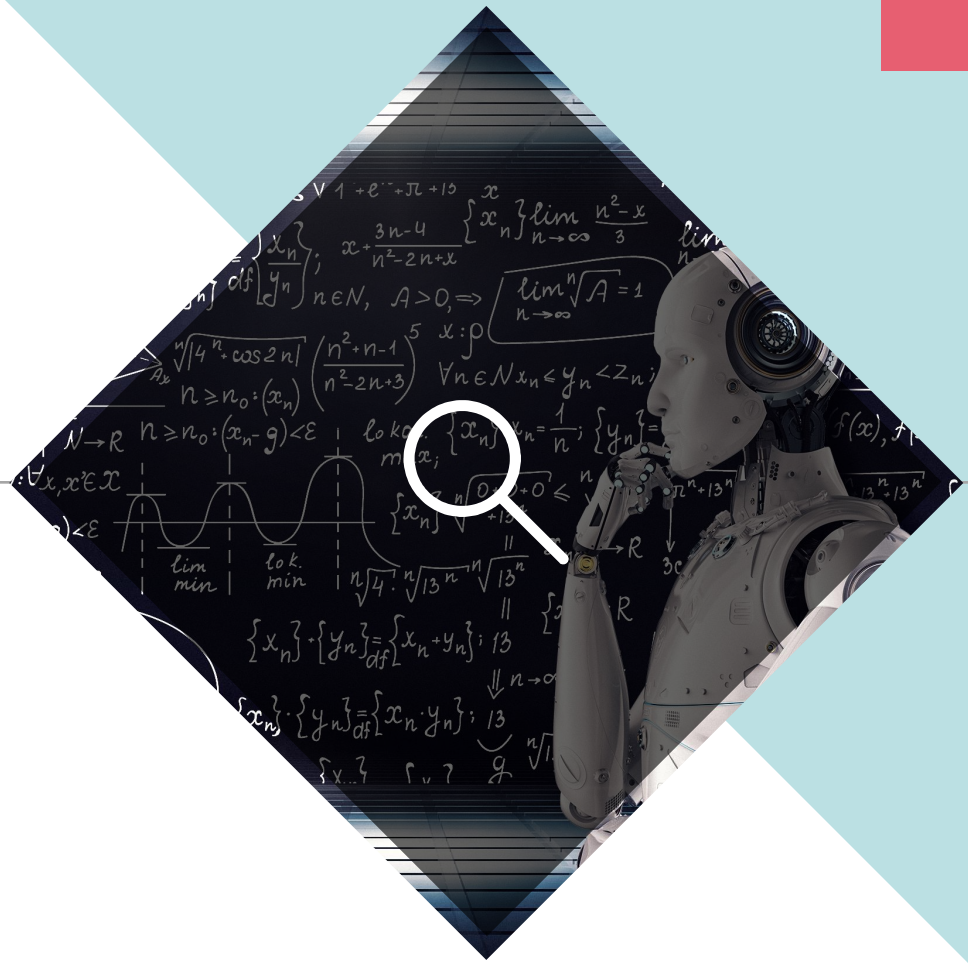


개발보드 실습
-빵판보드 이해

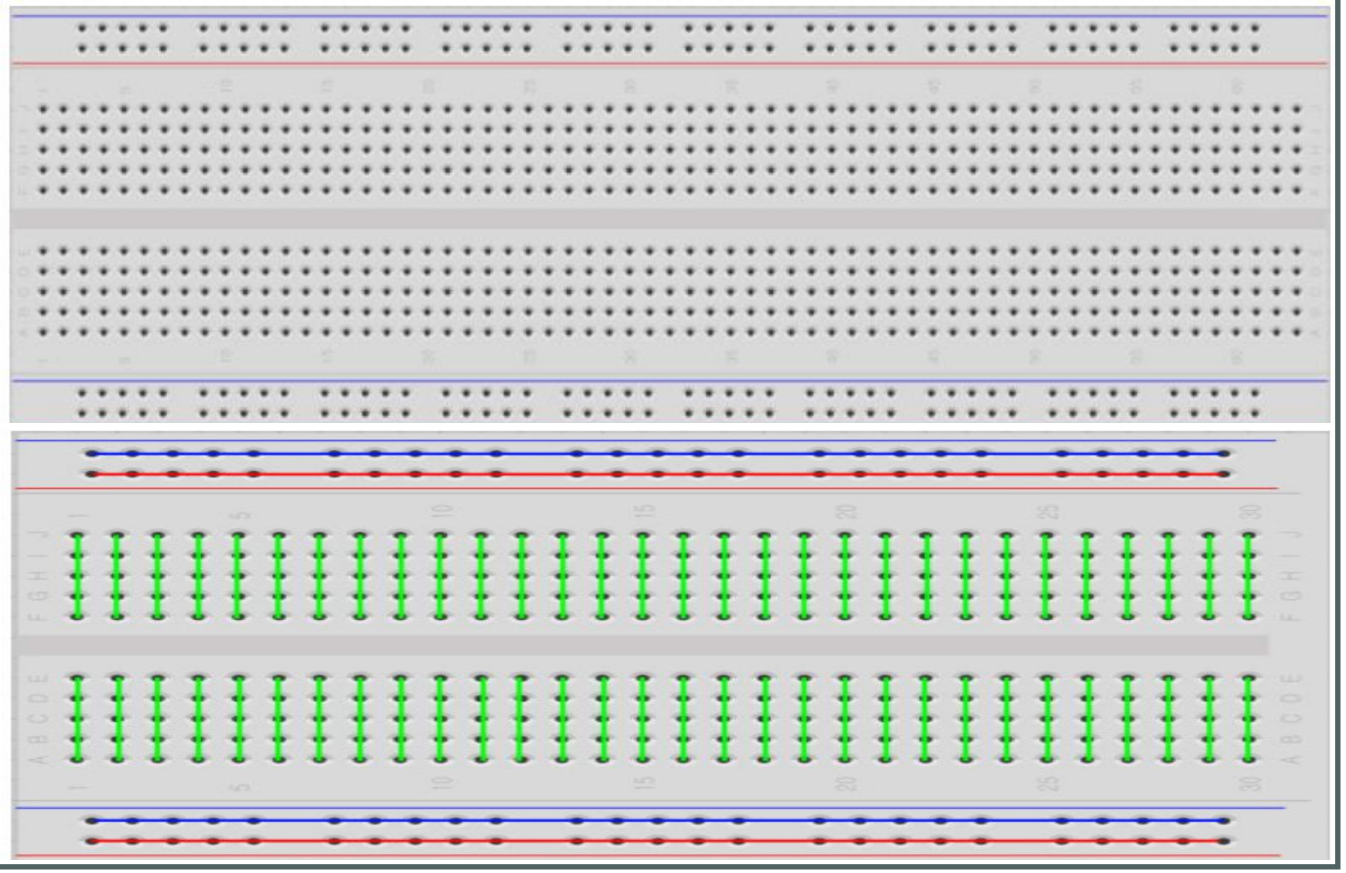
May 2025

BreadBoard 소개

Breadboard

: 빵판 또는 빵틀은 전자 회로의 시제품을 만드는데 사용하는 재활용할 수 있는 무납땜 장치. 버스 스트립으로 알려진 내부연결 전기단자의 스트립이 있고, 주장치 일부나 격리된 블록처럼 한쪽,양쪽은 전원선을 확장하도록 끼워져 있음. 현대의 무납땜 브레드보드는 천공아래 납이 도금된 스프링 클립이 있는 플라스틱 구멍뚫린 블록으로 구성됨. 두 개의 일련 패키지(Dual In-Line Package)인 직접회로(IC)는 블록의 중앙선을 벌려서 삽입, 내부연결 전선과 각각 부품 핀(축전기, 저항기, 코일 등)은 회로 위상을 완성하기 위해 남는 구멍에 연결할 수 있음

브레드보드의 외형 및 사용법
- 버스영역(버스띠)
  (파란색: -단자 or GND, 빨간색: + or 전원)
- IC영역/부품영역(단자띠)
  (녹색: 5칸씩 연결, 위아래 5칸 격리

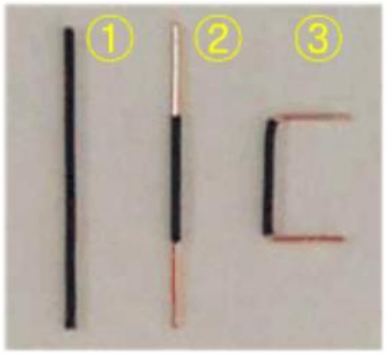


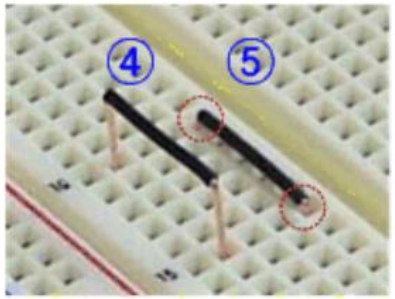
BreadBoard 부품 연결

1.전선 조립

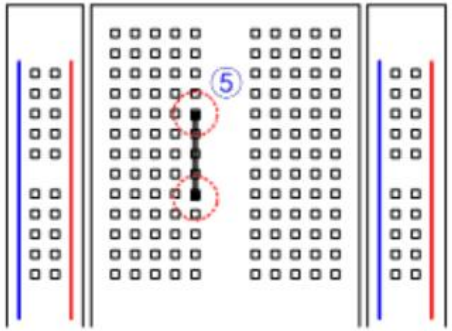
: 전선은 만들어 쓰는 방법이고, 기성품으로 가능

조립순서 : ①→②→③→④→⑤ (완성)





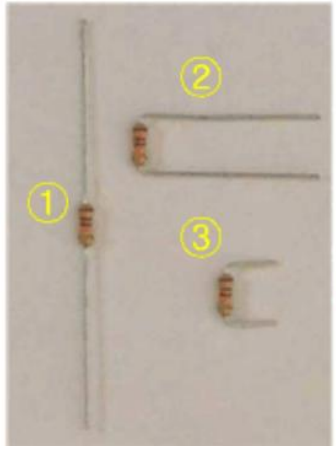
<실체도>

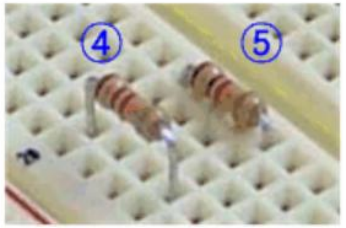


◯가 브레드보드와 연결된 부분입니다.

