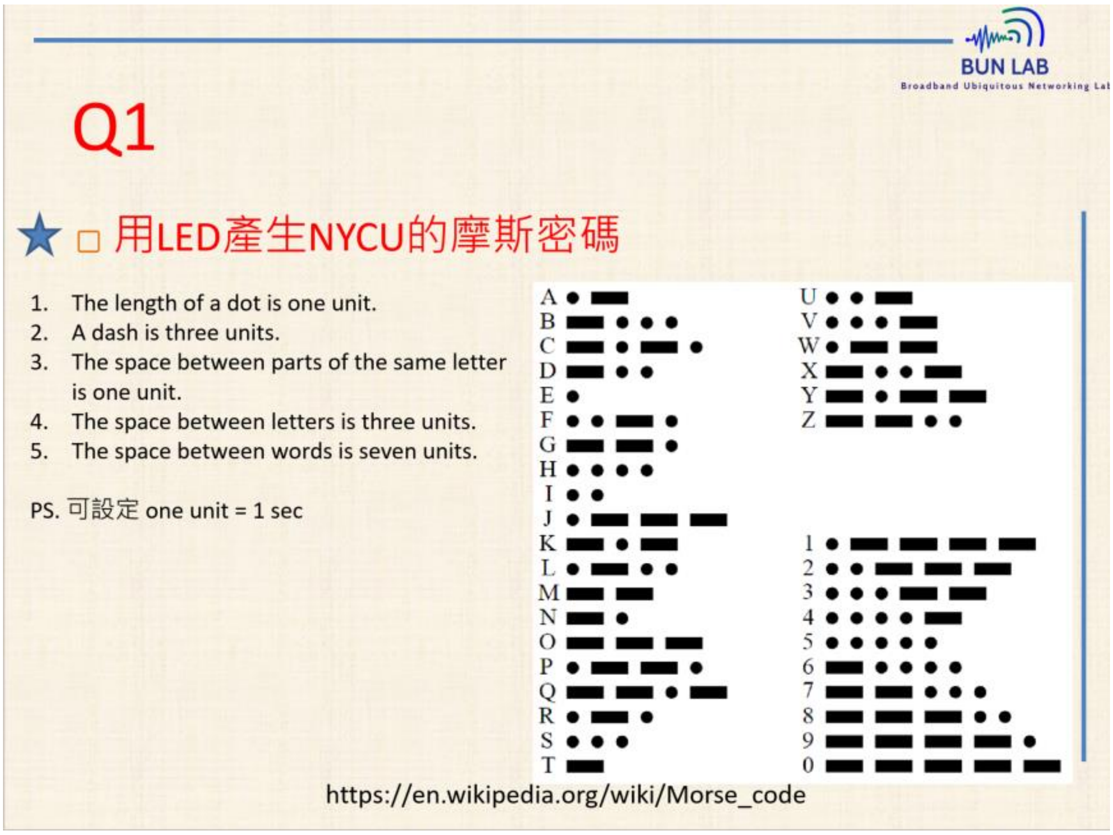
Communication Networks Lab
Topic IOT-Lab1

Q1:



BUN LAB
Broadband Ubiquitous Networking Lab

Q1

★ □ 用LED產生NYCU的摩斯密碼

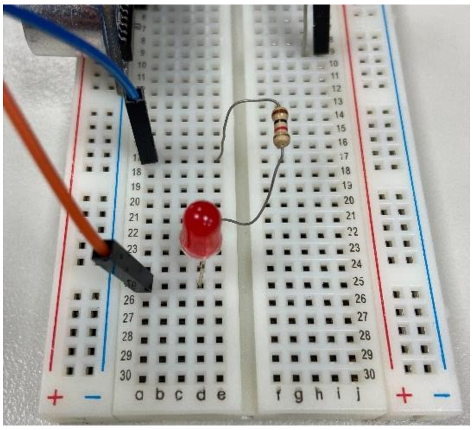
1. The length of a dot is one unit.
2. A dash is three units.
3. The space between parts of the same letter is one unit.
4. The space between letters is three units.
5. The space between words is seven units.

PS. 可設定 one unit = 1 sec

A	● ▬
B	▬ ● ● ●
C	▬ ● ▬ ●
D	▬ ● ●
E	●
F	● ● ▬ ●
G	▬ ▬ ●
H	● ● ● ●
I	● ●
J	● ▬ ▬ ▬
K	▬ ● ▬
L	● ▬ ● ●
M	▬ ▬
N	▬ ●
O	▬ ▬ ▬
P	● ▬ ▬ ●
Q	▬ ▬ ● ▬
R	● ▬ ●
S	● ● ●
T	▬
U	● ● ▬
V	● ● ● ▬
W	● ▬ ▬
X	▬ ● ● ▬
Y	▬ ● ▬ ▬
Z	▬ ▬ ● ●
1	● ▬ ▬ ▬ ▬
2	● ● ▬ ▬ ▬
3	● ● ● ▬ ▬
4	● ● ● ● ▬
5	● ● ● ● ●
6	▬ ● ● ● ●
7	▬ ▬ ● ● ●
8	▬ ▬ ▬ ● ●
9	▬ ▬ ▬ ▬ ●
0	▬ ▬ ▬ ▬ ▬

https://en.wikipedia.org/wiki/Morse_code

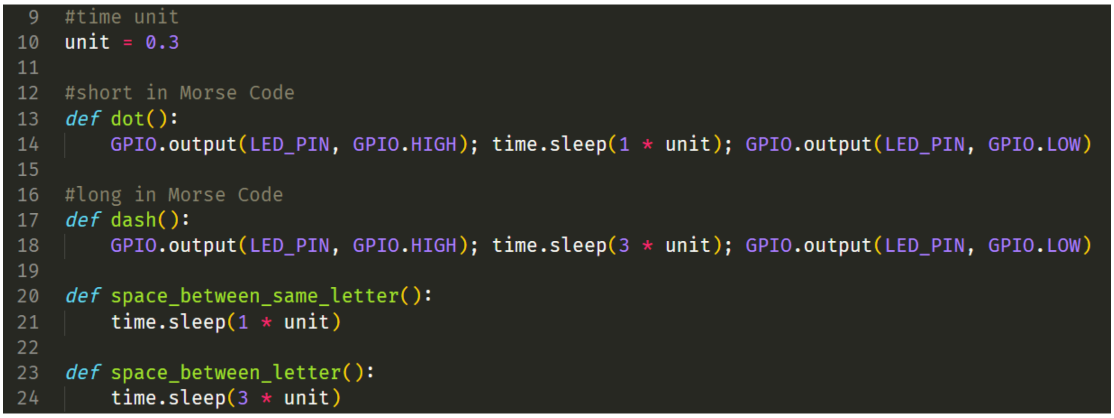
Circuit:



Blue wire connects to the pin12, and orange wire connects to ground on Raspberry Pi.

