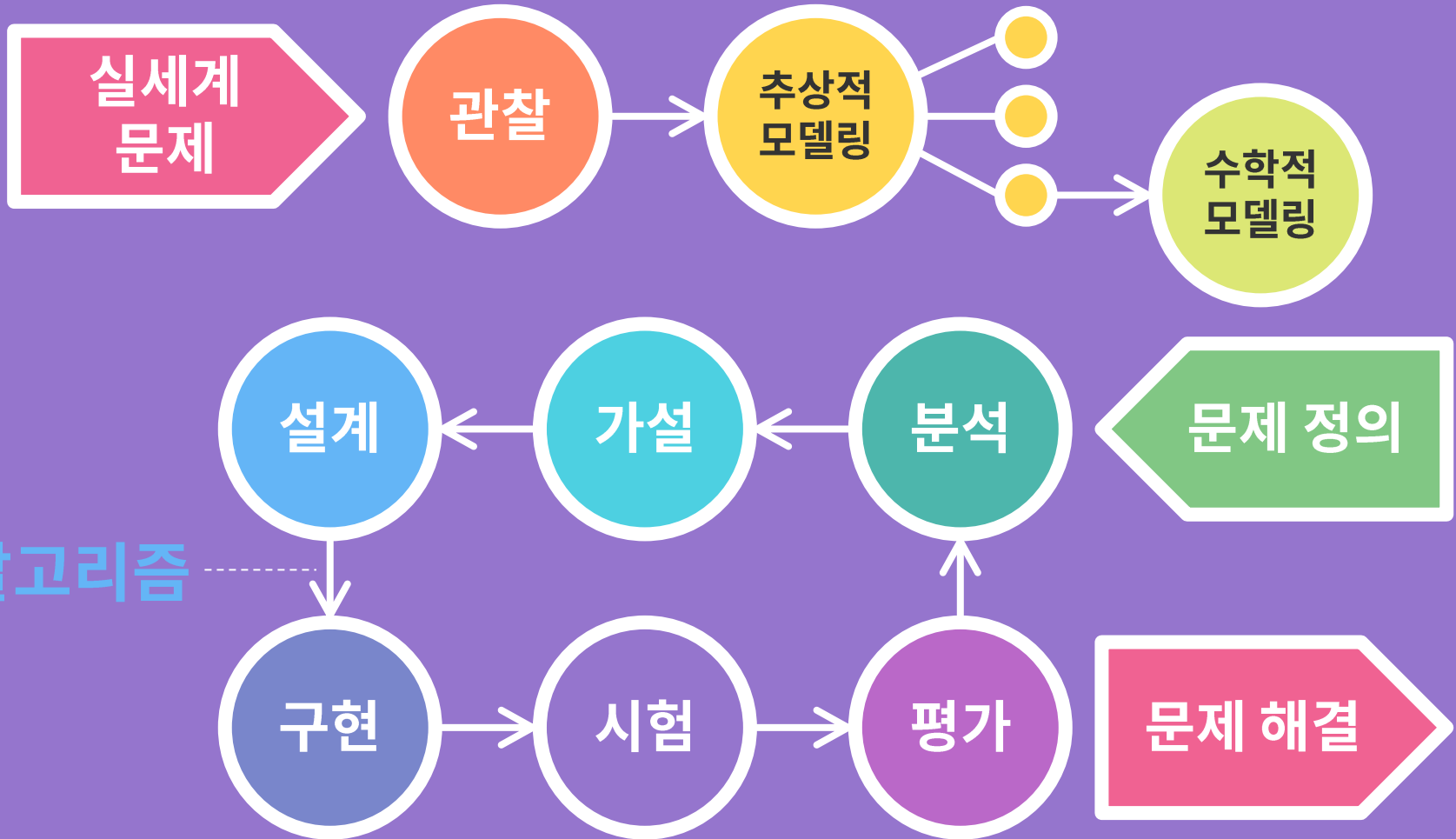
# 문제 해결 과정

# 문제 ?

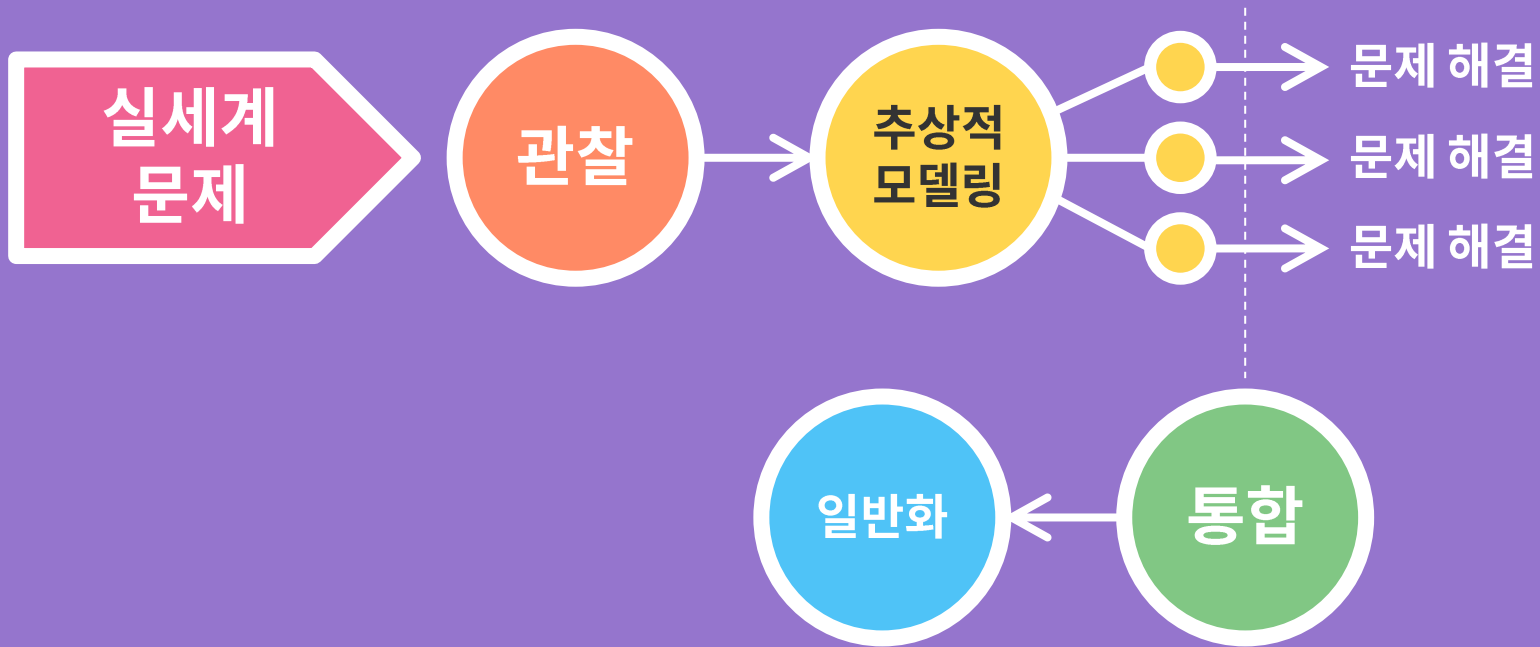




# 과정 ?



# 과정 ?



**아이들에게는 너무 어렵다 !**

# 문제 해결 과정의 단순화

1

## 문제 정의 또는 문제 표현

과학적 문제는 관찰과 데이터 분석을 통해서!

### 왜 내 로봇은 똑바로 가지 않을까?

중앙선에서 얼마나 벗어나는지?

양쪽 바퀴의 회전 수가 얼마나 다른지?

### 데이터를 분석·체계화하여 문제를 명확하게 표현

친구에게 설명하여 명확하지 않은 부분 모두 제거

또는 문제를 여러 개로 나누기

추상화(단순화)

문제 분해

# 문제 해결 과정의 단순화

1

## 문제 정의 또는 문제 표현

사회적 문제는 요구 사항 분석을 통해서!

**자신의 문제보다는 친구의 문제를 해결해 주자**

질문을 많이 하자

요구 사항을 말로 표현하면서 구체화

**요구 사항을 분석·체계화하여 문제를 명확하게 표현**

친구에게 설명하여 명확하지 않은 부분 모두 제거

또는 문제를 여러 개로 나누기

추상화(단순화)

문제 분해

문제 해결 과정의 단순화

## 2 해결책 찾고 구현하기

**생각을 모두 정리한 후에 구현하자**

알고리즘

책상에 앉아 바로 구현하기 시작하는 것 금지!

