

# Sensor practice

Week 10  
2022-05-04

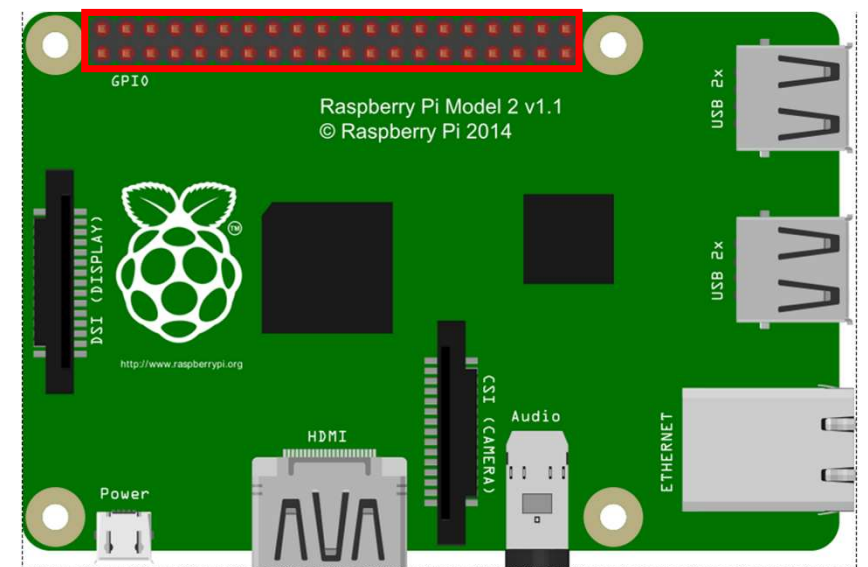
Handong Global University  
Smart Sensors and IoT Devices

**Raspberry Pi**

# 1. GPIO Interface

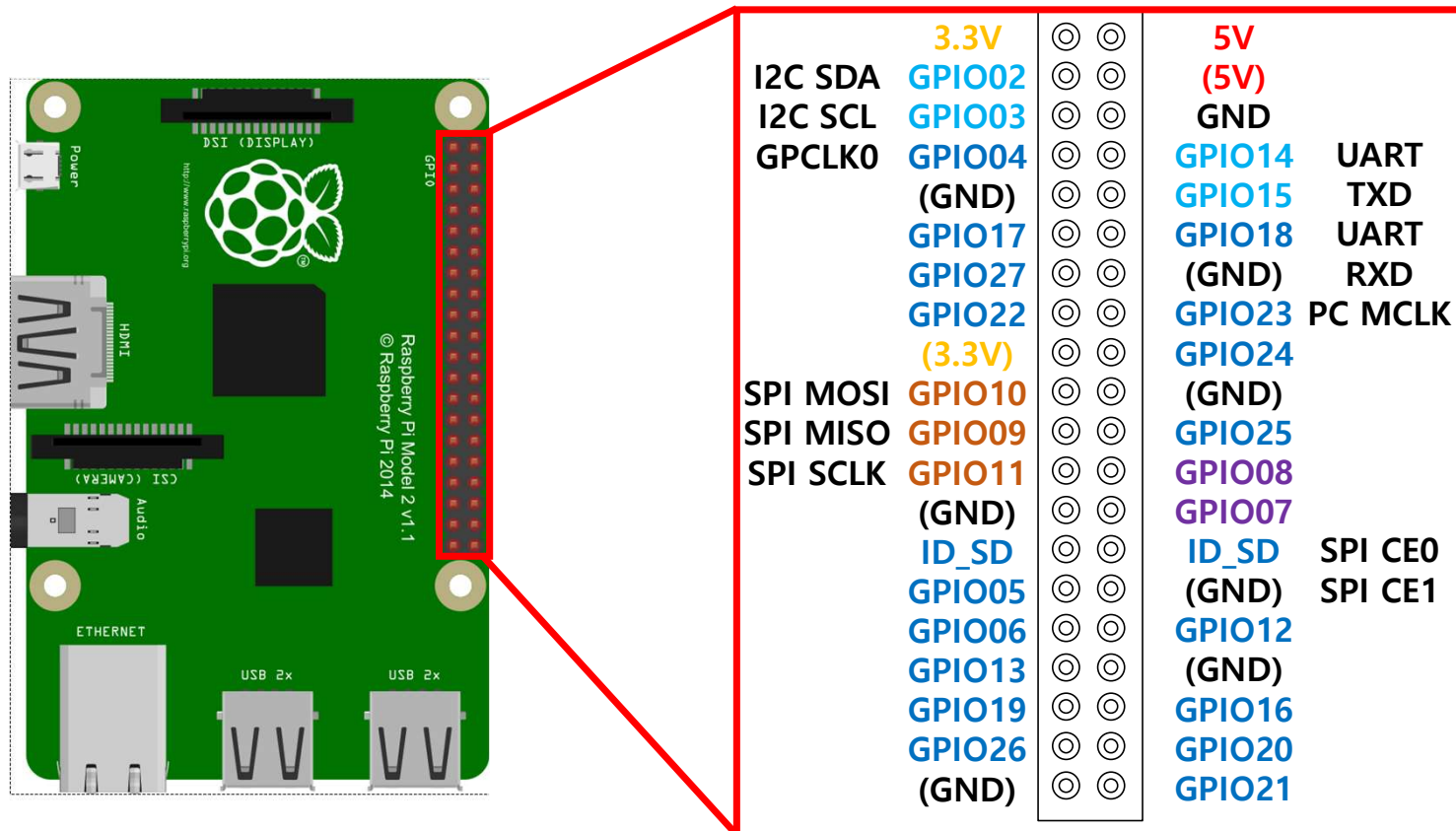
- GPIO (General Purpose Input / Output)

- CPU가 일반적인 용도로 사용 가능한 **디지털 입출력 포트 핀**
- 하드웨어 제어를 위한 신호 생성에 사용(output)
- 하드웨어로 들어오는 신호 받는데 사용(input)
- 라즈베리 파이는 총 **17개 핀의 GPIO 포트 사용 가능**
- 라즈베리 파이는 내부적으로 **3.3V 전원**에서 동작함
- GPIO핀도 3.3V 정도로 동작 시켜야 함



# 1. GPIO Interface

- GPIO (General Purpose Input / Output) Pin Mode



# 1. GPIO Interface

- GPIO 제어 소프트웨어

소프트웨어	프로그래밍 언어	특징
wiringPi	C/C++	C/C++/루비 등 다양한 바인딩 제공
bcm2835	C/C++	GPIO와 SPI 인터페이스 제공
<b>RPi.GPIO</b>	<b>파이썬</b>	<b>기본 파이썬 GPIO 제어 모듈</b>
<b>pigpio</b>	<b>파이썬</b>	<b>소프트웨어 PWM이 개선된 파이썬 GPIO 모듈</b>
Pi4J	자바	자바 GPIO 제어 모듈
ScratchGPIO	스크래치	기본 스크래치 GPIO 제어 모듈
WebIOPi	파이썬/웹/자바스크립트	GPIO에 대한 웹 인터페이스 제공

# 1. GPIO Interface

- Matlab 내 GPIO 주요 문법

문법	역할
raspi	라즈베리 파이 보드에 연결
configurePin	GPIO 핀을 디지털 입력, 디지털 출력 또는 PWM 출력으로 구성
readDigitalPin	GPIO 입력 핀에서 논리적 값 읽기
writeDigitalPin	GPIO 출력 핀에 논리적 값 쓰기
showPins	GPIO 핀 다이어그램 표시

- Matlab - 라즈베리 파이 통신 확인 및 보드 지정

모니터, 마우스, 키보드 연결 후 명령 프롬프트에서 'hostname -I' / 통신 확인 -> !ping 보드 네트워크  
보드 지정 -> mypi = raspi  
문제 발생 시 서버 업데이트 -> raspi.internal.updateServer('보드 네트워크')

# \* Raspberry Pi

## • 라즈베리 파이 통신

### 라즈베리 파이 통신 연결 확인 및 matlab 내 변수 선언

명령 창

MATLAB을 처음 사용한다면 [시작하기](#)를 참조하십시오.

```
>> !ping 169.254.49.177
```

Ping 169.254.49.177 32바이트 데이터 사용:

169.254.49.177의 응답: 바이트=32 시간=2ms TTL=64

169.254.49.177의 응답: 바이트=32 시간=2ms TTL=64

169.254.49.177의 응답: 바이트=32 시간=2ms TTL=64

169.254.49.177의 응답: 바이트=32 시간=2ms TTL=64

169.254.49.177에 대한 Ping 통계:

*fx* 패킷: 보냄 = 4, 받음 = 4, 손실 = 0 (0% 손실),  
왕복 시간(밀리초):

명령 창

MATLAB을 처음 사용한다면 [시작하기](#)를 참조하십시오.

```
>> mypi = raspi
```

mypi =

[raspi](#) - 속성 있음:

DeviceAddress: '169.254.49.177'

Port: 18734

BoardName: 'Raspberry Pi 3 Model B+'

AvailableLEDs: {'led0'}

AvailableDigitalPins: [4,5,6,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27]

AvailableSPIChannels: {'CE0','CE1'}

AvailableI2CBuses: {'i2c-1'}

AvailableWebcams: {}

I2CBusSpeed: 100000

AvailableCANInterfaces: {}

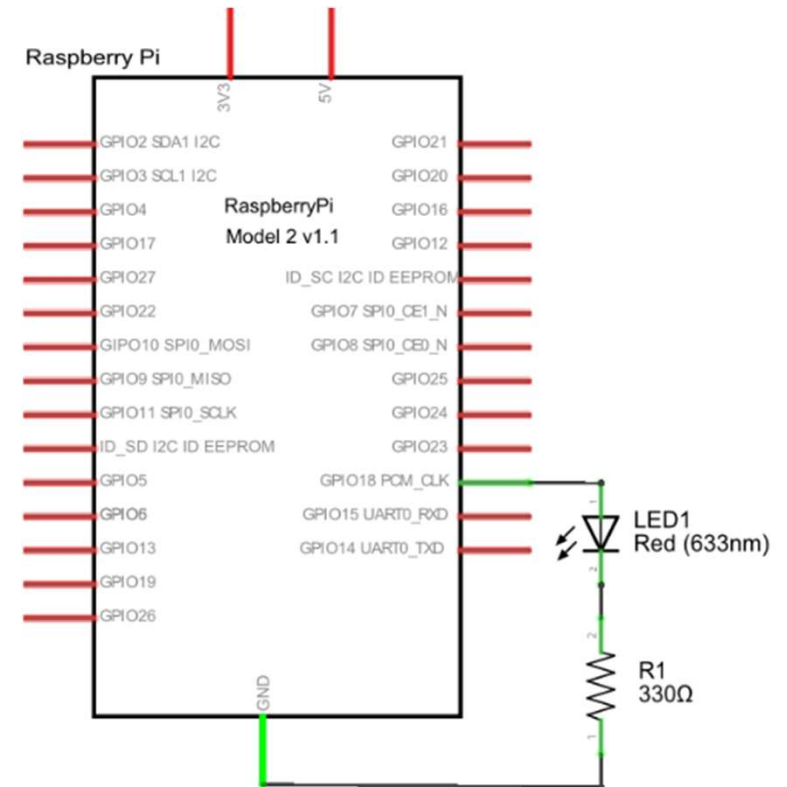
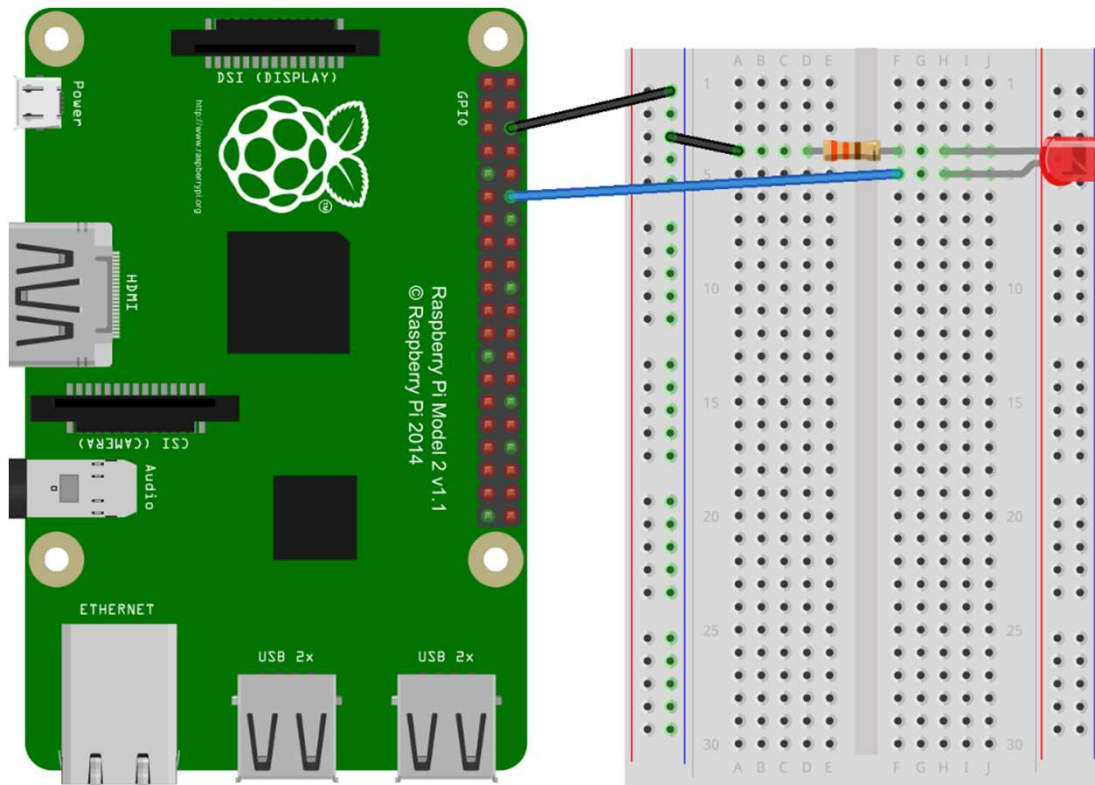
[Supported peripherals](#)