Code description:

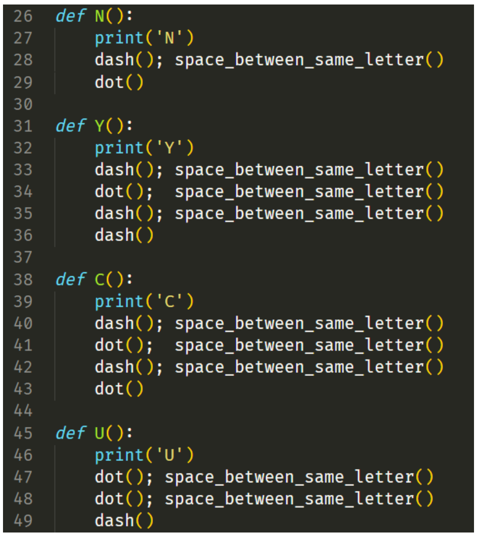
First: define the dot, dash and space in Morse Code.



```
 9  #time unit
10  unit = 0.3
11
12  #short in Morse Code
13  def dot():
14      GPIO.output(LED_PIN, GPIO.HIGH); time.sleep(1 * unit); GPIO.output(LED_PIN, GPIO.LOW)
15
16  #long in Morse Code
17  def dash():
18      GPIO.output(LED_PIN, GPIO.HIGH); time.sleep(3 * unit); GPIO.output(LED_PIN, GPIO.LOW)
19
20  def space_between_same_letter():
21      time.sleep(1 * unit)
22
23  def space_between_letter():
24      time.sleep(3 * unit)
```

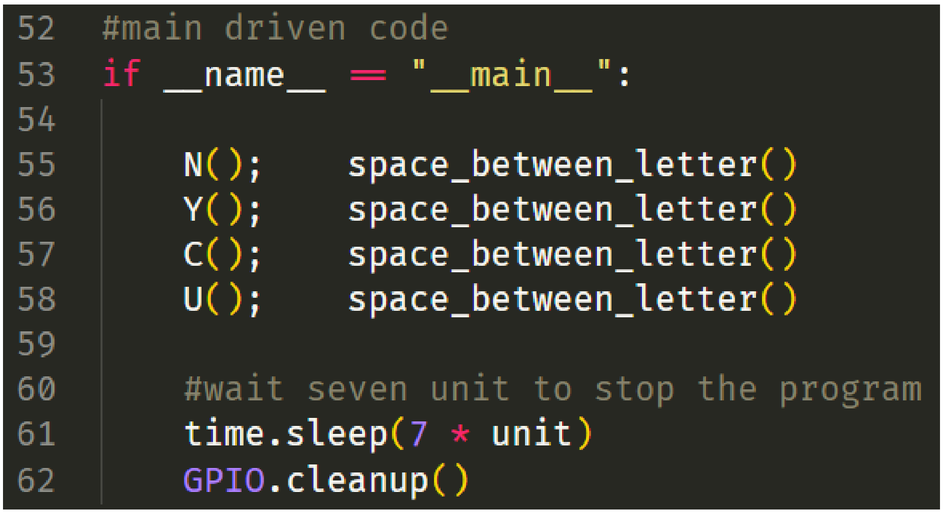
We can change the unit of time by simply changing the value on line 10.

Second: Construct the Morse Code of NYCU.



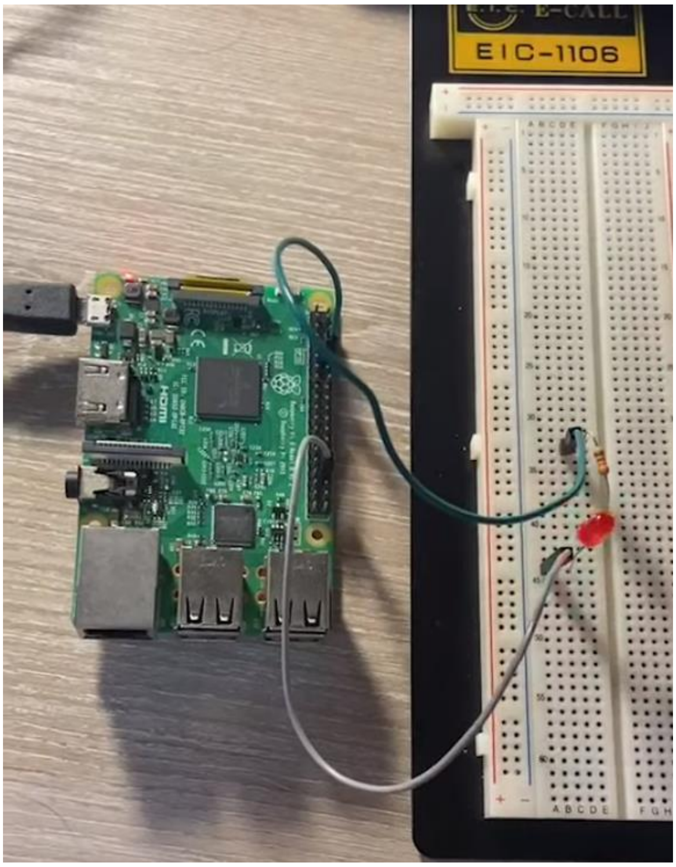
```
def N():
    print('N')
    dash(); space_between_same_letter()
    dot()

def Y():
    print('Y')
    dash(); space_between_same_letter()
    dot();  space_between_same_letter()
    dash(); space_between_same_letter()
    dash()

def C():
    print('C')
    dash(); space_between_same_letter()
    dot();  space_between_same_letter()
    dash(); space_between_same_letter()
    dot()

def U():
    print('U')
    dot(); space_between_same_letter()
    dot(); space_between_same_letter()
    dash()
```

Third: main driven code.