정리된 생각을 구현하기 전에 시뮬레이션 해 보자

**2인 1조로 짝 코딩 하자**

시뮬레이션

역할에 충실하게, 역할을 바꾸어 가며



3

## 시험, 디버깅, 수정

### 컴퓨터는 시킨 대로만 동작한다

디버깅!

명령어를 하나씩 친구에게 말하고 친구가 하나씩 몸으로 수행해 보면서 잘못된 부분을 찾자

### 생각대로 동작한다고 끝이 아니다

코드 리뷰

더 효율적으로 구현할 수 있는 방법은 없는지 생각해 보자  
발표를 통해 다른 친구는 어떻게 구현했는지 살펴 보자  
해결한 방법을 다른 문제에 적용하려면 어떻게 해야 할까?

추상화(일반화)

# 생각하는 아이로 키우려면?

## 답을 알려 주지 마세요

이유가 뭘까? 이렇게 하면 저렇게 되지 않을까? 생각 유도하기

## 생각할 수 있는 시간을 충분히 주고 기다려 주세요

답이 목적이 아니라 과정이 중요

## 팀을 짜서 같이 하게 하세요

코딩은 친구와 함께... 가르치는 것이 최고의 학습

컴퓨팅 사고의 목적은

비판적 사고력

논리적 사고력

# 왜 SW 교육을 하는가?

## 의도적 학습

즉각적인 피드백이 중요

SW는 즉각적인 피드백이 가능

생각하지 않는 즉각적 피드백은 독이 될 수 있다

## 로봇은?

몰입도 유지를 위한 수단

도구로서만 활용하고 목적이 되어서는 안 된다


**코딩에 관하여...**

# 프로그래밍 ?

컴퓨팅 문제를 해결하기 위한 방법을  
실행 가능한 컴퓨터 프로그램으로 만드는 것

문제 영역에 대한 다양한 전문 지식 필요

요구 사항 분석, 이해, 구조 설계, 알고리즘 만들기,  
알고리즘의 요구 사항 검증 (무결함, 자원 사용 등),  
알고리즘 구현,  
디버깅, 테스트,  
유지 보수, 빌드 시스템 등등



**코딩**



# 코딩 = 글쓰기

좋은 코드를 많이 읽고  
많은 코드를 작성해 본다

좋은 글을 많이 읽고  
많은 글을 작성해 본다

초고는 가슴으로 쓰고, 재고는 머리로 쓴다

파인딩 포스터

# 코딩을 학습하는 단계

1

## 순차

명령어 하나, 같은 명령 순서대로,  
다른 명령 섞어서 순서대로

2

## 횟수 반복

단순 반복 (한 가지 명령 반복),  
규칙(패턴) 있는 반복 (패턴을 반복)  
순차 반복 (단순 반복 + 순차, 단순 반복 + 단순 반복)  
규칙(패턴) 있는 순차 반복, 내포 반복\*

