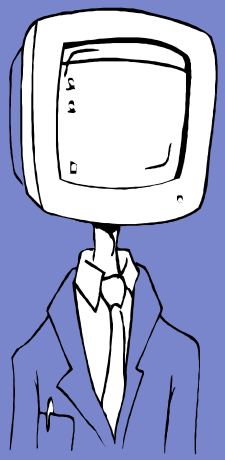
SW교육 연계 로봇 실험학교

중등 (심화)

로봇활용 SW교육의 필요성 및 현황
+ 로봇 제어 기초

광운대학교 로봇학부
박광현



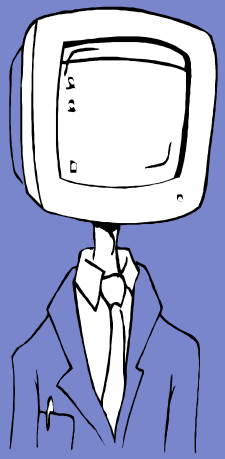
컴퓨팅 사고력 ?

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

초중등 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 정의 과정

불필요한 세부 사항 제거

가정 → 단순화

문제 정의 과정의 단순화

1 문제 정의 또는 문제 표현

사회·인문학적 문제는 **요구 사항 분석**을 통해서 !

자신의 문제보다는 다른 사람의 문제를 해결해 주자

관찰과 질문을 많이 하자

요구 사항을 말로 표현하면서 구체화 *추상화(단순화)*

항목 나열 → 우선 순위 표시 → 우선 순위 높은 것 2~3개만

요구 사항을 분석 · 체계화하여 문제를 명확하게 표현

다른 사람에게 설명하여 명확하지 않은 부분 수정

문제를 여러 개로 나누기 *문제 분해*

문제 정의 과정의 단순화

1 문제 정의 또는 문제 표현

과학·공학적 문제는 **관찰과 데이터 분석**을 통해서 !

왜 내 로봇은 똑바로 가지 않을까?

