

진로체험

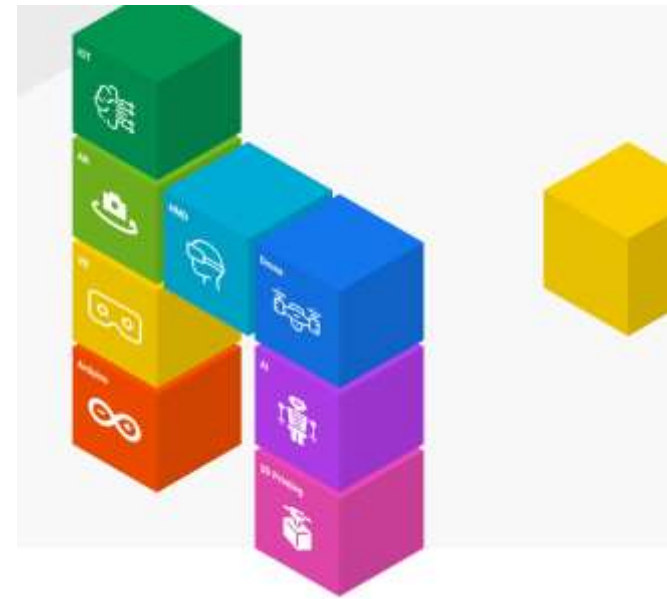
아두이노 시뮬레이션 1차시



www.helloapps.co.kr

070-4417-1559 / splduino@gmail.com

아두이노는 컴퓨터인가?



컴퓨터 vs 마이크로 컨트롤러



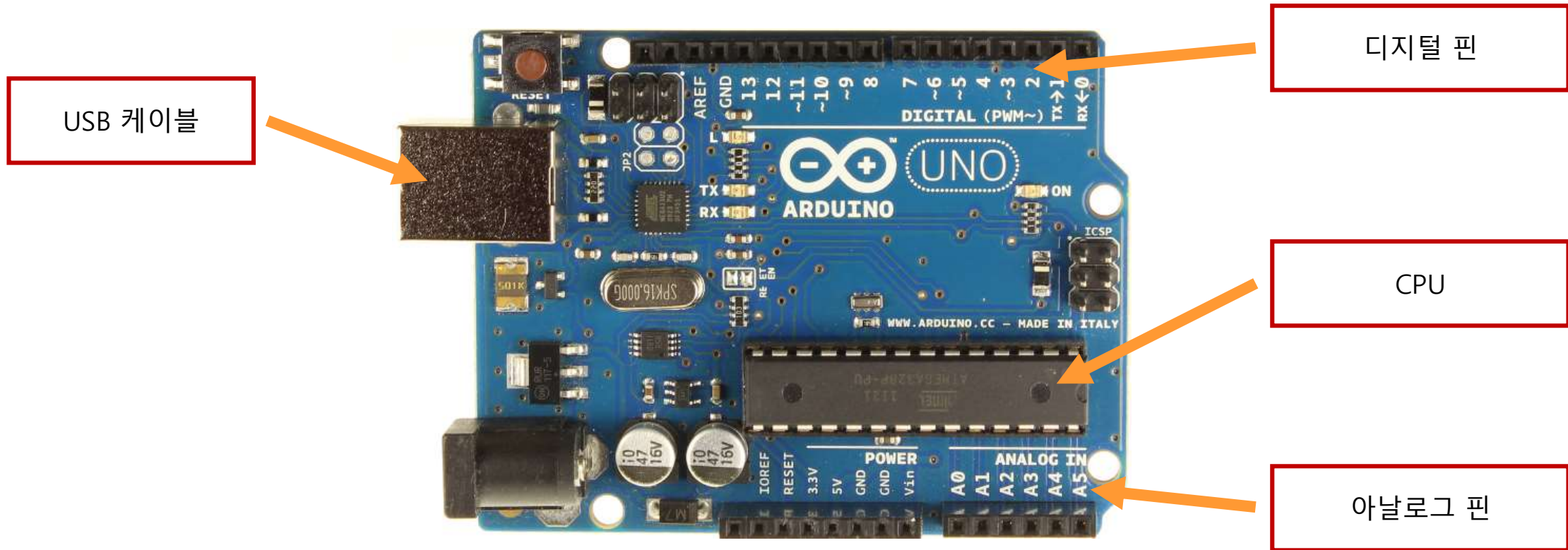
컴퓨터와 마이크로 컨트롤러 (마이컴)은 서로 다른 장치임

컴퓨터 (PC)	마이크로 컨트롤러 (마이컴)
<ul style="list-style-type: none">OS가 부팅된 후, 어플리케이션 SW가 실행됨 	<ul style="list-style-type: none">전원이 연결되자마자 메모리에 있는 SW가 바로 실행됨  <ul style="list-style-type: none">다양한 전자 기기에 내장되는 형태

아두이노 보드의 구성



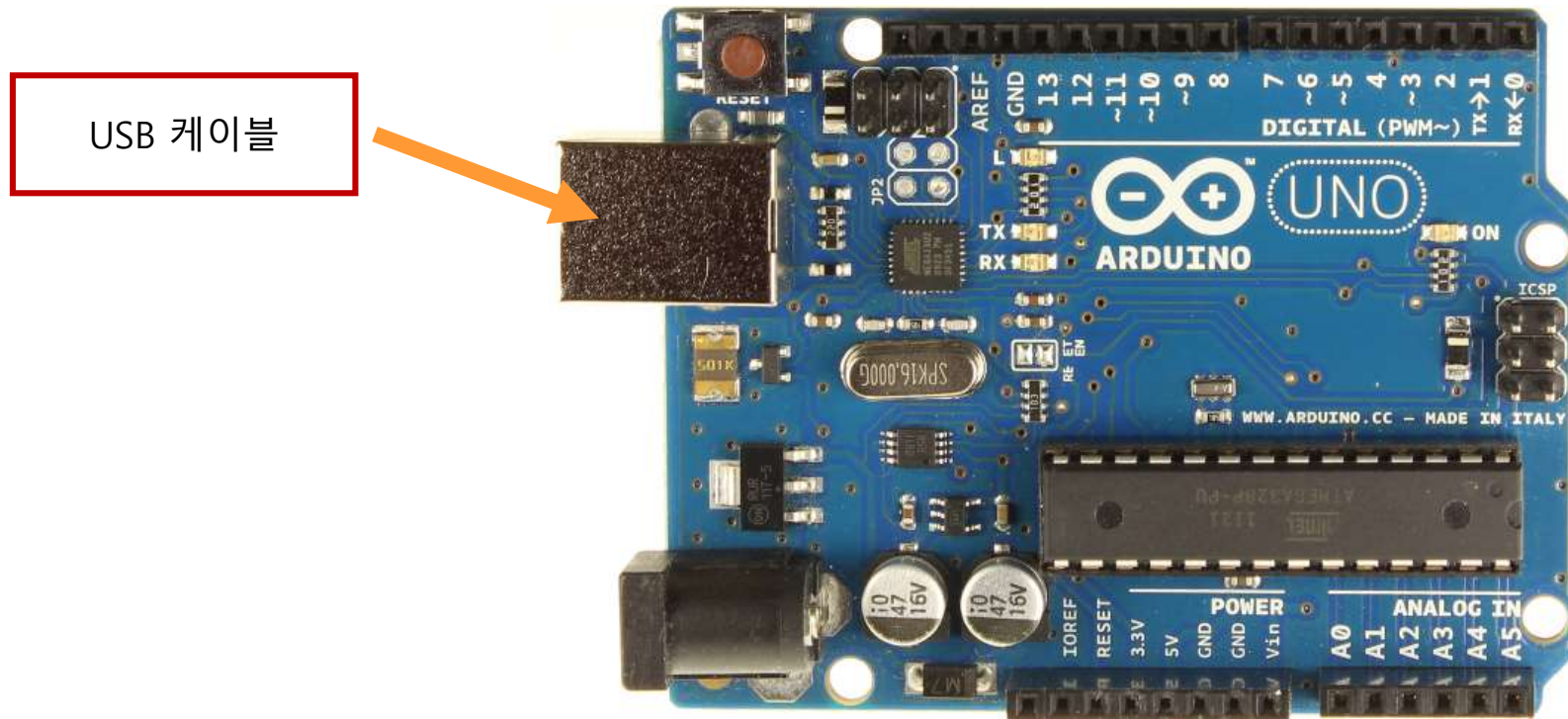
일반적으로 알고 있는 아두이노 우노 보드는 대표적인 레퍼런스 보드 중에 하나임



아두이노 보드의 구성



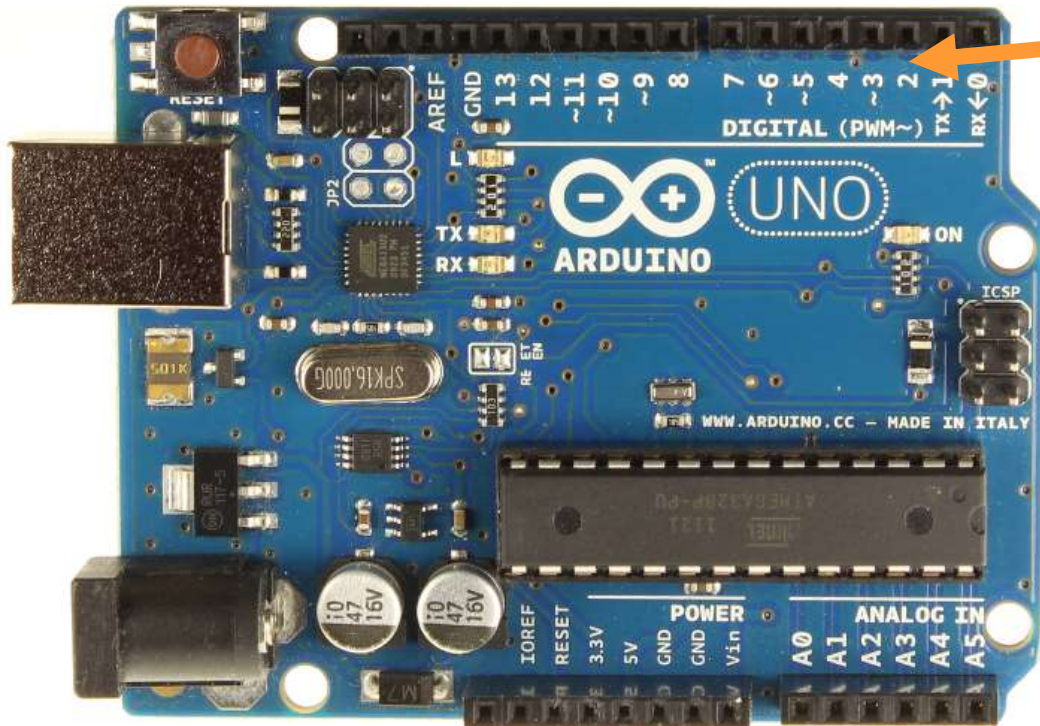
USB 케이블을 통해 프로그램을 업로드 하거나 PC와 데이터를 주고 받음



아두이노 보드의 구성



아두이노의 디지털 명령어를 사용하여 디지털 핀에 연결된 센서의 값을 읽거나 씀



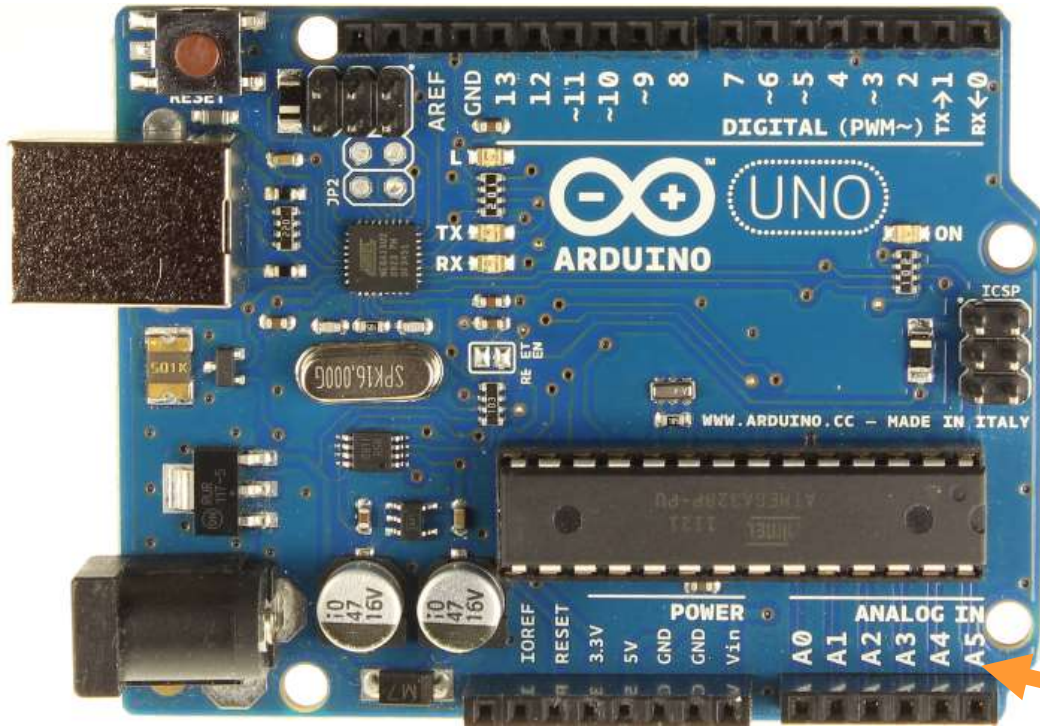
디지털 핀

디지털 읽기
디지털 쓰기
아날로그 쓰기

아두이노 보드의 구성



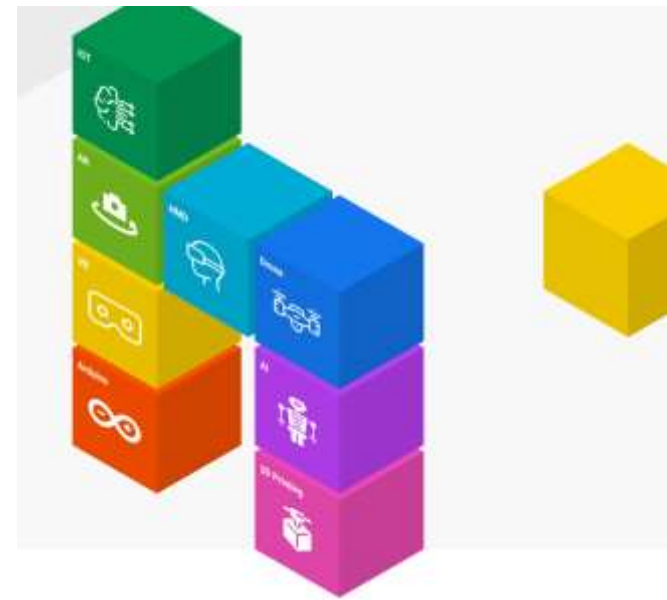
아두이노의 디지털 명령어를 사용하여 디지털 핀에 연결된 센서의 값을 읽거나 씀