3

## 디버깅

명령어 추가, 삭제, 수정  
순차를 반복으로 수정 (패턴 찾기)  
반복 내에서 추가, 삭제, 수정

\*고급 과정

4

## ~까지 반복

단순 반복  
규칙(패턴) 있는 반복  
순차 반복  
규칙(패턴) 있는 순차 반복, 내포 반복\*

# 코딩을 학습하는 단계

5

조건

만약 ~이라면  
만약 ~이라면, 아니면~  
~까지 반복과 조합

6

~동안 반복

7

논리

AND, OR, NOT

8

이벤트

이벤트 처리

# 코딩을 학습하는 단계

9

함수 호출

이미 존재하는 함수를 호출만

10

변수

11

for 반복

12

함수 만들기

함수 구현

13

병렬 처리\*

# 패턴 찾기가 중요

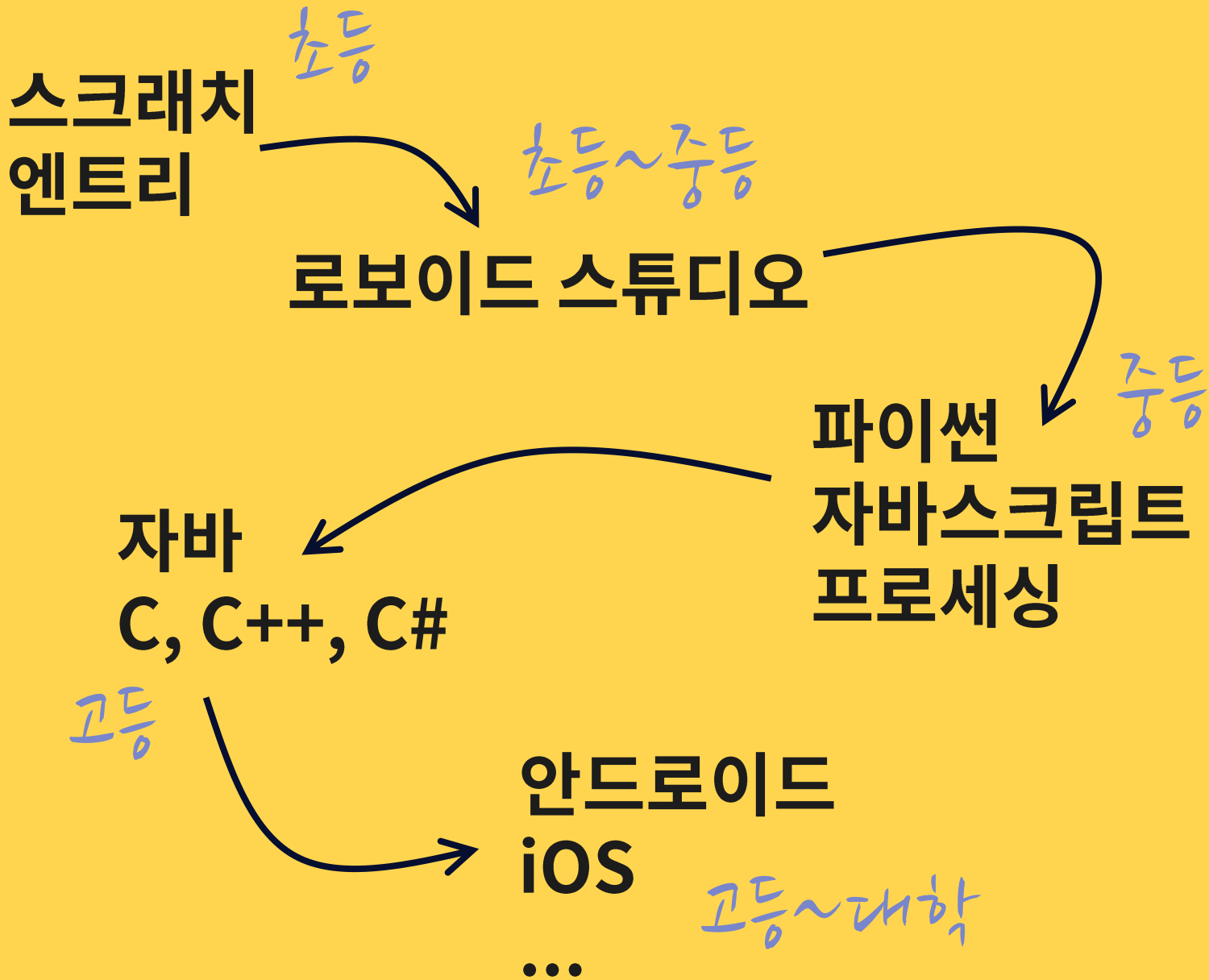
반복  
추상화  
함수

## 스스로 문법 유추하기

문법 위주로 가르치지 않는다  
예제에서 패턴을 찾아 문법을 만들어 보게 한다  
추후 실제 문법과 무엇이 다른지 비교한다

**학습 도구에 관하여...**





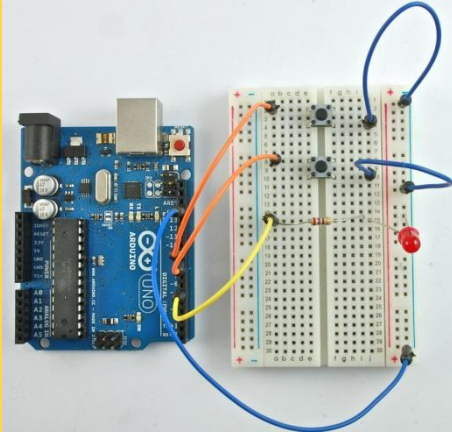
# 블록 코딩에서 텍스트 코딩으로 넘어가는 시기

한 번에 생각할 수 있는 단위는  
컴퓨터 화면에 보이는 범위 만큼으로 제한됨

블록 구성이 한 화면을 넘어가는 시기가 되면

- 함수를 만들어 생각의 범위를 분리하거나
- 좀더 함축적으로 표현할 수 있는 다른 그래픽 도구  
(예: 로보이드 스튜디오)로 넘어 가거나
- 텍스트 코딩으로 넘어가서  
생각의 범위를 넓혀 주어야 함