*fx* >>

## 2. LED 출력 실습

- LED 실습 회로

- 저항은 330 ~ 1k옴을 사용할 것





## 2. LED 출력 실습

- LED 출력 테스트 (명령 창)

```
>> mypi = raspi
...
>> writeDigitalPin(mypi, 18, 1)
>> writeDigitalPin(mypi, 18, 0)
```

- LED 출력 코드 예제

```
clc; clear all; close all

%%
mypi = raspi;

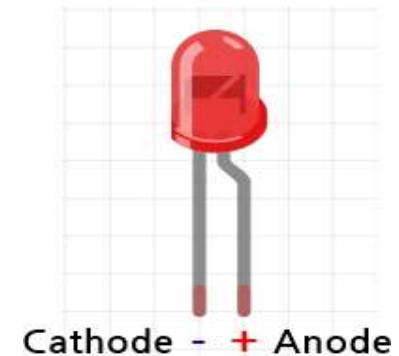
writeDigitalPin(mypi, 18, 1)
pause(0.1)
writeDigitalPin(mypi, 18, 0)
pause(0.1)
```

- LED 출력 코드 작성

- 3개의 LED 활용한 신호등 제작

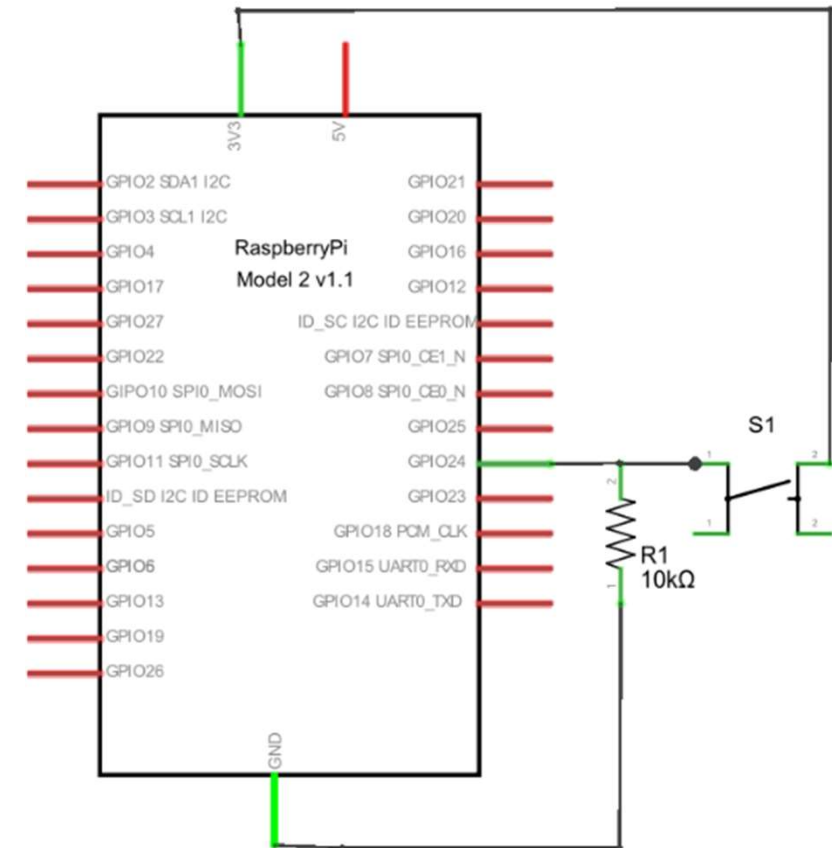
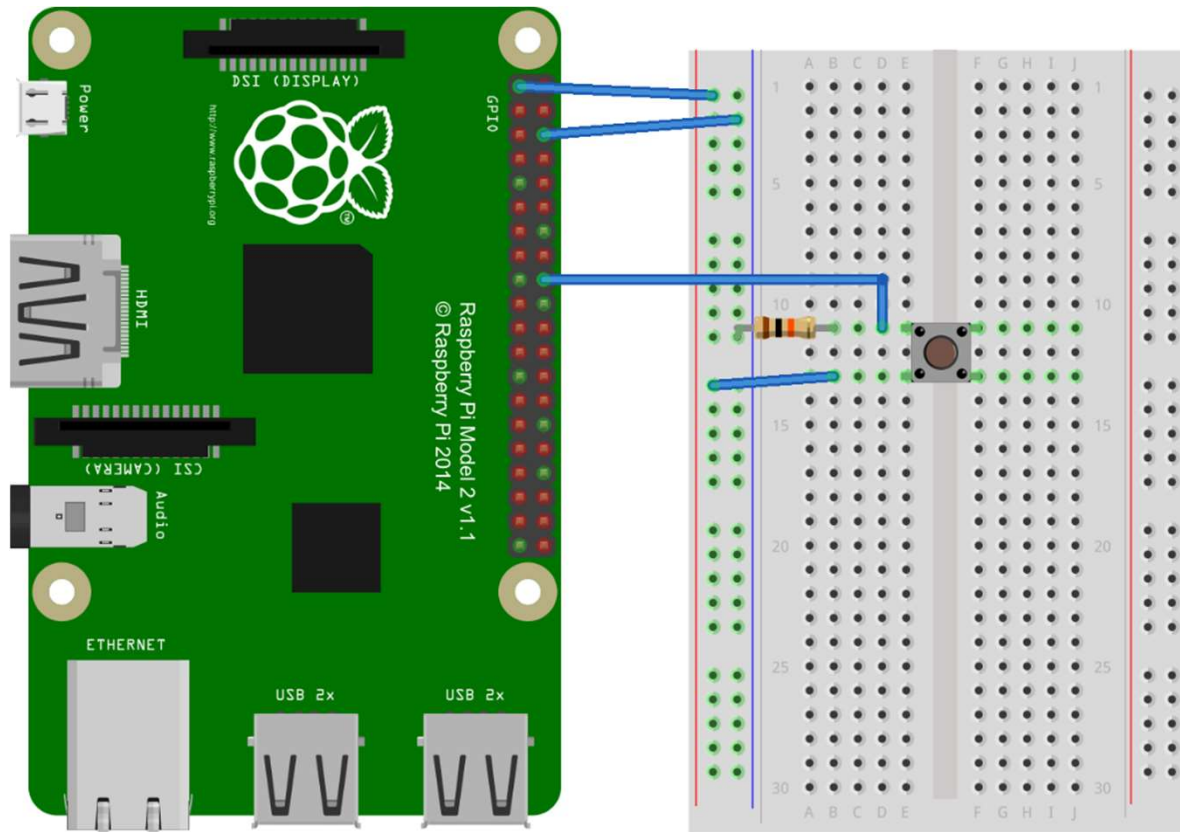
조건:

- (1) for문 사용 (2회 반복)
- (2) 청색 신호는 5초
- (3) 황색 신호는 2초
- (4) 적색 신호는 5초
- (5) 종료 시 LED 모두 꺼짐



### 3. Pushbutton 입력 실습

- Pushbutton 실습 회로



### 3. Pushbutton 입력 실습

- Pushbutton 입력 테스트 (명령 창)

```
>> mypi = raspi
>> configurePin(mypi, 24, 'digitalinput')
>> (버튼 누르지 않은 상태에서) readDigitalPin(mypi, 24)
...0
>> (버튼 누른 상태에서) readDigitalPin(mypi, 24)
...1
```

- Pushbutton 입력 코드 예제 (GPIO23)

```
clc; clear all; close all

%%
mypi = raspi;
count = 0;

while count < 10
    button = readDigitalPin(mypi, 23);
    if button
        count = count + 1;
    end
    pause(0.1);
    count
end
```

- Pushbutton 입력 코드 작성

- 버튼을 누를 때마다 깜박이는 LED

조건:

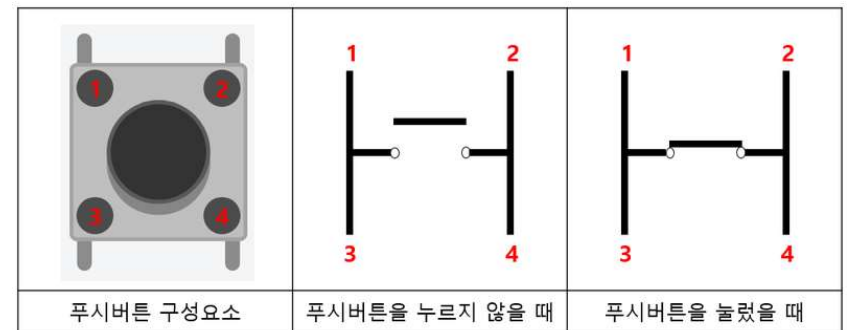
(1) Pushbutton은 GPIO 23pin

(2) LED는 GPIO 18pin

(3) while문 사용 (무한 루프)