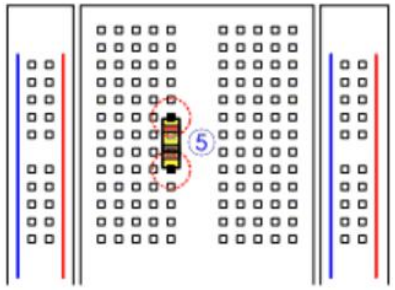
2.저항 조립

조립순서 : ①→②→③→④→⑤ (완성)





<실체도>

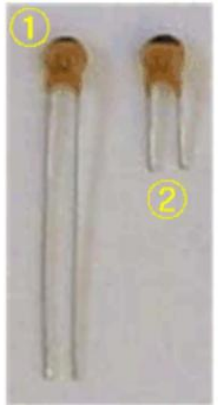


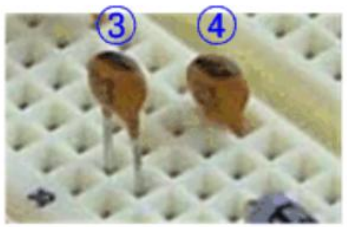
◯가 브레드보드와 연결된 부분입니다.

BreadBoard 부품 연결

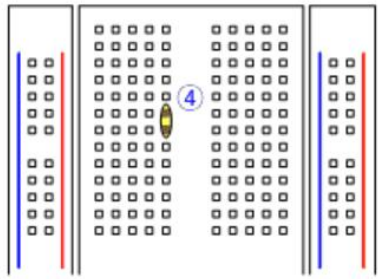
3.세라믹콘덴서 조립

조립순서 : ①→②→③→④ (완성)





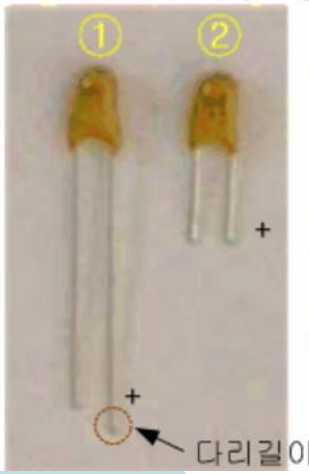
<실체도>

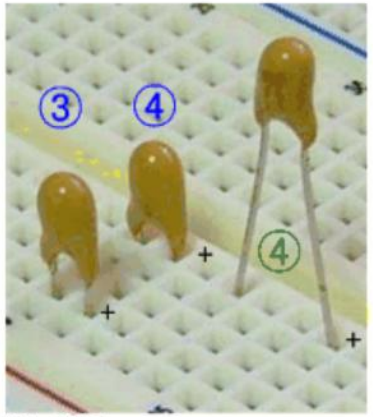


★ 세라믹 콘덴서는 극성의 구분이 없다

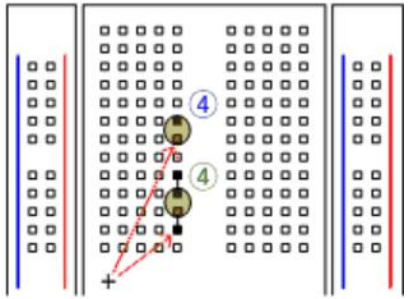
4.탄탈콘덴서, 전해콘덴서 조립

조립순서 : ①→②→③→④ (완성)

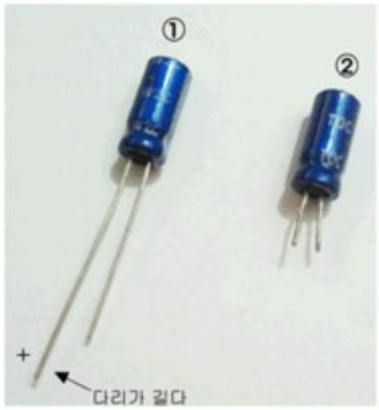


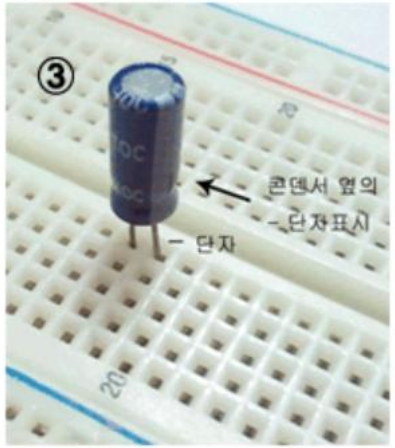


<실체도>

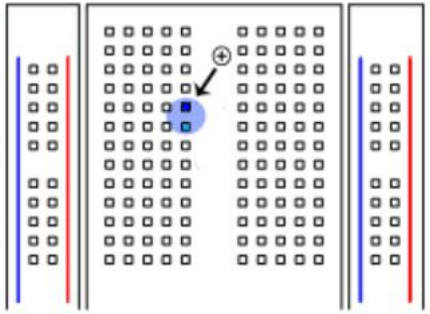


조립순서 ①→②→③ (완성)





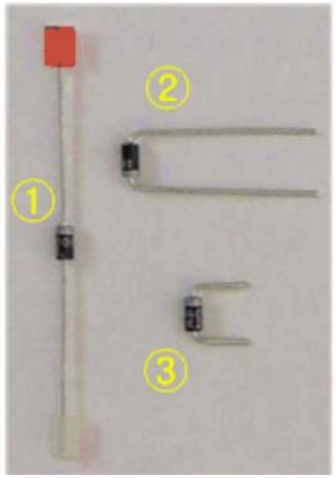
<실체도>



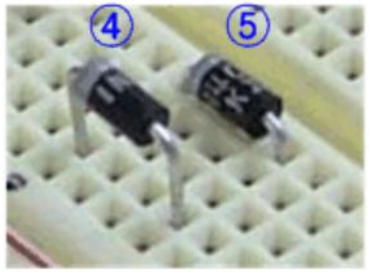
BreadBoard 부품 연결

5.다이오드 조립

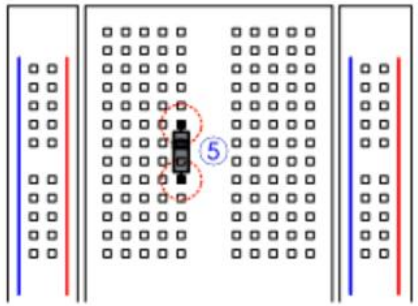
조립순서 : ①→②→③→④→⑤ (완성)



★ 부품방향에 주의!



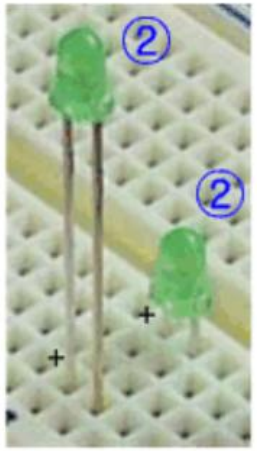
<실체도>



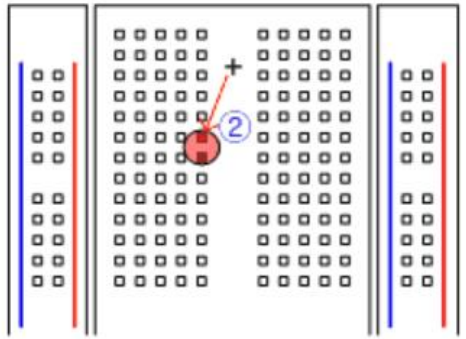
◯가 브레드보드와 연결된 부분입니다.

6.LED 조립

조립순서 : ①→② (완성)



<실체도>

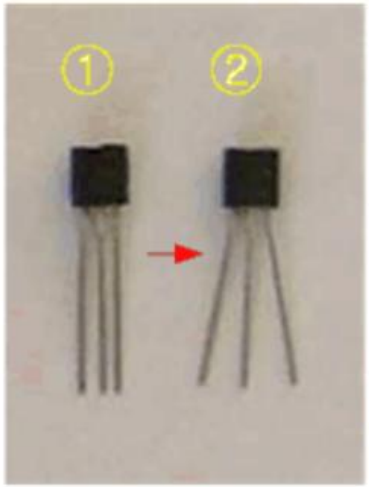


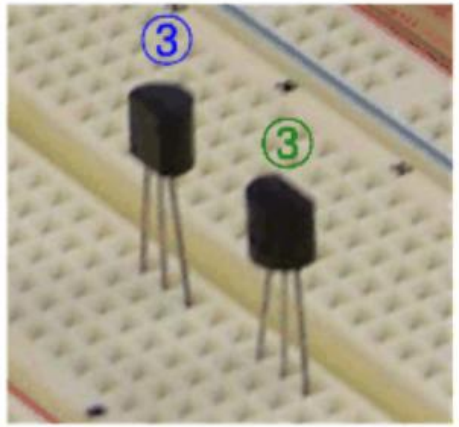
★ LED는 원 모양대로 사용하기도 한다

BreadBoard 부품 연결

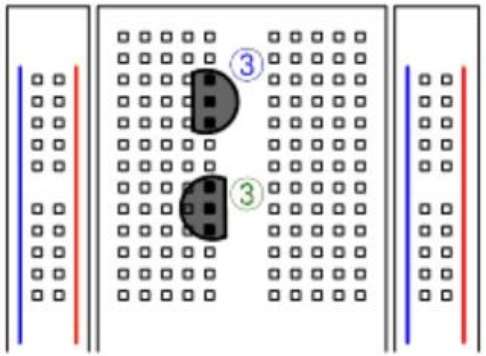
7.트랜지스터 조립

조립순서 : ①→②→③ (완성)