# 아두이노



**회로에 대한 이해 필요**

저항을 왜 연결해야 하나요?

**궁금증 해소 안되면**

**질문에 대한 동기 유발 감소**

```
Blink | Arduino 0023
File Edit Sketch Tools Help
Blink$
/*
 * Blink
 * Turns on an LED on for one second, then off for one second, repeats.
 * This example code is in the public domain.
 */

void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH); // set the LED on
  delay(1000);            // wait for a second
  digitalWrite(13, LOW);  // set the LED off
  delay(1000);            // wait for a second
}

Done uploading.
Binary sketch size: 1018 bytes (of a 32256 byte maximum)
```

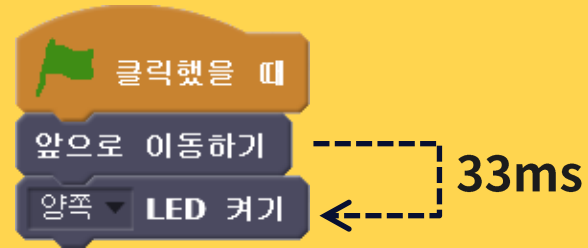
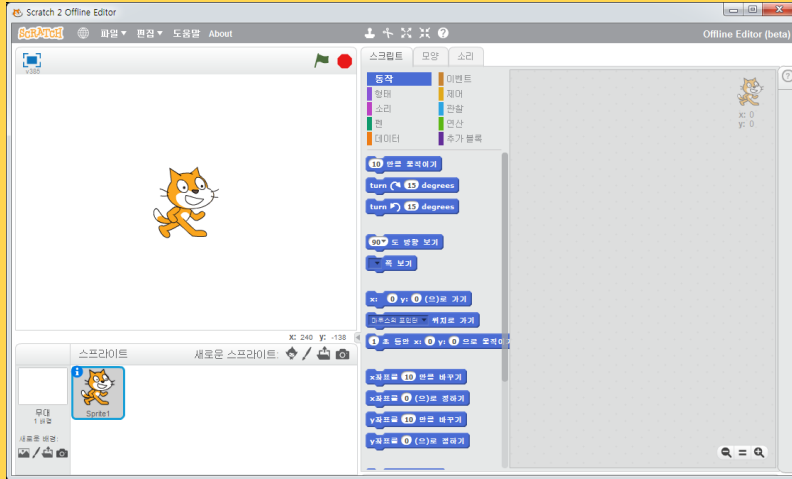
**void setup()**

**void loop()**

코딩 교육하기에는 한계가 있음

메이커 교육에는 좋은 도구이나  
소프트웨어 교육에는 부적절

# 스크래치



그래픽만으로 하는 것은 좋으나  
하드웨어 연결은 근본적인 문제점 존재

스크래치의 실행 엔진을 바꾸지 않으면 문제 해결 불가

**오늘의 로봇을 소개합니다**

# 크기 및 무게

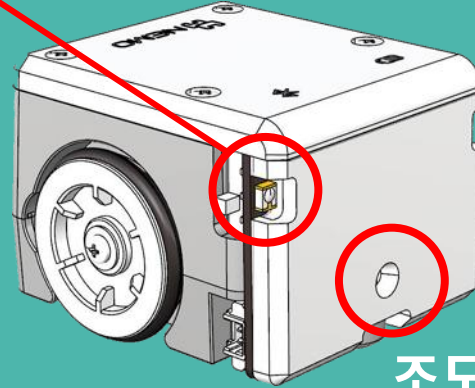
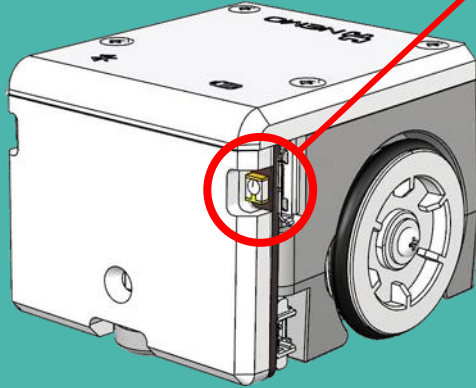