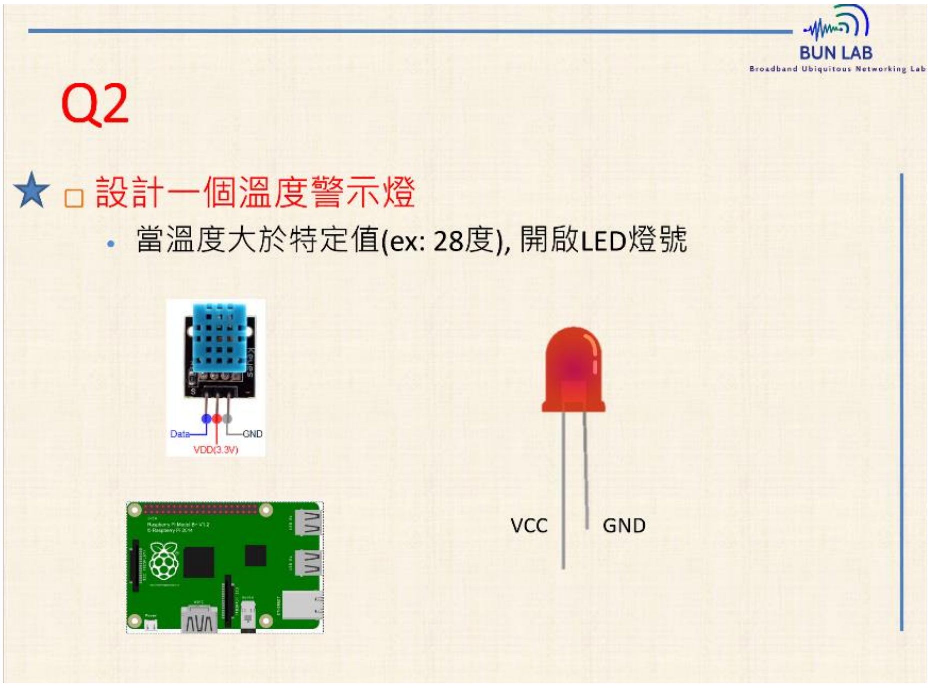
```
#main driven code
if __name__ == "__main__":

    N();    space_between_letter()
    Y();    space_between_letter()
    C();    space_between_letter()
    U();    space_between_letter()

    #wait seven unit to stop the program
    time.sleep(7 * unit)
    GPIO.cleanup()
```

Q1 result and demo:



https://youtube.com/shorts/x5zxI9CUN5E?feature=share

Q2:



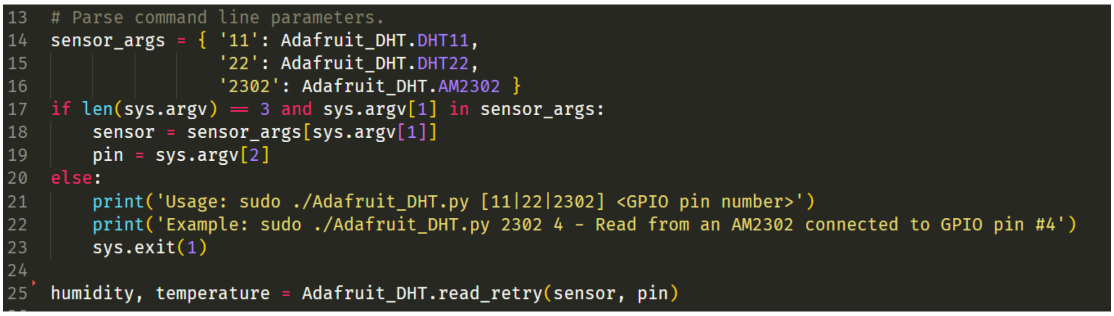
Q2

★ □ 設計一個溫度警示燈

- 當溫度大於特定值(ex: 28度), 開啟LED燈號

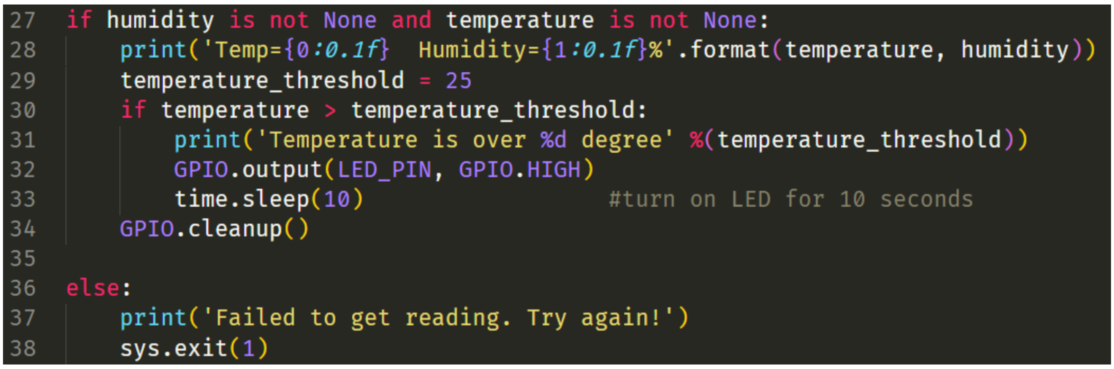
Code description:

First: parse the command line parameters to recognize which sensor we use and which pin do we connect with and then get the humidity and temperature from it.



```
# Parse command line parameters.
sensor_args = { '11': Adafruit_DHT.DHT11,
                '22': Adafruit_DHT.DHT22,
                '2302': Adafruit_DHT.AM2302 }
if len(sys.argv) == 3 and sys.argv[1] in sensor_args:
    sensor = sensor_args[sys.argv[1]]
    pin = sys.argv[2]
else:
    print('Usage: sudo ./Adafruit_DHT.py [11|22|2302] <GPIO pin number>')
    print('Example: sudo ./Adafruit_DHT.py 2302 4 - Read from an AM2302 connected to GPIO pin #4')
    sys.exit(1)

humidity, temperature = Adafruit_DHT.read_retry(sensor, pin)
```

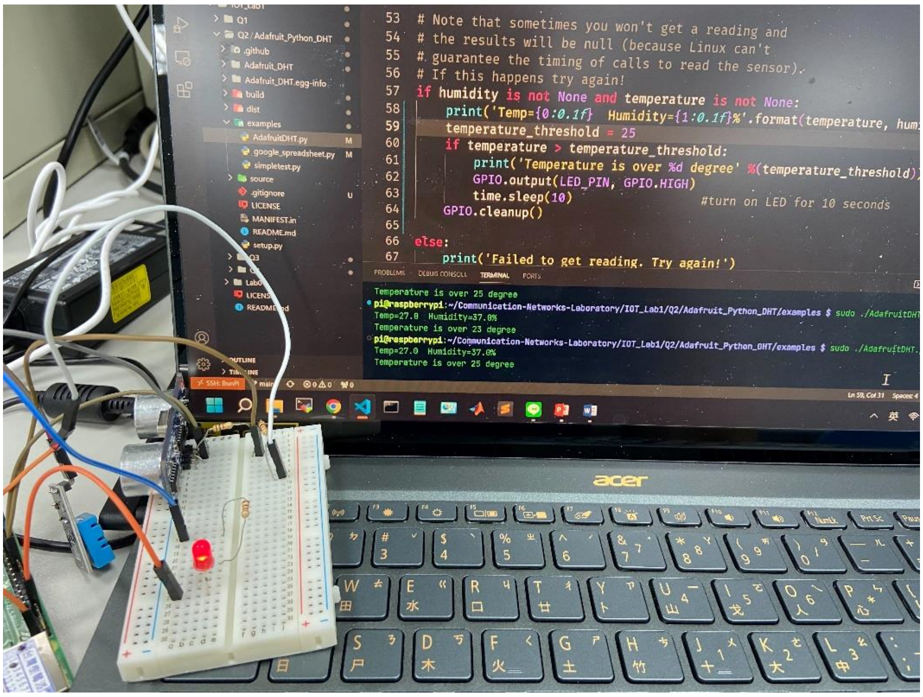
Second: Check whether the temperature is higher than temperature threshold to decide whether turn the LED on or off.



```
if humidity is not None and temperature is not None:
    print('Temp={0:0.1f}  Humidity={1:0.1f}%'.format(temperature, humidity))
    temperature_threshold = 25
    if temperature > temperature_threshold:
        print('Temperature is over %d degree' %(temperature_threshold))
        GPIO.output(LED_PIN, GPIO.HIGH)
        time.sleep(10)                  #turn on LED for 10 seconds
    GPIO.cleanup()

else:
    print('Failed to get reading. Try again!')
    sys.exit(1)
```