<실체도>

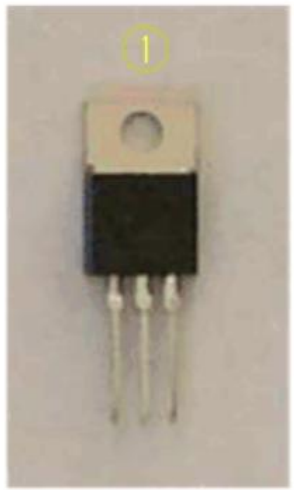


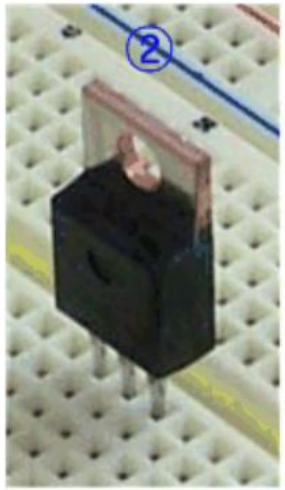
★ 부품방향에 주의!

8.대형 트랜지스터 조립

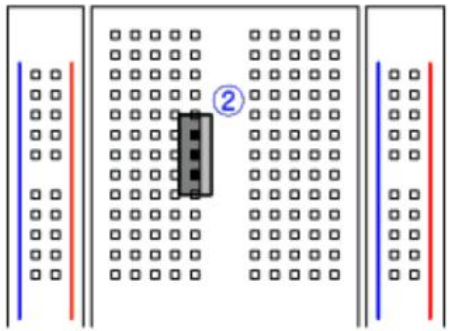
: 3단자 Regulator, Power Transister

조립순서 : ①→② (완성)





<실체도>

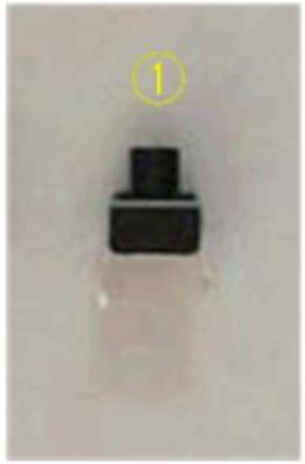


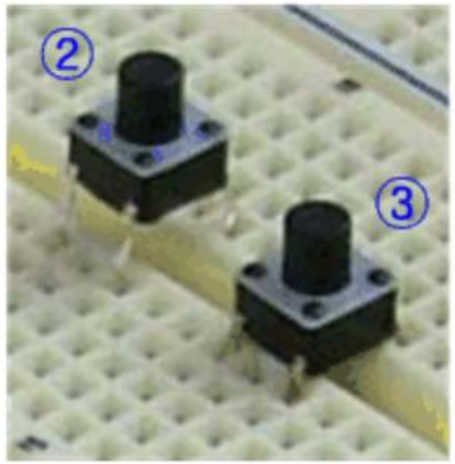
★ 부품은 원 모양대로 사용한다

BreadBoard 부품 연결

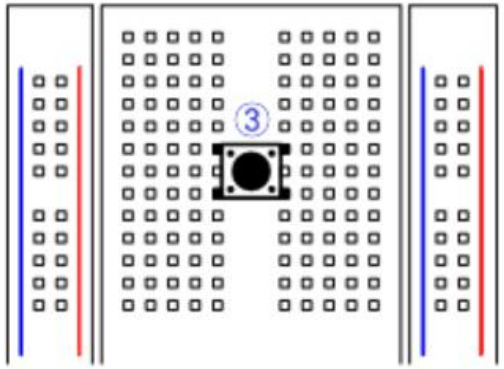
9.푸쉬스위치 조립

조립순서 : ①→②→③ (완성)





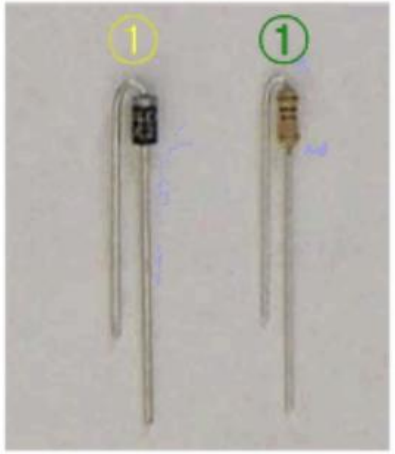
<실체도>

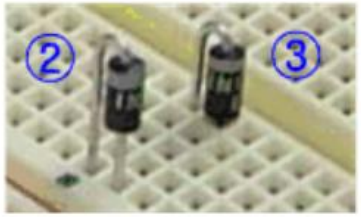


10.세로로 부품 조립

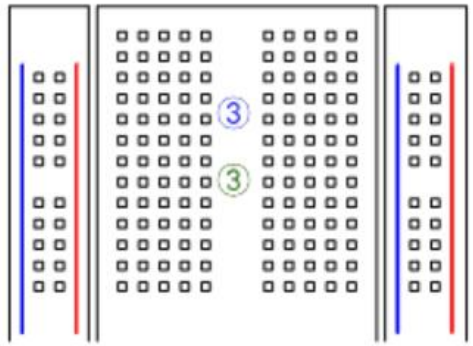
: 좁은 영역 사용시

조립순서 : ①→②→③ (완성)