아날로그 읽기

아날로그 핀

아두이노 디지털 명령어 기초



아두이노 디지털 명령어

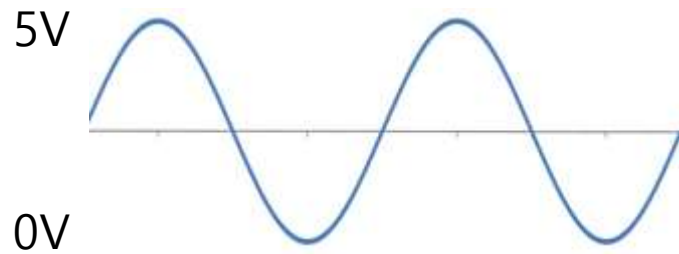
- 디지털 핀에 값을 쓸 때 사용하는 명령어
✓ 디지털 쓰기
- 디지털 핀에서 값을 읽어 올 때 사용하는 명령어
✓ 디지털 읽기



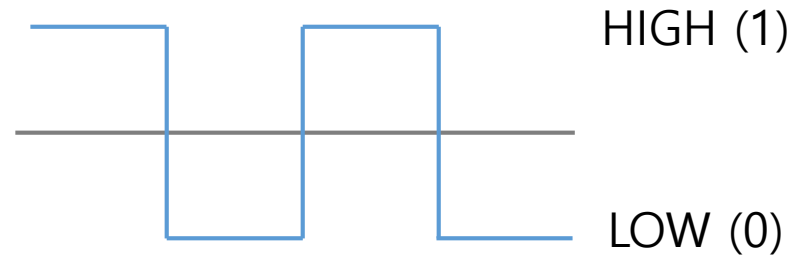
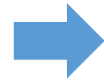
디지털 명령어에서의 값



- 0V와 5V 대신에 LOW와 HIGH라는 단어로 표시하는 이유



디지털 센서의 출력 또는 입력

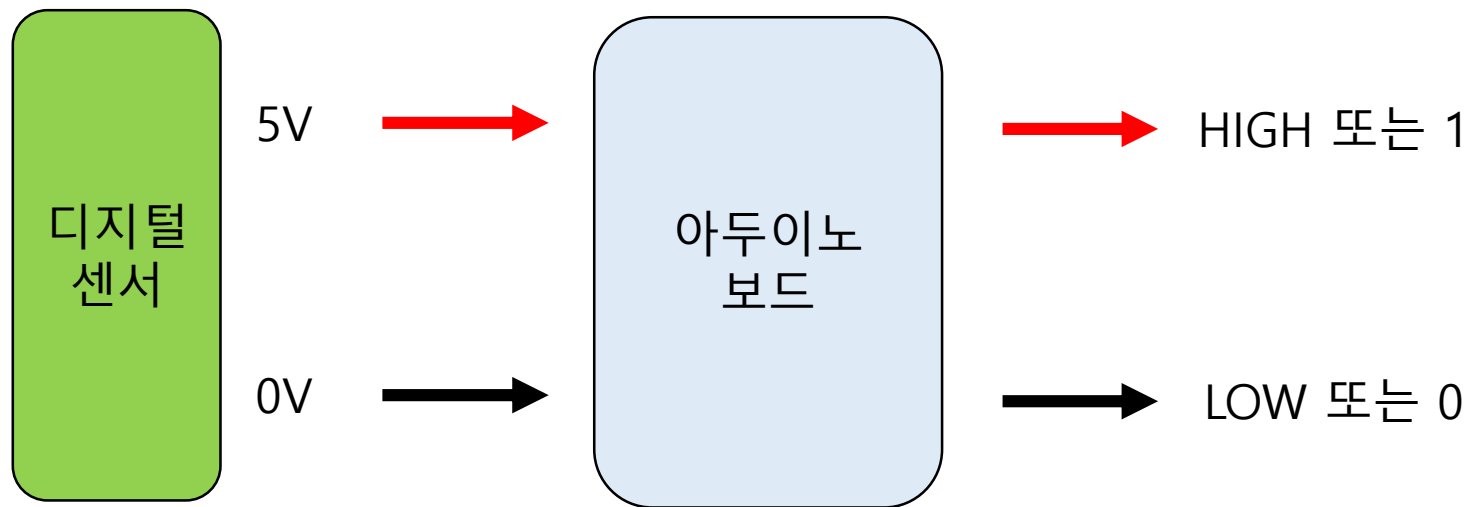


아두이노 보드에서의
디지털 센서값 처리

디지털 명령어에서의 값



- 아두이노 디지털 명령에서는 값이 0 또는 1을 사용하며, 0 대신에 LOW, 1 대신에 HIGH 라는 단어를 사용함



디지털 명령어에서의 값



- 디지털 핀에 값을 쓰는 방법

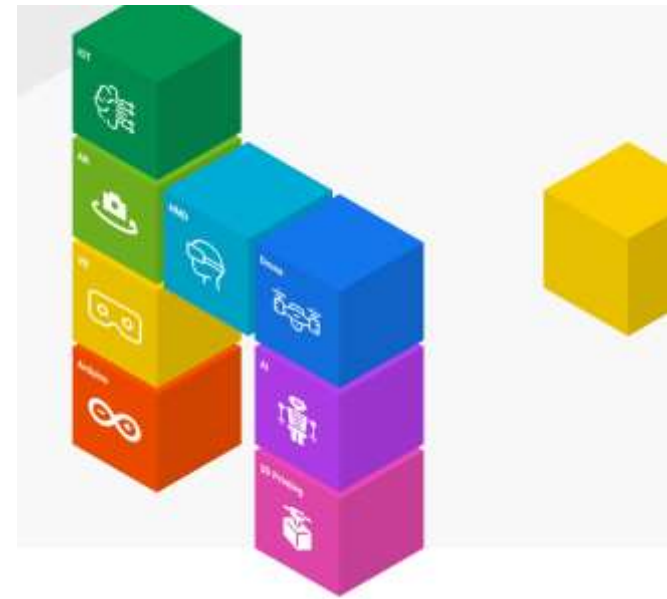
부품을 작동시킬 때 HIGH 라는 단어를 사용합니다.

예) 디지털 쓰기(13, HIGH)

부품 작동을 멈출 때 LOW 라는 단어를 사용합니다.

예) 디지털 쓰기(13, LOW)

코딩으로 LED 제어하기



디지털 명령어에서의 값



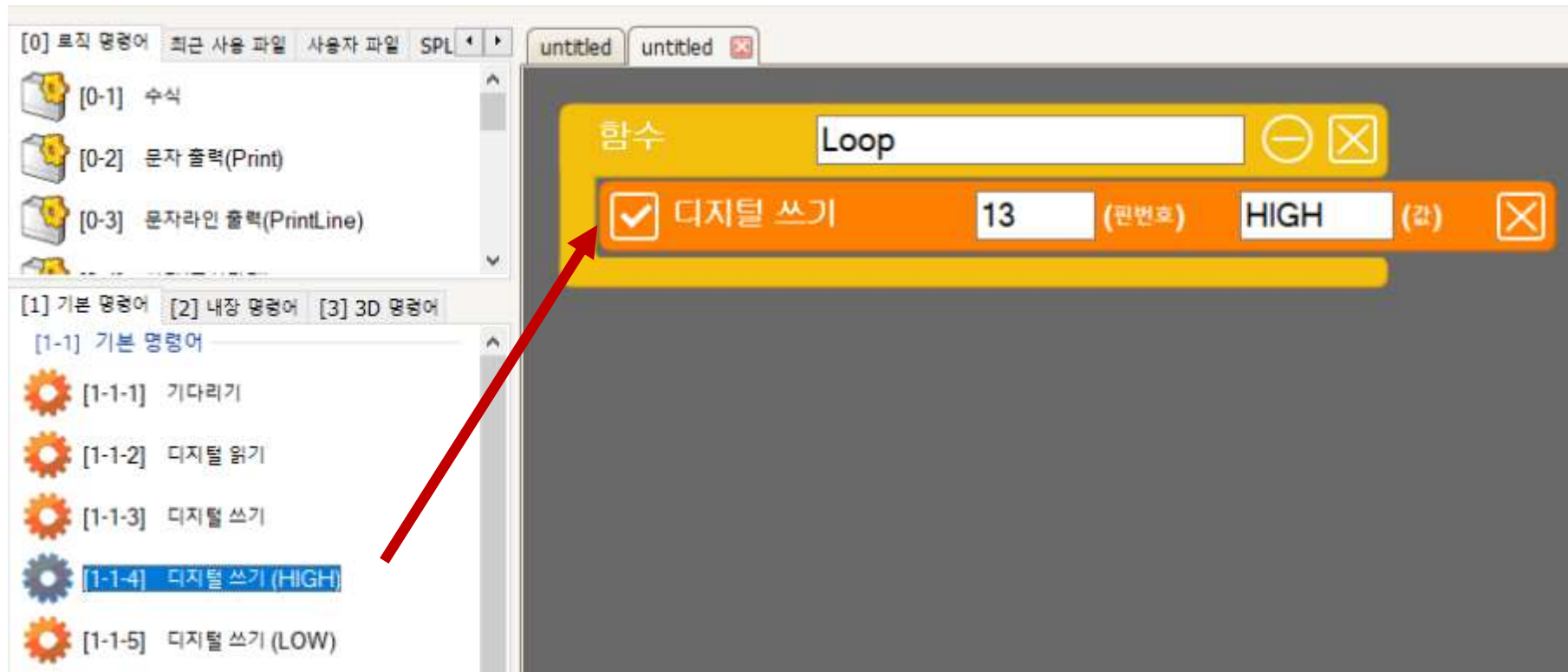
- 시나리오를 아두이노 로봇으로 선택합니다.



디지털 명령어에서의 값



- [1-1-4] 디지털 쓰기(High) 명령어를 Loop 함수 안에 드래그 하여 추가합니다.



디지털 쓰기(High) 명령어의 의미



디지털 쓰기 명령어에 HIGH 값을 자동으로 입력해 주는 명령어

☒ 디지털 쓰기 13 (핀번호) HIGH (값) ☐

HIGH 값이 있으면
LED가 켜짐

HIGH 값은 반드시
대문자로만
입력해야 함

디지털 13번 핀에
HIGH 값을 쓰라는 의미임

정상

HIGH

잘못된 예

High

잘못된 예

high

실행하기



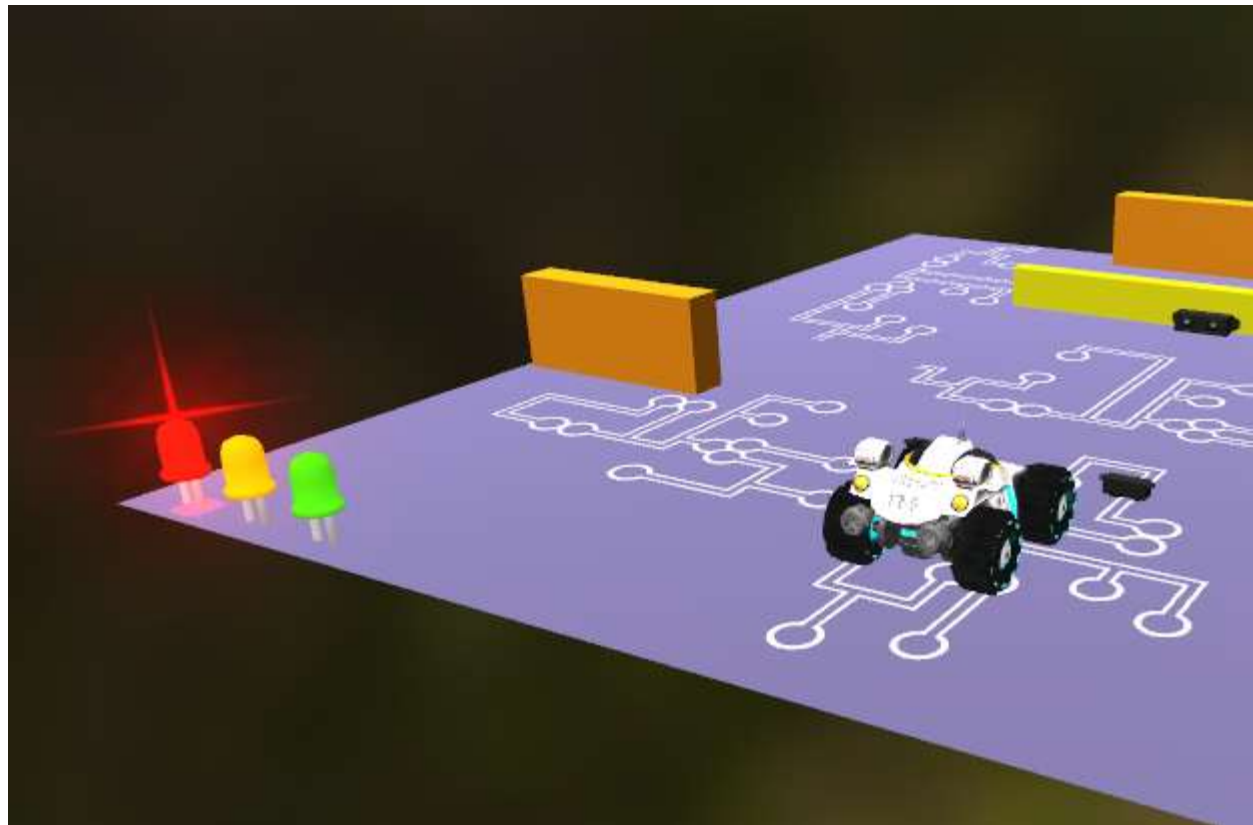
- 실행 버튼을 클릭합니다.