측정 도구로 주변 환경 및 현상을 측정 및 관찰

측정 데이터를 표 또는 그래프로 분석 *추상화(단순화)*

경향 파악 및 가정을 두어 문제 단순화

데이터를 분석 · 체계화하여 문제를 명확하게 표현

다른 사람에게 설명하여 명확하지 않은 부분 수정

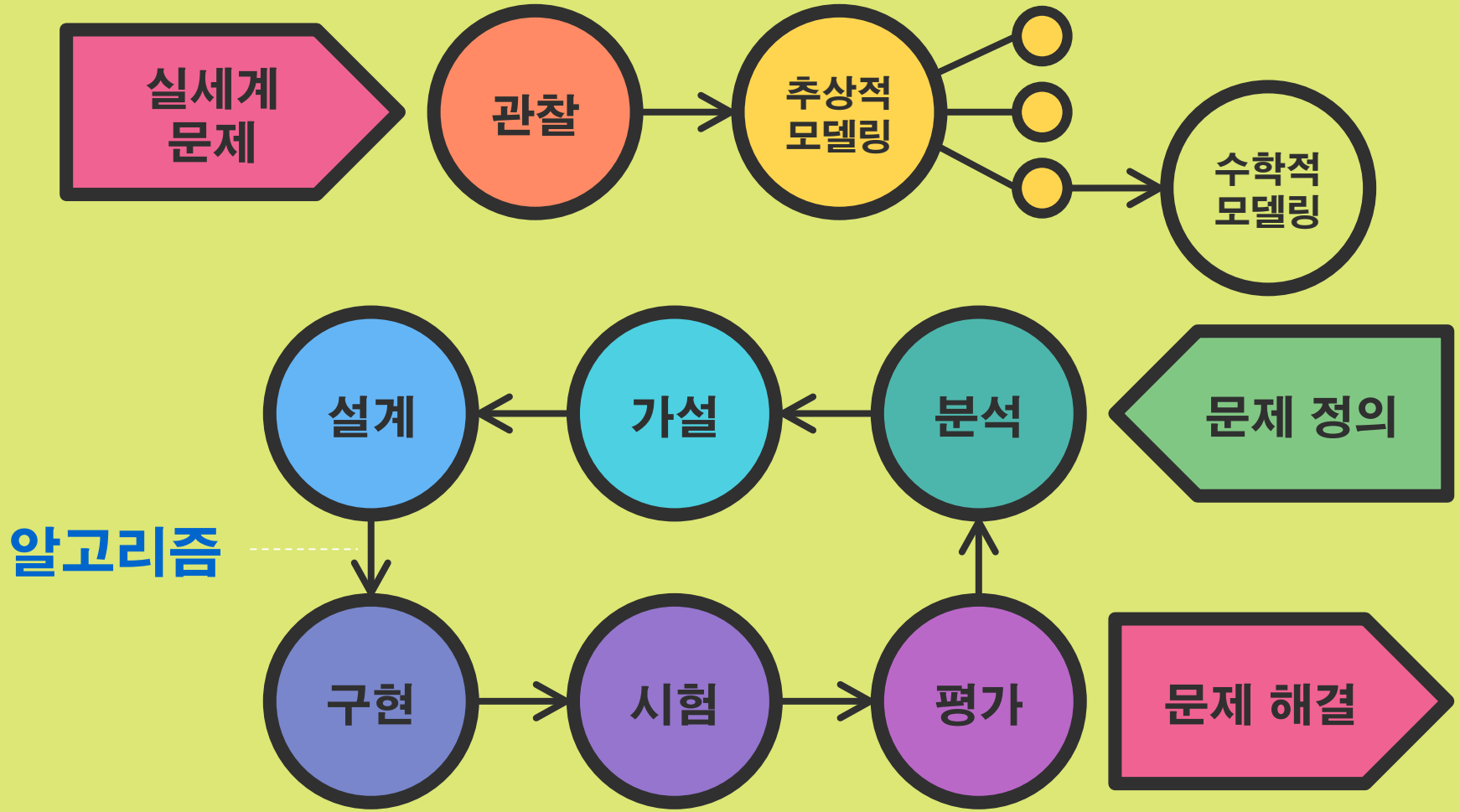
문제를 여러 개로 나누기 *문제 분해*

문제 정의 과정의 단순화

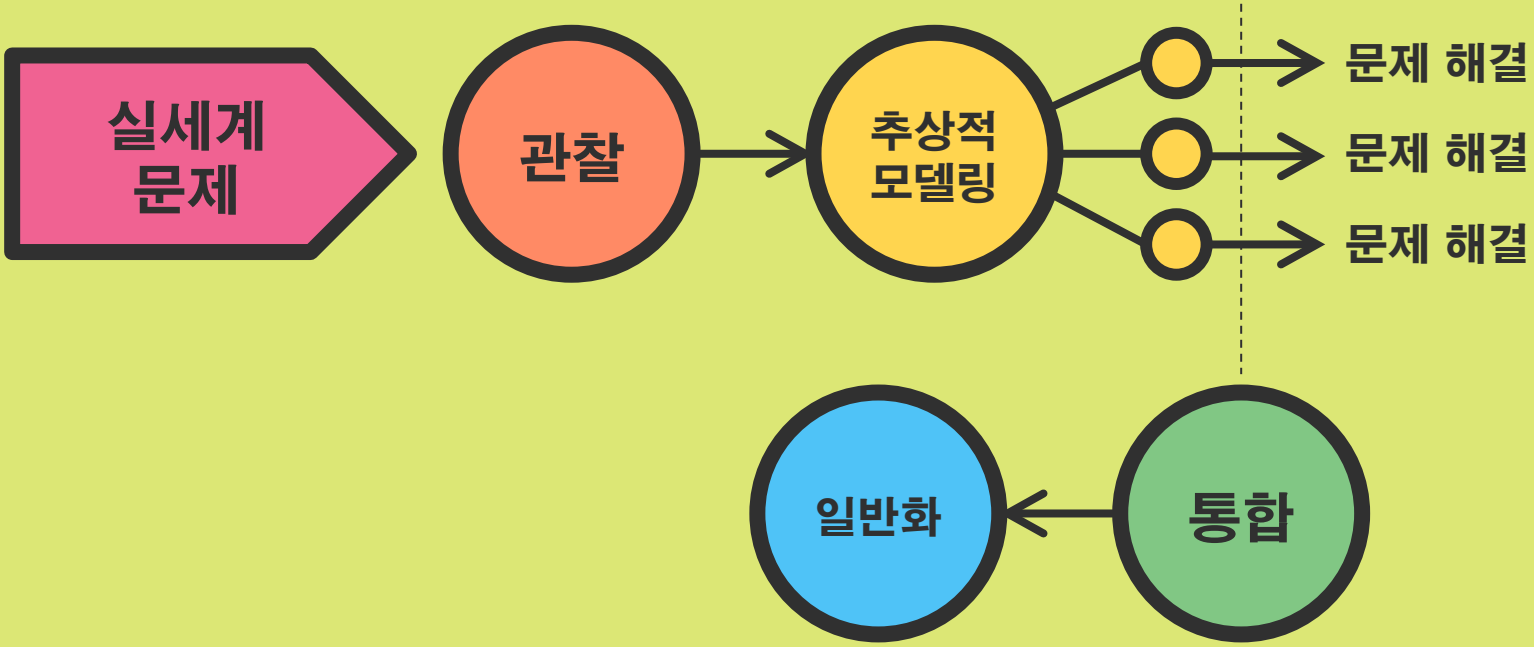
관찰 및 이해

명확한 표현

문제 해결 과정



문제 해결 과정



문제 해결 과정의 단순화

1 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

문제 해결 과정의 단순화

2 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)

문제 해결 과정의 단순화

생각 후 구현

생각, 생각, 생각

컴퓨팅 사고의 목적은

비판적 사고력

논리적 사고력

컴퓨터의 계산 능력을 활용한

문제 해결 과정

왜 SW 교육을 하는가?