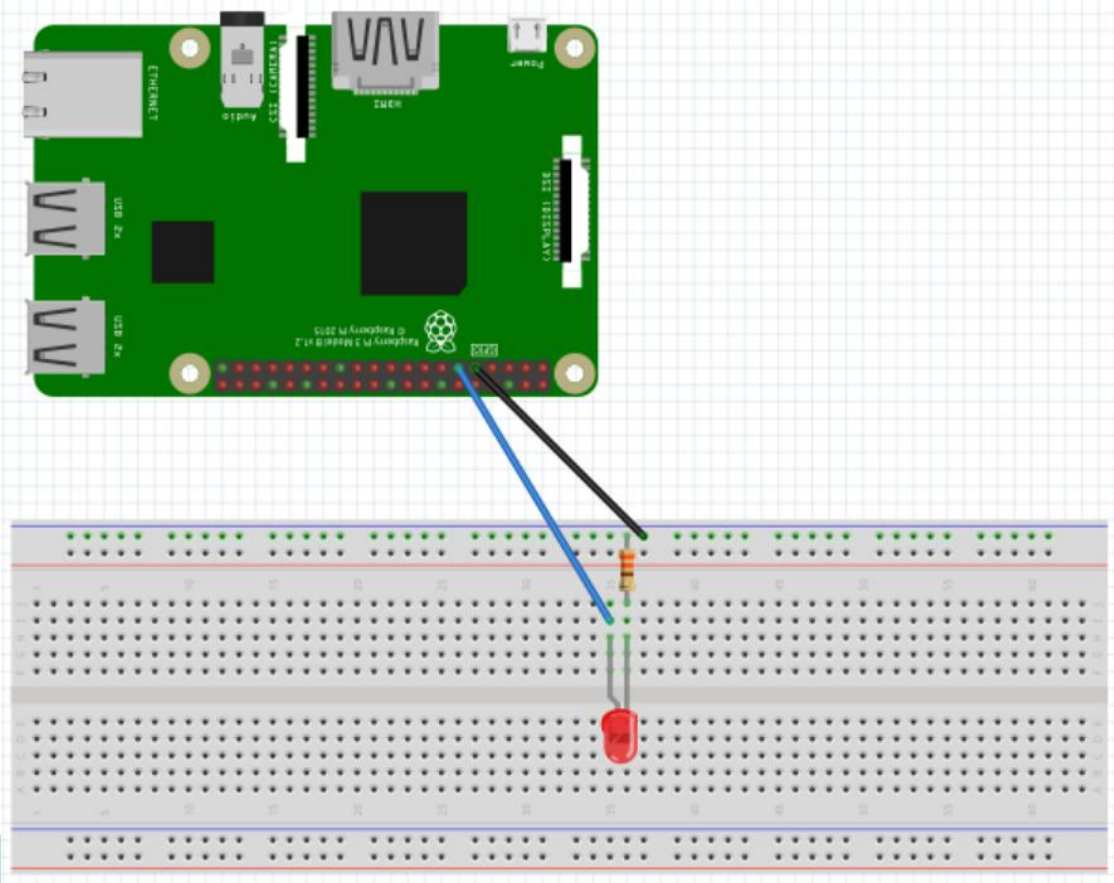
<실체도>

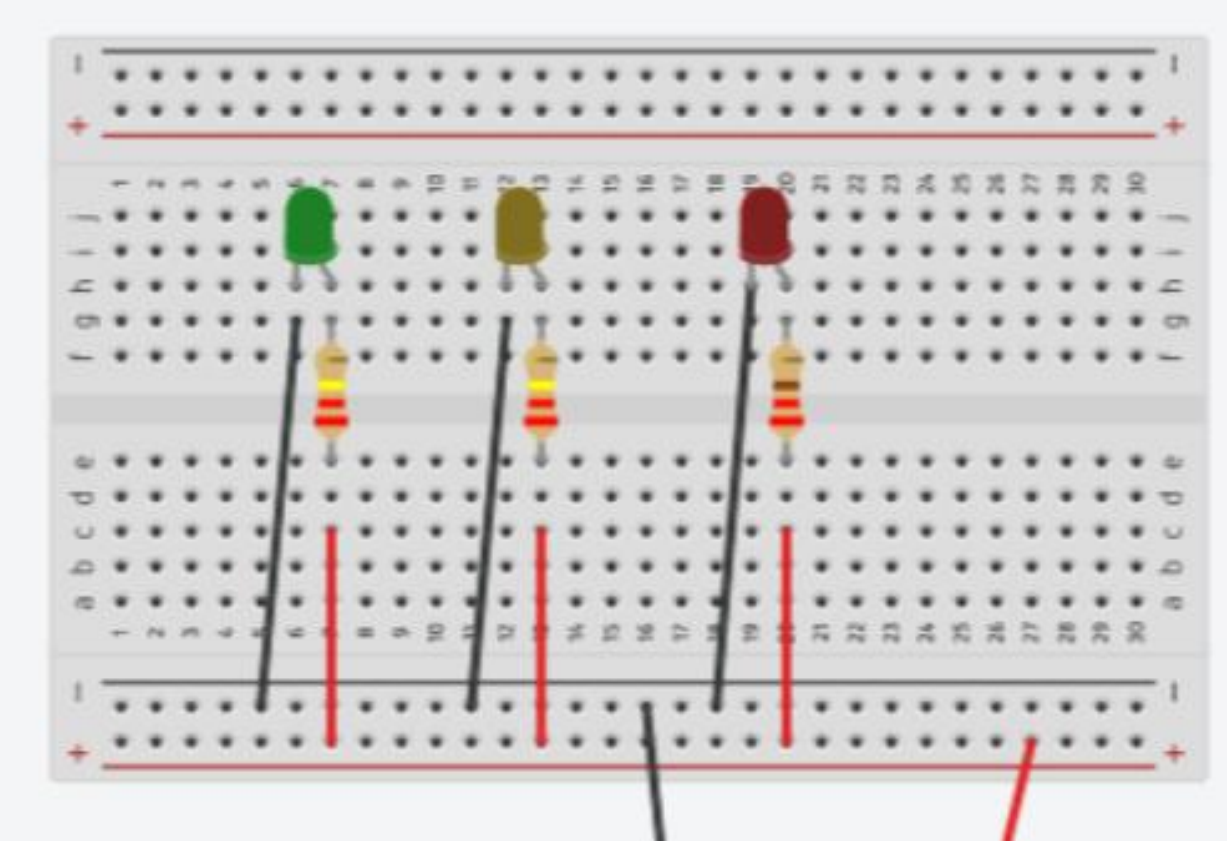


LED 회로 연결하기

<필요한 재료>

: 라즈베리파이(GPIO Port), LED, 저항(330옴, 고휘도 LED경우 1~2K), 점퍼케이블



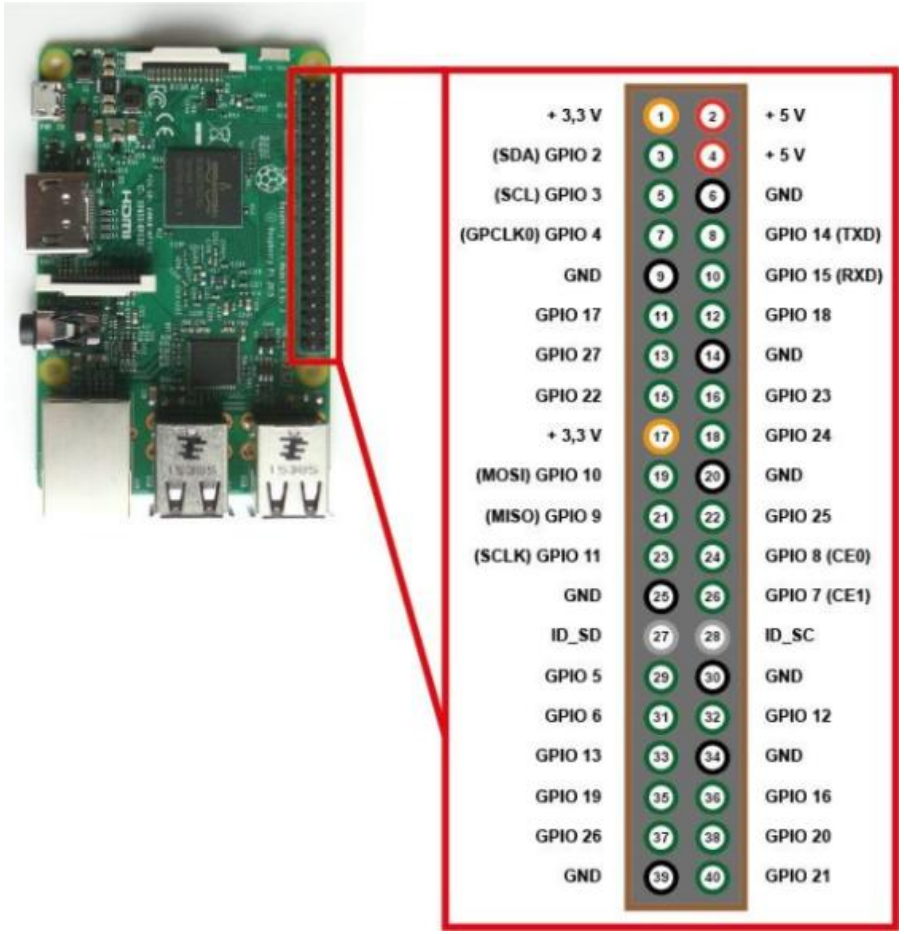