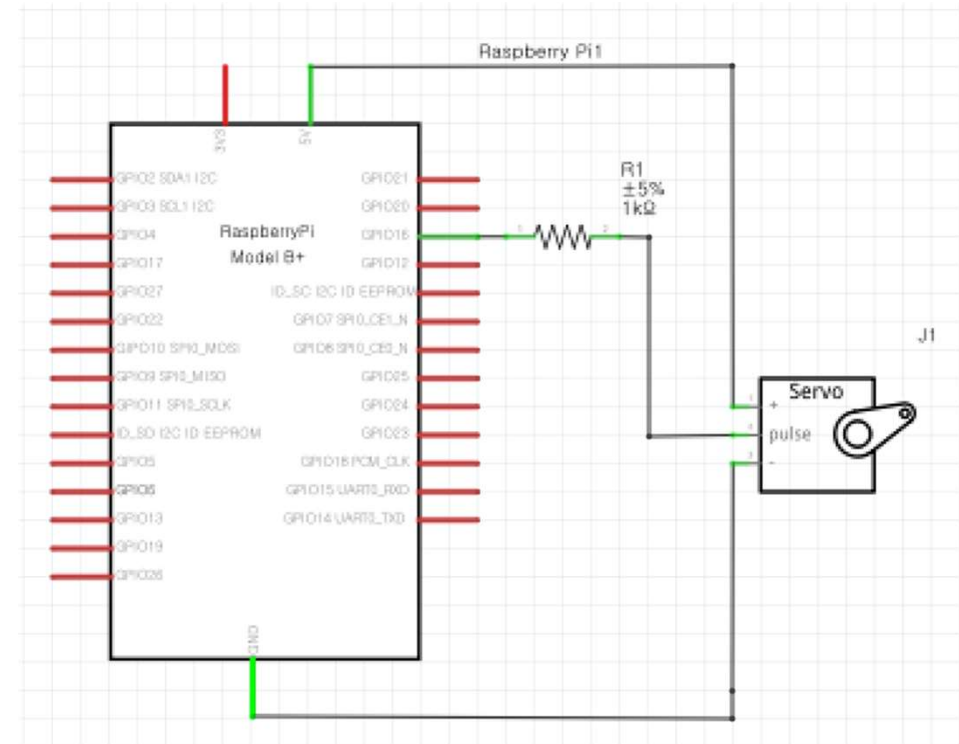
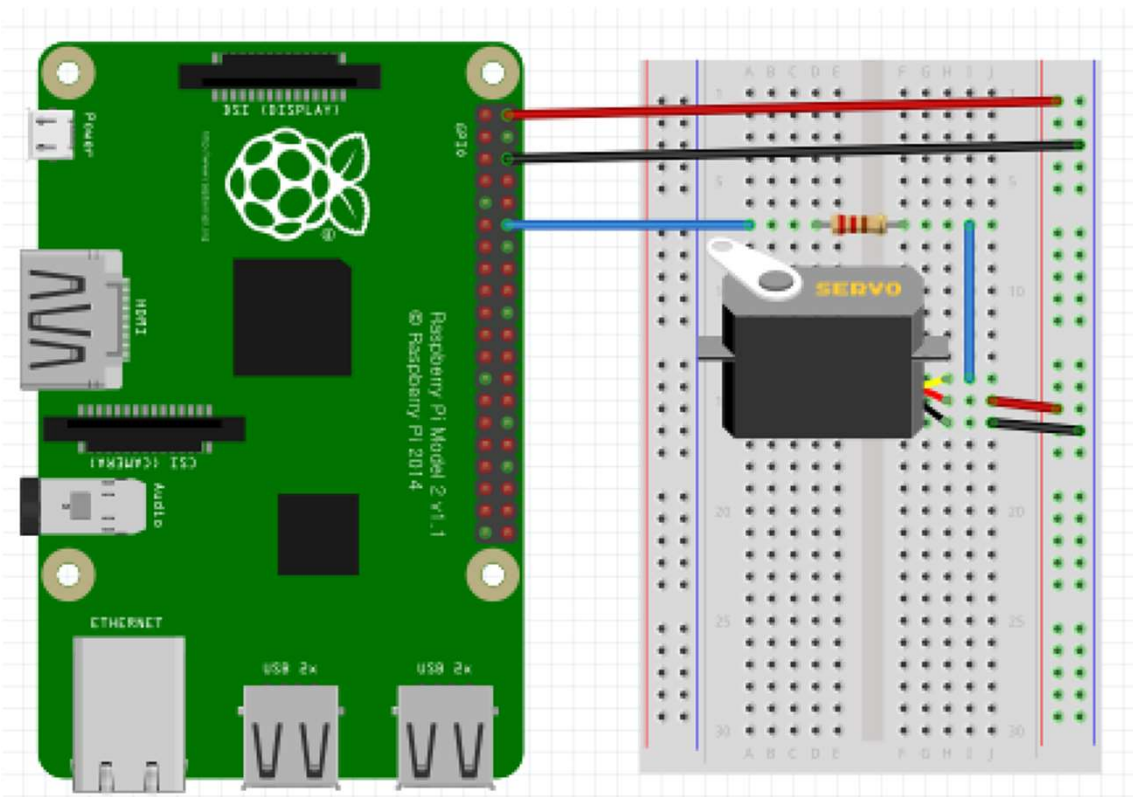
(4) 버튼을 클릭 시 LED 동작

(5) 10회 켜졌다가 꺼짐 반복



## 4. Servo motor 실습

- Servo motor 실습 회로도



## 4. Servo motor 실습

- Servo motor 테스트 (명령 창)

```
>> mypi = raspi
>> s = servo(mypi, 18)
>> writePosition(s, 180)
>> writePosition(s, 90)
>> writePosition(s, 45)
```

```
>> mypi = raspi
>> s = servo(mypi, 18, 'MinPulseDuration',7.00e-4,'MaxPulseDuration',2.3e-3)
>> writePosition(s, 180)
>> writePosition(s, 90)
>> writePosition(s, 45)
```

- Servo motor 코드 예제

```
clc; clear all; close all

%%
mypi = raspi;
s = servo(mypi, 18);
n = 0;

for i = 1:3
    for n = 1:18
        writePosition(s, n*10)
        pause(0.2)
    end
end
```

- Servo motor 코드 작성

- 버튼을 누르면 동작하는 모터

조건:

- (1) Pushbutton은 GPIO 23pin
- (2) LED는 GPIO 18pin
- (3) while문 사용 (무한 루프)
- (4) 버튼을 클릭 시 90도 이동 후 귀환

## 5. 라즈베리 파이 카메라 실습

### • 라즈베리 파이 카메라 실습 준비



	Camera Module v1	Camera Module v2	HQ Camera
Net price	\$25	\$25	\$50
Size	Around 25 × 24 × 9 mm		38 × 38 × 18.4mm (excluding lens)
Weight	3g	3g	
Still resolution	5 Megapixels	8 Megapixels	12.3 Megapixels
Video modes	1080p30, 720p60 and 640 × 480p60/90	1080p30, 720p60 and 640 × 480p60/90	1080p30, 720p60 and 640 × 480p60/90
Linux integration	V4L2 driver available	V4L2 driver available	V4L2 driver available
C programming API	OpenMAX IL and others available	OpenMAX IL and others available	
Sensor	OmniVision OV5647	Sony IMX219	<a href="#">Sony IMX477</a>
Sensor resolution	2592 × 1944 pixels	3280 × 2464 pixels	4056 × 3040 pixels
Sensor image area	3.76 × 2.74 mm	3.68 × 2.76 mm (4.6 mm diagonal)	6.287mm × 4.712 mm (7.9mm diagonal)
Pixel size	1.4 μm × 1.4 μm	1.12 μm × 1.12 μm	1.55 μm × 1.55 μm
Optical size	1/4"	1/4"	
Full-frame SLR lens equivalent	35 mm		
S/N ratio	36 dB		
Dynamic range	67 dB @ 8x gain		
Sensitivity	680 mV/lux-sec		
Dark current	16 mV/sec @ 60 C		
Well capacity	4.3 Ke-		
Fixed focus	1 m to infinity		N/A
Focal length	3.60 mm +/- 0.01	3.04 mm	Depends on lens
Horizontal field of view	53.50 +/- 0.13 degrees	62.2 degrees	Depends on lens
Vertical field of view	41.41 +/- 0.11 degrees	48.8 degrees	Depends on lens
Focal ratio (F-Stop)	2.9	2	Depends on lens



## 5. 라즈베리 파이 카메라 실습

### • 라즈베리 파이 카메라 실습 테스트 (명령 창)

```
>> mypi = raspi  
>> mycam = cameraboard(mypi,'Resolution','640x480');  
>> img = snapshot(mycam)  
>> image(img)
```

```
>> a = img(:, :, 1)  
>> b = img(:, :, 2)  
>> c = img(:, :, 3)  
>> image(a)  
>> image(b)  
>> image(c)
```

- 변수 a, b, c 확인 (640, 480)

### • 라즈베리 파이 카메라 코드 예제(1)

```
clc; clear all; close all  
  
%%  
mypi = raspi  
mycam = cameraboard(mypi,'Resolution','640x480');  
  
for i = 1:100  
    img = snapshot(mycam);  
    image(img);  
    drawnow;  
end
```



## 5. 라즈베리 파이 카메라 실습

### • 라즈베리 파이 카메라 코드 예제(2)

```
clc; clear all; close all
```

```
%%
```

```
mypi = raspi;
```

```
mycam = cameraboard(mypi,'Resolution', '640x480');
```

```
figure(1);
```

```
mycam.ImageEffect = 'sketch';
```

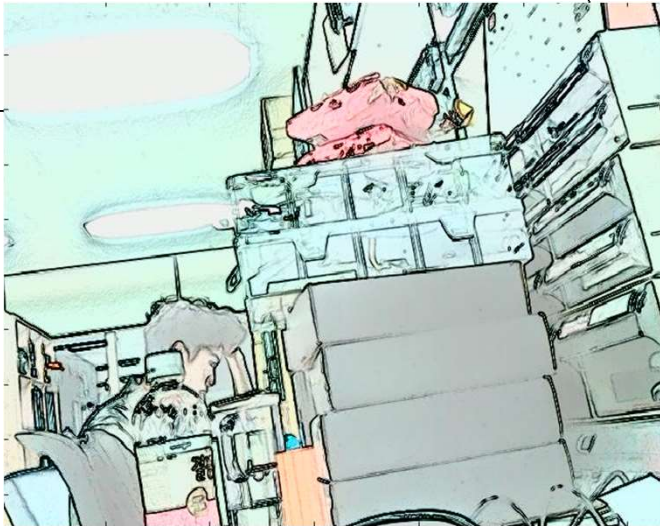
```
for i = 1:500
```

```
    img = snapshot(mycam);
```

```
    image(img);
```

```
    drawnow;
```

```
end
```



```
clc; clear all; close all
```

```
%%
```

```
mypi = raspi;
```

```
mycam = cameraboard(mypi,'Resolution', '640x480');
```

```
figure(1);
```

```
mycam.ImageEffect = 'negative';
```

```
for i = 1:500
```

```
    img = snapshot(mycam);
```

```
    image(img);
```

```
    drawnow;
```

```
end
```





## 5. 라즈베리 파이 카메라 실습

### • 라즈베리 파이 카메라 코드 예제(3)

```
clc; clear all; close all
```

```
mypi = raspi;
```

```
mycam = cameraboard(mypi,'Resolution', '640x480');
```

```
figure(1);
```

```
roi = [0 0 1 1];
```

```
mycam.ROI = [0 0 1 1];
```

```
for i = 1:300
```

```
    img = snapshot(mycam);
```

```
    if i > 20
```

```
        fc = (i - 5)*0.0025;
```

```
        roi(1:2) = [fc, fc];
```

```
        roi(3:end) = [1-fc, 1-fc];
```

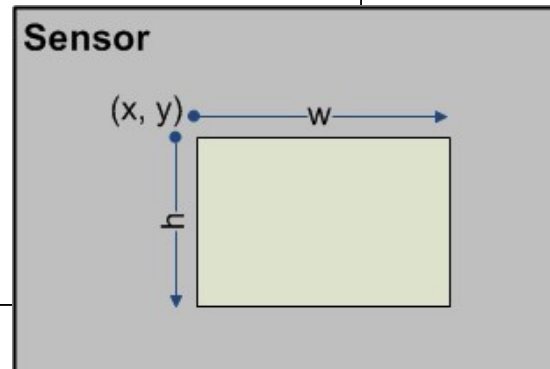
```
        mycam.ROI = roi;
```

```
        image(img);
```

```
        drawnow;
```

```
    end
```

```
end
```



## 6. 웹캠 실습

- 웹캠 실습 준비



<- Run time error 발생

**바로 컴퓨터에 직접연결**

카메라 활용만 확인

# 6. 웹캠 실습

- 웹캠 실습 테스트 (명령 창)

```
>> cam = webcam
>> webcamlist
>> cam.Resolution = '320x240';
>> preview(cam)
>> cam.Resolution = '640x480';
>> preview(cam)
>> img = snapshot(cam);
>> image(img)
```