의도적 학습

즉각적인 피드백이 중요

SW는 즉각적인 피드백이 가능

관찰과 생각 없는 즉각적인 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 포레스터

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

4. **~까지 반복**
   단순 반복
   규칙(패턴) 있는 반복
   순차 반복
   규칙(패턴) 있는 순차 반복, 내포 반복*

*고급 과정

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***

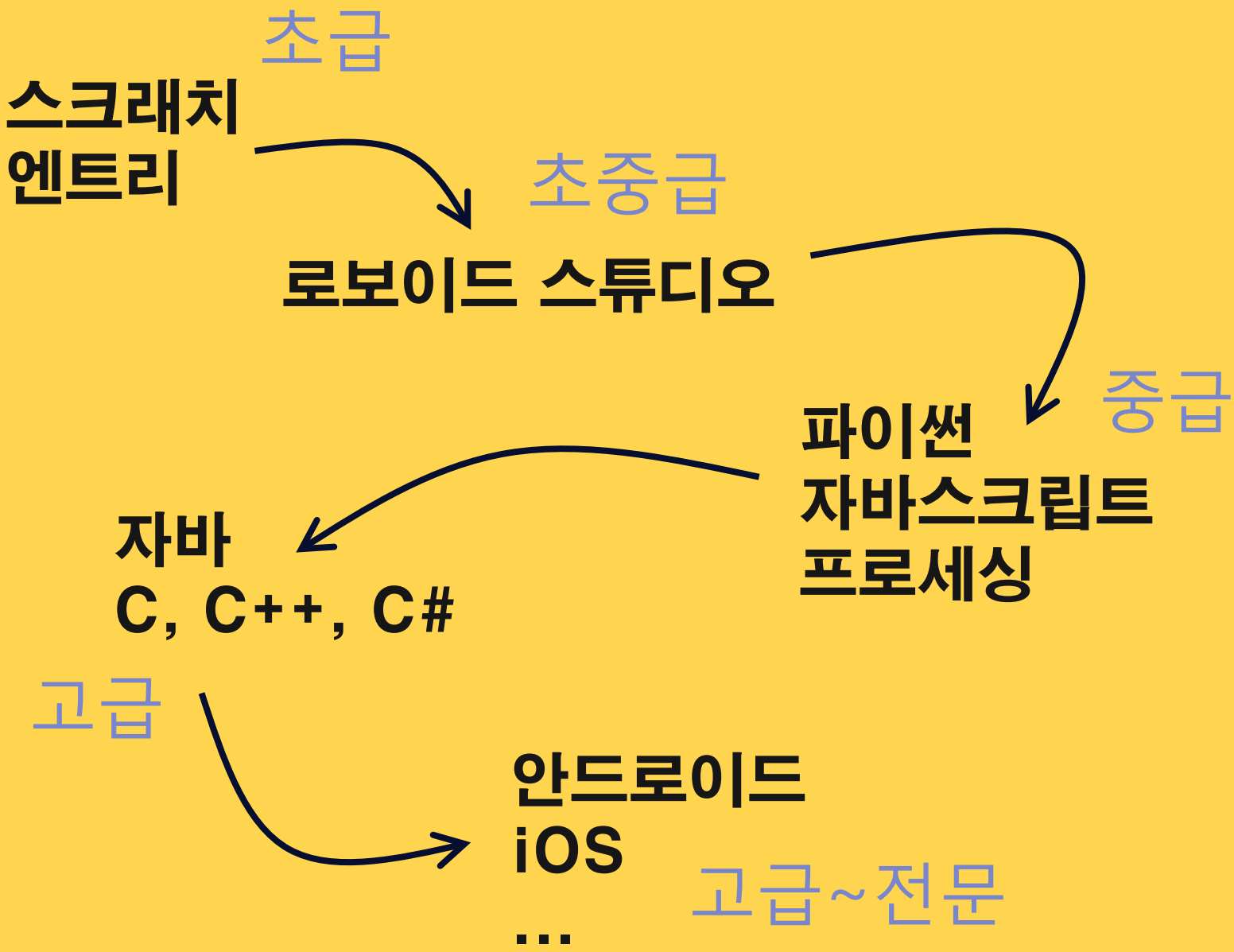
패턴 찾기가 중요

반복

추상화

함수

학습 도구에 관하여...


초급
스크래치
엔트리
초중급
로보이드 스튜디오
중급
파이썬
자바스크립트
프로세싱
자바
C, C++, C#
고급
안드로이드
iOS
...
고급~전문