크기 가로 35mm x 세로 40mm x 높이 30mm  
무게 30g

# 센서

1 ~ 30cm, 초당 100회 측정, 초당 50회 전송

## 전방 거리 센서 (근접 센서): 장애물 감지

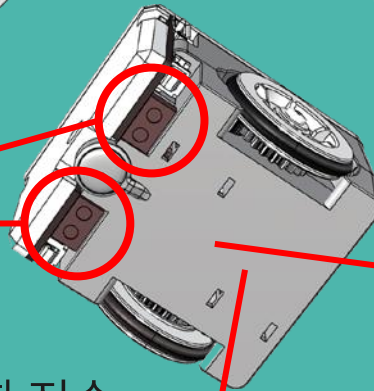


## 조도 센서

0 ~ 65,535룩스, 초당 10회 측정

## 바닥 센서: 라인 트레이싱

반사형 포트 인터럽터,  
초당 100회 측정, 초당 50회 전송



## 3축 가속도 센서

2g, 4g, 8g, 16g 범위 설정,  
7.81 ~ 1000Hz 반응 속도 설정

## 내부 온도 센서

섭씨 -40 ~ 87.5도, 0.5도 정밀도

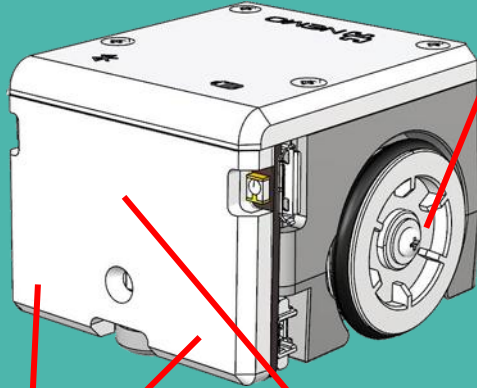
## 배터리 전압 센서

0.01V 정밀도

# 구동 장치

DC 기어드 모터

PWM 100단계 구동,  
최대 이동 속도 10cm/sec,  
위성 기어

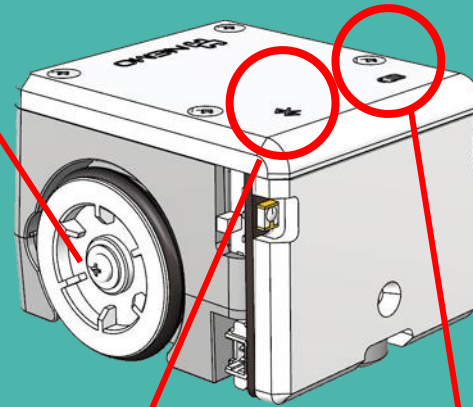


LED  
7가지 색

블루투스 연결 표시 및  
배터리 경고등

버저

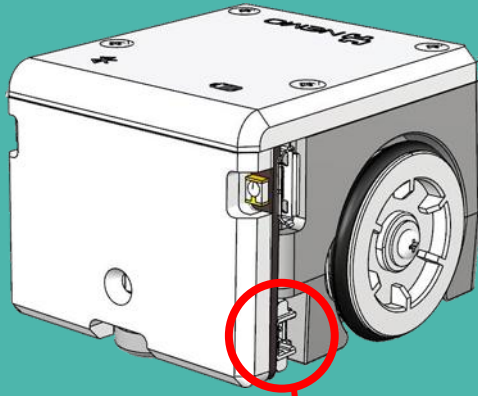
1Hz ~ 167.77215kHz, 모노톤, 0.01Hz 단위  
88 키, A3 ~ A7, 12 평균율, 정밀도 +/- 0.1cent