Q2 result and demo:

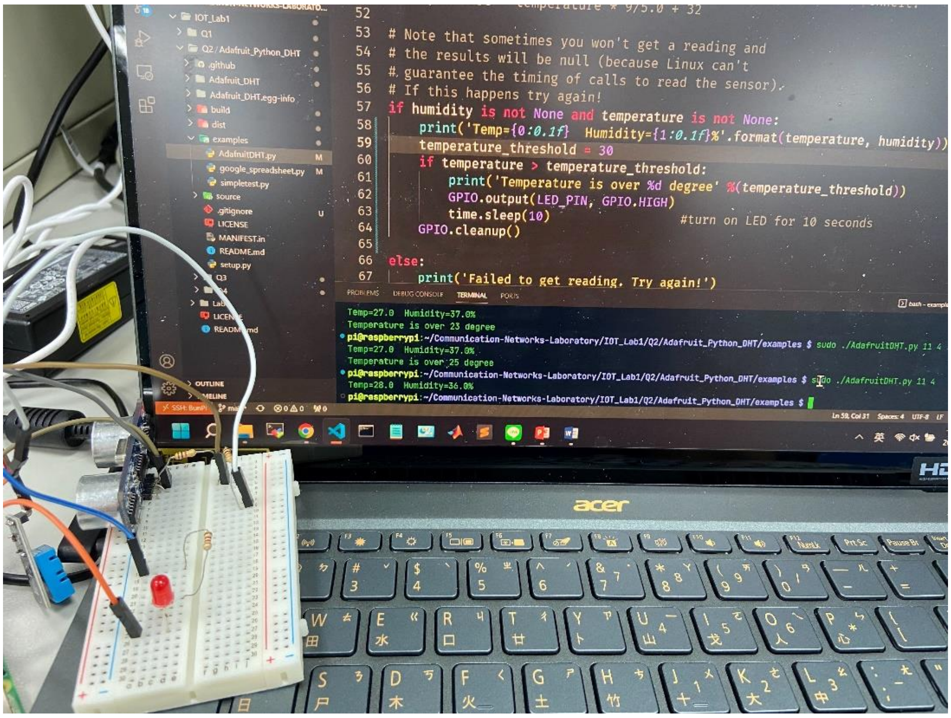
(a) When temperature is over temperature-threshold(25 Celsius), turn on LED.



```
# Note that sometimes you won't get a reading and
# the results will be null (because Linux can't
# guarantee the timing of calls to read the sensor).
# If this happens try again!
if humidity is not None and temperature is not None:
    print('Temp={0:0.1f}  Humidity={1:0.1f}%'.format(temperature, hum
    temperature_threshold = 25
    if temperature > temperature_threshold:
        print('Temperature is over %d degree' %(temperature_threshold)
        GPIO.output(LED_PIN, GPIO.HIGH)
        time.sleep(10)                  #turn on LED for 10 seconds
    GPIO.cleanup()

else:
    print('Failed to get reading. Try again!')
```

```
Temperature is over 25 degree
pi@raspberrypi:~/Communication-Networks-Laboratory/IOT_Lab1/Q2/Adafruit_Python_DHT/examples $ sudo ./AdafruitDHT
Temp=27.0  Humidity=37.0%
Temperature is over 23 degree
pi@raspberrypi:~/Communication-Networks-Laboratory/IOT_Lab1/Q2/Adafruit_Python_DHT/examples $ sudo ./AdafruitDHT.
Temp=27.0  Humidity=37.0%
Temperature is over 25 degree
```