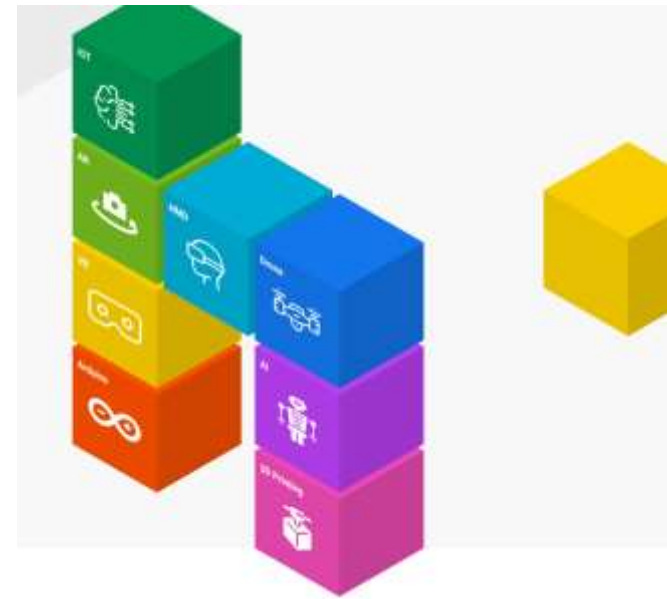
실행하기



RED 컬러 LED가 켜져 있게 됩니다.
마우스로 화면이 이동시킬 수 있습니다.



시뮬레이션 연결 환경 이해하기

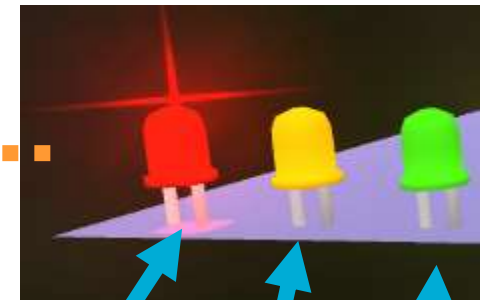
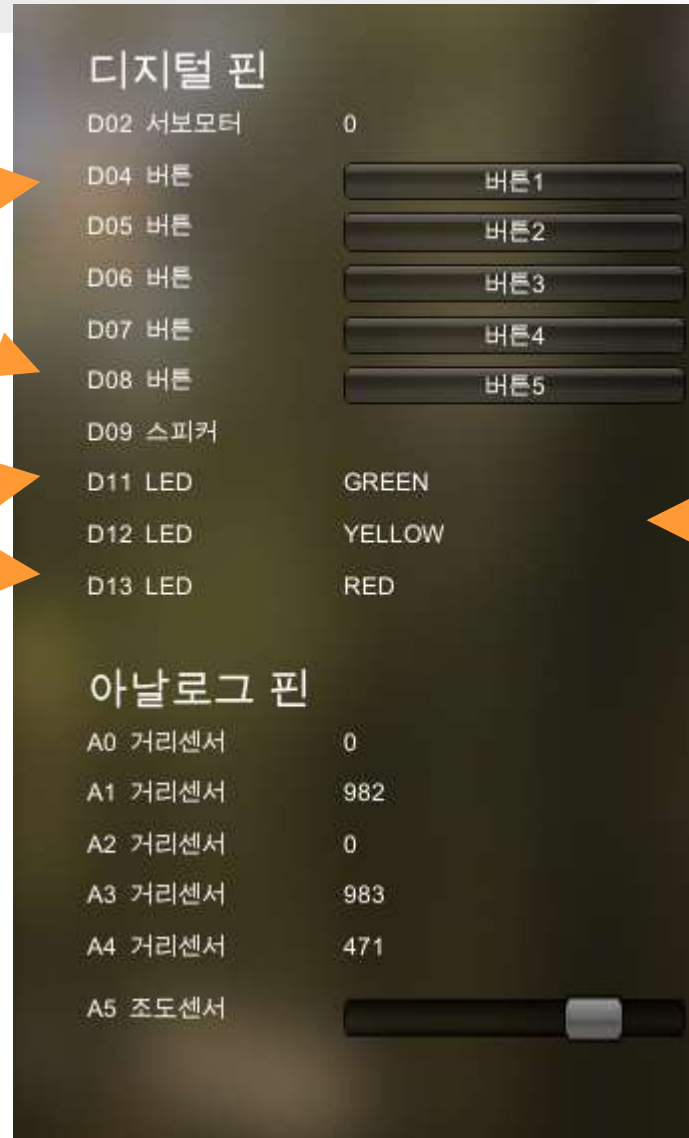


시뮬레이션 연결 환경 이해하기



4번부터 8번까지 5개의
버튼이 연결되어 있음

3개의 LED가
11번 부터 13번까지 연결되어
있음



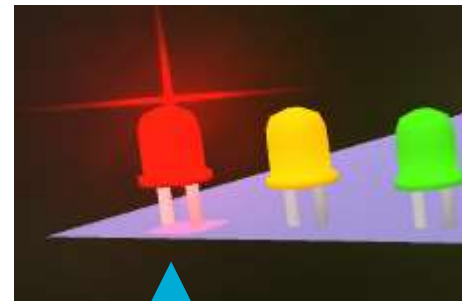
시뮬레이션 연결 환경 이해하기



13번에 연결된 빨간색 LED가 켜진 상태로 표시됩니다.

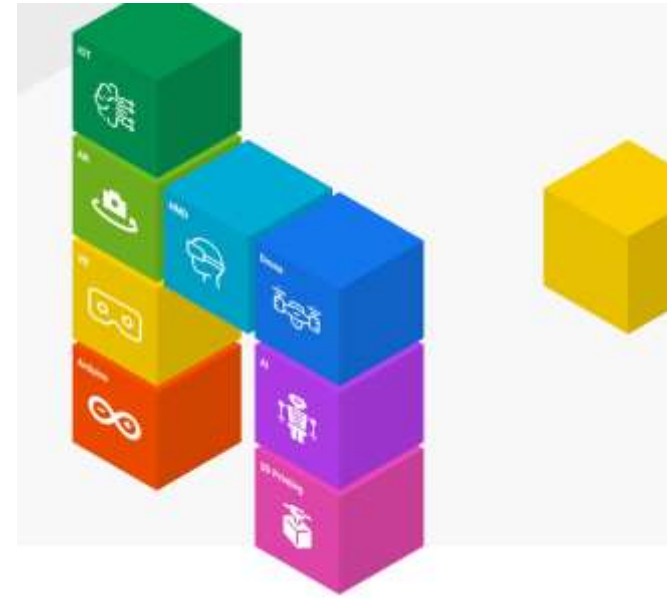


13번

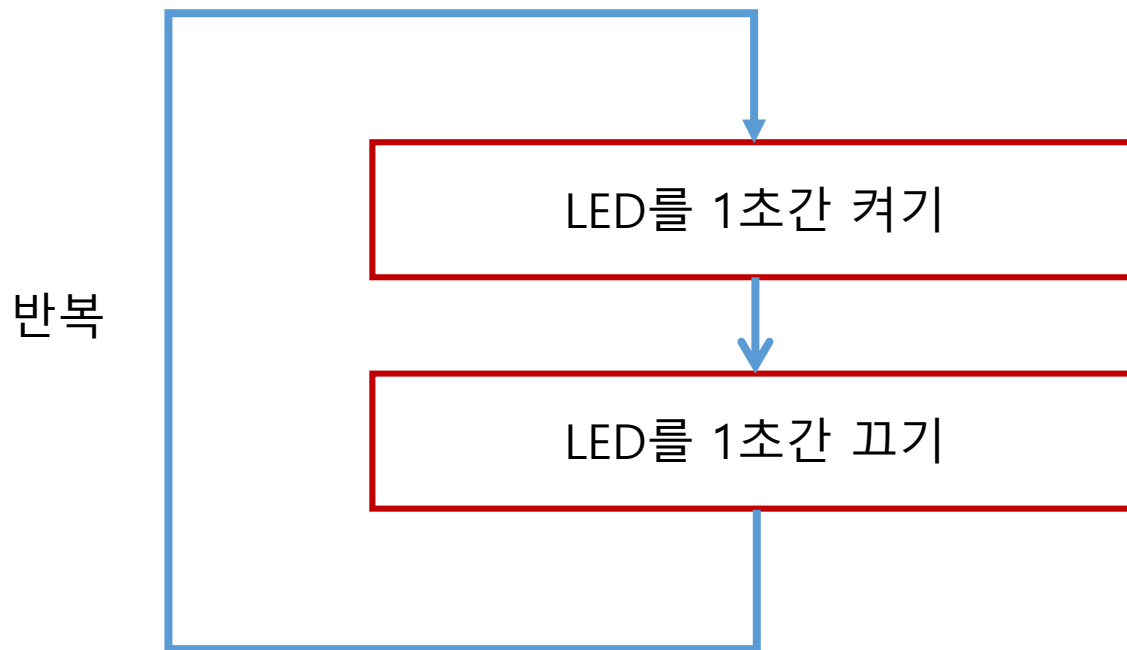


13번

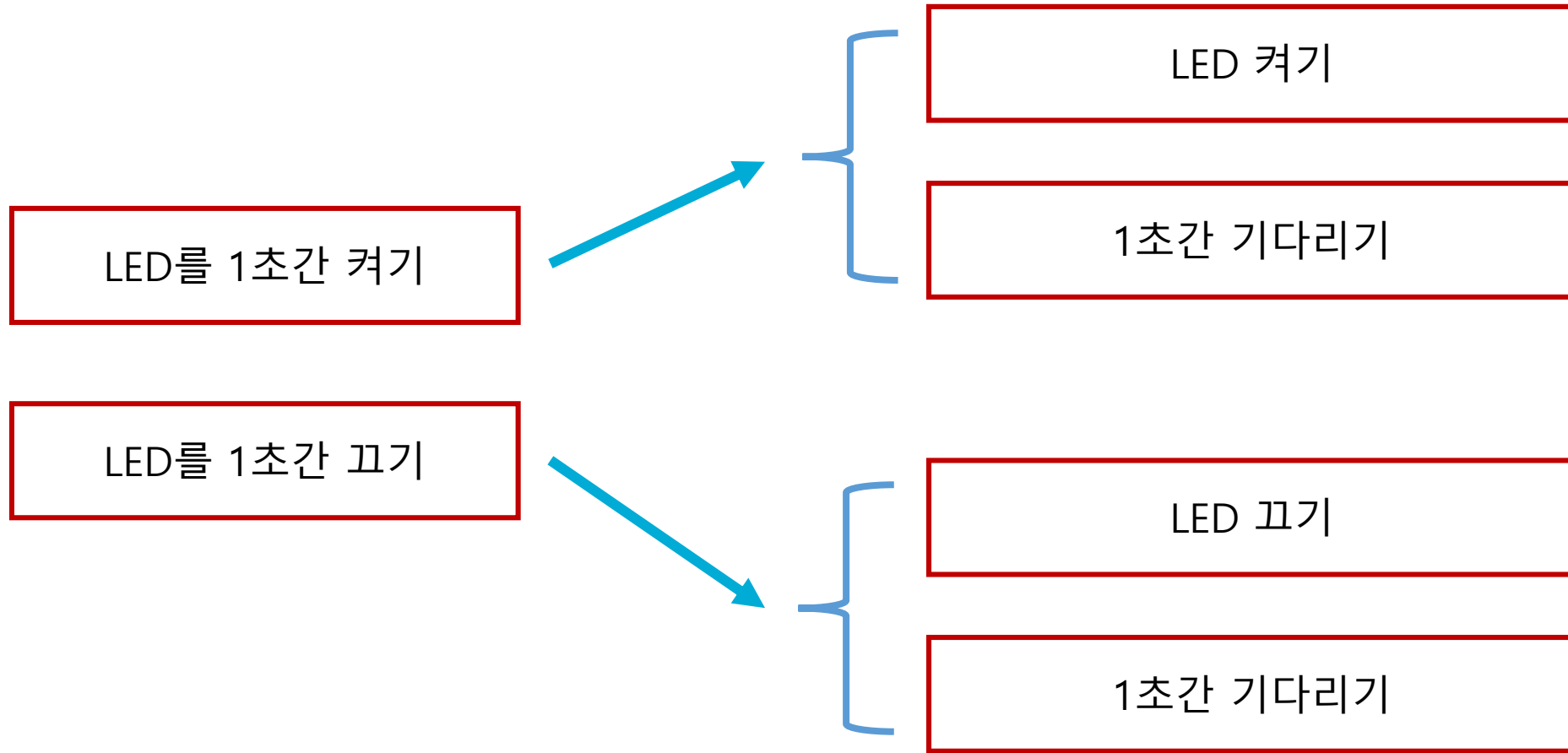
LED를 반복하여 점멸시키기



LED를 1초 간격으로 점멸 시키기



LED를 1초 간격으로 점멸 시키기



LED를 1초 간격으로 점멸 시키기

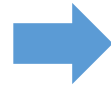


LED 켜기

1초간 기다리기

LED 끄기

1초간 기다리기



디지털 쓰기(13, HIGH)

기다리기(1000)

디지털 쓰기(13, LOW)

기다리기(1000)