- 웹캠 실습 코드 예제

```
clc; clear all; close all

%%
cam = webcam;

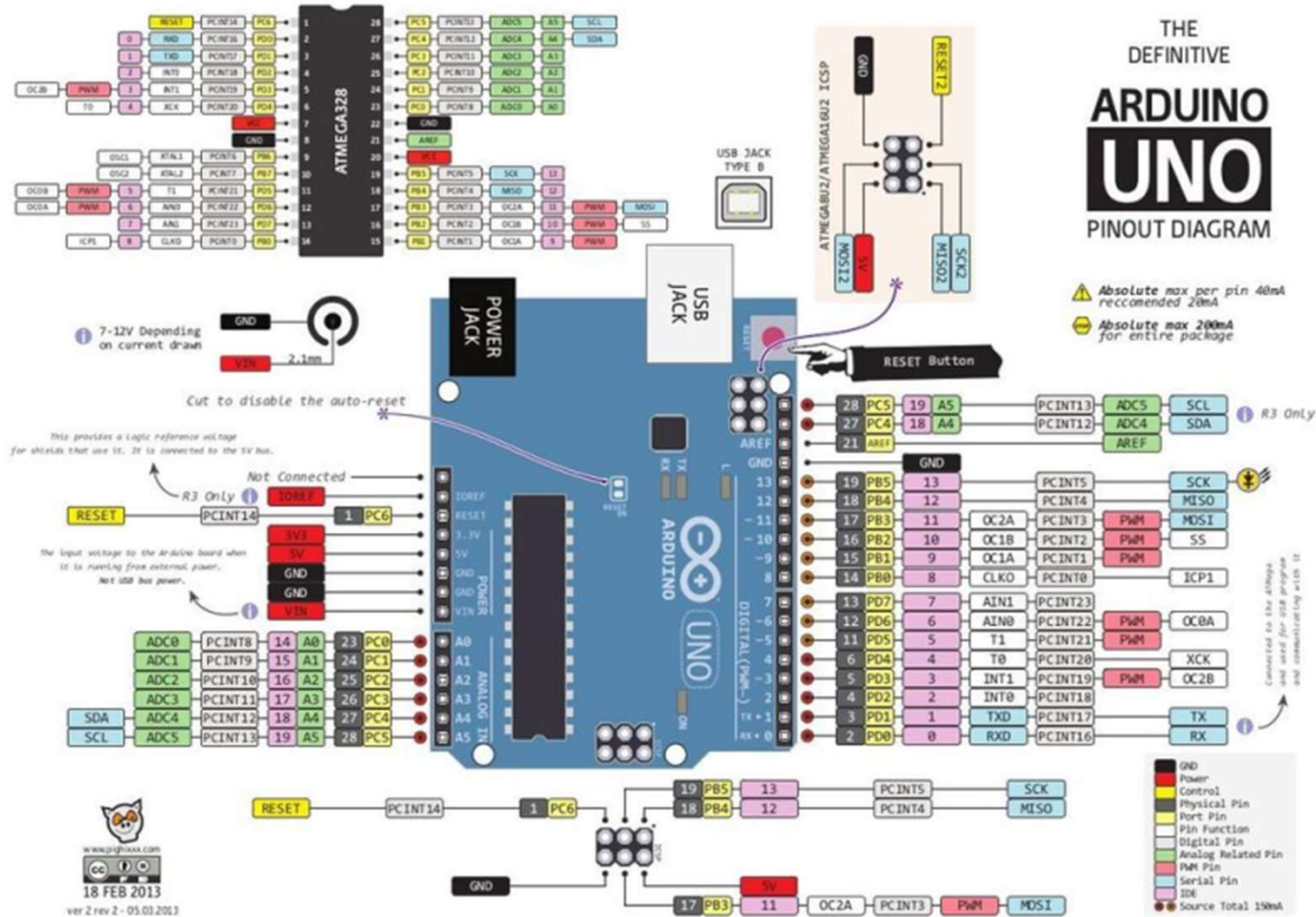
for i = 1:100
    img = snapshot(cam);
    image(img);
    drawnow;
end
```

Possible Device-Specific Properties	Description
BacklightCompensation	Configures backlight compensation to adjust the camera to capture images dependent on environmental conditions. The valid values are specified as a double.
Brightness	Indicates the brightness level, which adjusts for the amount of lighting on the image.
Contrast	Indicates contrast level, which adjusts for the difference between brightest and dimmest areas in the image.
ColorEnable	Specifies the color enable setting. Values are on and off.
Gain	Indicates a multiplier for the RGB color values. The value 0 is normal. Positive values are brighter and negative values are darker.
Gamma	Indicates gamma measurement.
Hue	Indicates hue setting, which adjusts the color tint of the image through the red-yellow-blue spectrum.
PowerLineFrequency	Reduces flicker caused by the frequency of a power line.
Saturation	Indicates saturation level, which adjusts the amount of color in the image.
Sharpness	Indicates sharpness level, which adjusts the clarity of the image.
WhiteBalance	Indicates color temperature in degrees Kelvin.
Pan	Control panning, measured in degrees.
Tilt	Control tilting, measured in degrees.
Roll	Control rolling, measured in degrees.
Zoom	Control zooming, measured in millimeters.
Exposure	Specify exposure to fine-tune the highlight and shadow details in the image.
Iris	Specify iris setting, in units of f-stop x 10.
Focus	Set focus, as the distance to the optimally focused target, in millimeters.

**Arduino**

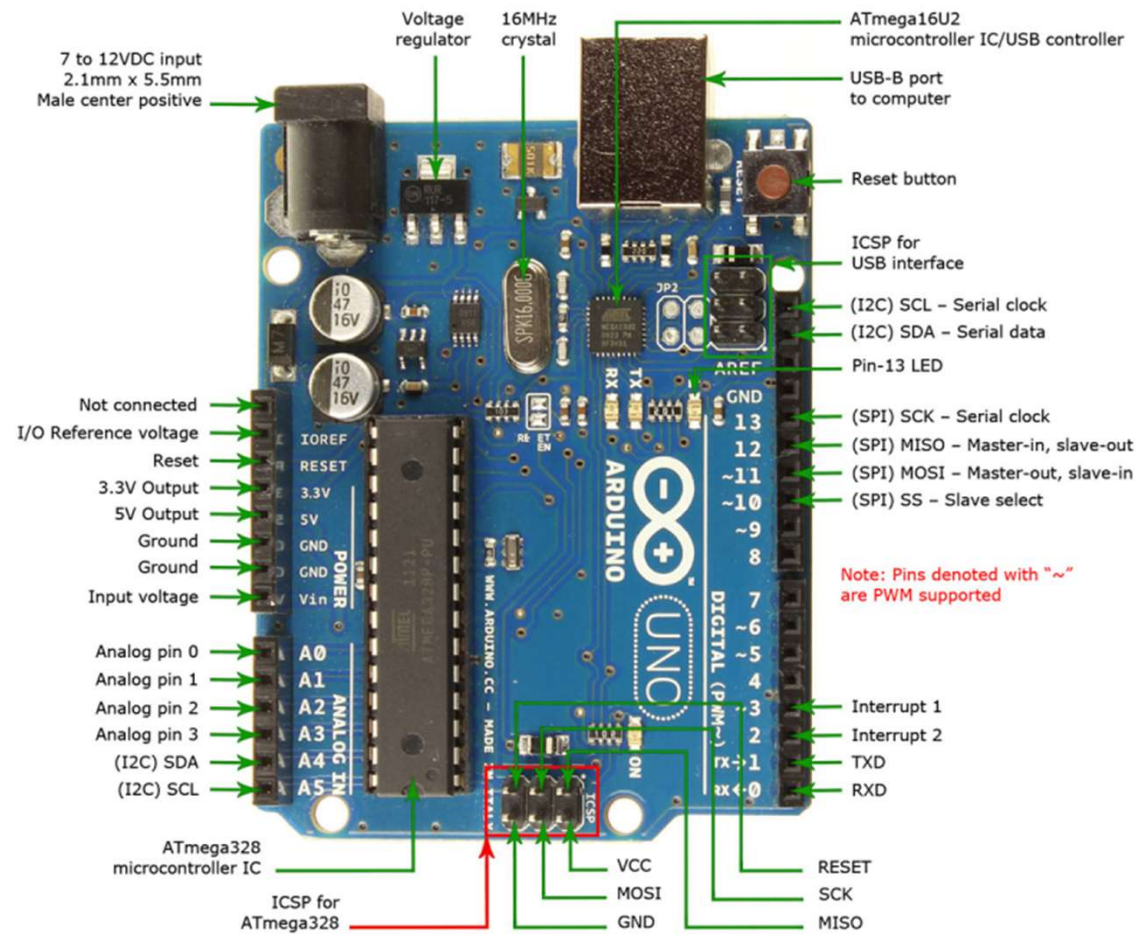
# 1. 아두이노 우노 보드 핀 맵

- 아두이노 우노 보드 핀 맵



# 1. 아두이노 우노 보드 핀 맵

## • 아두이노 우노 보드 핀 맵





## 2. 아두이노 IDE 코딩

### • 아두이노 IDE 동작 방식

// 전역변수를 선언할 수 있다.

```
void setup() {  
    // setup()는 아두이노 IDE를 시작을 불러온다.  
    /*  
    * 해당 블록안에서는 pinMode(), 라이브러리 등을 사용할 수 있다.  
    * setup은 오직 한 번만 실행되고, 아두이노 보드를  
    * 실행 혹은 리셋을 시킬 수 있다.  
    */  
}
```

```
void loop() {  
    /*  
    * setup블록이 생성된 이후에,  
    * 값을 초기화 시켜주고,  
    * loop(), 즉 무한 반복을 하면서 아두이노 보드를 컨트롤해준다.  
    */  
}
```

```
1 void setup(){  
2     // 11번 핀을 OUTPUT으로 설정합니다  
3     pinMode(11,OUTPUT);  
4 }  
5  
6 void loop(){  
7     // 11번 핀에 1(HIGH) 신호를 보냅니다. : HIGH는 전류를 흘려보내는 걸 의미합니다  
8     digitalWrite(11,1);  
9 }
```

```
1 void setup(){  
2     // Serial 모니터를 9600 bps로 설정합니다  
3     Serial.begin(9600);  
4     // 2번 핀을 INPUT으로 설정합니다  
5     pinMode(2,INPUT);  
6 }  
7  
8 void loop(){  
9     // 2번핀에 HIGH가 입력되면 OFF, LOW가 입력되면 ON을 출력하는 코드를 작성합니다  
10    if(!digitalRead(2)){  
11        Serial.print("ON\n");  
12    }  
13    else{  
14        Serial.print("OFF\n");  
15    }  
16 }
```

## 2. 아두이노 IDE 코딩

- 아두이노 함수

### 1) 매개변수 X, 반환값 X

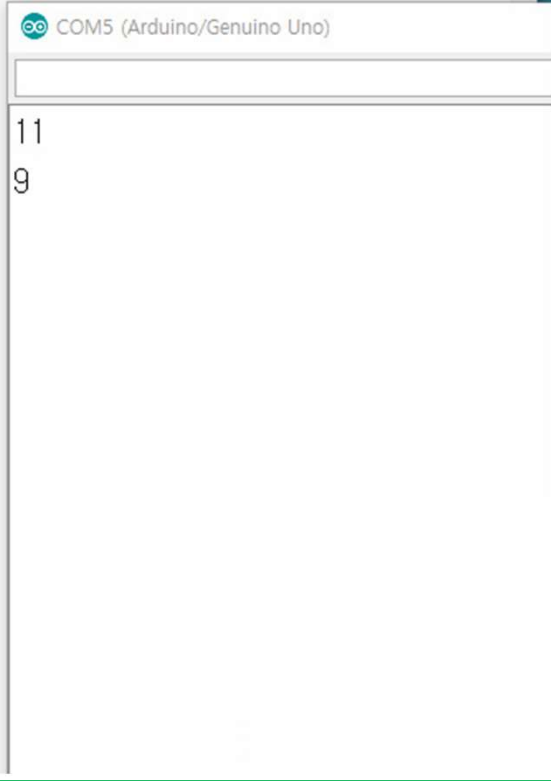
```
sketch_nov29b $  
void setup() {  
  Serial.begin(9600);  
}  
  
void loop() {  
  show();  
  show();  
  show();  
  Serial.end();  
}  
  
void show() {  
  Serial.println("hello codingrun");  
}
```



The serial monitor window shows the output of the 'show()' function being called three times in the loop, resulting in the text 'hello codingrun' being printed three times.

### 2) 매개변수 O, 반환값 X

```
void setup() {  
  Serial.begin(9600);  
}  
  
void loop() {  
  sum(1, 10);  
  sum(2, 7);  
  Serial.end();  
}  
  
void sum(int a, int b){  
  int total;  
  total = a + b;  
  Serial.println(total);  
  return;  
}
```



The serial monitor window shows the output of the 'sum' function being called twice in the loop. The first call with parameters (1, 10) results in the output '11', and the second call with parameters (2, 7) results in the output '9'.