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
- 또는 텍스트 코딩으로 넘어가서

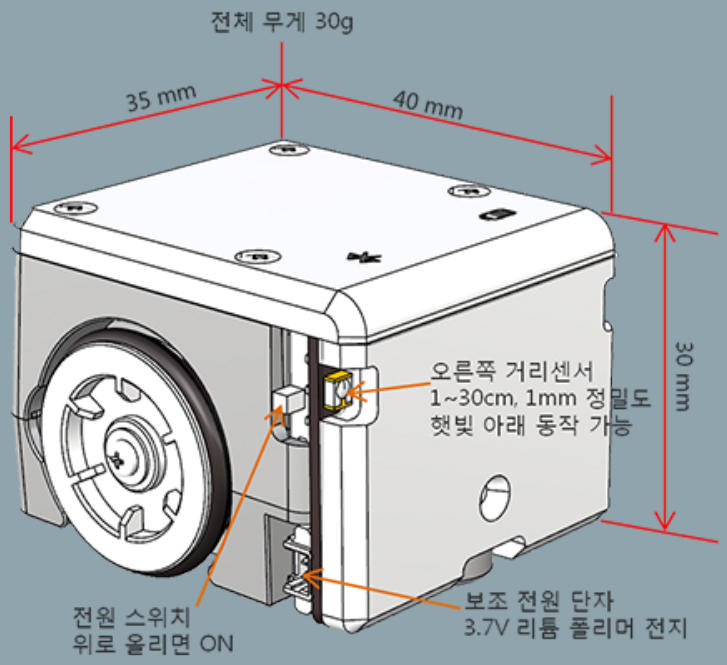
생각의 범위를 넓혀 주어야 함

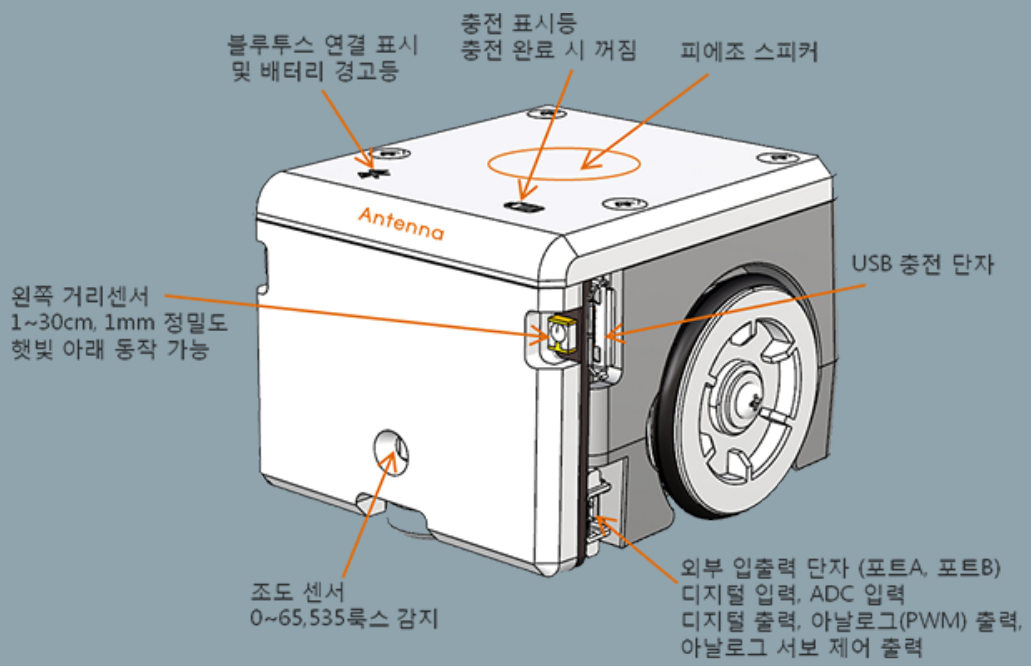
로봇활용 SW교육에서 고려해야 하는 사항들

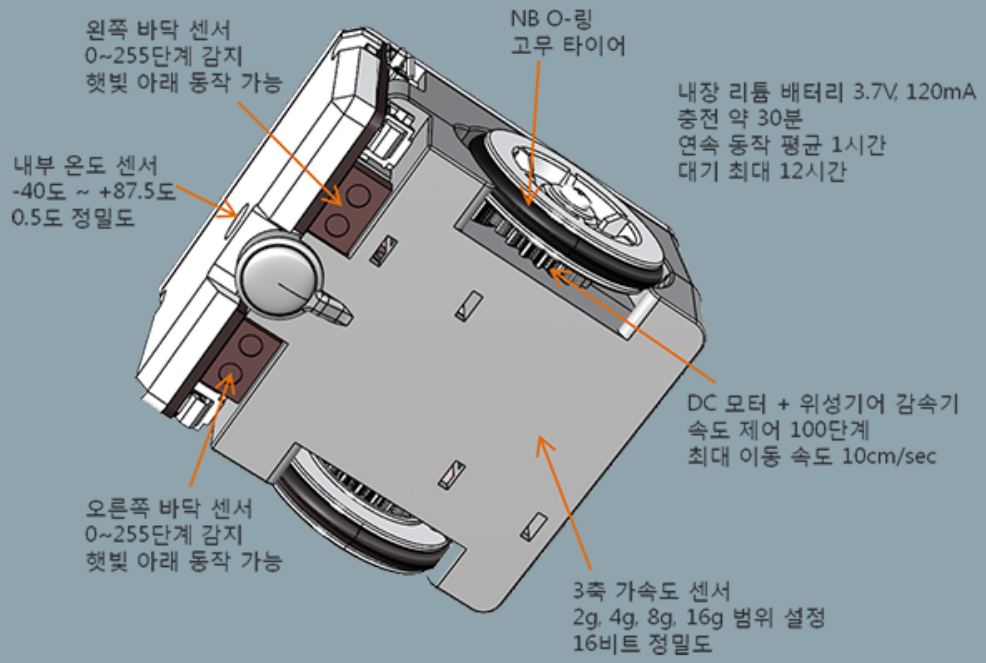
- 동력전달 메커니즘, 센서와 액추에이터 구동 원리 **X**
  → 로봇 교육을 하는 것이 아님
- HW 창작은 최대한 배제
  → 키트 제품은 수업 전에 미리 조립해 둘 것
- 모든 센서 값을 실시간 모니터링할 수 있는 제품을 선택 (가능한 무선으로) → 데이터 관찰이 중요
- 활용할 수 있는 SW 도구 및 추후 개발 계획 확인

햄스터 로봇 활용

햄스터 HW 구성


전체 무게 30g
35 mm
40 mm
30 mm
오른쪽 거리센서
1~30cm, 1mm 정밀도
햇빛 아래 동작 가능
전원 스위치
위로 올리면 ON
보조 전원 단자
3.7V 리튬 폴리머 전지


블루투스 연결 표시
및 배터리 경고등
충전 표시등
충전 완료 시 꺼짐
피에조 스피커
Antenna
USB 충전 단자
왼쪽 거리센서
1~30cm, 1mm 정밀도
햇빛 아래 동작 가능
조도 센서
0~65,535룩스 감지
외부 입출력 단자 (포트A, 포트B)
디지털 입력, ADC 입력
디지털 출력, 아날로그(PWM) 출력,
아날로그 서보 제어 출력


왼쪽 바닥 센서
0~255단계 감지
햇빛 아래 동작 가능
NB O-링
고무 타이어
내장 리튬 배터리 3.7V, 120mA
충전 약 30분
연속 동작 평균 1시간
대기 최대 12시간
내부 온도 센서
-40도 ~ +87.5도
0.5도 정밀도
DC 모터 + 위성기어 감속기
속도 제어 100단계
최대 이동 속도 10cm/sec
오른쪽 바닥 센서
0~255단계 감지
햇빛 아래 동작 가능
3축 가속도 센서
2g, 4g, 8g, 16g 범위 설정
16비트 정밀도