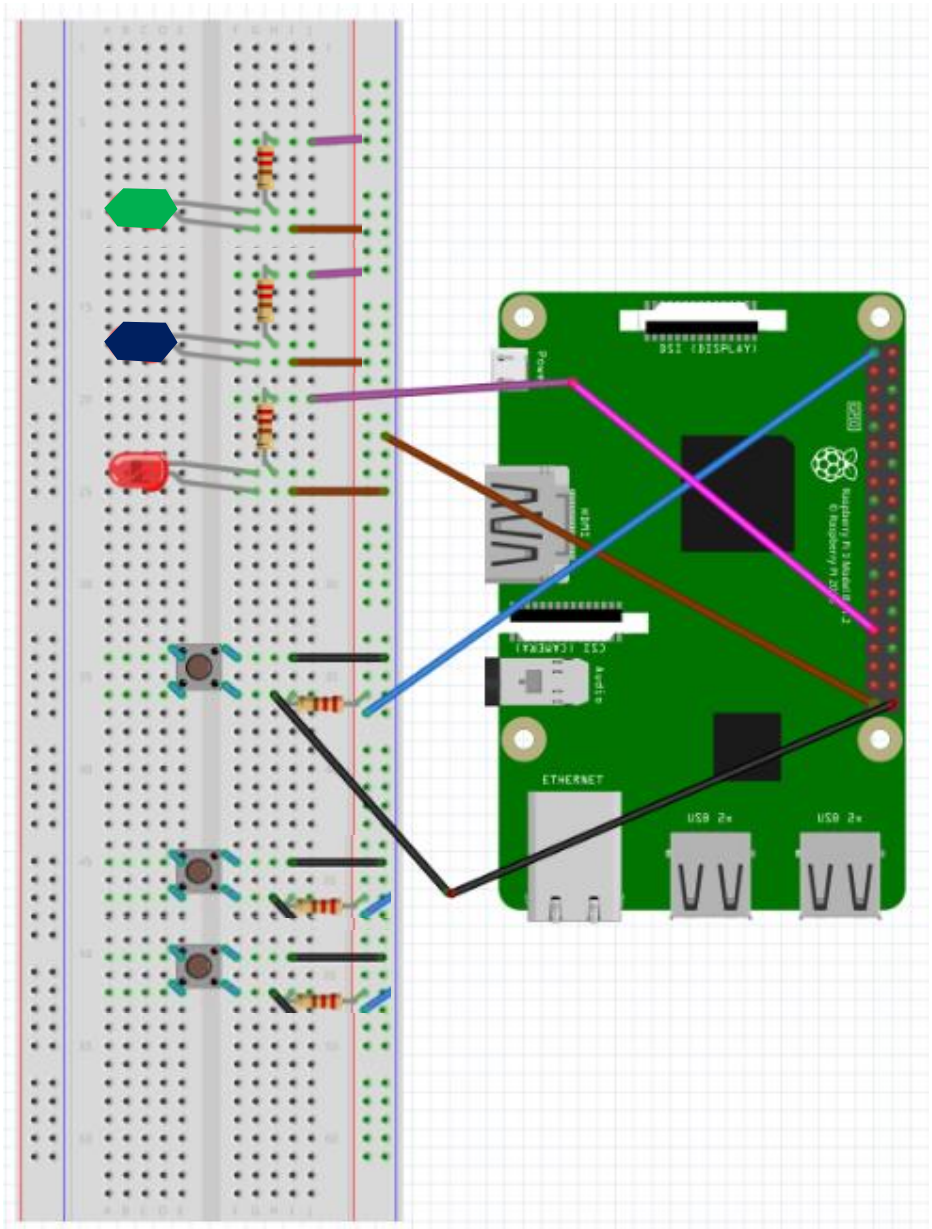
실습하기

<LED_RGB turn on/off and Push button>

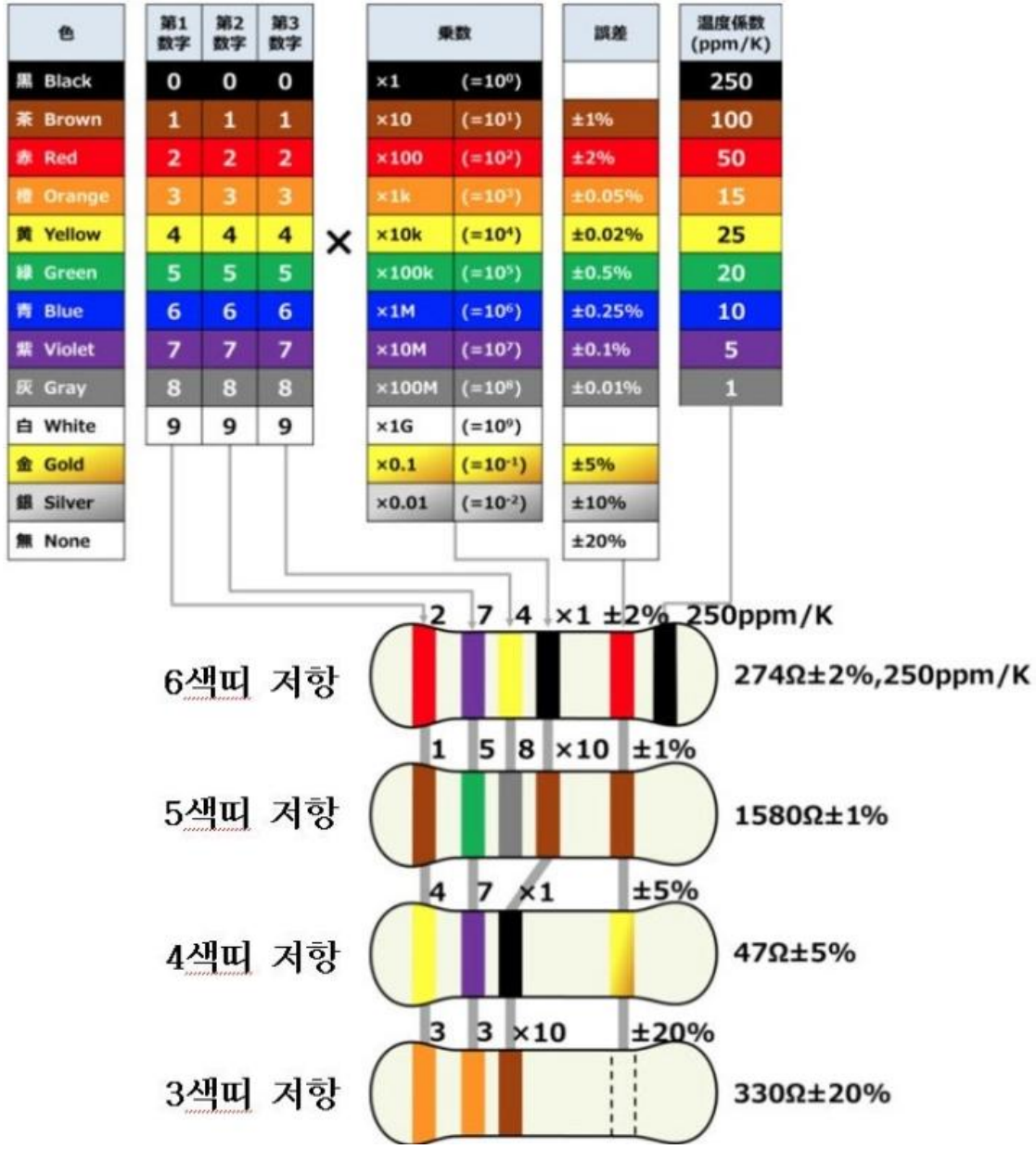
: LED → GPIO16,GPIO20,GPIO21 (Res: 330ohm)

: Switch → GPIO13,GPIO19,GPIO26 (PU Res: 10Kohm)