## 2. 아두이노 IDE 코딩

- 아두이노 함수

### 3) 매개변수 X, 반환값 O

```
sketch_nov29b $
void setup() {
  Serial.begin(9600);
}

void loop() {
  int returnvalue;
  returnvalue = ban();
  Serial.println(returnvalue);
  Serial.end();
}

int ban(){
  int a = 10;
  Serial.print("return value : ");
  return a;
}
```

COM5 (Arduino/Genuino Uno)

return value : 10

### 4) 매개변수 O, 반환값 O

```
sketch_nov29a $
void setup() {
  Serial.begin(9600);
}

void loop() {
  int total;
  total = sum(1, 10);
  Serial.println(total);
  total = sum(2, 7);
  Serial.println(total);
  Serial.end();
}

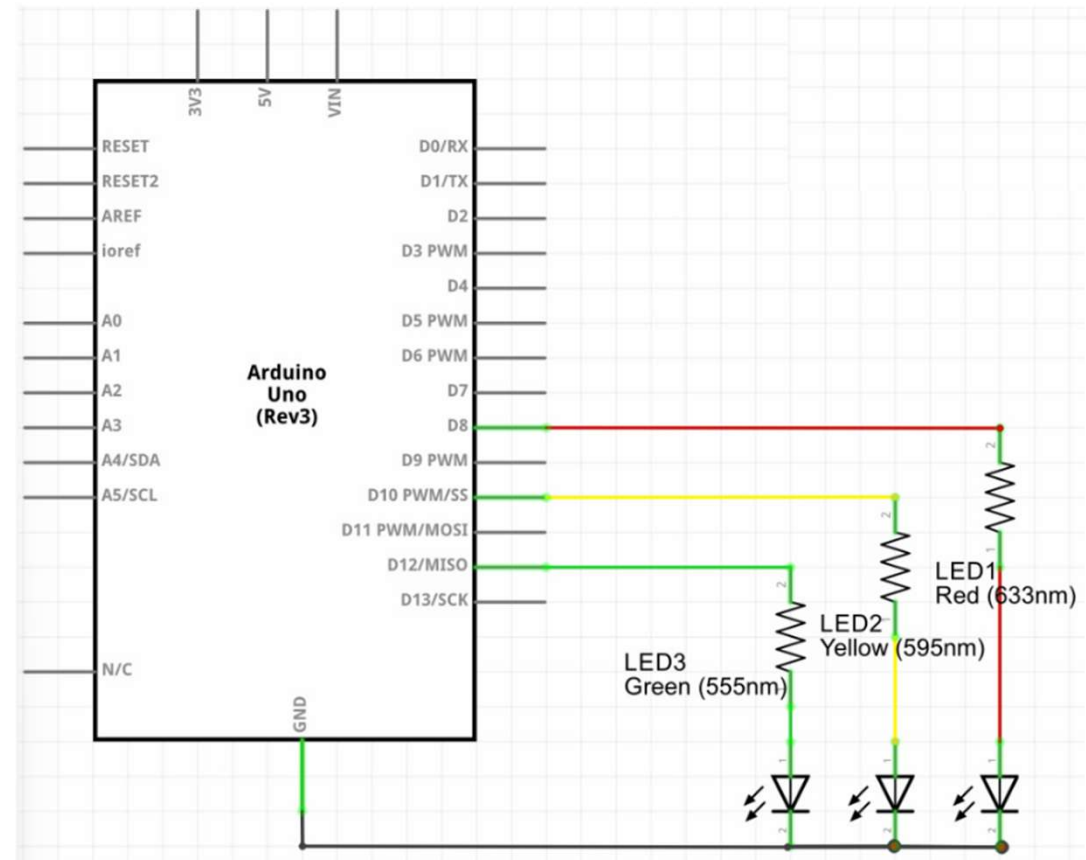
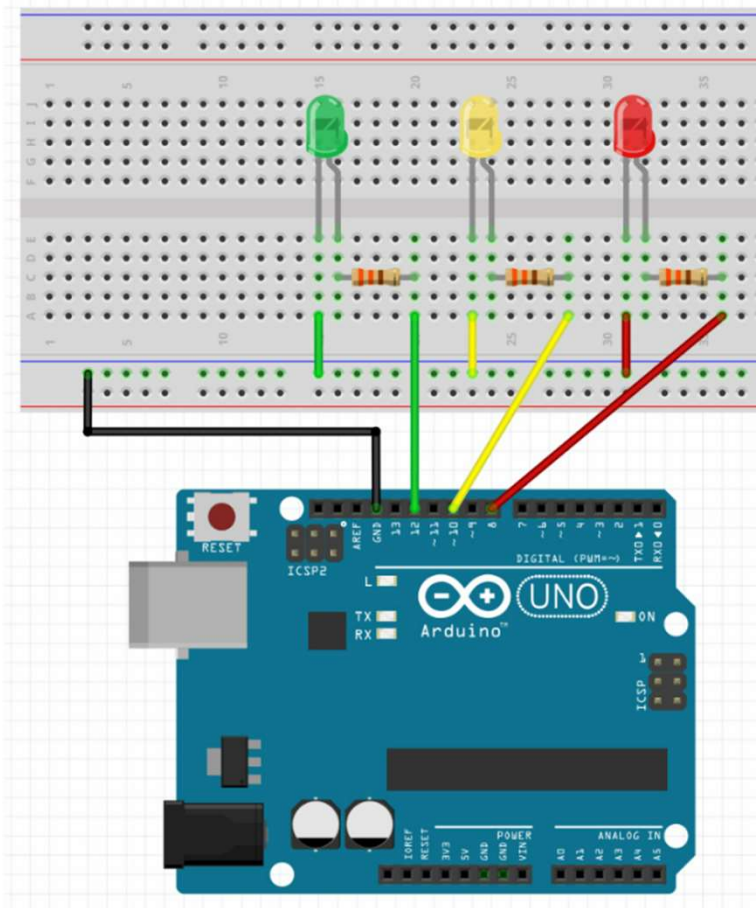
int sum(int a, int b){
  int total;
  total = a + b;
  return total;
}
```

COM5 (Arduino/Genuino Uno)

11  
9

### 3. LED/Pushbutton 실습

- LED/Pushbutton 실습 준비



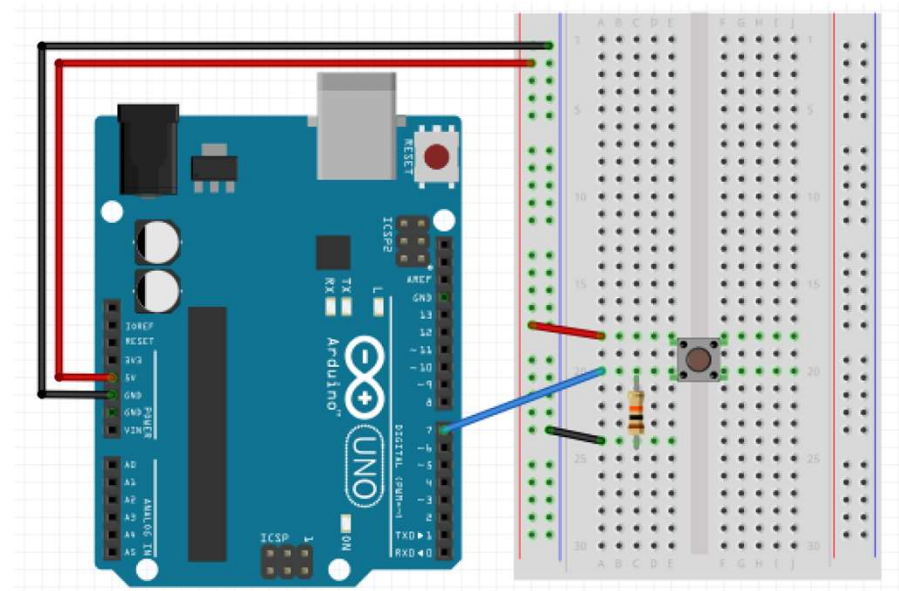
### 3. LED/Pushbutton 실습

- LED/Pushbutton 테스트

```
int green = 12; int yellow = 10; int red = 8;
```

```
void setup() {  
  pinMode(green, OUTPUT);  
  pinMode(yellow, OUTPUT);  
  pinMode(red, OUTPUT);  
}
```

```
void loop() {  
  digitalWrite(green, HIGH);  
  digitalWrite(yellow, LOW);  
  digitalWrite(red, LOW);  
  delay(100);  
  digitalWrite(green, LOW);  
  digitalWrite(yellow, HIGH);  
  digitalWrite(red, LOW);  
  delay(100);  
  digitalWrite(green, LOW);  
  digitalWrite(yellow, LOW);  
  digitalWrite(red, HIGH);  
  delay(100);  
}
```



```
int button = 6;
```

```
void setup() {  
  Serial.begin(9600);  
  pinMode(button, INPUT);  
}
```

```
void loop() {  
  int button_value = digitalRead(button);  
  Serial.println(button_value);  
}
```

## 4. LED/Pushbutton 실습

- LED/Pushbutton 코드 작성

- Pushbutton을 이용한 수동 신호 제어와 error 상황 판별

조건:

(1) 4개의 LED(빨, 초, 노, 흰), 3개의 Pushbutton 사용

(2) 각 Pushbutton은 3개의 LED(빨, 초, 노)를 제어 -> Pushbutton을 누르면 On

(3) Pushbutton을 눌렀을 때 serial monitor에서 on은 '1', off는 '0', error가 없으면 'ok' 표시

ex) 빨: 1, 초: 0, 노: 0, ok

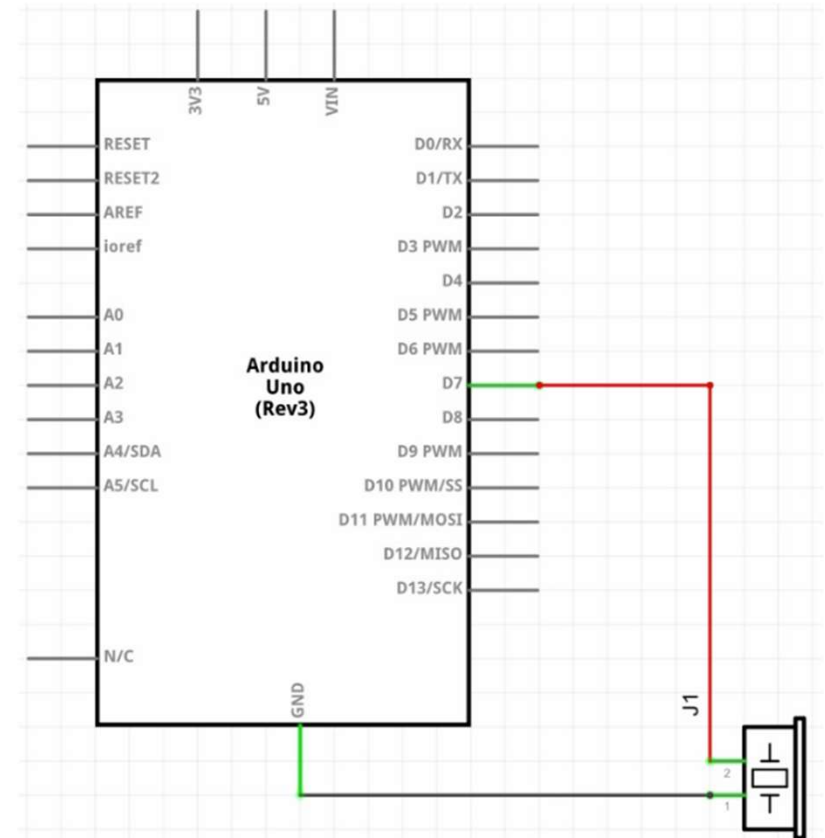
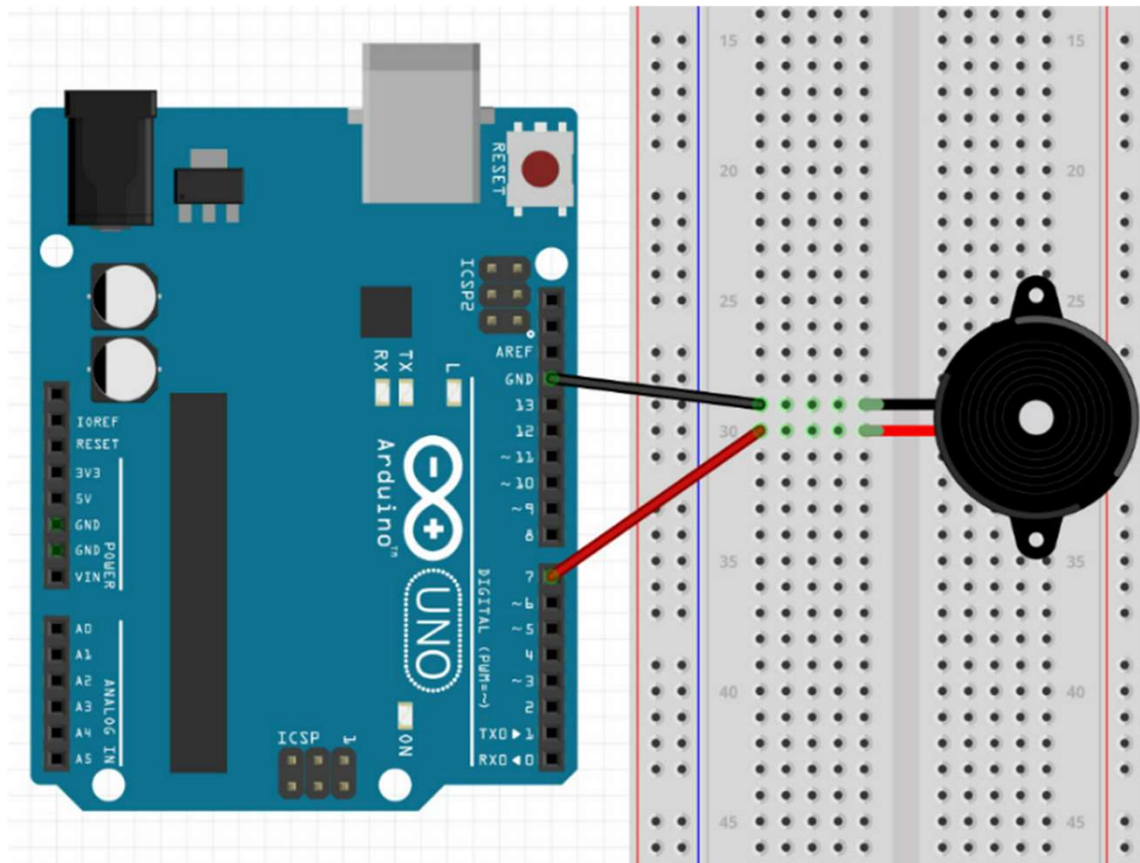
(4) Pushbutton을 동시에 2개 이상 누르면 흰색 LED가 켜짐