RobotCoding 설치

1. 스크래치 오프라인 에디터 설치
   https://scratch.mit.edu/scratch2download/

2. RobotCoding 프로그램 설치
   http://hamster.school

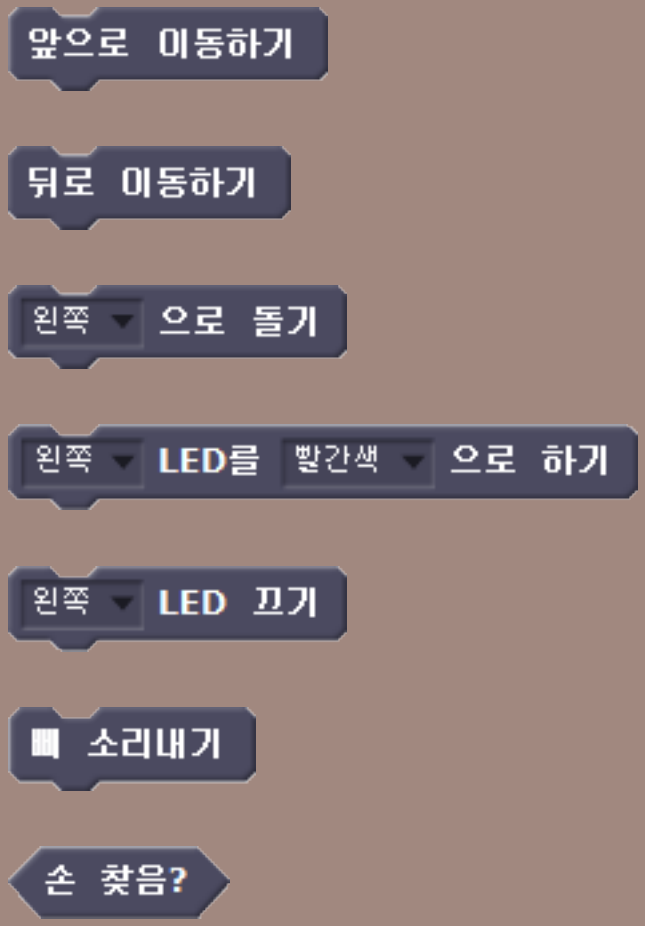
   소프트웨어 탭
   → 스크래치 + 엔트리 + 자바스크립트 내려받기