LED를 1초 간격으로 점멸 시키기



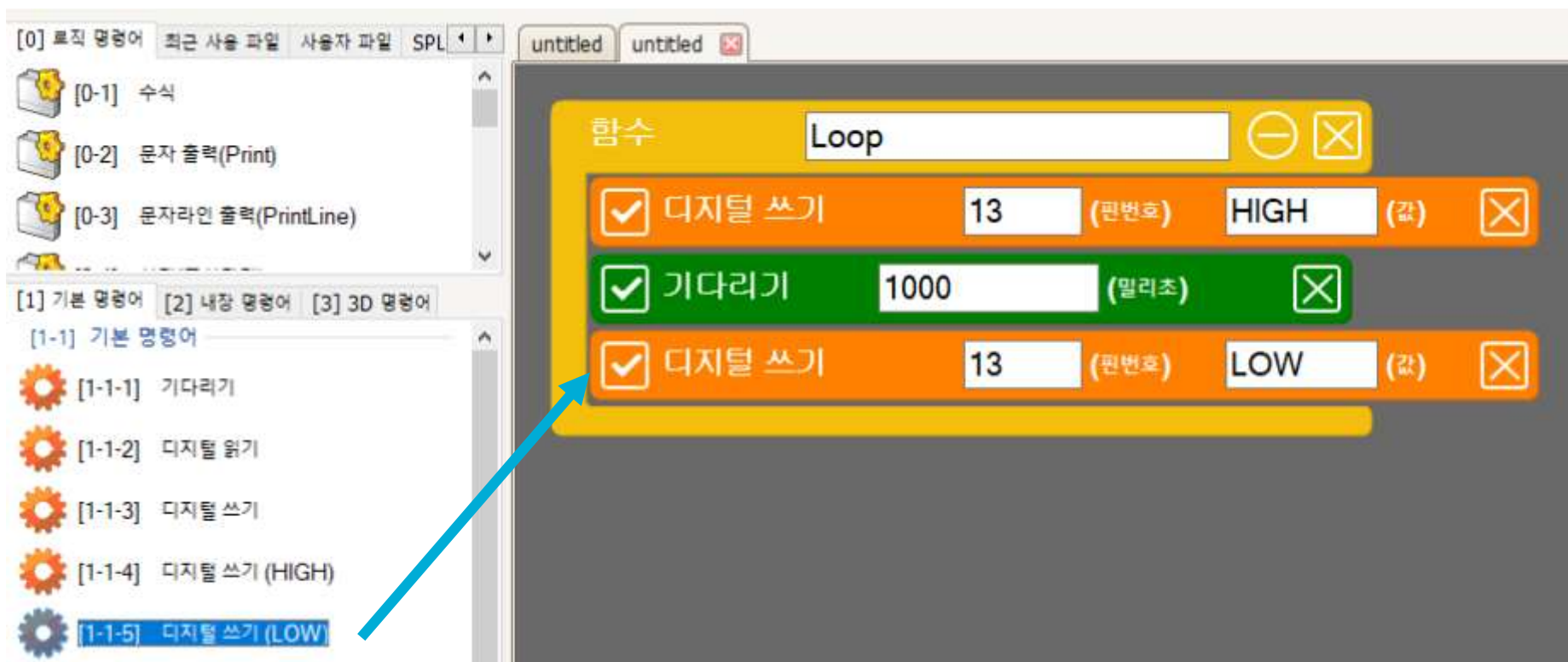
기다리기 명령어를 추가합니다.

The screenshot shows the Hello Apps software interface. On the left is a command palette with categories: [0] 로직 명령어, [1] 기본 명령어, [2] 내장 명령어, and [3] 3D 명령어. Under [1-1] 기본 명령어, the '기다리기' (Delay) command is highlighted with a blue arrow. The main workspace shows a yellow '함수' (Function) block named 'Loop'. Inside the loop, there are two orange blocks: '디지털 쓰기' (Digital Write) with pin 13 and HIGH level, and a green '기다리기' (Delay) block with a value of 1000 milliseconds. A blue arrow points from the '기다리기' command in the palette to the '기다리기' block in the workspace.

명령어 복사하기



LED를 끄기 위해 디지털 쓰기(LOW) 명령어를 추가합니다.

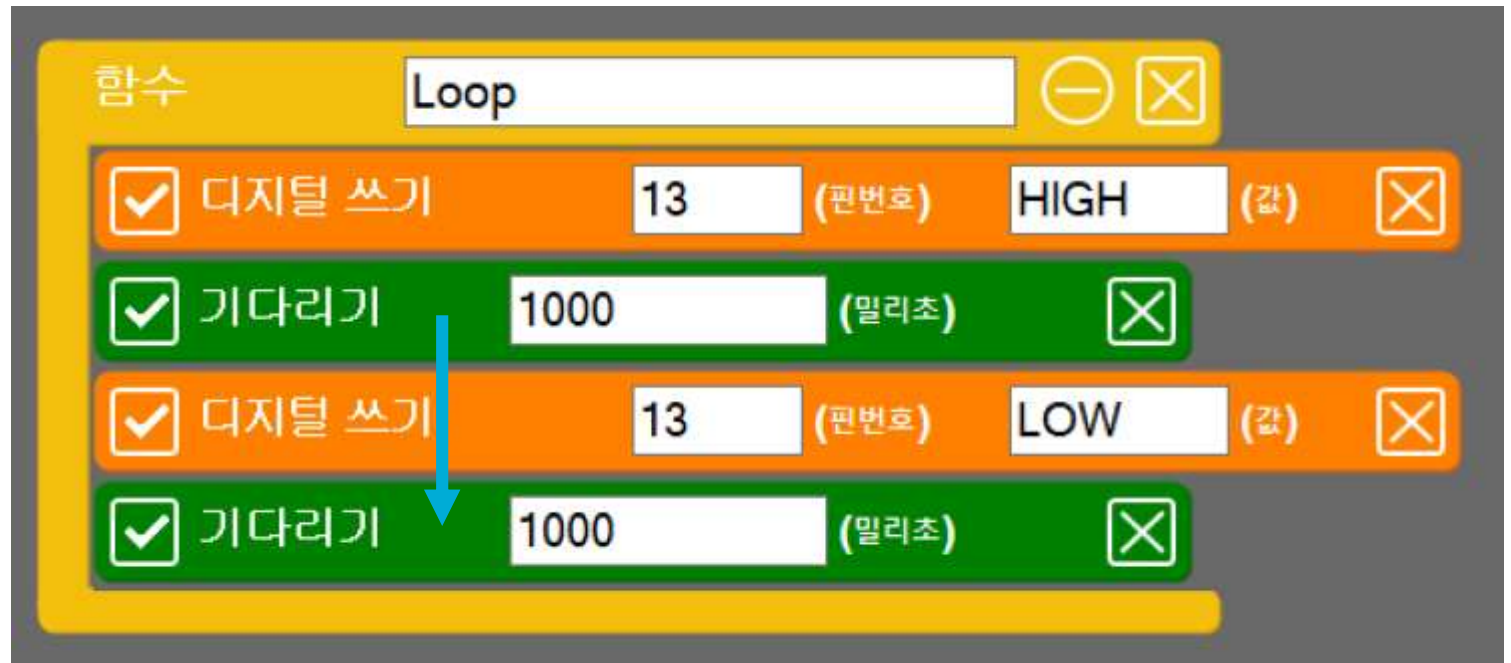


명령어 복사하기



블록 명령어를 Shift 키를 누른 채 이동하면 해당 명령어가 복사됩니다.
기다리기 명령어를 복사해 봅니다.

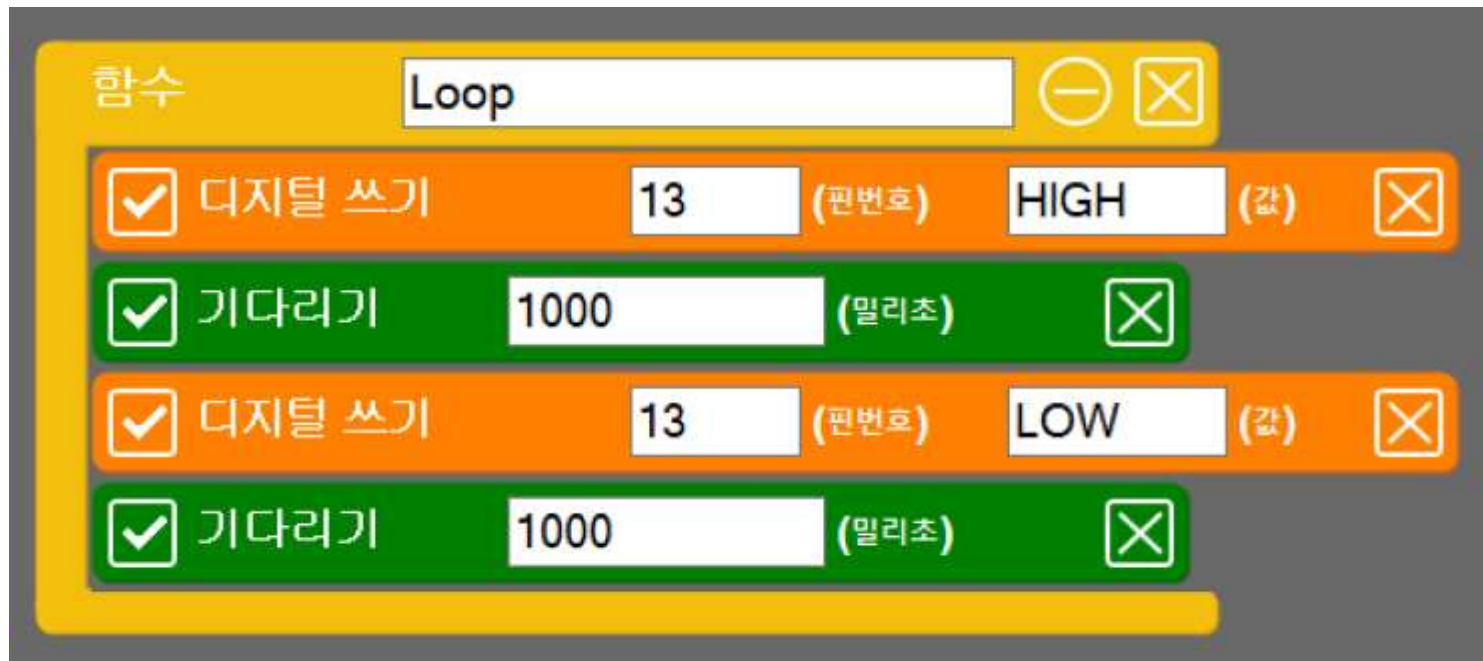
기다리기 명령어를 Shift 키를 누른
상태로 이동하면 복사되어
추가됩니다.



LED를 1초 간격으로 점멸 시키기



완성된 코드입니다.



실행하기



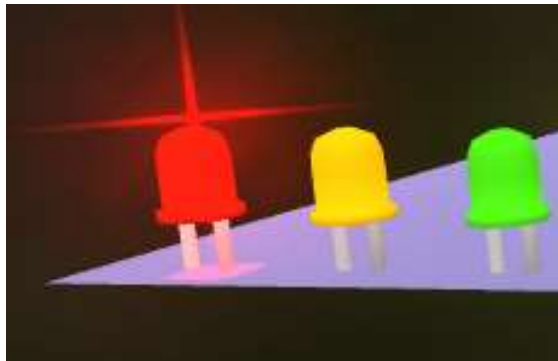
- 실행 버튼을 클릭합니다.



LED를 1초 간격으로 점멸 시키기



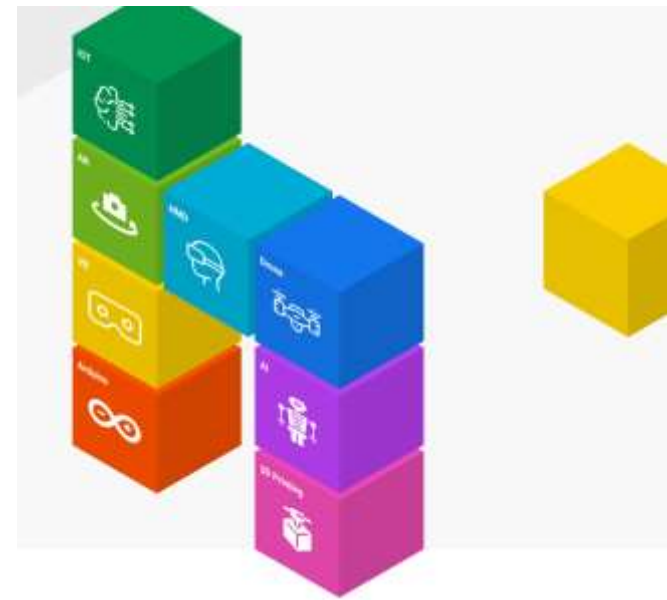
13번에 연결된 빨간색 LED가 1초 간격으로 점멸합니다.