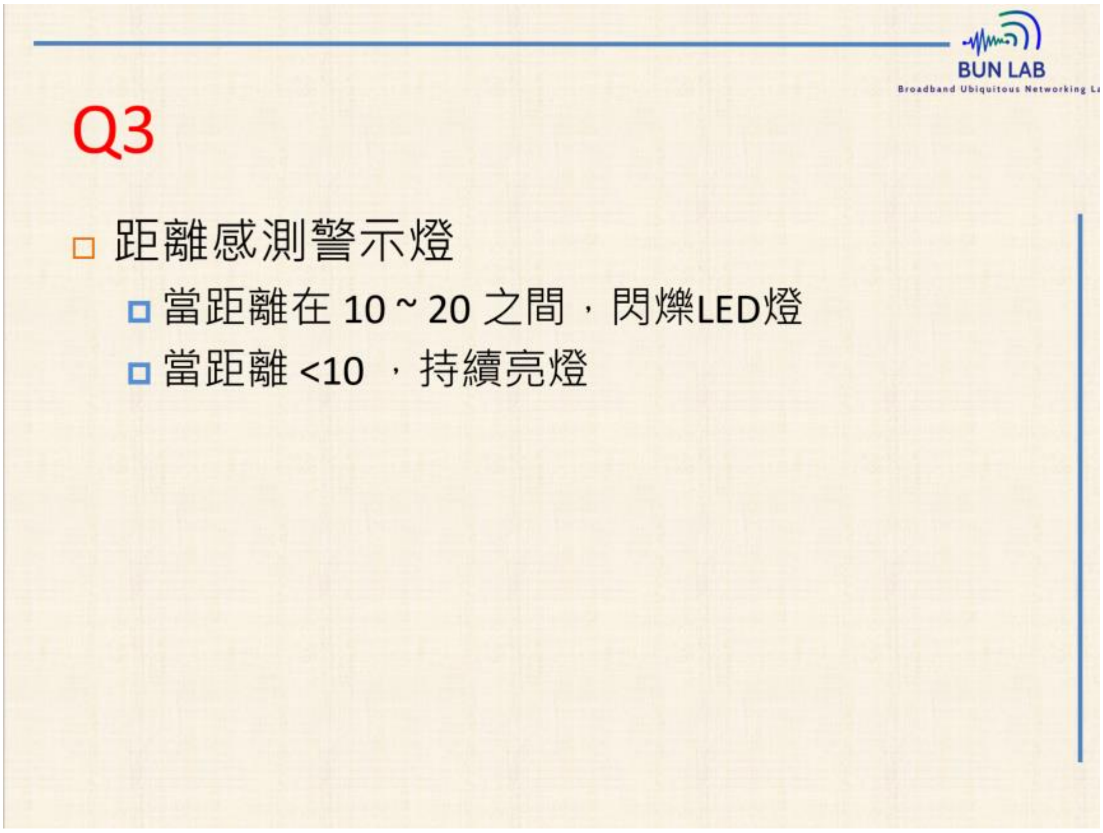
(b) When temperature is lower than temperature-threshold(30 Celsius), turn off.



```
# Note that sometimes you won't get a reading and
# the results will be null (because Linux can't
# guarantee the timing of calls to read the sensor).
# If this happens try again!
if humidity is not None and temperature is not None:
    print('Temp={0:0.1f}  Humidity={1:0.1f}%'.format(temperature, humidity))
    temperature_threshold = 30
    if temperature > temperature_threshold:
        print('Temperature is over %d degree' %(temperature_threshold))
        GPIO.output(LED_PIN, GPIO.HIGH)
        time.sleep(10)                          #turn on LED for 10 seconds
    GPIO.cleanup()

else:
    print('Failed to get reading. Try again!')
```

```
Temp=27.0  Humidity=37.0%
Temperature is over 23 degree
pi@raspberrypi:~/Communication-Networks-Laboratory/IOT_Lab1/Q2/Adafruit_Python_DHT/examples $ sudo ./AdafruitDHT.py 11 4
Temp=27.0  Humidity=37.0%
Temperature is over 25 degree
pi@raspberrypi:~/Communication-Networks-Laboratory/IOT_Lab1/Q2/Adafruit_Python_DHT/examples $ sudo ./AdafruitDHT.py 11 4
Temp=28.0  Humidity=36.0%
pi@raspberrypi:~/Communication-Networks-Laboratory/IOT_Lab1/Q2/Adafruit_Python_DHT/examples $
```

Q3:



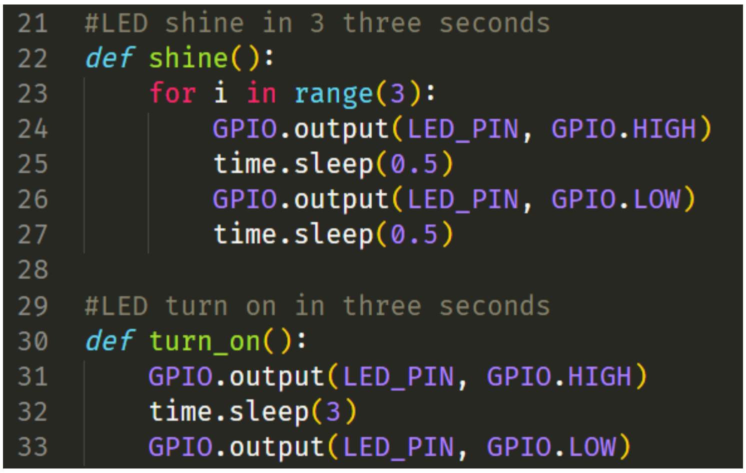
BUN LAB
Broadband Ubiquitous Networking La

Q3

- 距離感測警示燈
  - 當距離在 10 ~ 20 之間，閃爍LED燈
  - 當距離 <10 ，持續亮燈

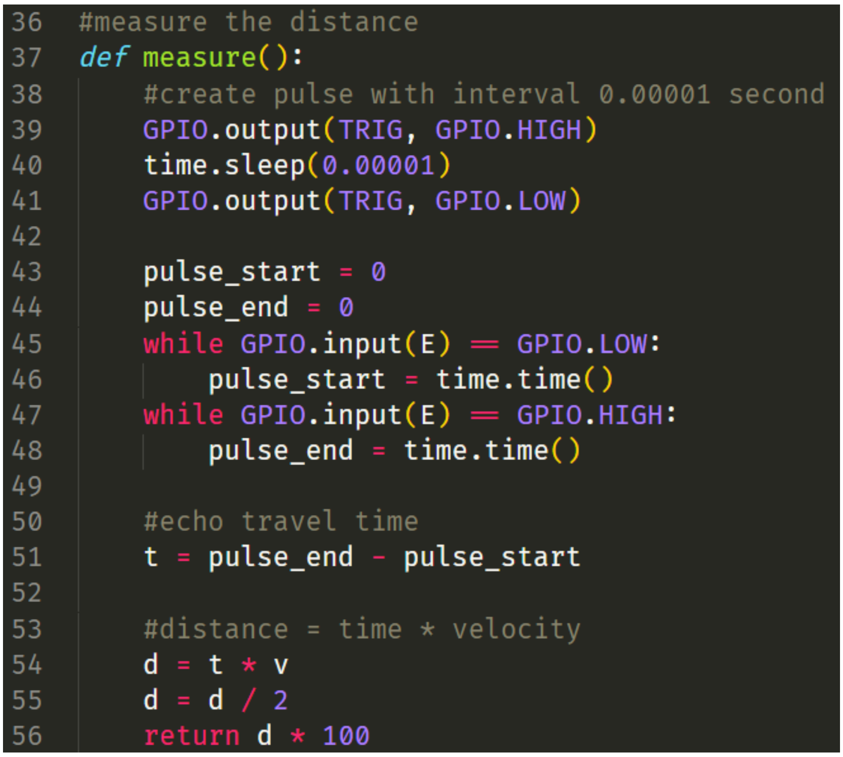
Code description:

First: define two methods for the LED: shine and just turn on in three seconds because I measure the distance one time per three second