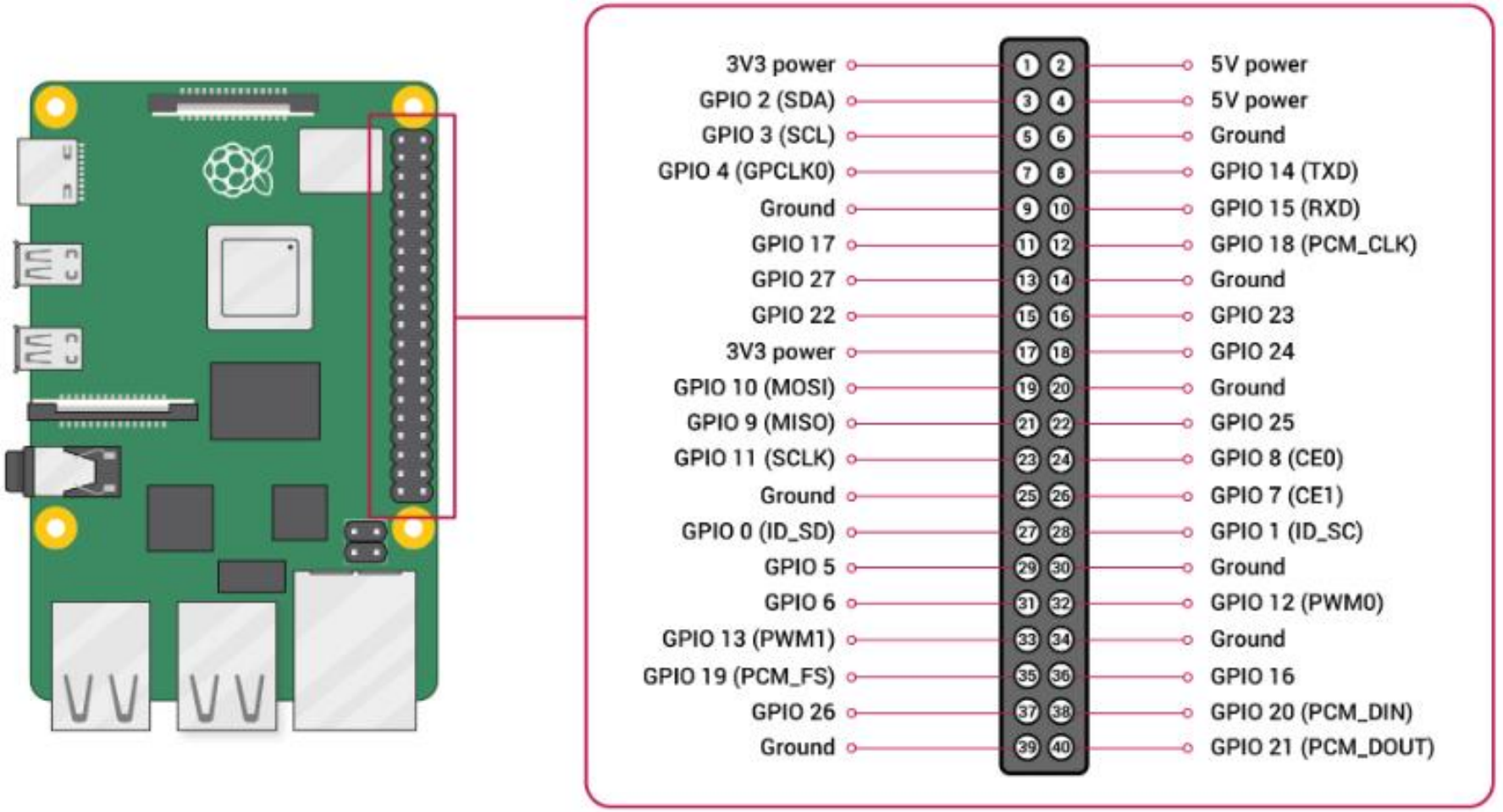


저항띠 읽기

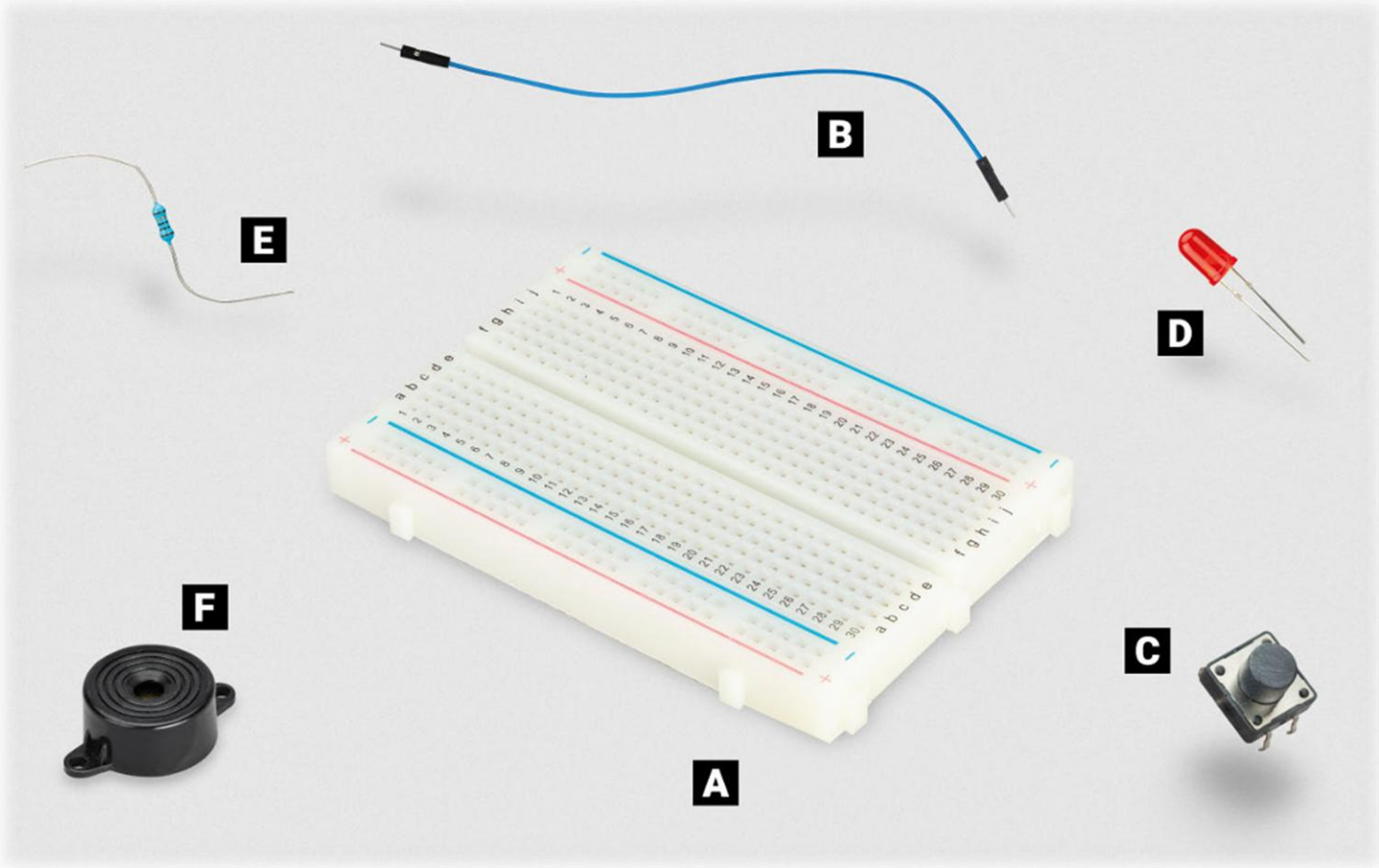

色
第1数字
第2数字
第3数字
乗数
誤差
温度係数 (ppm/K)
黒 Black 0 0 0 ×1 (=10⁰) 250
茶 Brown 1 1 1 ×10 (=10¹) ±1% 100
赤 Red 2 2 2 ×100 (=10²) ±2% 50
橙 Orange 3 3 3 ×1k (=10³) ±0.05% 15
黄 Yellow 4 4 4 ×10k (=10⁴) ±0.02% 25
緑 Green 5 5 5 ×100k (=10⁵) ±0.5% 20
青 Blue 6 6 6 ×1M (=10⁶) ±0.25% 10
紫 Violet 7 7 7 ×10M (=10⁷) ±0.1% 5
灰 Gray 8 8 8 ×100M (=10⁸) ±0.01% 1
白 White 9 9 9 ×1G (=10⁹)
金 Gold ×0.1 (=10⁻¹) ±5%
銀 Silver ×0.01 (=10⁻²) ±10%
無 None ±20%
2 7 4 ×1 ±2% 250ppm/K
6색띠 저항
274Ω±2%,250ppm/K
1 5 8 ×10 ±1%
5색띠 저항
1580Ω±1%
4 7 ×1 ±5%
4색띠 저항
47Ω±5%
3 3 ×10 ±20%
3색띠 저항
330Ω±20%

라즈베리파이 5 GPIO

https://www.raspberrypi.com/documentation/computers/raspberry-pi.html#gpio



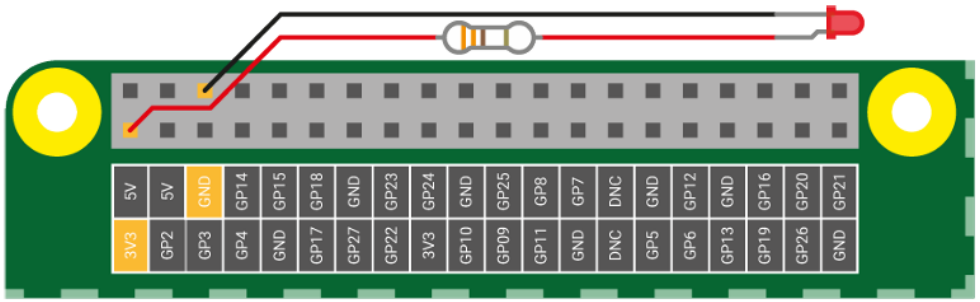
실습에 필요한 준비물

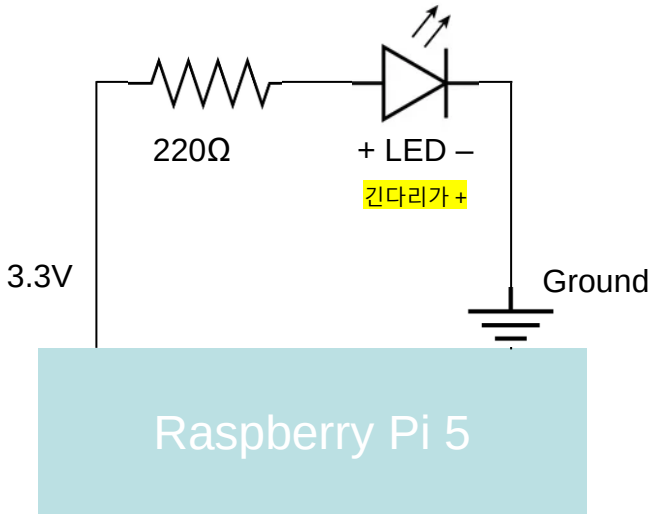


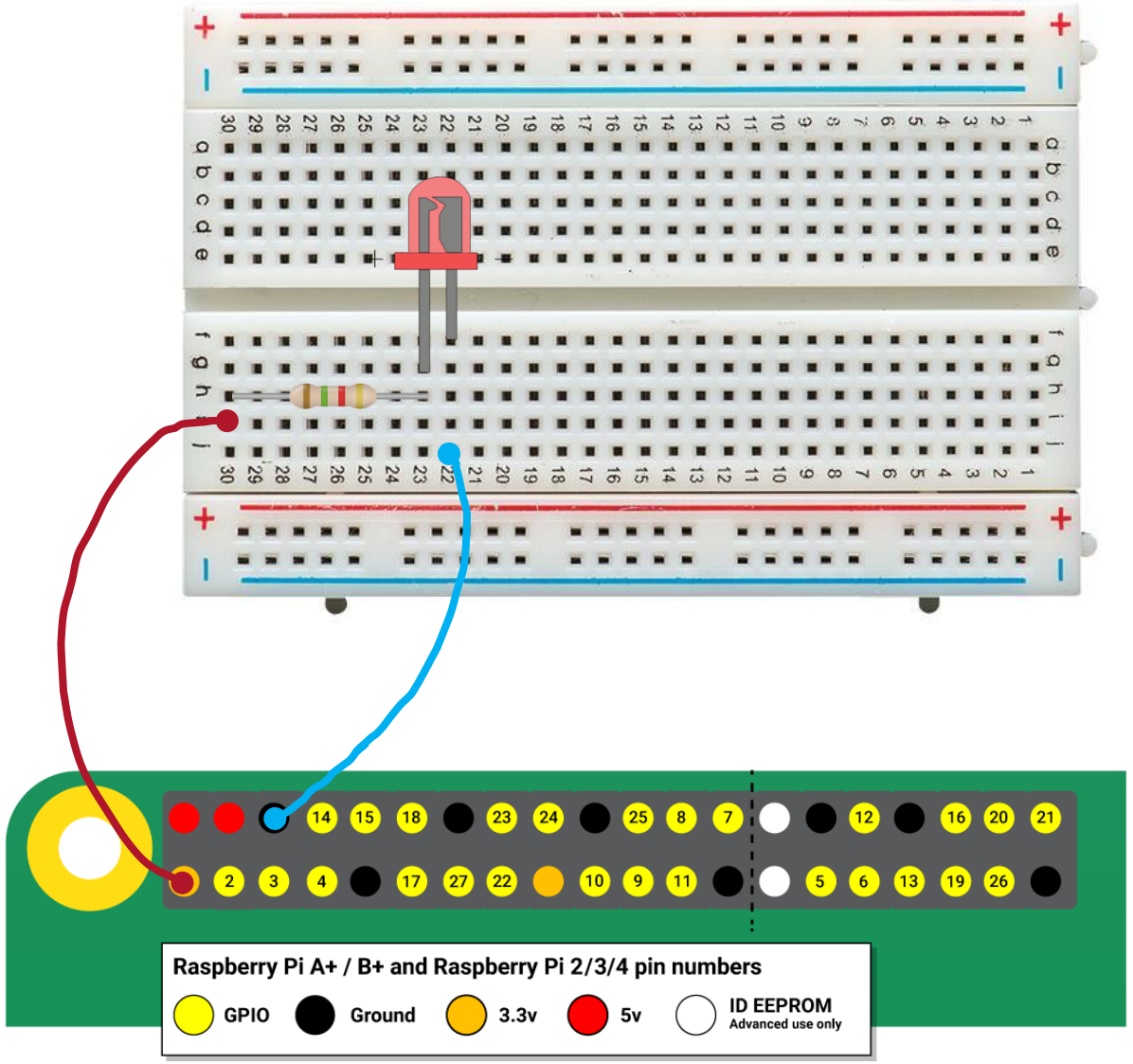
- A Breadboard
- B Jumper wire
- C Momentary switch
- D Light-emitting diode (LED)
- E Resistor
- F Piezoelectric buzzer