(5) Pushbutton을 동시에 2개 이상 눌렀다면 누른 색과 error를 serial monitor에 표시

ex) 빨: 1, 초: 1, 노: 0, error

## 4. Piezo buzzer 실습

- Piezo buzzer 실습 준비



## 4. Piezo buzzer 실습

### • Piezo buzzer 테스트

```
int piezo = 7;

void setup() {
  pinMode(piezo, OUTPUT);
  tone(piezo, 2093);
  delay(1000);

  noTone(piezo);
}

void loop() {
}
```

Piezo buzzer는  
1초 동안의 진동 횟수,  
즉 고유 주파수를 통해  
소리를 생성한다.

옥타브 및 음계별 표준 주파수

( 단위 : Hz )

음계 \ 옥타브	1	2	3	4	5	6	7	8
C(도)	32.7032	65.4064	130.8128	261.6256	523.2511	1046.502	2093.005	4186.009
C#	34.6478	69.2957	138.5913	277.1826	554.3653	1108.731	2217.461	4434.922
D(레)	36.7081	73.4162	146.8324	293.6648	587.3295	1174.659	2349.318	4698.636
D#	38.8909	77.7817	155.5635	311.1270	622.2540	1244.508	2489.016	4978.032
E(미)	41.2034	82.4069	164.8138	329.6276	659.2551	1318.510	2637.020	5274.041
F(파)	43.6535	87.3071	174.6141	349.2282	698.4565	1396.913	2793.826	5587.652
F#	46.2493	92.4986	184.9972	369.9944	739.9888	1479.978	2959.955	5919.911
G(솔)	48.9994	97.9989	195.9977	391.9954	783.9909	1567.982	3135.963	6271.927
G#	51.9130	103.8262	207.6523	415.3047	830.6094	1661.219	3322.438	6644.875
A(라)	55.0000	110.0000	220.0000	440.0000	880.0000	1760.000	3520.000	7040.000
A#	58.2705	116.5409	233.0819	466.1638	932.3275	1864.655	3729.310	7458.620
B(시)	61.7354	123.4708	246.9417	493.8833	987.7666	1975.533	3951.066	7902.133

## 4. Piezo buzzer 실습

- Piezo buzzer 코드 작성

- 옥타브 및 음계별 표준 주파수를 활용한 음악 재생 시스템

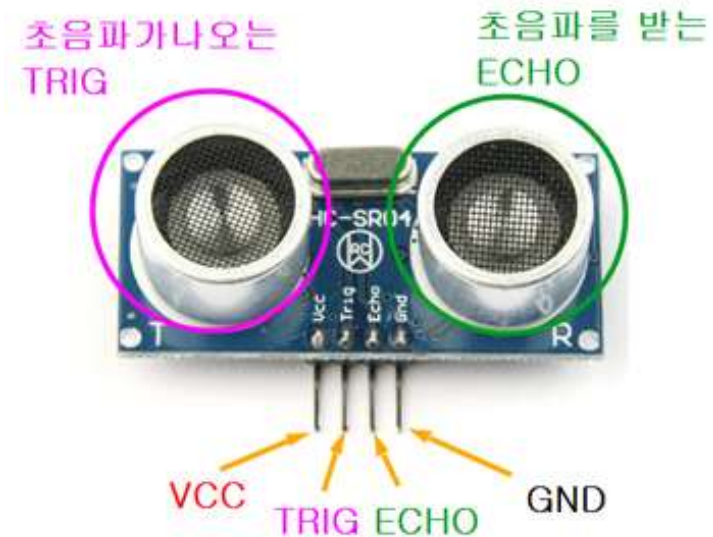
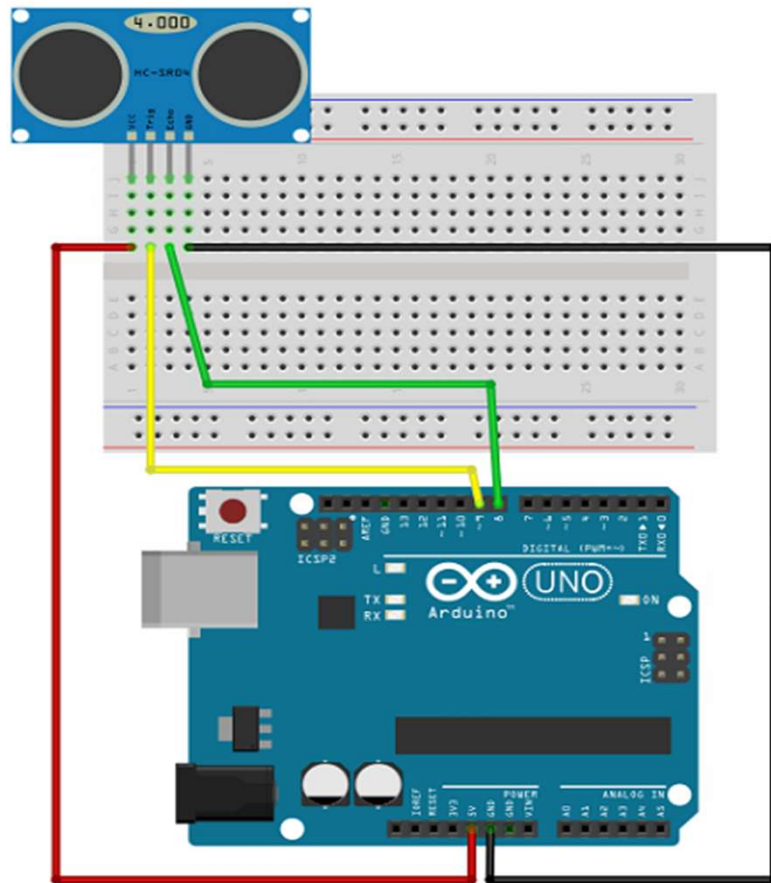
조건:

- (0) 음악은 아두이노의 함수를 사용하여 작성하기
- (1) piezo buzzer와 LED, pushbutton을 1개씩 사용하기
- (2) pushbutton을 누르면 음악 재생
- (3) 음악 재생 중에 pushbutton를 또 누르면 음악 정지
- (4) 정지 상태에서 pushbutton를 다시 누르면 음악 처음부터 재생
- (5) 본인이 정한 옥타브에서 '파' 이상의 값이 동작될 경우 LED on
- (6) 음악이 재생되는 동안 serial monitor에서 음악의 주파수 값을 디스플레이



## 5. 초음파 센서 실습

- 초음파 센서 실습 준비



- 1) 측정범위: 2cm ~ 4M
- 2) 측정각도: 15도
- 3) 사용전압: 5V
- 4) 사용전류 15mA



## 5. 초음파 센서 실습

### • 초음파 센서 테스트

#### define VS int

상수표현
10
#define A 10
const int B = 10;

#### long VS int (+ unsigned)

자료형		크기	표현범위	
			최소	최대
문자형	char	1바이트	아스키코드	
정수형	byte	1바이트	0	255
	int	2바이트	-32768	32767
	long	4바이트	-2147483648	2147483647
실수형	float	4바이트	-3.4028235E+38	3.4028235E+38
	double	4바이트	-3.4028235E+38	3.4028235E+38

```
#define TRIG 9
#define ECHO 8

void setup() {
  Serial.begin(9600);
  pinMode(TRIG, OUTPUT);
  pinMode(ECHO, INPUT);
}

void loop() {
  long duration, distance;

  digitalWrite(TRIG, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG, LOW);

  duration = pulseIn(ECHO, HIGH);
  //340m/ms * 1000 * 측정시간 (ms) / 2
  distance = duration * 17 / 1000;

  Serial.println(duration);
  Serial.print("Distance : ");
  Serial.print(distance);
  Serial.println("Cm");
}
```

## 5. 초음파 센서 실습

- 초음파 센서 코드 작성

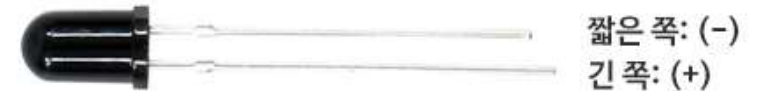
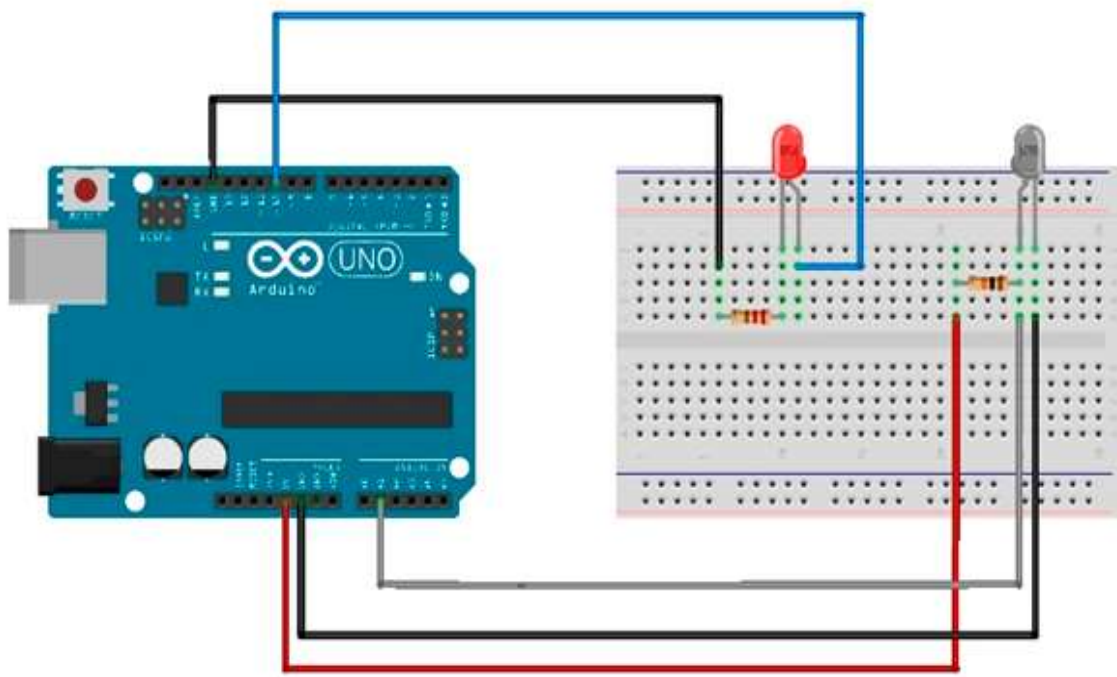
- 초음파 센서를 활용한 거리 감지와 위급 상황 알림 시스템