주의 사항

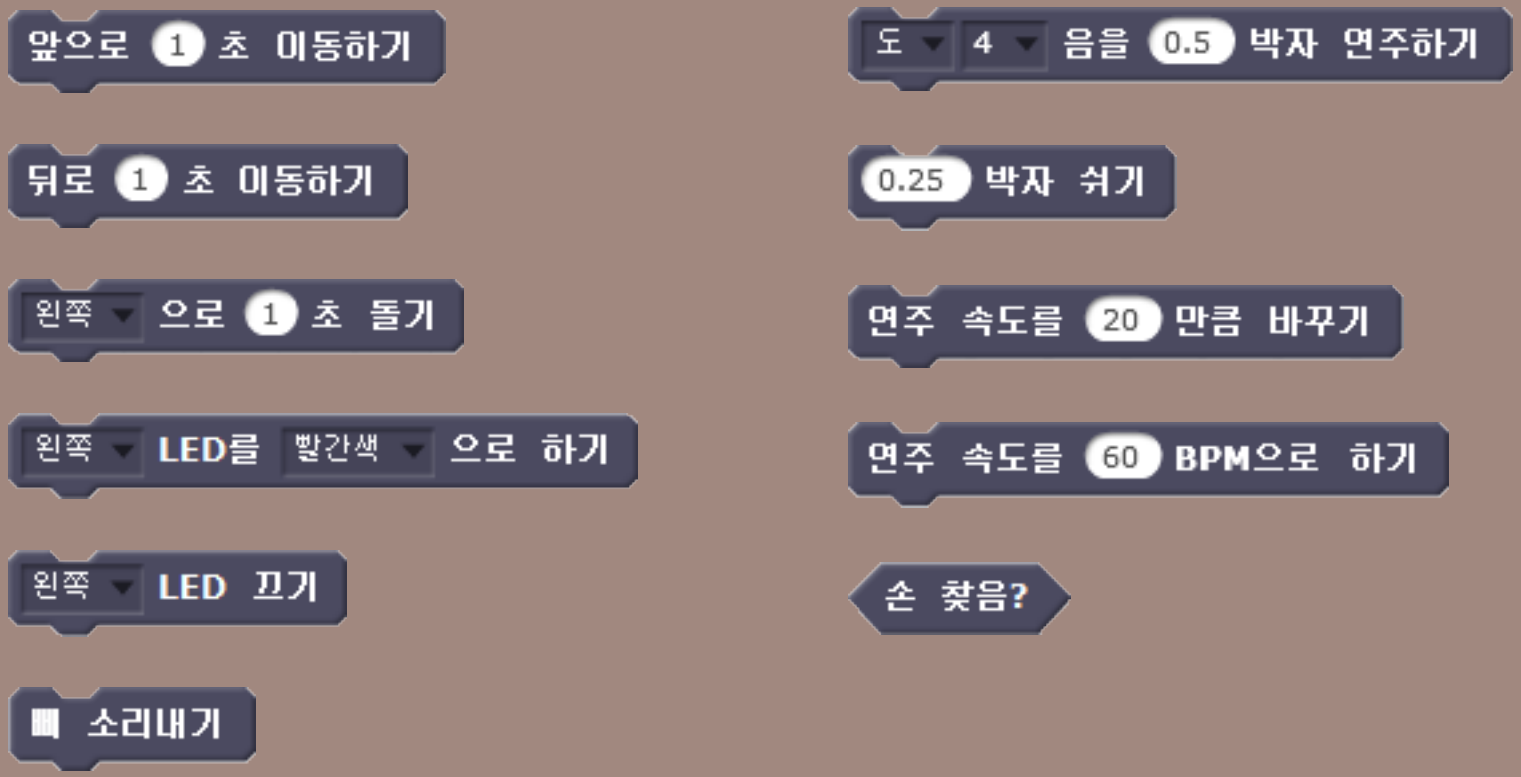
SW 설치가 끝난 후에 USB 동글을 PC에 연결하세요

RobotCoding 설치

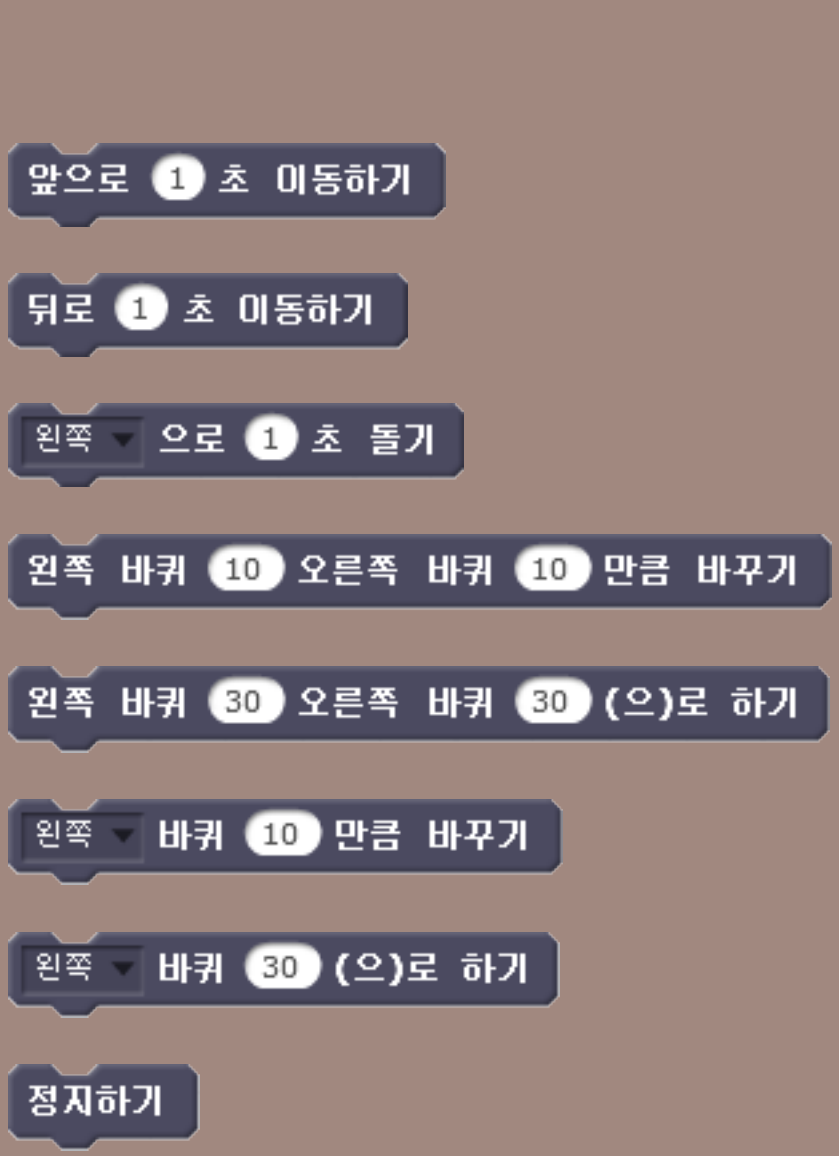
햄스터 스크래치 블록 – 초급

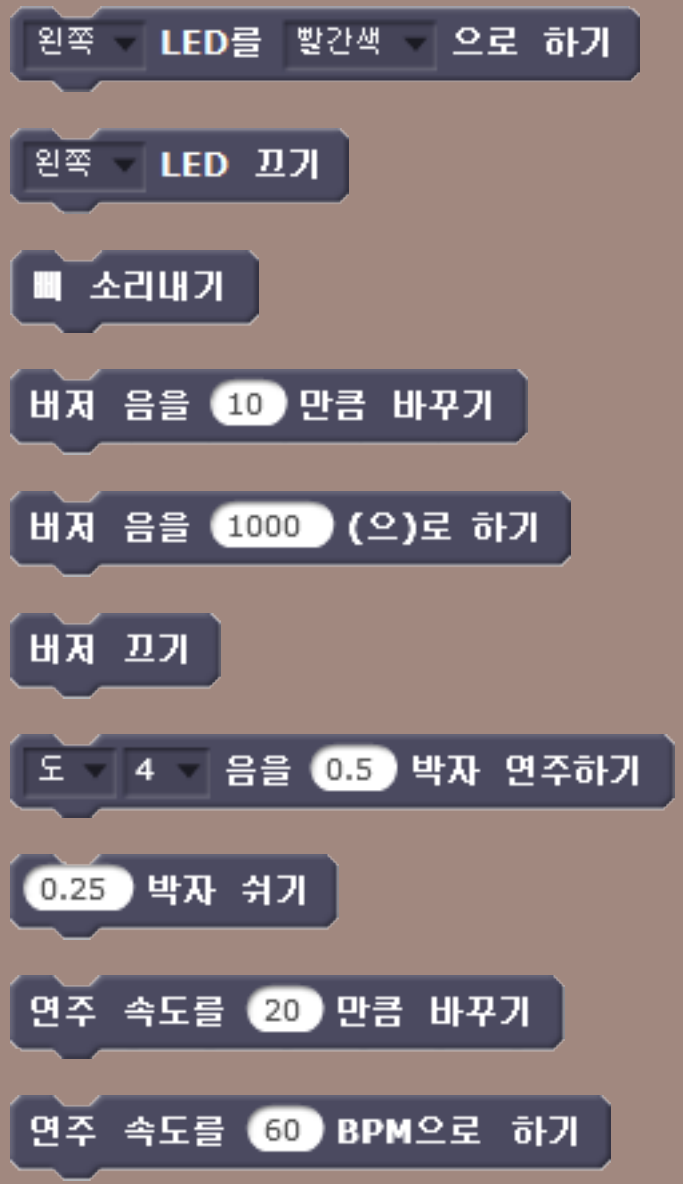


햄스터 스크래치 블록 – 중급

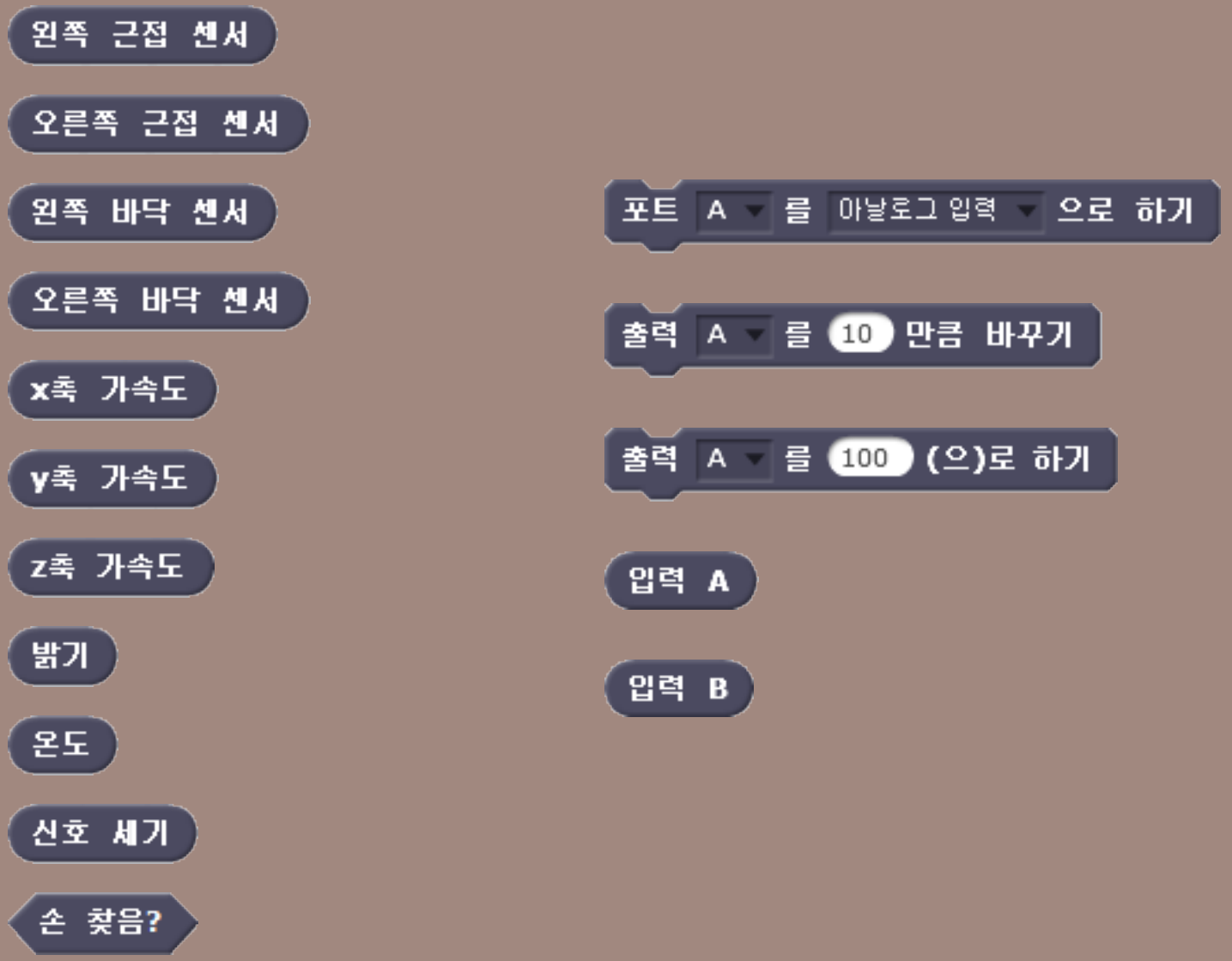


햄스터 스크래치 블록 – 고급

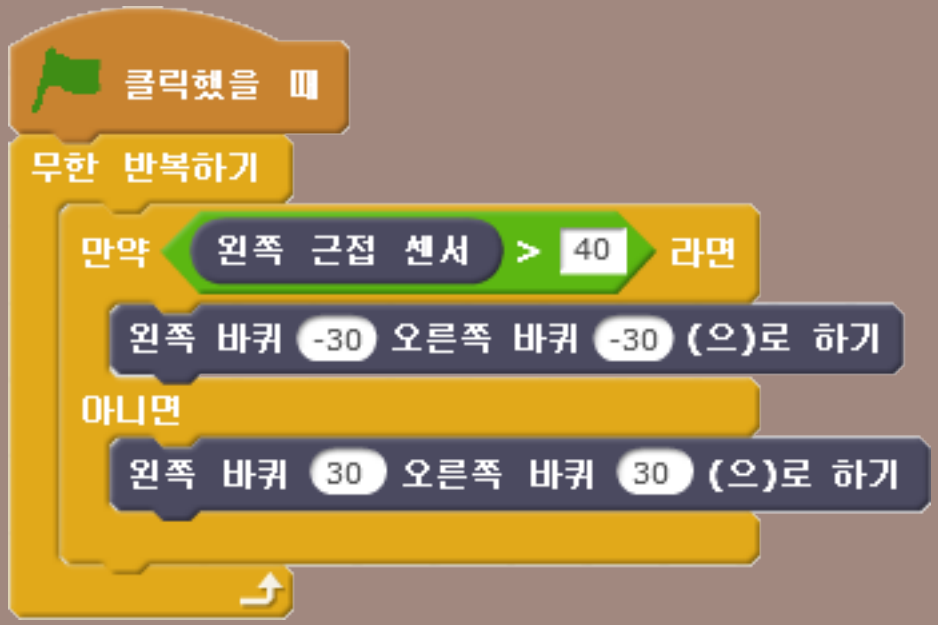




햄스터 스크래치 블록 – 고급



실습 1: 장애물과 거리 유지하기



햄스터 C언어 설치

1. Dev-C++ 설치
   http://sourceforge.net/projects/orwelldevcpp/

2. 햄스터용 C라이브러리 내려 받기
   http://hamster.school

   소프트웨어 탭
   → C언어 내려 받기

햄스터 C언어 설치


Hamster School
www.hamster.school/ko/software/
Hamster
새 소식
하드웨어
소프트웨어
교육 자료
레퍼런스
구입 및 A/S
디바이스 드라이버
디바이스 드라이버 설치
그래픽 언어
스크래치 + 엔트리 + 자바스크립트
로보이드 스튜디오
스크립트 언어
프로세싱
파이썬
고급 언어
C 언어
C++
자바
내려 받기
디바이스 드라이버
디바이스 드라이버
내려 받기
소프트웨어
예상 시기
세부 사항
스크래치 + 엔트리 + 자바스크립트
윈도우 OS 완료
내려 받기
로보이드 스튜디오
윈도우 OS 완료
내려 받기
프로세싱
1단계 완료
내려 받기
파이썬
1단계 완료
내려 받기
Node.js
2015년 10월 30일
C
윈도우 OS 완료
내려 받기
C++
1단계 완료
내려 받기
C#
???
자바
1단계 완료
내려 받기
안드로이드
1단계 완료
11월 말 배포
아이폰
2015년 말
MatLab
???
ROS
???

햄스터 C언어 – 기본 함수들

```
#include “hamster.h”
```

- 통신 연결
  `void hamster_create(void)`
- 모든 연결 해제
  `void hamster_dispose_all(void)`
- 1000분의 1초 기다리기
  `void hamster_wait(int milliseconds)`
- 디바이스 값 읽기
  `int hamster_read(int device_id)`
- 디바이스 값 쓰기