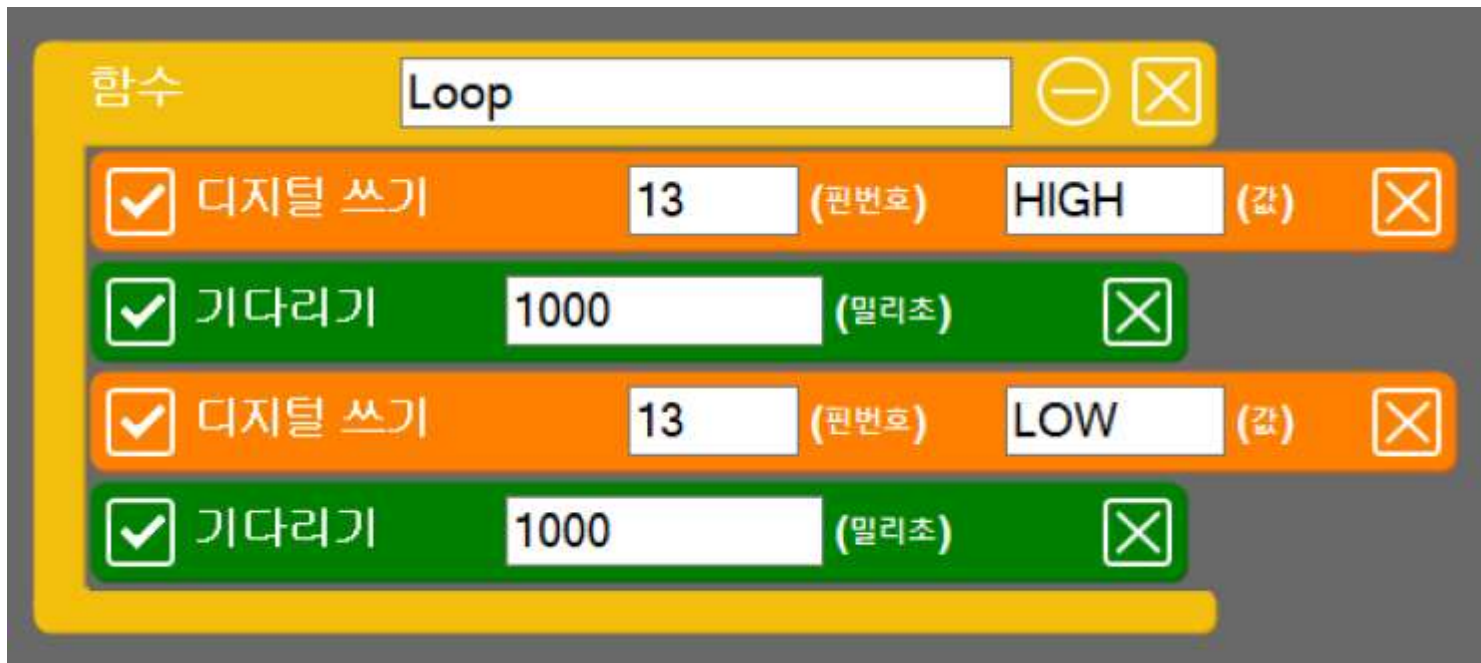
1초 후



LED를 더 빠른 속도로 점멸시키기

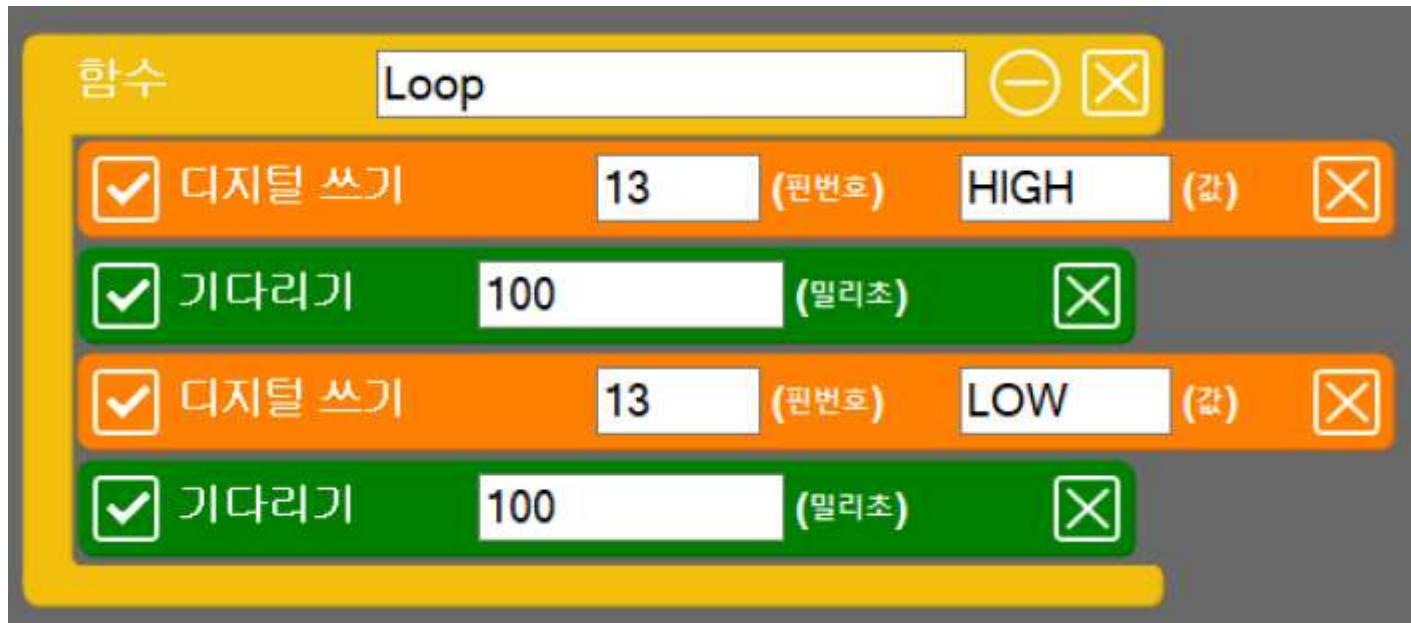


LED를 더 빠른 속도로 점멸시키기



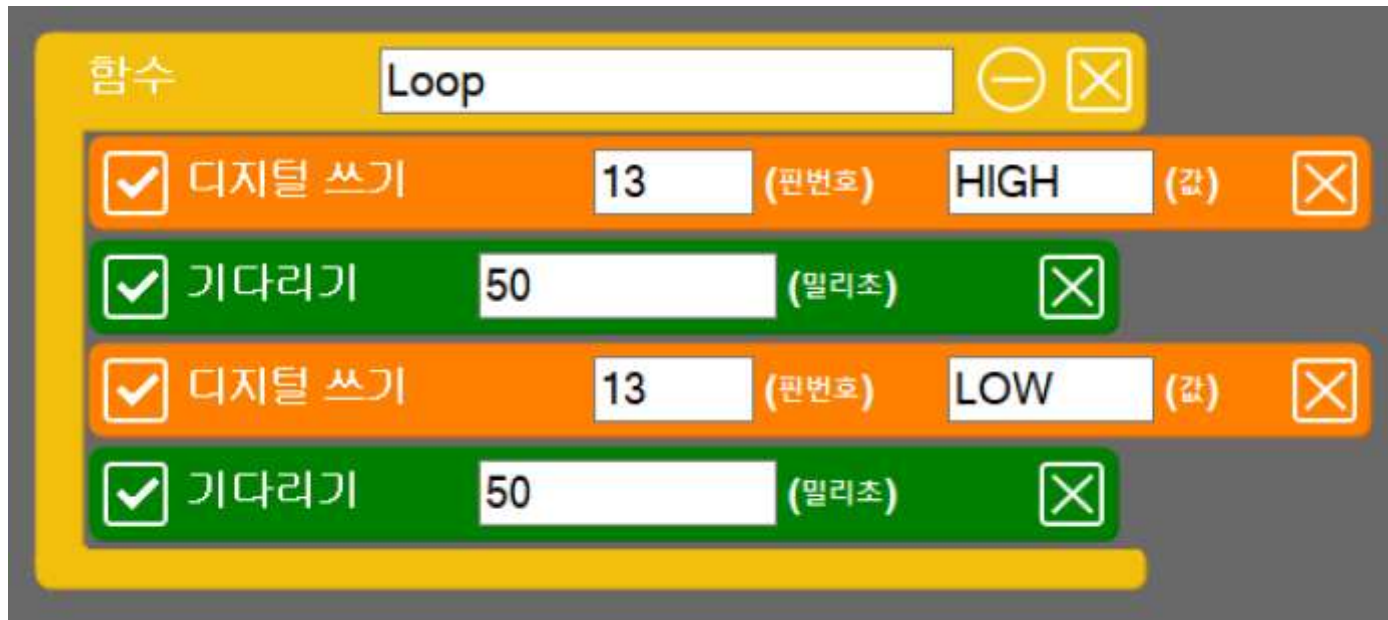
LED를 더 빠른 속도로
점멸시키기 위해서는
어느 명령어의 값을
수정해 주어야 할까요?

0.1초 간격으로 점멸시키기



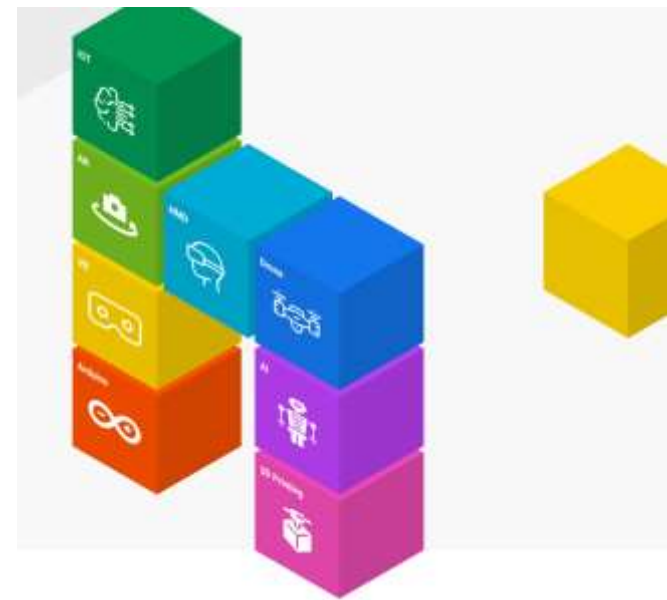
100밀리초 -> 0.1초

0.05초 간격으로 점멸시키기

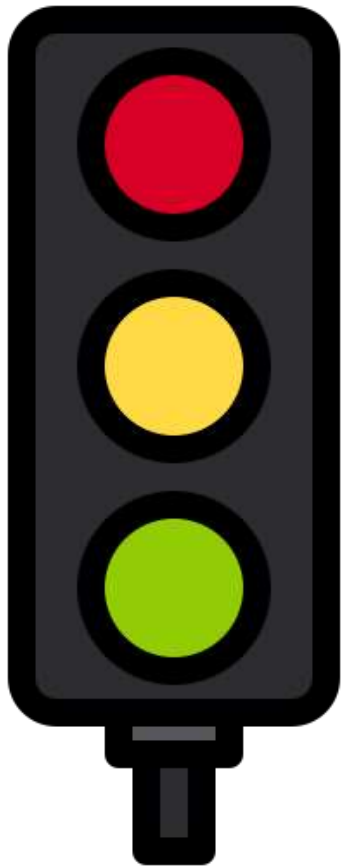


50밀리초 -> 0.05초

신호등 만들어 보기



신호등 만들기



초록불 2초



노란불 0.5초

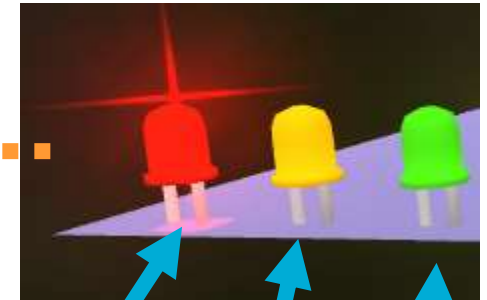
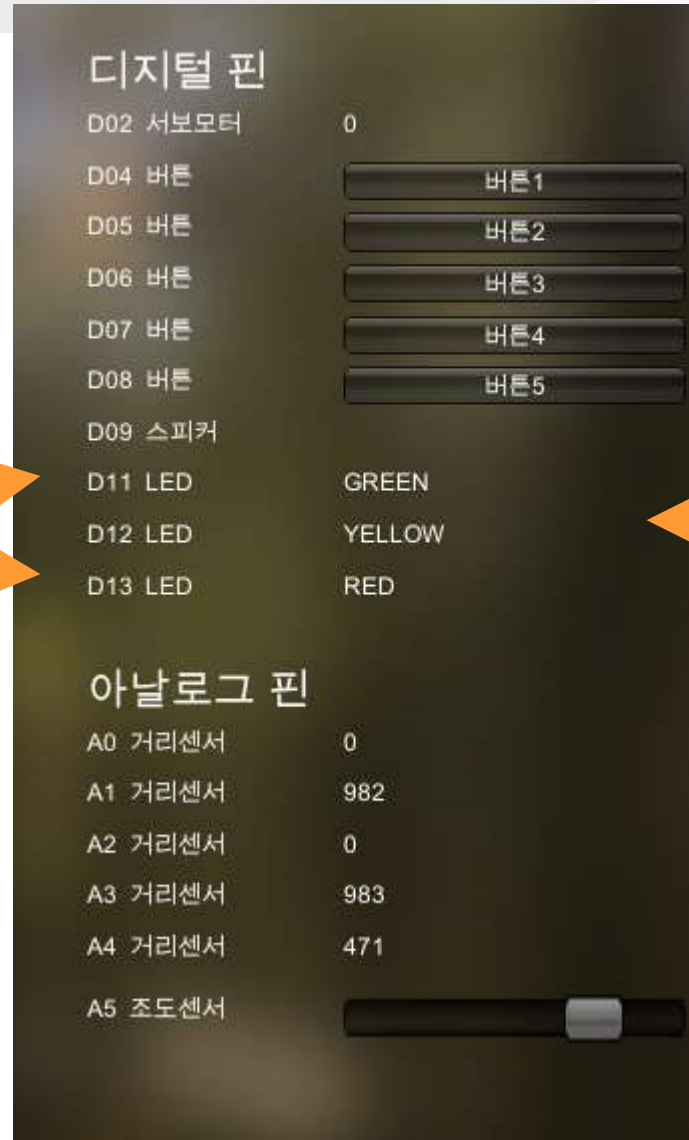


빨간불 2초

시뮬레이션 연결 환경 이해하기



3개의 LED가
11번 부터 13번까지 연결되어
있음

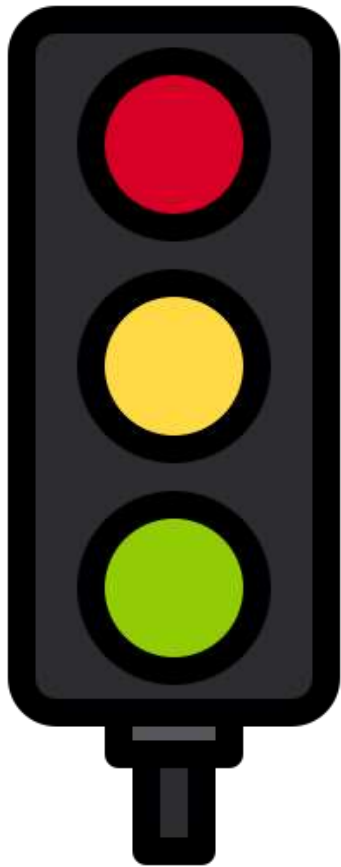


13번

12번

11번

신호등 만들기



초록불 2초



노란불 0.5초



빨간불 2초

Green LED 켜기

Green LED 끄기

Yellow LED 켜기

Yellow LED 끄기

Red LED 켜기

Red LED 끄기

신호등 만들기



함수 Loop

<input checked="" type="checkbox"/> 디지털 쓰기	11	(핀번호)	HIGH	(값)	<input type="checkbox"/>
<input checked="" type="checkbox"/> 기다리기	2000	(밀리초)	<input type="checkbox"/>		
<input checked="" type="checkbox"/> 디지털 쓰기	11	(핀번호)	LOW	(값)	<input type="checkbox"/>
<input checked="" type="checkbox"/> 디지털 쓰기	12	(핀번호)	HIGH	(값)	<input type="checkbox"/>
<input checked="" type="checkbox"/> 기다리기	500	(밀리초)	<input type="checkbox"/>		
<input checked="" type="checkbox"/> 디지털 쓰기	12	(핀번호)	LOW	(값)	<input type="checkbox"/>
<input checked="" type="checkbox"/> 디지털 쓰기	13	(핀번호)	HIGH	(값)	<input type="checkbox"/>
<input checked="" type="checkbox"/> 기다리기	2000	(밀리초)	<input type="checkbox"/>		
<input checked="" type="checkbox"/> 디지털 쓰기	13	(핀번호)	LOW	(값)	<input type="checkbox"/>