조건:

- (1) piezo buzzer와 LED 2개 사용, 초음파 센서 사용
- (2) 초음파 센서의 거리를 cm 단위로 serial monitor에 디스플레이  
ex) **288 cm**
- (3) 위급 상황용 LED와 정상 상태용 LED를 설치 후 처음에는 정상 상태용 LED on
- (4) 10cm 이하에서 거리 지정하여 해당 거리 이하의 값이 될 경우 위급 상황용 LED on
- (5) 동시에 piezo buzzer가 긴급 상황을 알림 (알림 음은 임의로)

## 6. 화염 감지 센서 실습

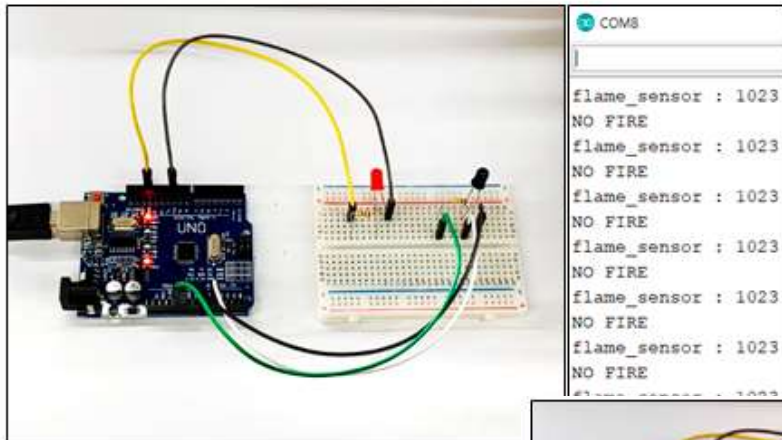
- 화염 감지 센서 실습 준비



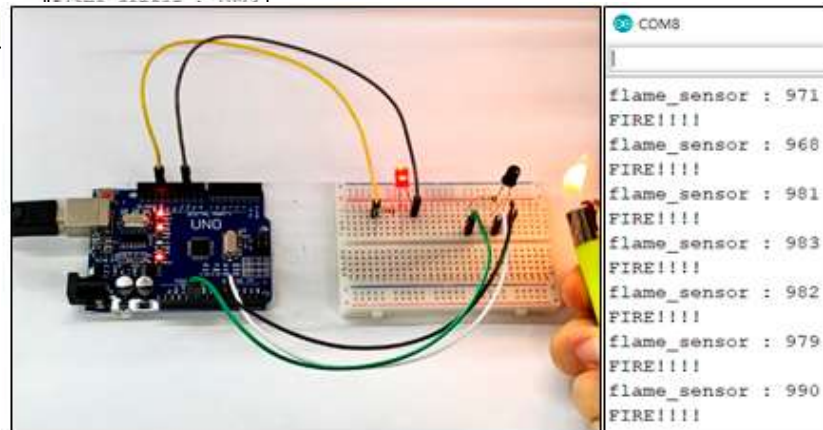
크기	37 * 5 mm
용도	불꽃의 파장 감지
동작전압	DC 3 ~ 5V
감지 파장	770nm ~ 1,300nm

## 6. 화염 감지 센서 실습

### • 화염 감지 센서 테스트



[ 정상시 ]



[ 불꽃이 감지될 때 ]

```
int flame = A1;  
int LED = 10;  
  
void setup() {  
  Serial.begin(9600);  
  pinMode(flame, INPUT);  
  pinMode(LED, OUTPUT);  
}  
  
void loop() {  
  int val = analogRead(flame);  
  Serial.print("flame_sensor : ");  
  Serial.println(val);  
  
  if(val < 1023) {  
    digitalWrite(LED, HIGH);  
    Serial.println("FIRE!!!!");  
  }  
  else {  
    digitalWrite(LED, LOW);  
    Serial.println("No FIRE");  
  }  
  delay(200);  
}
```

## 6. 화염 감지 센서 실습

- 화염 감지 센서 코드 작성

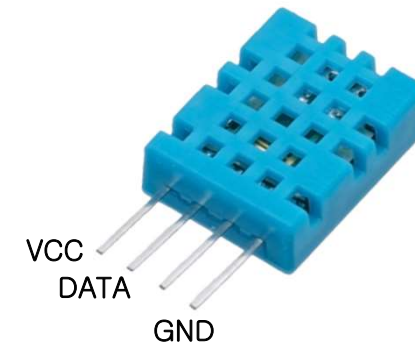
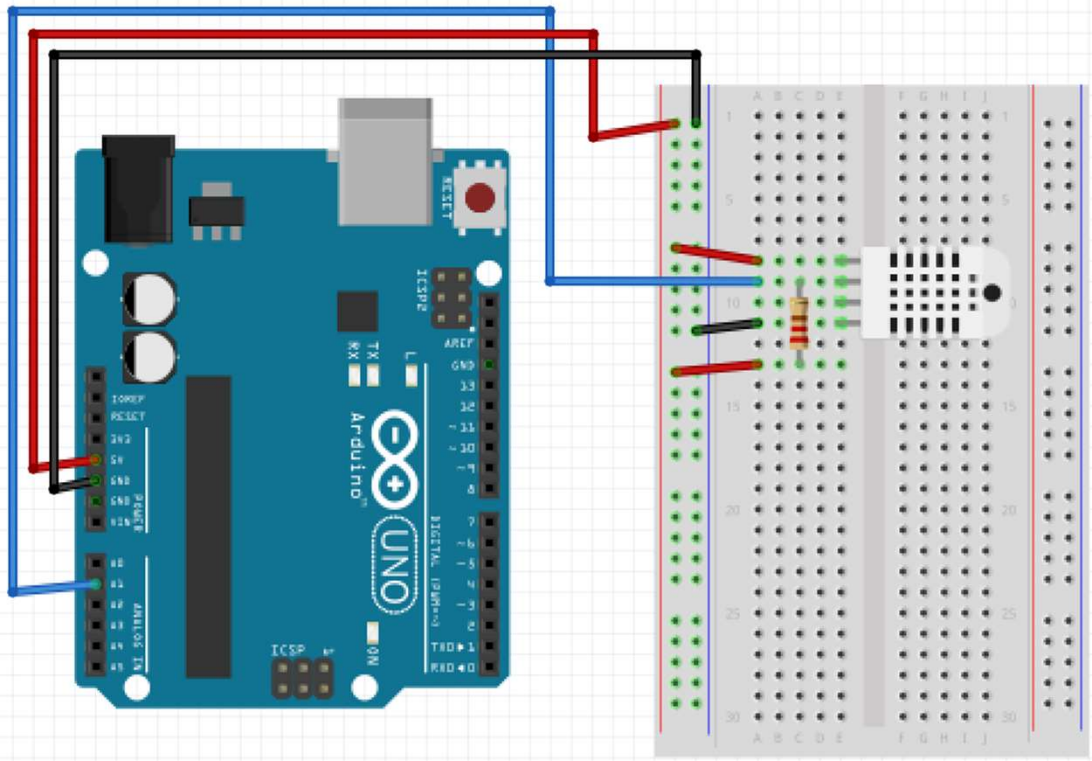
- 화염 감지 센서를 활용한 화재 경보 시스템

조건:

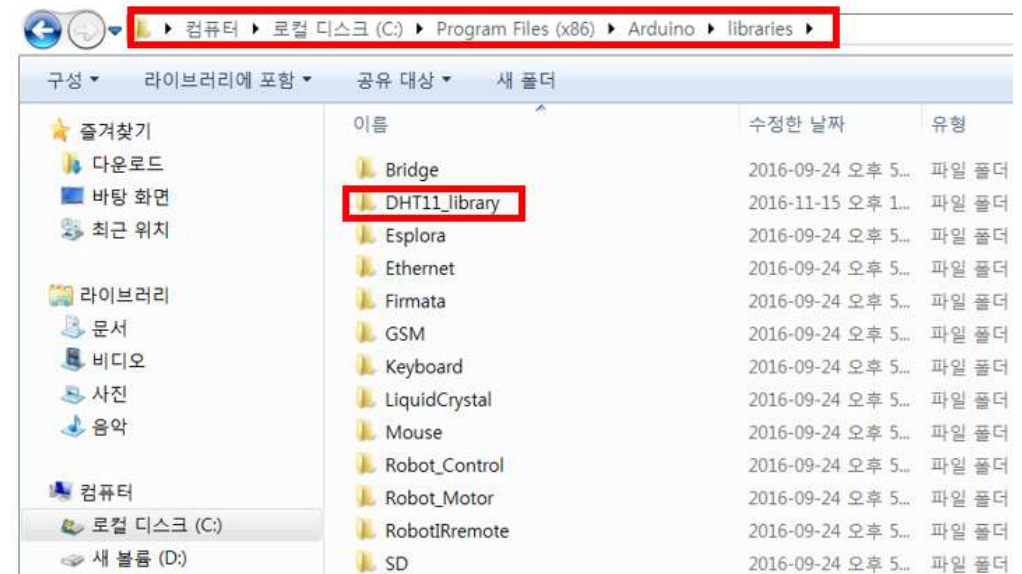
- (1) piezo buzzer와 LED 2개 사용, 화염 감지 센서 사용
- (2) 화염 감지 센서의 수치와 현재의 상태를 serial monitor에 디스플레이  
ex) 1023, 안전
- (3) 화재 경보용 LED와 정상 상태용 LED를 설치 후 처음에는 정상 상태용 LED on
- (4) 지정한 임계 값 이하의 값이 될 경우 화재 경보용 LED on
- (5) 동시에 piezo buzzer가 긴급 상황을 알림 (알림 음은 임의로)