  `int hamster_write(int device_id, int data)`
- 디바이스 값 초기화
  `void hamster_reset(void)`

햄스터 C언어 – 디바이스 ID 상수 값

- HAMSTER_LEFT_WHEEL -100 ~ 100 %
- HAMSTER_RIGHT_WHEEL -100 ~ 100 %
- HAMSTER_BUZZER 0 ~ 167772.15 Hz
- HAMSTER_LEFT_LED 0 ~ 7
- HAMSTER_RIGHT_LED 0 ~ 7
- HAMSTER_NOTE 0 ~ 88

- HAMSTER_LEFT_PROXIMITY 0 ~ 255
- HAMSTER_RIGHT_PROXIMITY 0 ~ 255
- HAMSTER_LEFT_FLOOR 0 ~ 255
- HAMSTER_RIGHT_FLOOR 0 ~ 255
- HAMSTER_ACCELERATION -32768 ~ 32767
- HAMSTER_LIGHT 0 ~ 65535
- HAMSTER_TEMPERATURE -40 ~ 88 ºC

실습 2: 앞뒤로 움직이고 정지

```
#include “hamster.h”

int main(int argc, char *argv[]) {
        hamster_create(); // 통신 연결

        hamster_write(HAMSTER_LEFT_WHEEL, 50); // 전진
        hamster_write(HAMSTER_RIGHT_WHEEL, 50); // 전진
        hamster_wait(500); // 0.5초 쉬기

        hamster_write(HAMSTER_LEFT_WHEEL, -50); // 후진
        hamster_write(HAMSTER_RIGHT_WHEEL, -50); // 후진
        hamster_wait(500); // 0.5초 쉬기

        hamster_write(HAMSTER_LEFT_WHEEL, 0); // 정지
        hamster_write(HAMSTER_RIGHT_WHEEL, 0); // 정지

        hamster_dispose_all(); // 모든 연결 해제
        return 0;
}
```

실습 3: LED 켜고 끄기

HAMSTER_LED_OFF
HAMSTER_LED_RED
HAMSTER_LED_GREEN
HAMSTER_LED_BLUE
HAMSTER_LED_CYAN
HAMSTER_LED_MAGENTA
HAMSTER_LED_YELLOW
HAMSTER_LED_WHITE

```
#include “hamster.h”