실습 #1 – LED 기초

3.3V output 핀을 사용해 LED 켜보기

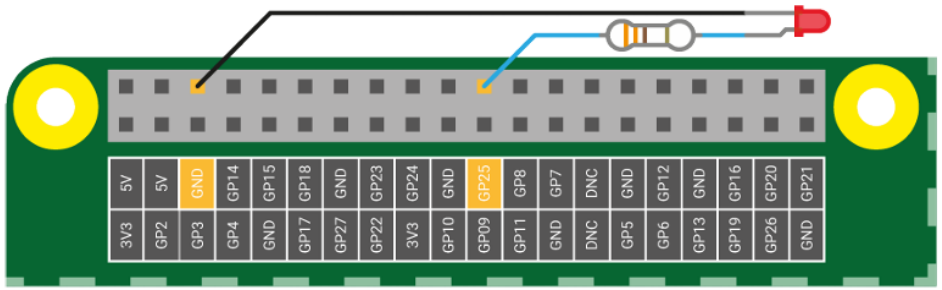


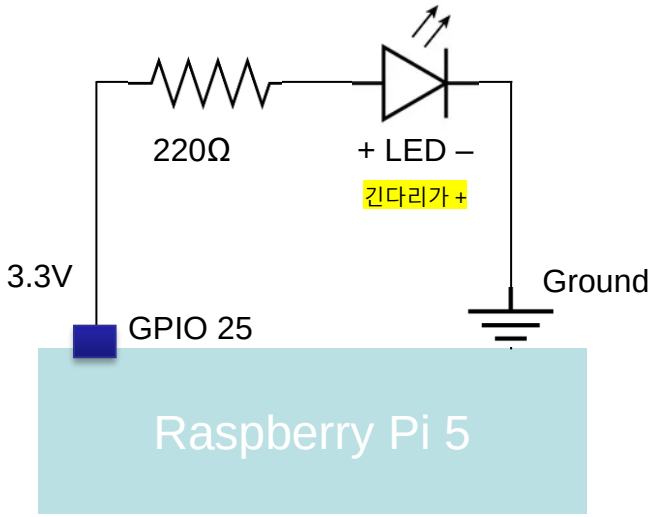


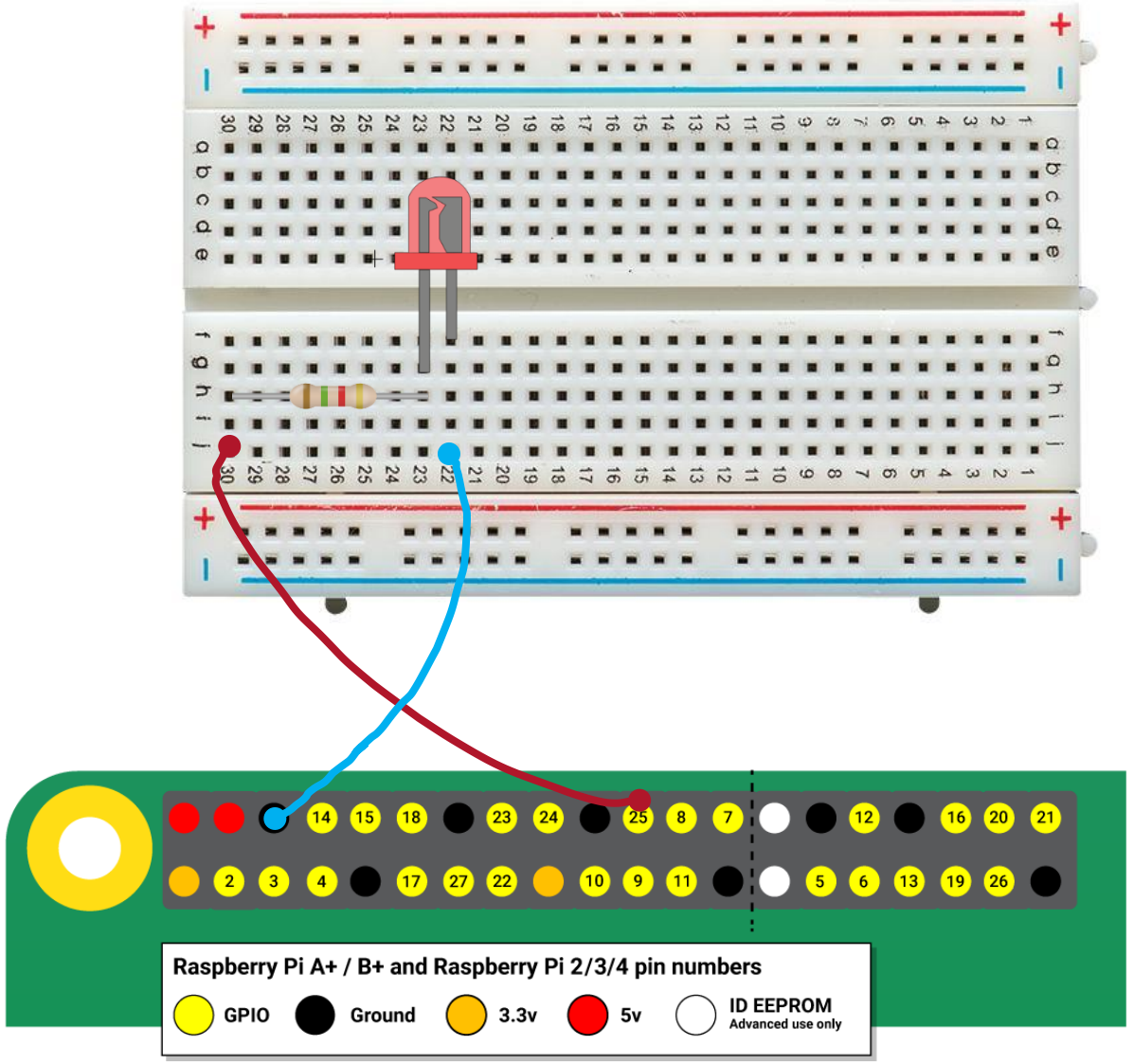


실습 #1 – LED 기초

GPIO out 핀을 사용해 LED 켜보기





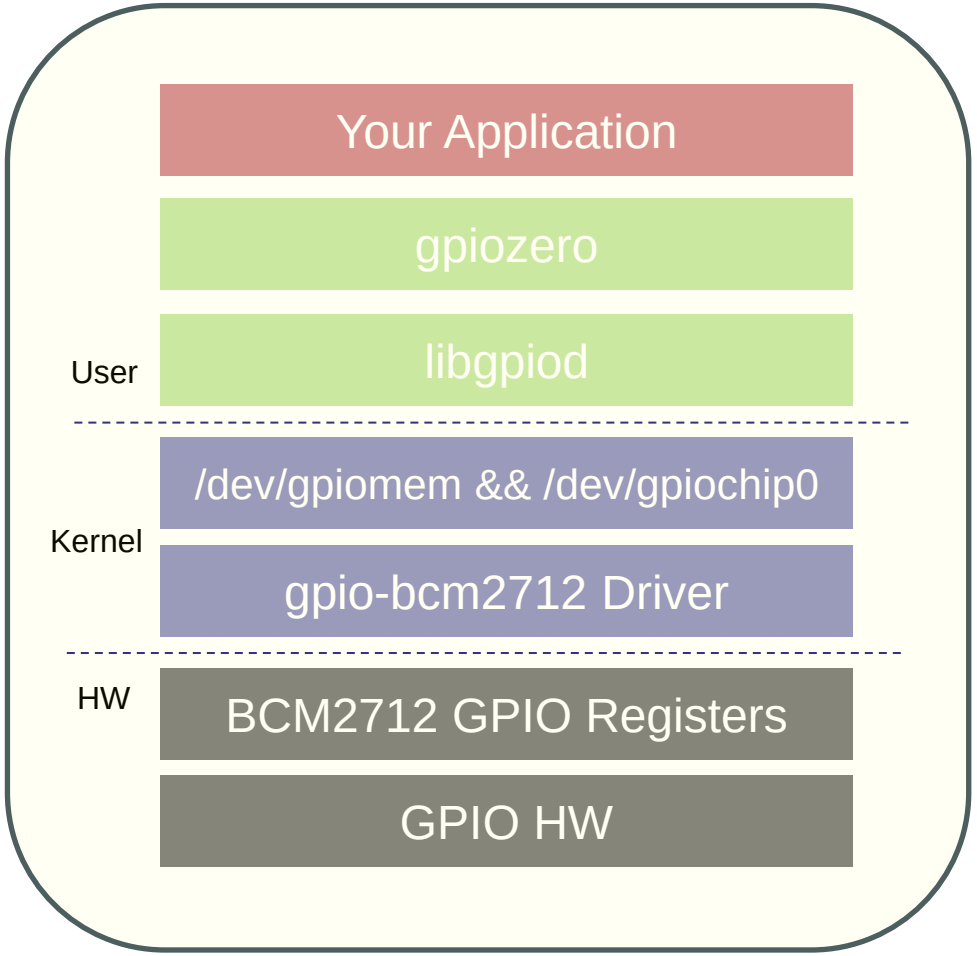


실습 #1 – LED 기초

https://github.com/gpiozero/gpiozero
https://gpiozero.readthedocs.io

GPIO out 핀을 사용해 LED 켜보기 – Python sample code

```
# 필요한 라이브러리 import
from gpiozero import LED

# led 변수에 LED 객체를 GPIO 25 핀을 할당해 초기화
led = LED(25)

# LED on
led.on()
```