충전 표시등

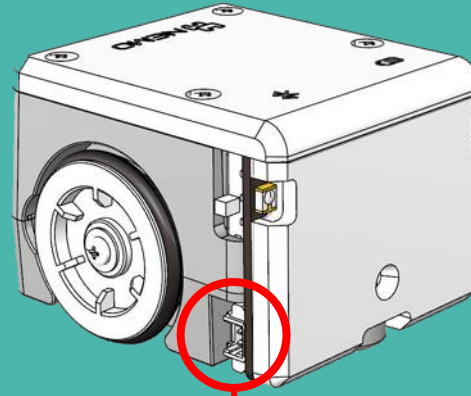


# 외부 입출력



**포트 A/B**

5가지 모드: 디지털 입력  
ADC 입력 (8비트, 0 ~ 3.6V 기준)  
디지털 출력  
아날로그 PWM 출력 (255단계)  
아날로그 서보 제어 (0 ~ 180도)



**VCC, GND**

# 기타



통신 방식  
배터리  
충전 단자  
PC 연결  
스마트 폰 연결

블루투스 4.0 BLE (접속 범위 15m 이내)  
충전 30분, 연속 동작 평균 1시간, 대기 최대 12시간  
마이크로 USB (스마트 폰 충전 단자)  
USB Dongle 통해 시리얼 통신  
블루투스 4.0 BLE 연결

# 지원 SW : 8월 중순까지 완료되는 SW들



ENTRY



# 회사 소개

뭐든지 물어 보세요  
광운대학교 박광현 교수  
akaii@kw.ac.kr

## 로보메이션 하드웨어만 판매

햄스터 로봇 구입 문의 : 로보메이션에서는 직접 유통하지 않으므로  
roboidfactory@gmail.com 으로 구매 문의

## 카이맥스 교육 솔루션 판매

햄스터 로봇 + 교재 패키지 문의 : 김혜주 부서장  
hjkhera@naver.com

## SK텔레콤

알버트 로봇 구입 문의 : 황은동 부장  
edhwang@sk.com

실습 #1

스크래치

# 오늘의 문제

## 초고령 사회, 반려 로봇

<https://www.youtube.com/watch?v=V-WTCm7kOP0>



# 따라 해보기 #1

로봇 움직여 보기

# 따라 해보기 #2

일정 거리 유지하기



# 따라 해보기 #3

뒤집으면 노래 연주하기

# 오늘의 문제

## 초고령 사회, 반려 로봇

<https://www.youtube.com/watch?v=V-WTCm7kOP0>



실습 #2

프로세싱

# 따라 해보기 #1

로봇 움직여 보기

```
import org.roboid.robot.*;
import processing.hamster.*;

Hamster hamster;

void setup() {
  hamster = new Hamster(this);
}

// don't forget 'draw'
void draw() {
}

void control() {
  // move forward
  hamster.write(Hamster.LEFT_WHEEL, 50);
  hamster.write(Hamster.RIGHT_WHEEL, 50);
  delay(500);

  // move backward
  hamster.write(Hamster.LEFT_WHEEL, -50);
  hamster.write(Hamster.RIGHT_WHEEL, -50);
  delay(500);

  // stop
  hamster.write(Hamster.LEFT_WHEEL, 0);
  hamster.write(Hamster.RIGHT_WHEEL, 0);
}
```

# 따라 해보기 #2

손 따라 오기

```
import org.roboid.robot.*;
import processing.hamster.*;

void setup() {
  new Hamster(this);
}

void draw() {

void execute(Robot hamster) {
  // left wheel
  int proximity = hamster.read(Hamster.LEFT_PROXIMITY);
  if(proximity > 15) {
    hamster.write(Hamster.LEFT_WHEEL, (40 - proximity) * 4);
  } else {
    hamster.write(Hamster.LEFT_WHEEL, 0);
  }

  // right wheel
  proximity = hamster.read(Hamster.RIGHT_PROXIMITY);
  if(proximity > 15) {
    hamster.write(Hamster.RIGHT_WHEEL, (40 - proximity) * 4);
  } else {
    hamster.write(Hamster.RIGHT_WHEEL, 0);
  }
}
```

# 토론 주제

컴퓨팅 사고 - 창의 컴퓨팅 - 메이커

보편적 교육 - 전문적 교육



# 추후 일정

블록 코딩

8월 3일 ~ 4일

13:00 ~ 18:00

대회의실

파이썬

8월 8일, 15일, 22일

13:00 ~ 18:00

강의실 A

**수고하셨습니다 !**

**akaii@kw.ac.kr**