실행하기



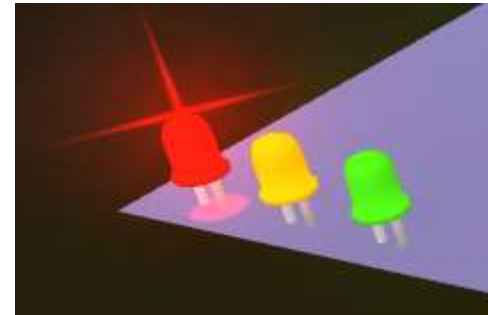
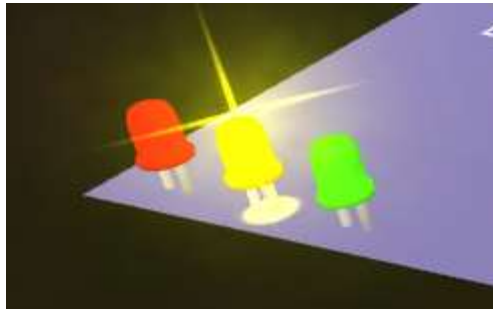
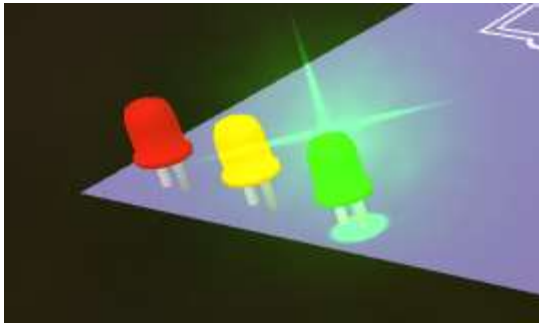
- 실행 버튼을 클릭합니다.



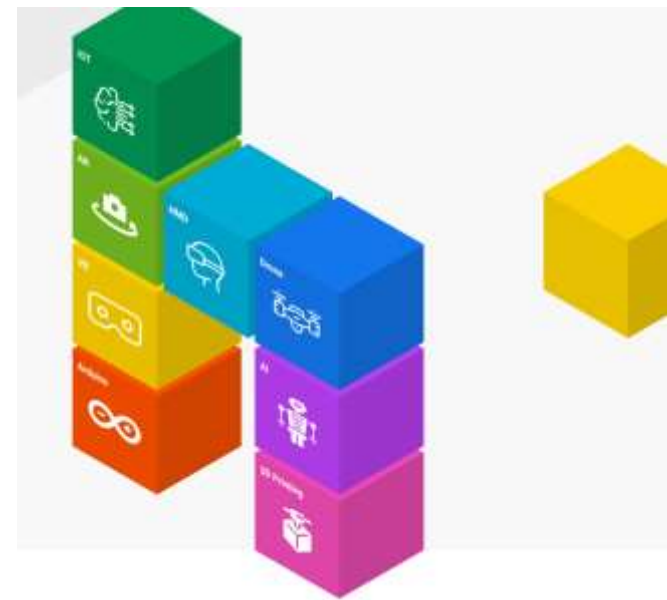
신호등 만들기



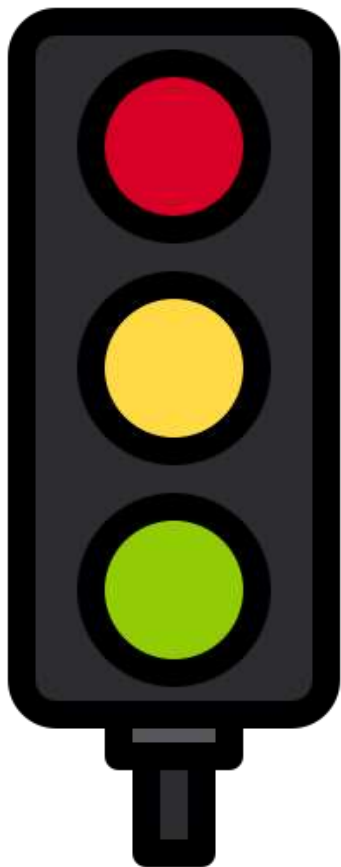
- 실행 결과



신호등 만들어 보기 응용



신호등 만들기 응용 실습



초록불 2초



초록불 0.5초 점멸



노란불 0.5초



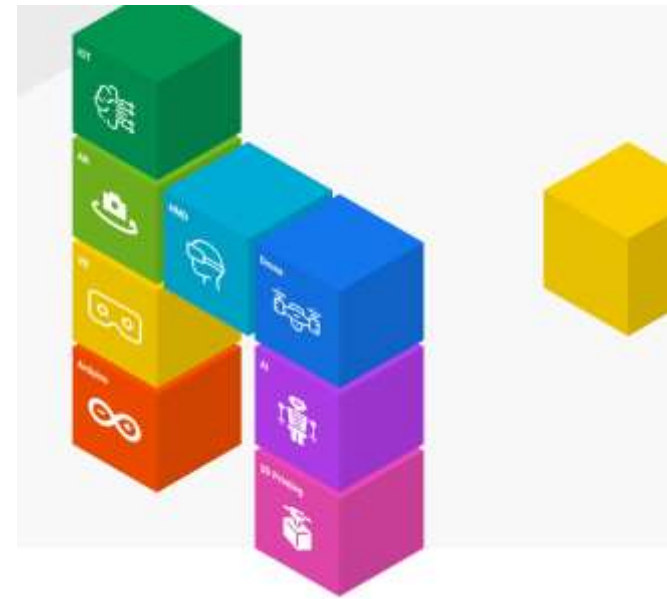
빨간불 2초

신호등 만들기 응용 실습

함수 Loop

<input checked="" type="checkbox"/> 디지털 쓰기	11	(핀번호)	HIGH	(값)	<input type="checkbox"/>
<input checked="" type="checkbox"/> 기다리기	2000	(밀리초)	<input type="checkbox"/>		
<input checked="" type="checkbox"/> 디지털 쓰기	11	(핀번호)	LOW	(값)	<input type="checkbox"/>
<input checked="" type="checkbox"/> 기다리기	100	(밀리초)	<input type="checkbox"/>		
<input checked="" type="checkbox"/> 디지털 쓰기	11	(핀번호)	HIGH	(값)	<input type="checkbox"/>
<input checked="" type="checkbox"/> 기다리기	100	(밀리초)	<input type="checkbox"/>		
<input checked="" type="checkbox"/> 디지털 쓰기	11	(핀번호)	LOW	(값)	<input type="checkbox"/>
<input checked="" type="checkbox"/> 기다리기	100	(밀리초)	<input type="checkbox"/>		
<input checked="" type="checkbox"/> 디지털 쓰기	11	(핀번호)	HIGH	(값)	<input type="checkbox"/>
<input checked="" type="checkbox"/> 기다리기	100	(밀리초)	<input type="checkbox"/>		
<input checked="" type="checkbox"/> 디지털 쓰기	12	(핀번호)	HIGH	(값)	<input type="checkbox"/>
<input checked="" type="checkbox"/> 기다리기	500	(밀리초)	<input type="checkbox"/>		
<input checked="" type="checkbox"/> 디지털 쓰기	12	(핀번호)	LOW	(값)	<input type="checkbox"/>
<input checked="" type="checkbox"/> 디지털 쓰기	13	(핀번호)	HIGH	(값)	<input type="checkbox"/>
<input checked="" type="checkbox"/> 기다리기	2000	(밀리초)	<input type="checkbox"/>		
<input checked="" type="checkbox"/> 디지털 쓰기	13	(핀번호)	LOW	(값)	<input type="checkbox"/>

버튼 값을 화면에 출력하기



버튼 값을 화면에 출력하기



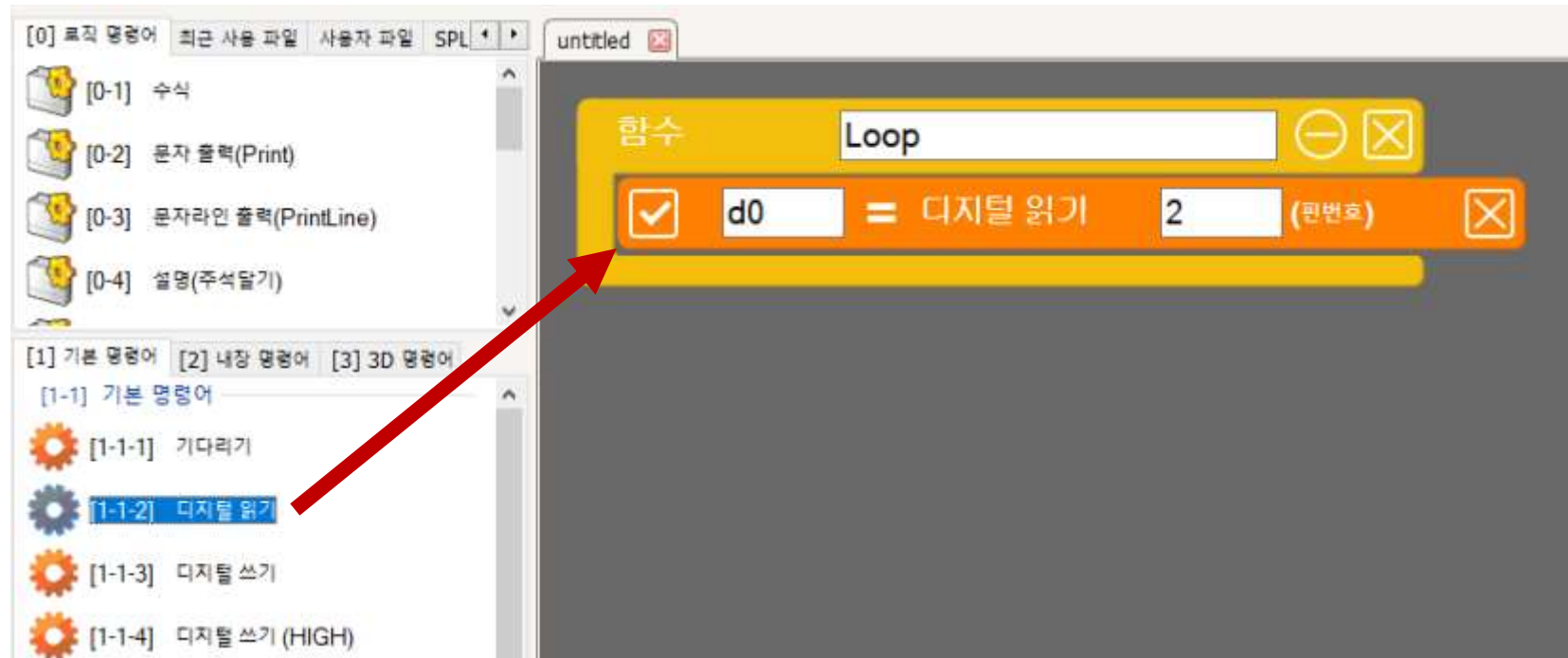
디지털 핀	
D02 서보모터	0
D04 버튼	버튼1
D05 버튼	버튼2
D06 버튼	버튼3
D07 버튼	버튼4
D08 버튼	버튼5
D09 스피커	
D11 LED	GREEN
D12 LED	YELLOW
D13 LED	RED
아날로그 핀	
A0 거리센서	0
A1 거리센서	982
A2 거리센서	0
A3 거리센서	983
A4 거리센서	471
A5 조도센서	

버튼1이 디지털 4번 핀에
연결되어 있습니다.

버튼 값을 화면에 출력하기



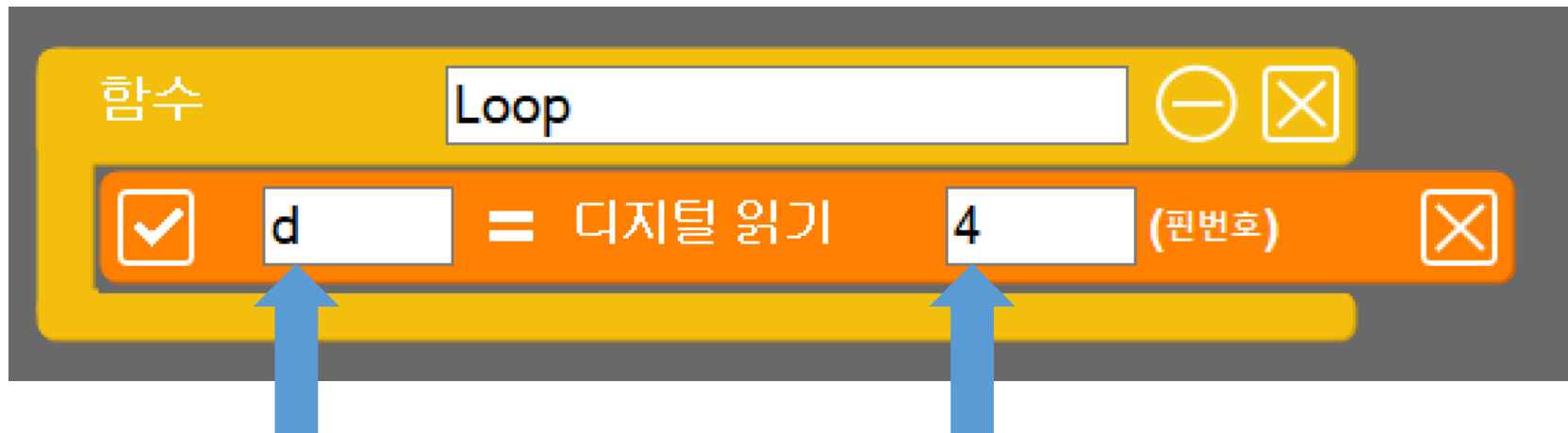
[1-1-2] 디지털 읽기 명령어를 Loop 함수 안에 추가합니다.