## 7. DHT 실습

- DHT 실습 준비

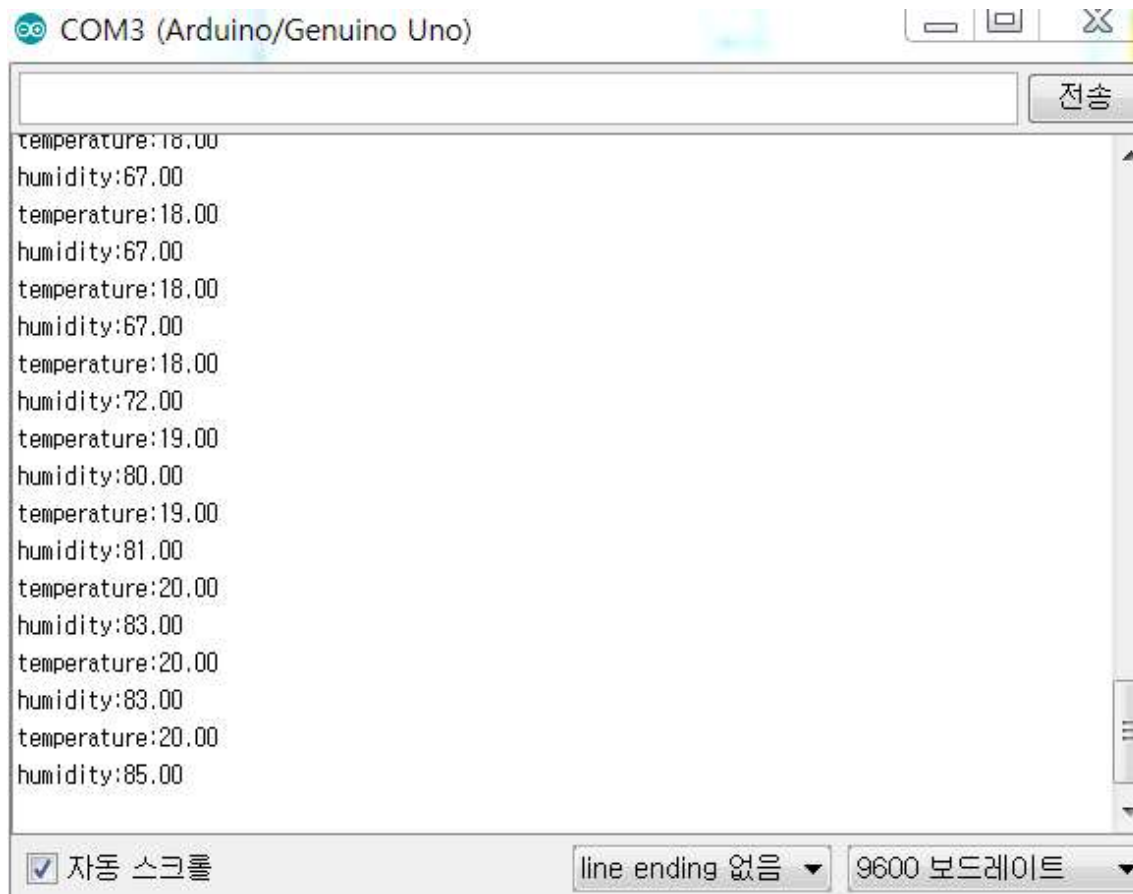


### DHT11 라이브러리 추가 – 아두이노 설치 경로에 넣기



## 7. DHT 실습

- DHT 테스트



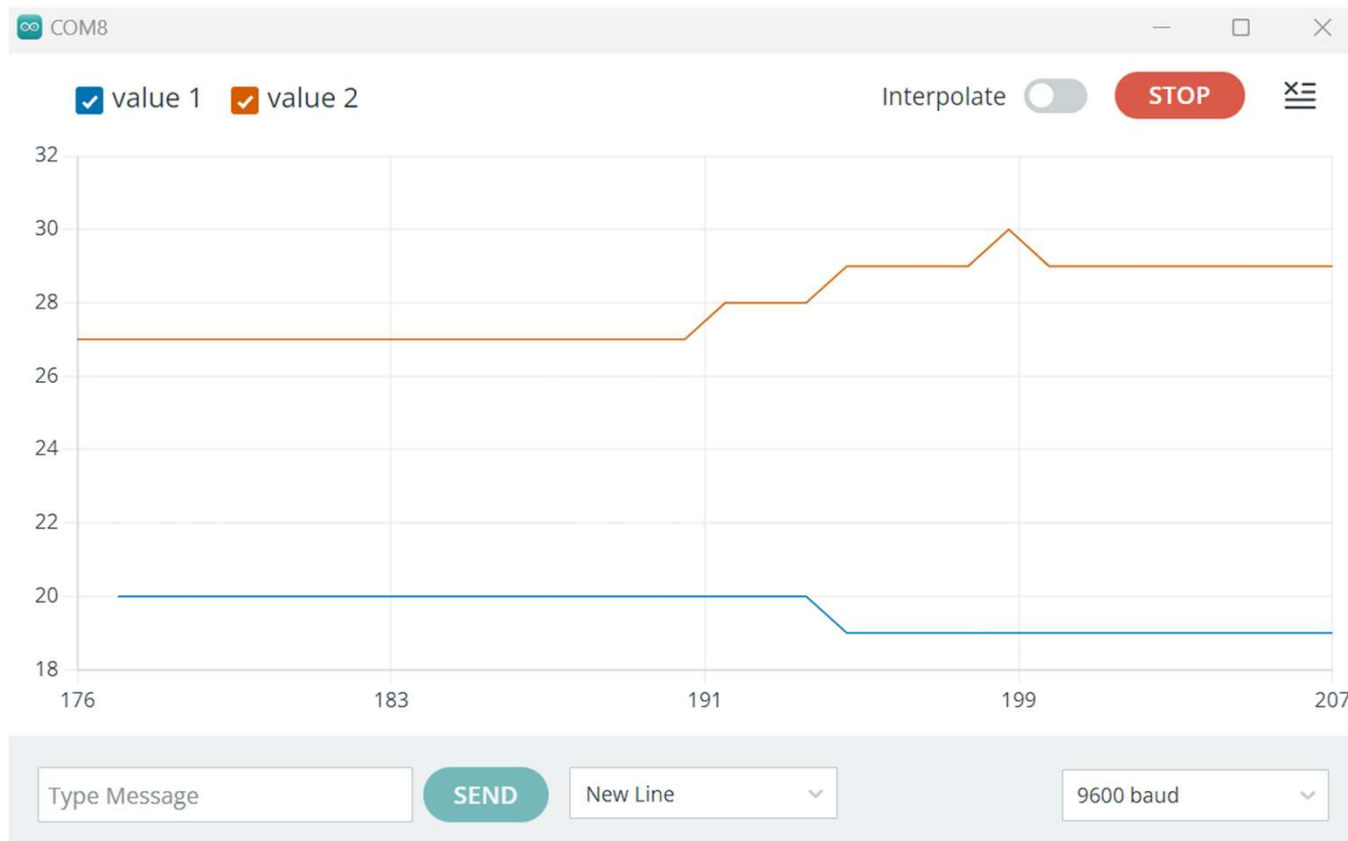
```
#include <DHT11.h>
int pin = A1;
DHT11 dht11(pin);

void setup() {
  Serial.begin(9600);
}

void loop() {
  int i;
  float humi, temp;
  if((i = dht11.read(humi, temp)) == 0) {
    Serial.print("humidity: ");
    Serial.println(humi);
    Serial.print("temperature: ");
    Serial.println(temp);
  }
  else {
    Serial.print("Error: ");
    Serial.print(i);
  }
  delay(2000);
}
```

# 7. DHT 실습

## • DHT 테스트



```
#include <DHT11.h>
int pin = A1;
DHT11 dht11(pin);

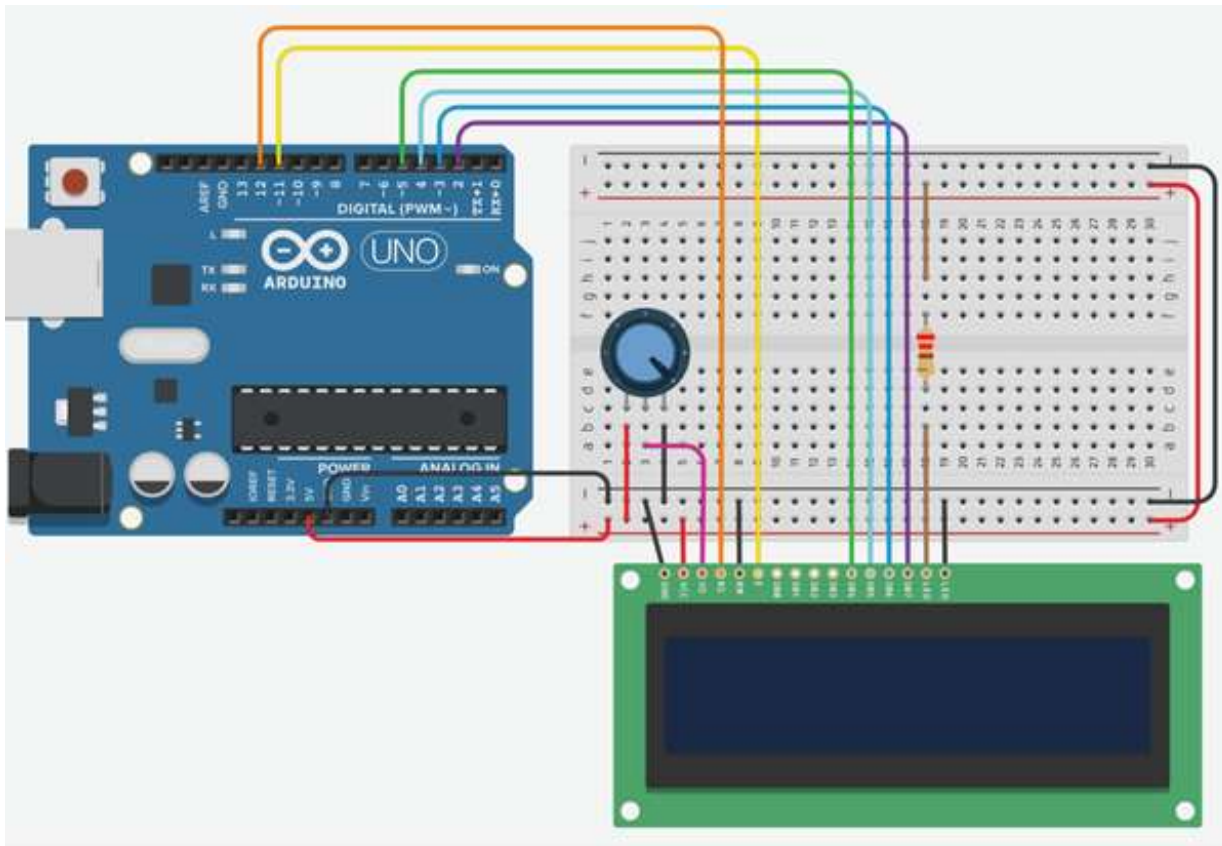
void setup() {
  Serial.begin(9600);
}

void loop() {
  int i;
  float humi, temp;
  if((i = dht11.read(humi, temp)) == 0) {
    Serial.print(humi);
    Serial.print(",");
    Serial.println(temp);
  }
  else {
    Serial.print("Error: ");
    Serial.print(i);
  }
  delay(2000);
}
```



## 8. 디스플레이 LCD 실습

- 디스플레이 LCD 실습 준비



액정 디스플레이(LCD)

## 8. 디스플레이 LCD 실습

### • 디스플레이 LCD 테스트

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  lcd.begin(16, 2);
  lcd.print("Hello, Arduino!!");
}

void loop() {
  lcd.setCursor(0, 1);
  lcd.print("1234567890");
}
```

```
1 #include <LiquidCrystal.h>
2
3 // 아두이노 연결된 핀번호로 LCD모듈 초기화
4 LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
5
6 void setup() {
7   lcd.begin(16, 2); // 16x2 LCD모듈 설정
8   Serial.begin(9600); // 시리얼 통신 초기화
9 }
10
11 void loop() {
12   int readValue = analogRead(A0); // 온도센서(TMP36) 값 측정
13   float voltage = readValue*5.0/1024.0; // 전압 값 변환
14   float temperature = voltage*100-50; // 온도 값 변환
15   String tempStr = String(temperature); // 온도 값을 문자열로 변환
16   lcd.print("TEMP: " + tempStr); // LCD에 문자열 출력
17   Serial.println(tempStr); // 시리얼 모니터에 문자열 출력
18
19   delay(500); // 500ms(밀리초) 지연
20   lcd.clear(); // LCD 초기화
21 }
```



## 8. 디스플레이 LCD 실습

- 디스플레이 LCD 코드 작성

- 온도와 습도/거리 측정 디스플레이 장치

조건:

(1) 초음파 센서와 DHT와 디스플레이 LCD 사용하기

(2) DHT로부터 받은 온도와 습도 정보를 serial monitor에 디스플레이

ex) **humidity: 28**

**temperature: 20**

(3) 온도, 습도 정보 이후에 초음파 센서의 측정 거리를 디스플레이

ex) **140**

**distance: 12cm**

**과제**

# 1. 과제

1) 아두이노 사용 -> 코드 작성 과제 5개

2) 추가 점수용

- 7가지 센서 중 5가지 센서를 사용한 종합 시스템

조건:

(1) LED, Pushbutton, Piezo buzzer, 초음파 센서, 화염 감지 센서, DHT, 디스플레이 중 5가지 사용

(2) 사용하는 센서가 유기적으로 연결되어 있어야 함. (trigger)

ex) LED, Pushbutton, Piezo buzzer, 초음파 센서, 화염 감지 센서를 사용한다고 했을 때

LED, Pushbutton, 초음파 센서 / Piezo buzzer, 화염 감지 센서가 따로 동작하면 안됨!

(3) 다음 주 자신이 만든 시스템 간략히 소개