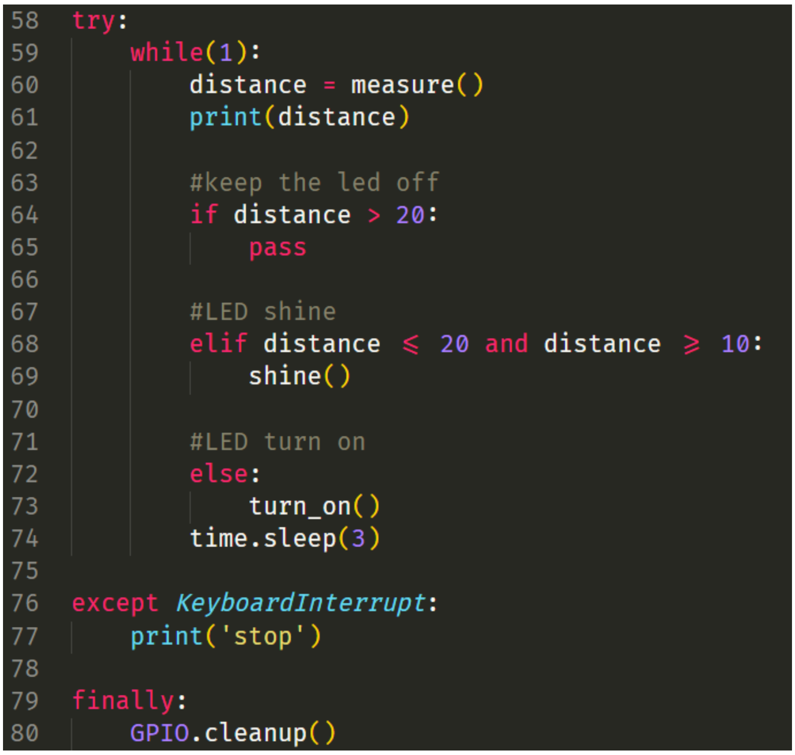
```python
#LED shine in 3 three seconds
def shine():
    for i in range(3):
        GPIO.output(LED_PIN, GPIO.HIGH)
        time.sleep(0.5)
        GPIO.output(LED_PIN, GPIO.LOW)
        time.sleep(0.5)

#LED turn on in three seconds
def turn_on():
    GPIO.output(LED_PIN, GPIO.HIGH)
    time.sleep(3)
    GPIO.output(LED_PIN, GPIO.LOW)
```

Second: Using trigger pin to create a pulse, and catch the signal by echo pin to determine the distance that ultrasound travel.



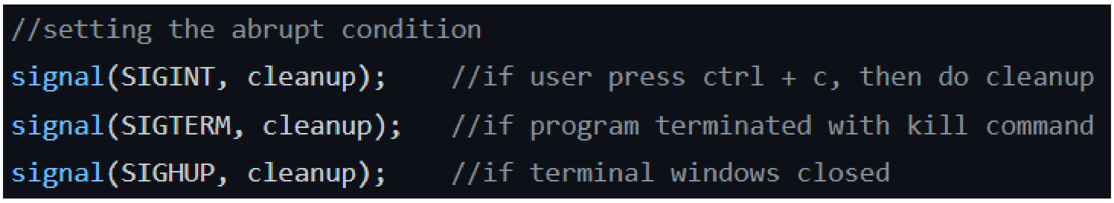
```python
#measure the distance
def measure():
    #create pulse with interval 0.00001 second
    GPIO.output(TRIG, GPIO.HIGH)
    time.sleep(0.00001)
    GPIO.output(TRIG, GPIO.LOW)

    pulse_start = 0
    pulse_end = 0
    while GPIO.input(E) == GPIO.LOW:
        pulse_start = time.time()
    while GPIO.input(E) == GPIO.HIGH:
        pulse_end = time.time()

    #echo travel time
    t = pulse_end - pulse_start

    #distance = time * velocity
    d = t * v
    d = d / 2
    return d * 100
```

Third: main code just to decide what method to perform due to different distance. Besides, we can use try except to avoid GPIO.cleanup( ) not be executed error.



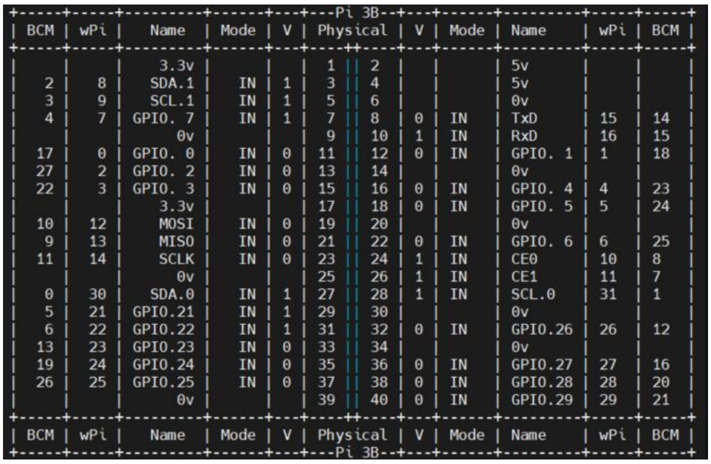
```python
try:
    while(1):
        distance = measure()
        print(distance)

        #keep the led off
        if distance > 20:
            pass

        #LED shine
        elif distance <= 20 and distance >= 10:
            shine()

        #LED turn on
        else:
            turn_on()
        time.sleep(3)

except KeyboardInterrupt:
    print('stop')

finally:
    GPIO.cleanup()
```

In C, we can also use <signal.h> to execute cleanup function after ctrl + C was hit.



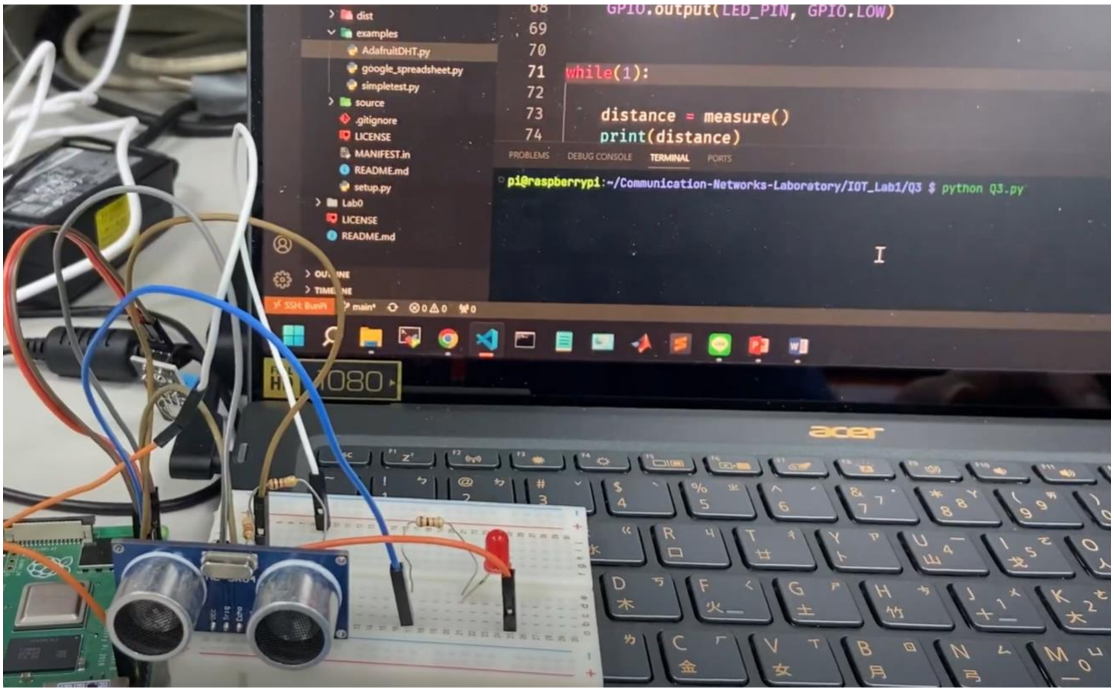
```c
//setting the abrupt condition
signal(SIGINT, cleanup);    //if user press ctrl + c, then do cleanup
signal(SIGTERM, cleanup);   //if program terminated with kill command
signal(SIGHUP, cleanup);    //if terminal windows closed
```