버튼 값을 화면에 출력하기



디지털 읽기 명령어의 변수 값과 핀번호를 다음과 같이 수정합니다.



버튼 값을 화면에 출력하기



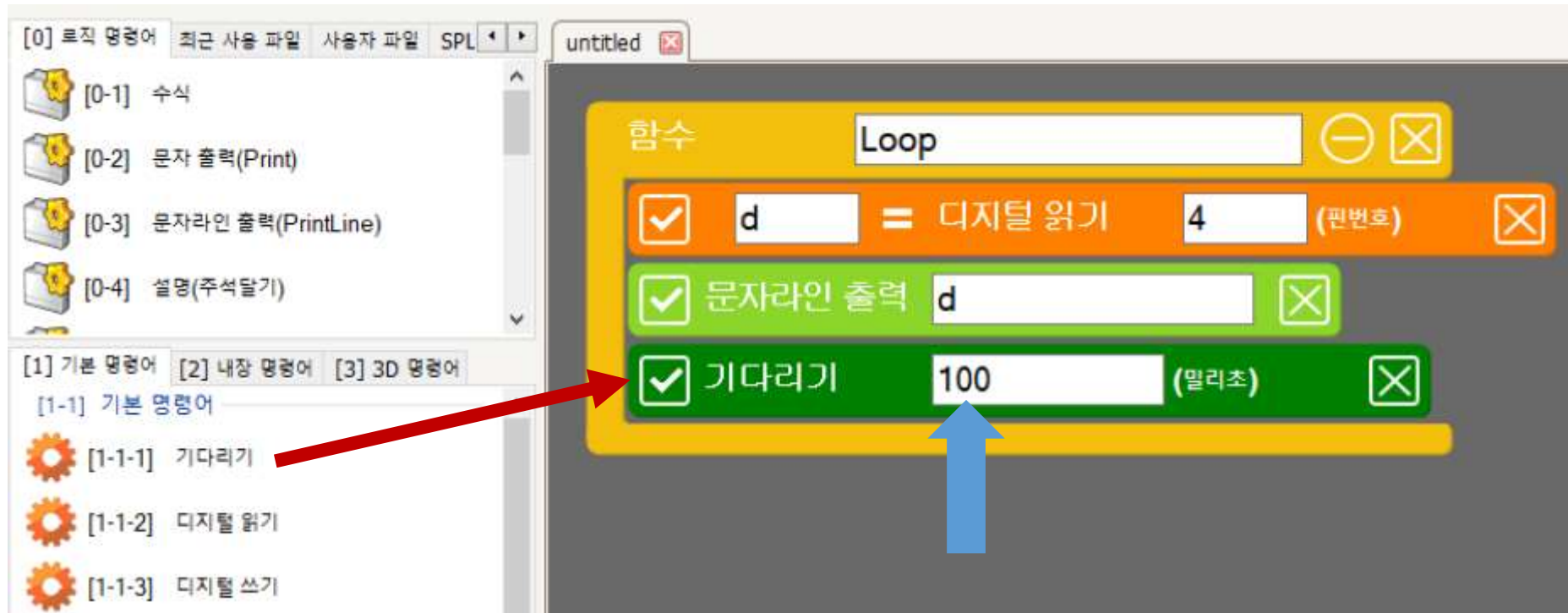
문자라인 출력(PrintLine) 명령어를 추가합니다.



버튼 값을 화면에 출력하기



기다리기 명령어를 추가한 후, 값을 100 밀리초 (0.1초)로 수정해 줍니다.



값을 100으로 수정합니다 (0.1초)

실행하기



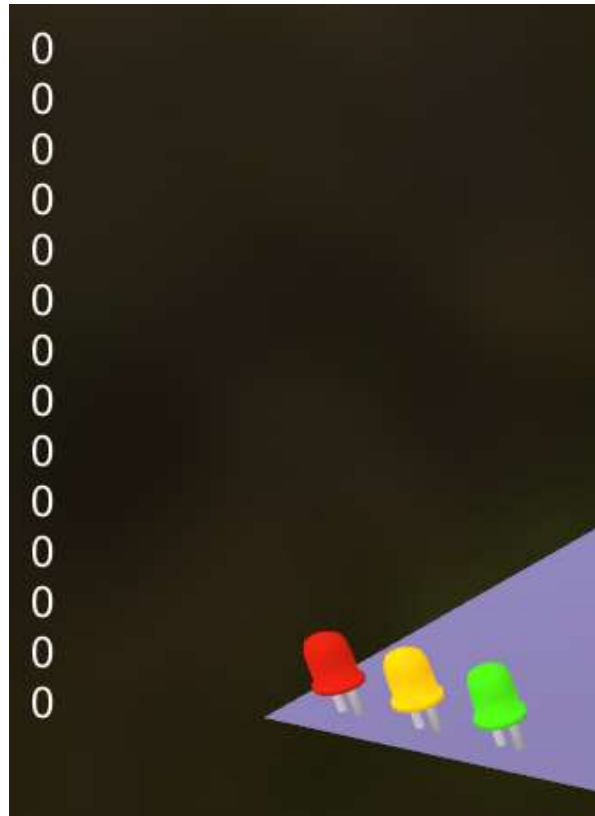
- 실행 버튼을 클릭합니다.



버튼1(4번 핀)의 값을 화면에 출력하기



버튼1이 눌러지지 않은 상태에서는 0 값이 출력됩니다.



0은 LOW와 같은 의미입니다.

버튼1(4번 핀)의 값을 화면에 출력하기



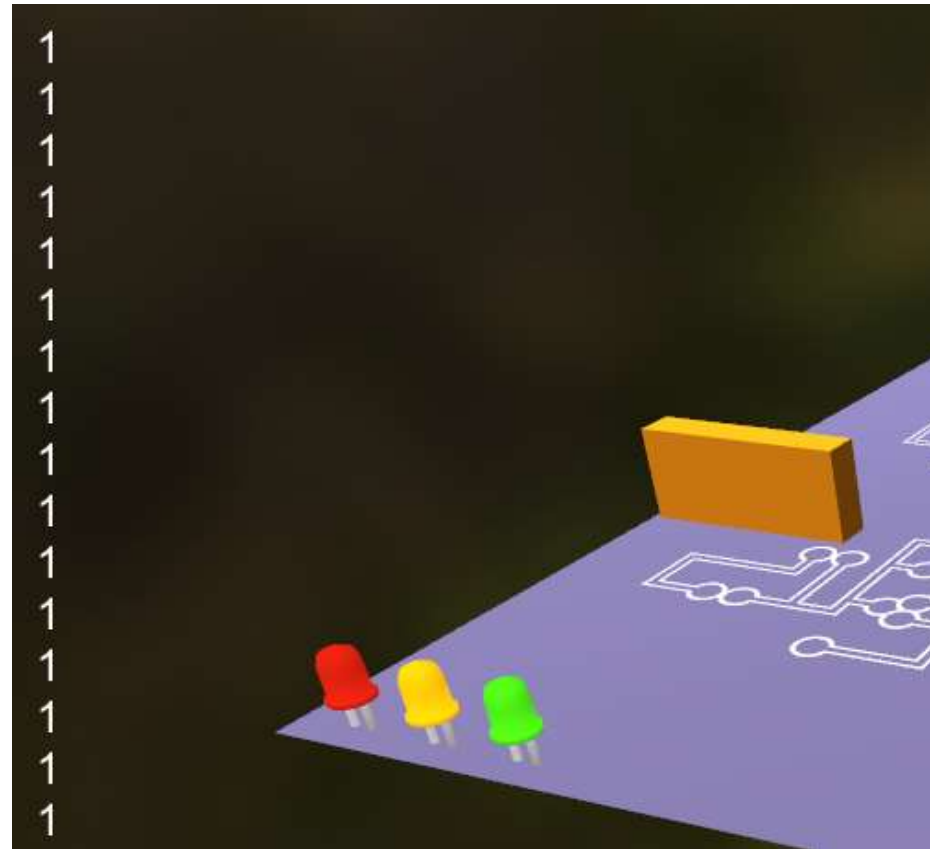
4번 핀에 연결된 버튼1을
마우스로 클릭합니다.

버튼1(4번 핀)의 값을 화면에 출력하기

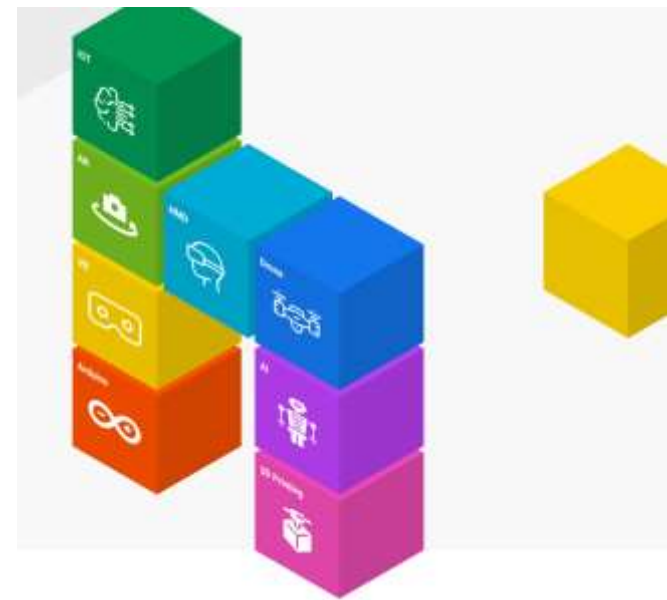


버튼1을 누르고 있으면 1 값이 읽혀집니다.

1은 HIGH와 같은 의미입니다.



버튼으로 LED 켜기

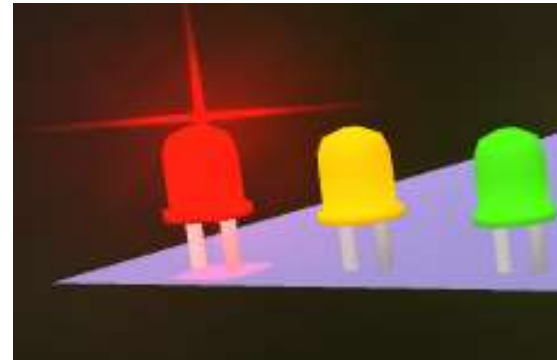


버튼으로 LED 켜기



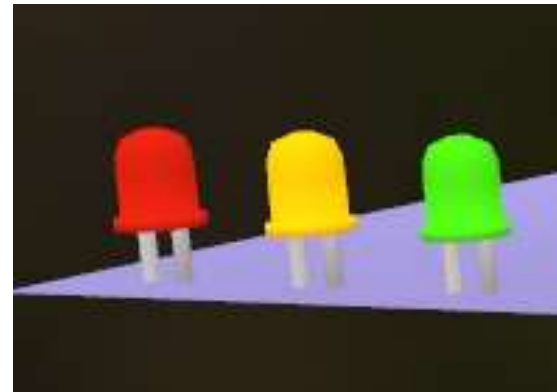
버튼1을 누르면 13번 LED가 켜지고 그렇지 않으면 꺼지는 기능을 구현해 봅니다.

버튼1을
마우스로 클릭



13번 핀의 LED가
켜짐

버튼을 클릭하지 않음



13번 핀의 LED가
꺼짐

if – else 명령어로 값 비교하기



if와 else 블록을 이용하여 조건이 만족되는 경우와 그렇지 않은 경우에 따라서 명령어를 실행시킬 수 있습니다.

비교할 조건 수식을 이곳에 입력

if 조건이 참(true)이면
if 블록 안의 명령어가
실행됨

if 조건이 거짓(false)이면
else 블록 안의 명령어가
실행됨

if 비교하기 true

왼쪽에 있는 명령어 아이콘을
이곳에 마우스로 드래그하여 넣어 주세요.