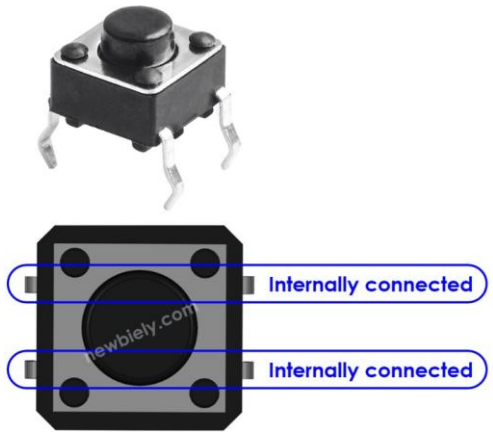


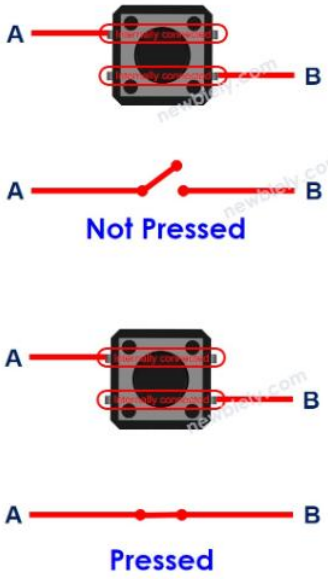
Raspberry Pi GPIO Architecture

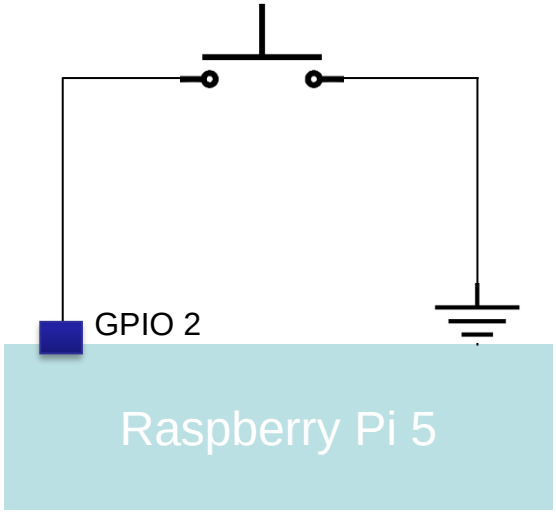
실습 #1 – LED 실습

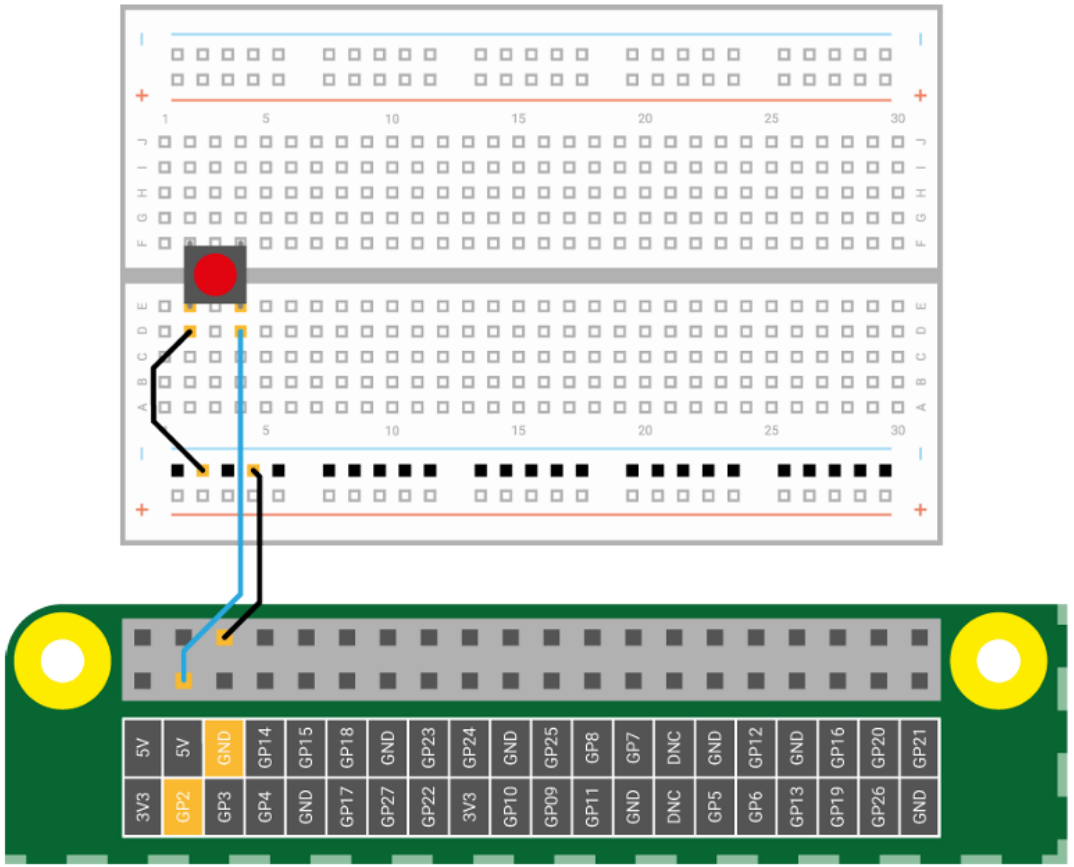
1.
2.
3.
4.
5.

실습 #2 – Button 기초









실습 #2 – Button 기초

https://github.com/gpiozero/gpiozero
https://gpiozero.readthedocs.io

Button 하나를 GPIO 2번핀에 연결해 Button 이 눌렸을 때 python print() 함수를 사용해 “pressed” 를 출력하고 눌렀다 때는 순간 “released” 를 출력 – Python sample code

콜백 + pause

```python
from gpiozero import Button
from signal import pause

btn = Button(2, pull_up=True)

# 1) 이름 있는 함수로 콜백 정의
def on_press():
    print("pressed")

def on_release():
    print("released")

# 2) 콜백 설정
btn.when_pressed  = on_press
btn.when_released = on_release

pause()
```

대기 wait_for_*()

```python
from gpiozero import Button

# BCM 2번 핀, 내부 풀업
btn = Button(2, pull_up=True)

# 무한 루프 안에서 순차적으로 대기 → 처리
while True:
    # 버튼이 눌릴 때까지 대기 → 눌리면 실행
    btn.wait_for_press()
    print("pressed")

    # 버튼이 떼어질 때까지 대기 → 떼어지면 실행
    btn.wait_for_release()
    print("released")
```

Polling 루프

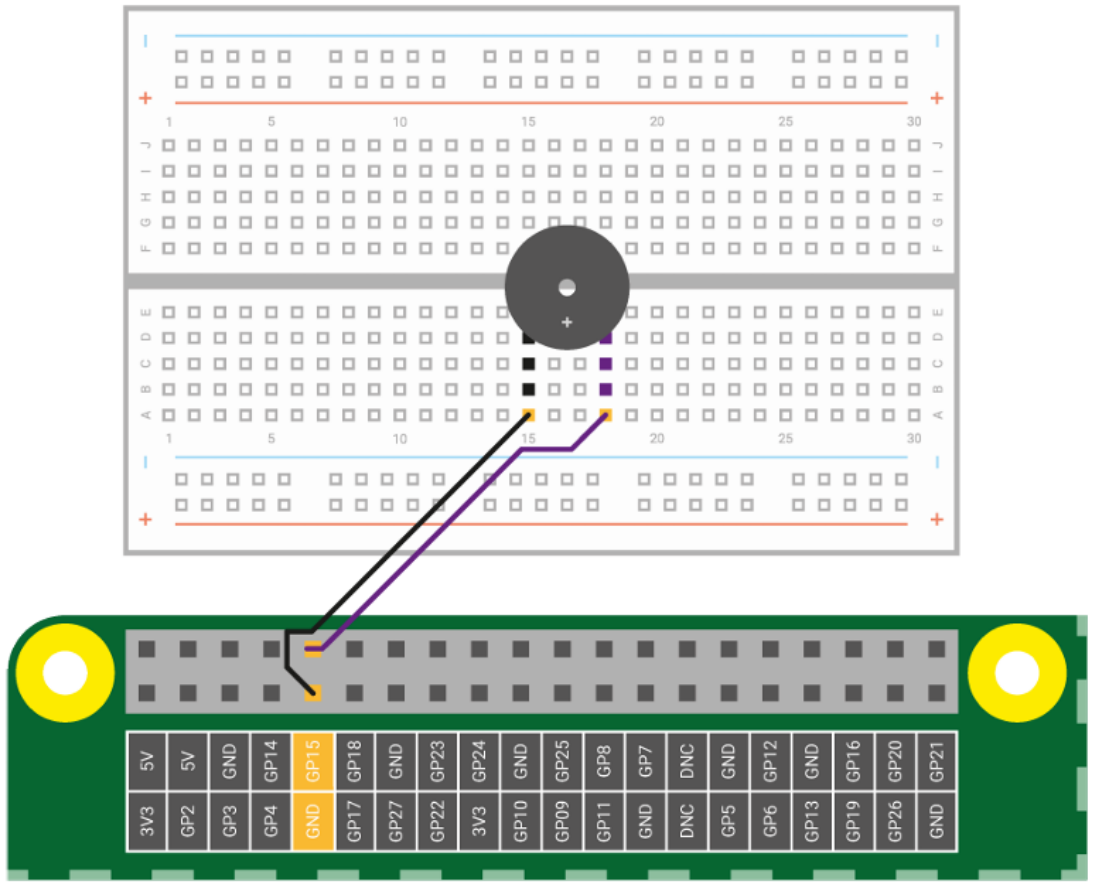
```python
from gpiozero import Button
from time import sleep

# BCM 2번 핀, 내부 풀업
btn = Button(2, pull_up=True)
# 이전 상태 초기화 (눌림 여부)
prev = btn.is_pressed

while True:
    # 현재 눌림 상태 읽기 (True=눌림, False=떼어짐)
    cur = btn.is_pressed

    # 상태 변화가 있을 때만 출력
    if cur != prev:
        if cur:
            print("pressed")  # 눌리면 출력
        else:
            print("released") # 떼어지면 출력
        prev = cur            # 이전 상태 갱신

    # 짧게 대기하여 CPU 과부하 방지
    sleep(0.01)
```

실습 #2 – Button

1.

2.

실습 #3 – Buzzor 기초



실습 #3 – Buzzor

1.
2.

실습 #4 – 두더지 게임

1 단계

-
- 

2 단계

-
- 

THANK YOU