int main(int argc, char *argv[]) {
        hamster_create(); // 통신 연결

        // LED 빨간색, 초록색으로 켜기
        hamster_write(HAMSTER_LEFT_LED, HAMSTER_LED_RED);
        hamster_write(HAMSTER_RIGHT_LED, HAMSTER_LED_GREEN);
        hamster_wait(1000); // 1초 쉬기

        // LED 끄기
        hamster_write(HAMSTER_LEFT_LED, HAMSTER_LED_OFF);
        hamster_write(HAMSTER_RIGHT_LED, HAMSTER_LED_OFF);

        hamster_dispose_all(); // 모든 연결 해제
        return 0;
}
```

실습 4: 버저 소리내기

```
#include “hamster.h”

int main(int argc, char *argv[]) {
	int buzzer = 500, i;
	hamster_create(); // 통신 연결

	for(i = 0; i < 200; ++i) {
		buzzer += 10;
		if(buzzer > 700) buzzer = 500;
		hamster_write(HAMSTER_BUZZER, buzzer);
		hamster_wait(20); // 0.02초 쉬기
	}

	hamster_dispose_all(); // 모든 연결 해제
	return 0;
}
```

실습 5: 손 대면 멈추기

```
#include “hamster.h”

int main(int argc, char *argv[]) {
	hamster_create(); // 통신 연결

	hamster_write(HAMSTER_LEFT_WHEEL, 40); // 전진
	hamster_write(HAMSTER_RIGHT_WHEEL, 40); // 전진

	while(hamster_read(HAMSTER_LEFT_PROXIMITY) < 40) {
		hamster_wait(20);
	}

	hamster_reset(); // 모든 디바이스 값 초기화

	hamster_dispose_all(); // 모든 연결 해제
	return 0;
}
```

미션: 전진하다가 정지선을 만나면 정지하기

HINT

- `HAMSTER_LEFT_FLOOR`
- `HAMSTER_RIGHT_FLOOR`

수고하셨습니다

akaii@kw.ac.kr