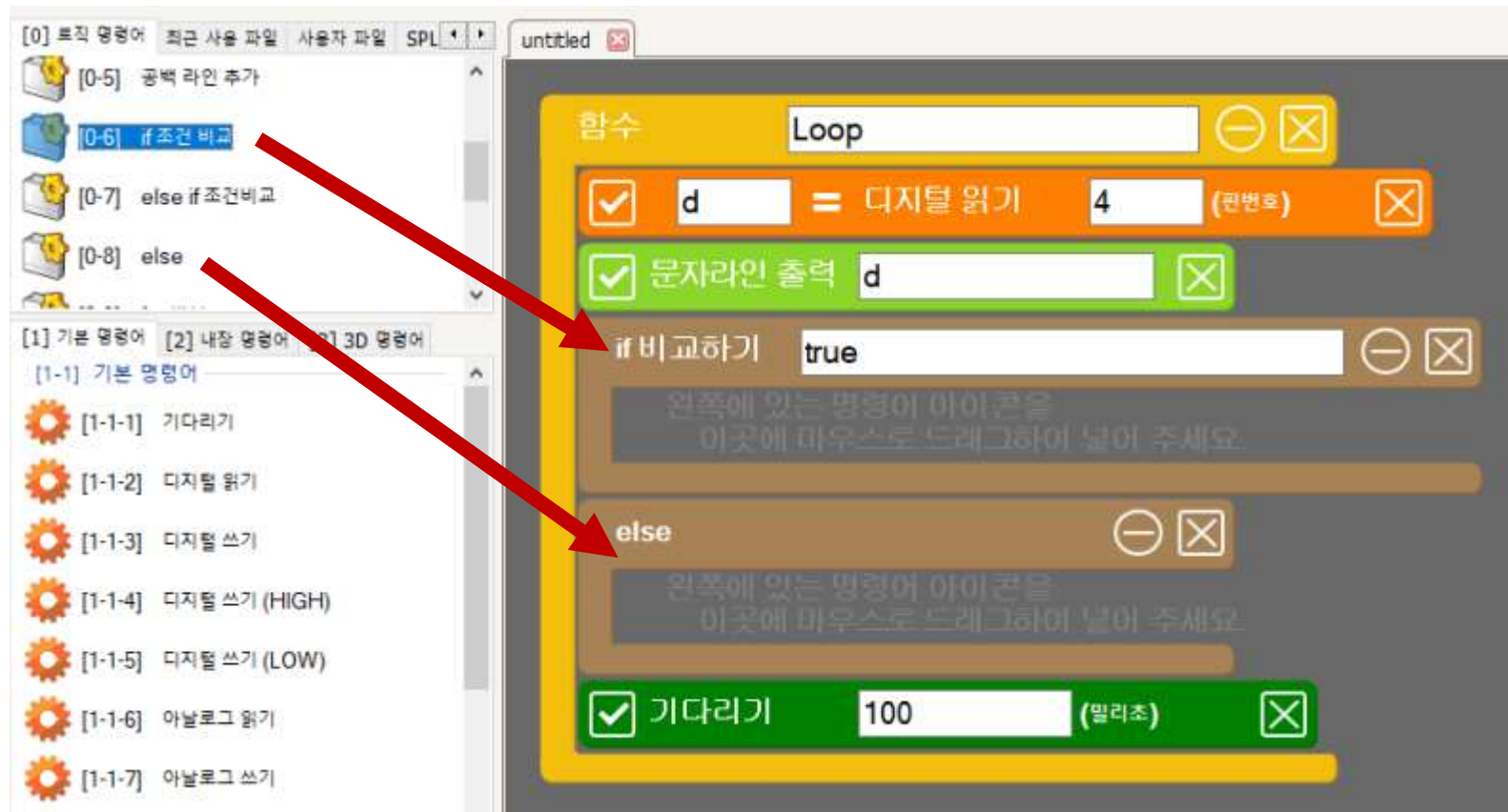
else

왼쪽에 있는 명령어 아이콘을
이곳에 마우스로 드래그하여 넣어 주세요.

if – else 명령어 추가하기



if와 else 블록 명령어를 추가합니다. (기다리기 명령어 위에 순서대로 추가합니다)



if – else 명령어 추가하기

if 조건문의 수식을 다음과 같이 수정합니다.



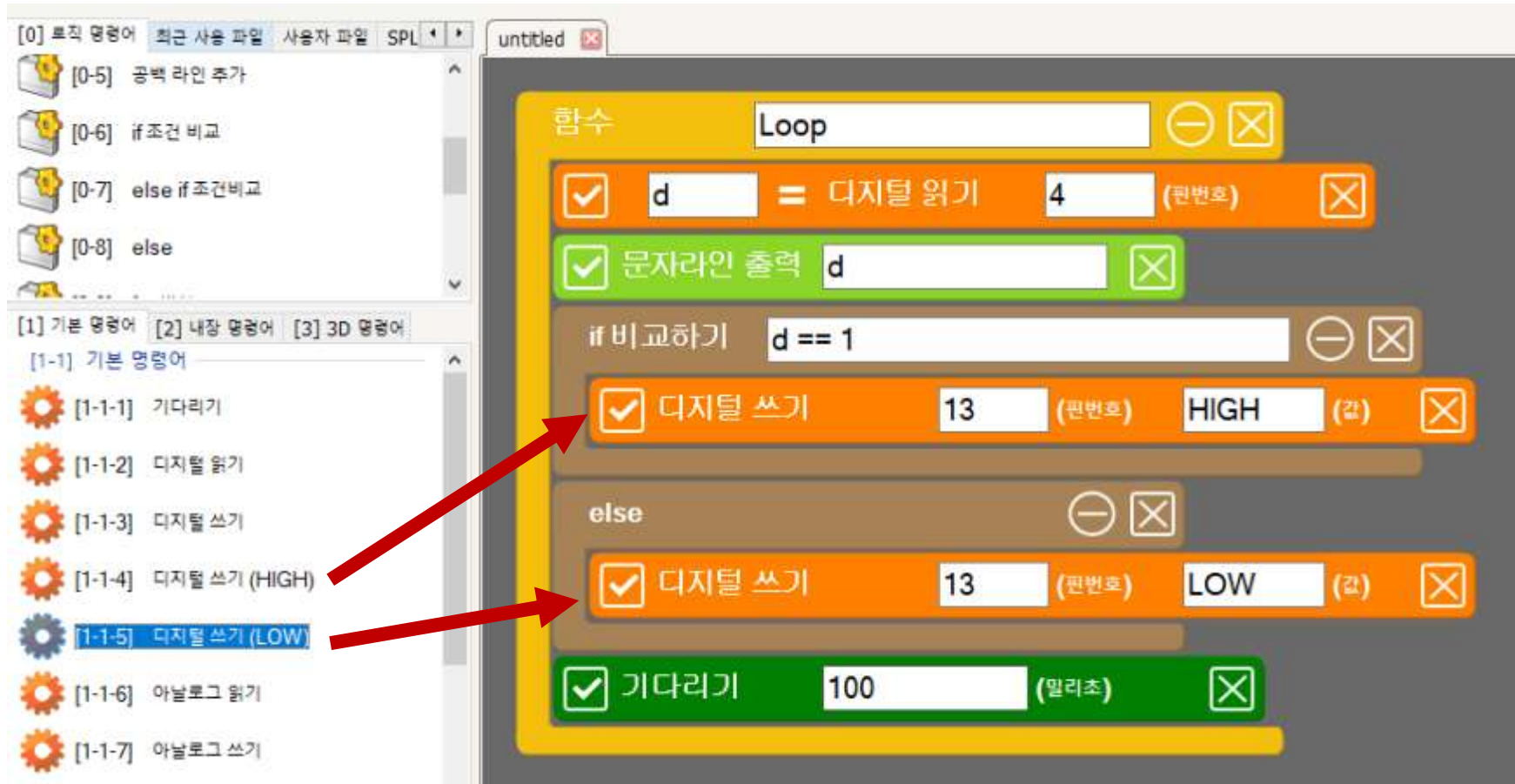
d == 1

= 기호가 두 번 반복됩니다.

==

LED 명령어 추가하기

if 블록 및 else 블록안에 각각 LED를 켜고 끄는 명령어를 추가합니다.



LED 명령어 추가하기



if 블록 및 else 블록안에 각각 LED를 켜고 끄는 명령어를 추가합니다.

LED 켜기

LED 끄기



실행하기



- 실행 버튼을 클릭합니다.



버튼으로 LED 제어하기



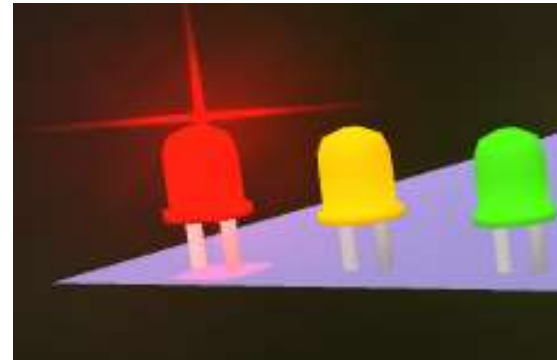
4번 핀에 연결된 버튼1을
마우스로 클릭합니다.

버튼으로 LED 켜기



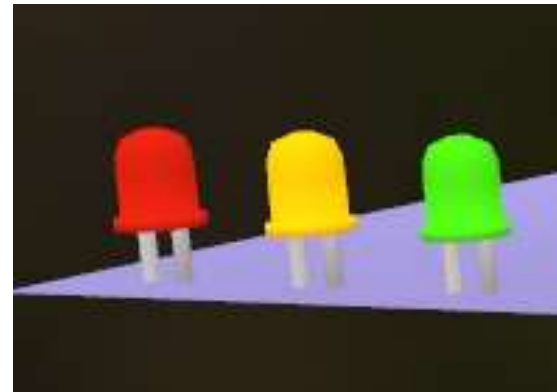
버튼1을 누르면 13번 LED가 켜지고 그렇지 않으면 LED가 꺼지게 됩니다.

버튼1을
마우스로 클릭



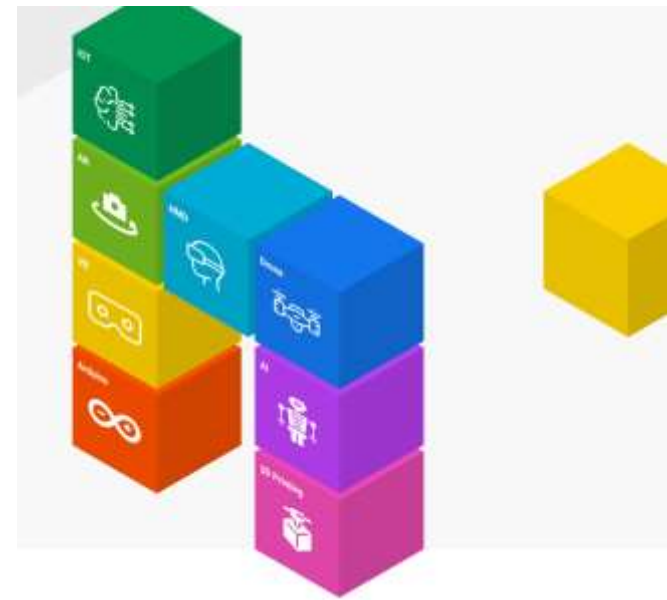
13번 핀의 LED가
켜짐

버튼을 클릭하지 않음



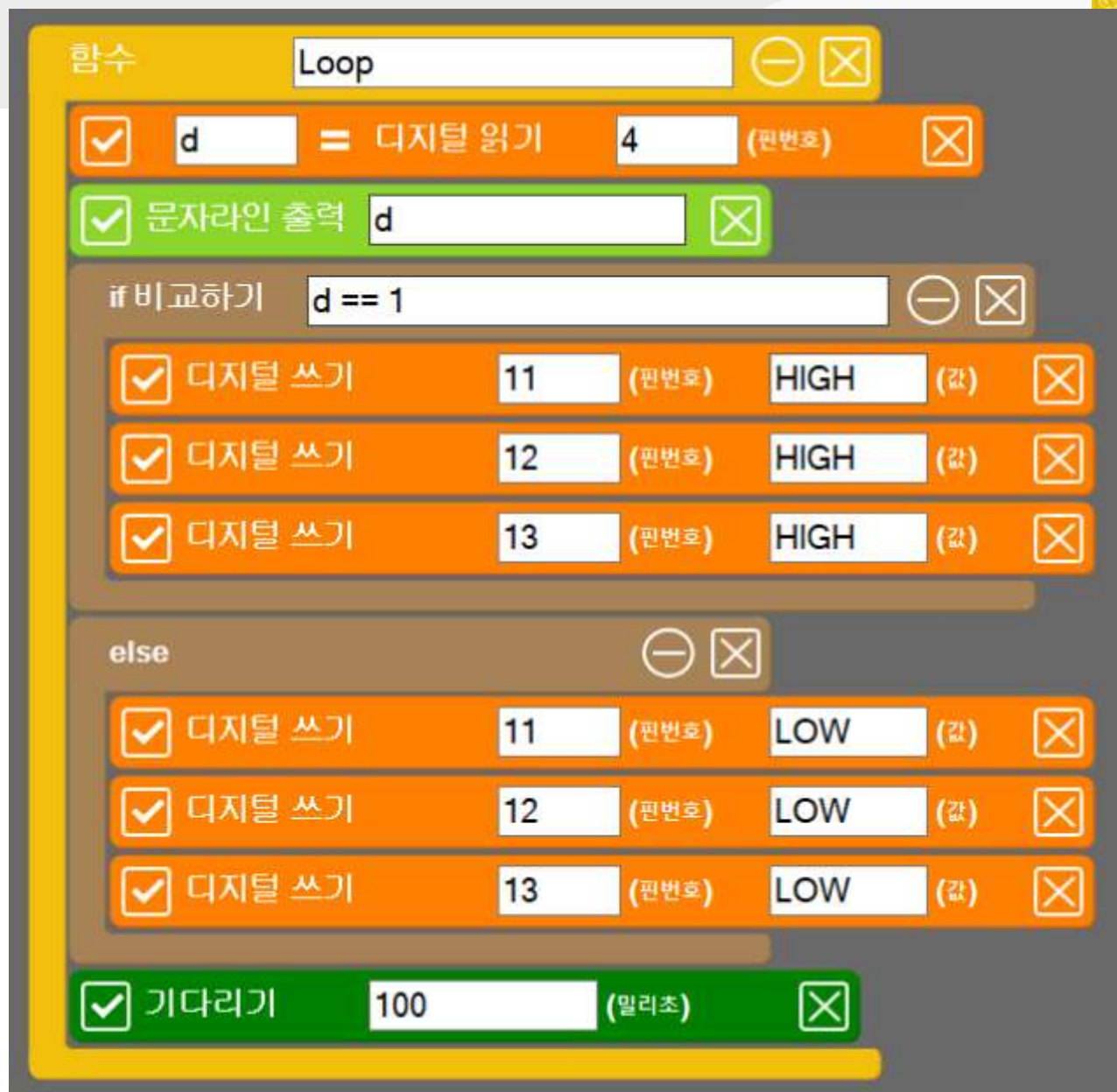
13번 핀의 LED가
꺼짐

LED 전체 켜기 실습



LED 전체 켜기 실습

3개의 LED를 모두 켭니다.



The image shows a code editor window with a yellow border. The main block is a '함수' (Function) block named 'Loop'. It contains the following logic:

- ☒ `d` = 디지털 읽기 `4` (핀번호)
- ☒ 문자라인 출력 `d`
- if 비교하기** `d == 1`
 - ☒ 디지털 쓰기 `11` (핀번호) `HIGH` (값)
 - ☒ 디지털 쓰기 `12` (핀번호) `HIGH` (값)
 - ☒ 디지털 쓰기 `13` (핀번호) `HIGH` (값)
- else**
 - ☒ 디지털 쓰기 `11` (핀번호) `LOW` (값)
 - ☒ 디지털 쓰기 `12` (핀번호) `LOW` (값)
 - ☒ 디지털 쓰기 `13` (핀번호) `LOW` (값)
- ☒ 기다리기 `100` (밀리초)

LED 전체 켜기 실습



3개의 LED를 모두 켭니다.