It’s really important to keep our GPIO pinMode in IN mode, so that it will not be dangerous to our circuit or device. After install wiringPi package, we can type gpio readall to monitor all of the pinMode to better realize the state of our RPi.



```
+-----+-----+---------+------+---+---Pi 3B--+---+------+---------+-----+-----+
| BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
+-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
|     |     |    3.3v |      |   |  1 || 2  |   |      | 5v      |     |     |
|   2 |   8 |   SDA.1 |   IN | 1 |  3 || 4  |   |      | 5v      |     |     |
|   3 |   9 |   SCL.1 |   IN | 1 |  5 || 6  |   |      | 0v      |     |     |
|   4 |   7 | GPIO. 7 |   IN | 1 |  7 || 8  | 0 | IN   | TxD     | 15  | 14  |
|     |     |      0v |      |   |  9 || 10 | 1 | IN   | RxD     | 16  | 15  |
|  17 |   0 | GPIO. 0 |   IN | 0 | 11 || 12 | 0 | IN   | GPIO. 1 | 1   | 18  |
|  27 |   2 | GPIO. 2 |   IN | 0 | 13 || 14 |   |      | 0v      |     |     |
|  22 |   3 | GPIO. 3 |   IN | 0 | 15 || 16 | 0 | IN   | GPIO. 4 | 4   | 23  |
|     |     |    3.3v |      |   | 17 || 18 | 0 | IN   | GPIO. 5 | 5   | 24  |
|  10 |  12 |    MOSI |   IN | 0 | 19 || 20 |   |      | 0v      |     |     |
|   9 |  13 |    MISO |   IN | 0 | 21 || 22 | 0 | IN   | GPIO. 6 | 6   | 25  |
|  11 |  14 |    SCLK |   IN | 0 | 23 || 24 | 1 | IN   | CE0     | 10  | 8   |
|     |     |      0v |      |   | 25 || 26 | 1 | IN   | CE1     | 11  | 7   |
|   0 |  30 |   SDA.0 |   IN | 1 | 27 || 28 | 1 | IN   | SCL.0   | 31  | 1   |
|   5 |  21 | GPIO.21 |   IN | 1 | 29 || 30 |   |      | 0v      |     |     |
|   6 |  22 | GPIO.22 |   IN | 1 | 31 || 32 | 0 | IN   | GPIO.26 | 26  | 12  |
|  13 |  23 | GPIO.23 |   IN | 0 | 33 || 34 |   |      | 0v      |     |     |
|  19 |  24 | GPIO.24 |   IN | 0 | 35 || 36 | 0 | IN   | GPIO.27 | 27  | 16  |
|  26 |  25 | GPIO.25 |   IN | 0 | 37 || 38 | 0 | IN   | GPIO.28 | 28  | 20  |
|     |     |      0v |      |   | 39 || 40 | 0 | IN   | GPIO.29 | 29  | 21  |
+-----+-----+---------+------+---+----++----+---+------+---------+-----+-----+
| BCM | wPi |   Name  | Mode | V | Physical | V | Mode | Name    | wPi | BCM |
+-----+-----+---------+------+---+---Pi 3B--+---+------+---------+-----+-----+
```

Q3 result and demo:

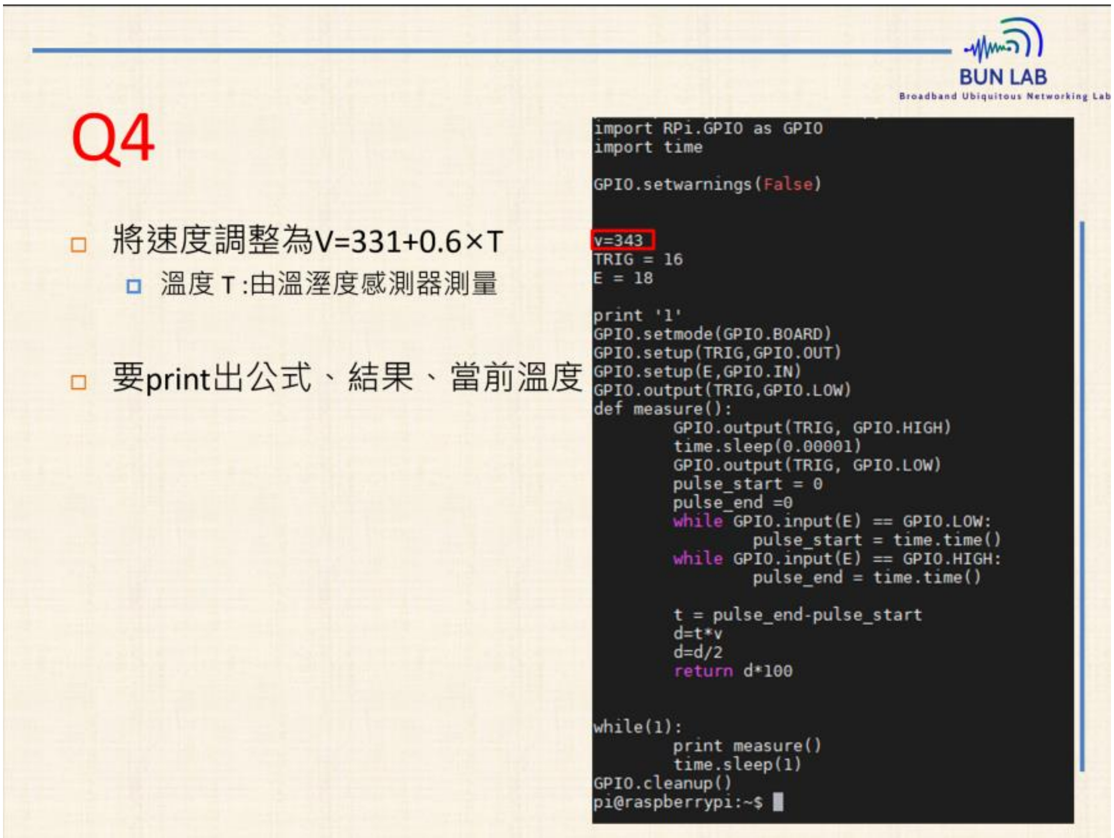


```
    GPIO.output(LED_PIN, GPIO.LOW)

while(1):

    distance = measure()
    print(distance)
```

```
pi@raspberrypi:~/Communication-Networks-Laboratory/IOT_Lab1/Q3 $ python Q3.py
```

https://youtu.be/fkxidR9ER6A

Q4:



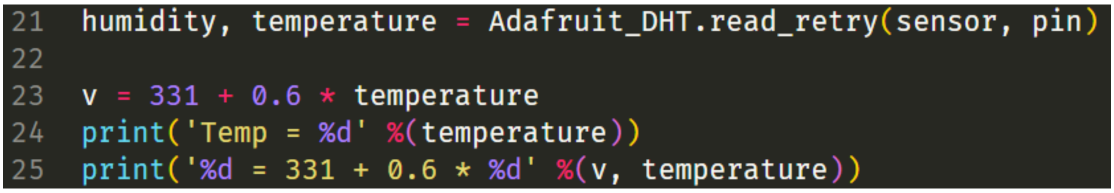
Q4

- 將速度調整為V=331+0.6×T
  - 溫度 T :由溫溼度感測器測量
- 要print出公式、結果、當前溫度

```
import RPi.GPIO as GPIO
import time

GPIO.setwarnings(False)

v=343
TRIG = 16
E = 18

print '1'
GPIO.setmode(GPIO.BOARD)
GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(E,GPIO.IN)
GPIO.output(TRIG,GPIO.LOW)
def measure():
        GPIO.output(TRIG, GPIO.HIGH)
        time.sleep(0.00001)
        GPIO.output(TRIG, GPIO.LOW)
        pulse_start = 0
        pulse_end =0
        while GPIO.input(E) == GPIO.LOW:
                pulse_start = time.time()
        while GPIO.input(E) == GPIO.HIGH:
                pulse_end = time.time()

        t = pulse_end-pulse_start
        d=t*v
        d=d/2
        return d*100


while(1):
        print measure()
        time.sleep(1)
GPIO.cleanup()
pi@raspberrypi:~$ 
```

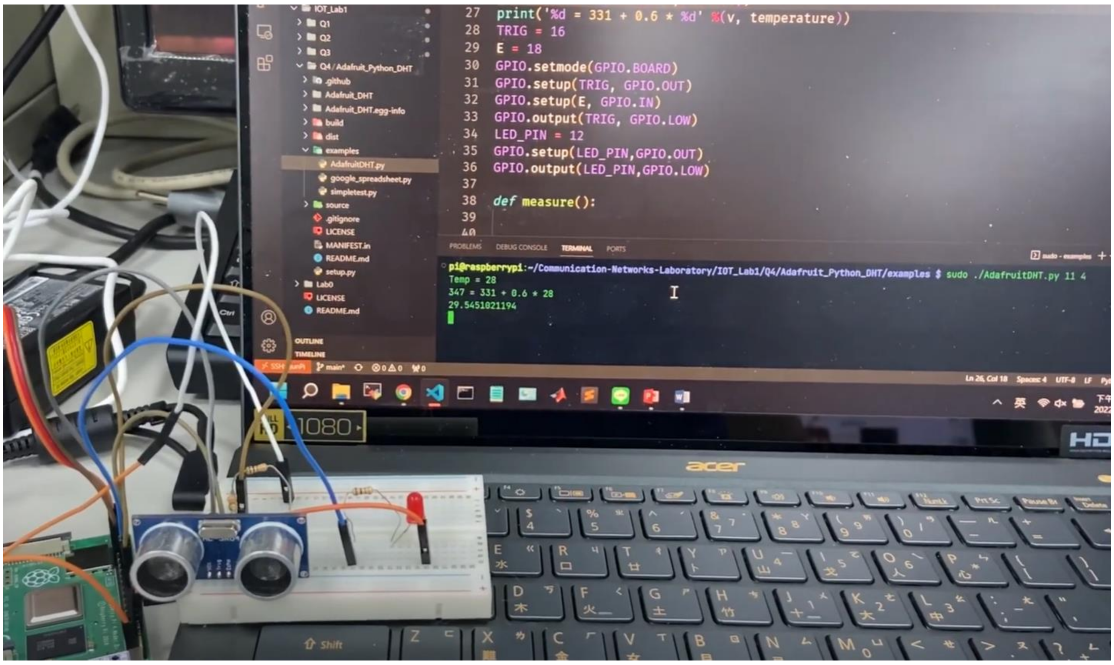
Code description:

It’s quite same as Q2 and Q3, just replace v with the formula.



```python
humidity, temperature = Adafruit_DHT.read_retry(sensor, pin)

v = 331 + 0.6 * temperature
print('Temp = %d' %(temperature))
print('%d = 331 + 0.6 * %d' %(v, temperature))
```

Q4 result and demo:



```python
print('%d = 331 + 0.6 * %d' %(v, temperature))
TRIG = 16
E = 18
GPIO.setmode(GPIO.BOARD)
GPIO.setup(TRIG, GPIO.OUT)
GPIO.setup(E, GPIO.IN)
GPIO.output(TRIG, GPIO.LOW)
LED_PIN = 12
GPIO.setup(LED_PIN,GPIO.OUT)
GPIO.output(LED_PIN,GPIO.LOW)

def measure():
```

```
pi@raspberrypi:~/Communication-Networks-Laboratory/IOT_Lab1/Q4/Adafruit_Python_DHT/examples $ sudo ./AdafruitDHT.py 11 4
Temp = 28
347 = 331 + 0.6 * 28
29.5451021194
```

https://www.youtube.com/watch?v=Bd-AdQ53wXw

Feedback:

After taking Electronic Circuit Lab in last full semester, playing around with Raspberry Pi and small circuit is fun to me. I am looking forward to the next week experiment! At first, using ttl to connect Raspberry Pi is annoying because the editor is limit only to vim and nano. Therefore, with TA’s help. I use ssh to connect to Raspberry Pi, and then using my favorite IDE-Vscode to increase the coding speed. I have used C with wiringPi to play with Raspberry Pi before, it’s quite different to play it with python, but I actually have fun in this first lab. By the way, I think Lab0 is a little bit difficult, it should just contain basic syntax for python, however it contains some advanced algorithm such as dynamic programming and prefix sum. It might not be friendly to the python beginner.