1. 實驗目的

學習如何使用 raspberry pi，並用 python code 來控制 LED 需要在何時亮、獲取溫溼度感測器所感應到的溫度和濕度值、獲取超音波感測器所得知的距離。還要知道要如何接線，要接到哪些對應的腳位。

Q1 要能控制 LED 燈每次亮暗的時間間隔

Q2 要能獲取溫溼度感測器的溫、濕度值，並根據輸入的 threshold 來控制 LED 燈要亮還是暗

Q3 要能獲取超音波感測器的距離，並根據距離大小來控制 LED 燈要亮還是暗

Q4 要能獲取溫溼度感測器的溫、濕度值，利用溫度來估算速度，再根據距離大小來控制 LED 燈要亮還是暗

2. 實驗過程 (Code + 說明)

Q1

```
# import package
import RPi.GPIO as GPIO
import time

# do not display warning messages
GPIO.setwarnings(False)
# numbered in order according to GPIO pin
GPIO.setmode(GPIO.BOARD)
# LED pin is 12 (GPIO18)
LED_PIN = 12
GPIO.setup(LED_PIN, GPIO.OUT)

while True:
  # s
  for i in range(3) :
    # set the LED_PIN to a high logic level (turn on the light)
    GPIO.output(LED_PIN, GPIO.HIGH)
    # lasting 0.1 seconds
    time.sleep(0.1)
    # set the LED_PIN to a low logic level (turn off the light)
    GPIO.output(LED_PIN, GPIO.LOW)
    # lasting 0.1 seconds
    time.sleep(0.1)
  time.sleep(0.3)

  # o
  for i in range(3) :
```

```
    # set the LED_PIN to a high logic level (turn on the light)
    GPIO.output(LED_PIN, GPIO.HIGH)
    # lasting 0.3 seconds
    time.sleep(0.3)
    # set the LED_PIN to a low logic level (turn off the light)
    GPIO.output(LED_PIN, GPIO.LOW)
    # lasting 0.1 seconds
    time.sleep(0.1)
  time.sleep(0.3)

  # s
  for i in range(3) :
    # set the LED_PIN to a high logic level (turn on the light)
    GPIO.output(LED_PIN, GPIO.HIGH)
    # lasting 0.1 seconds
    time.sleep(0.1)
    # set the LED_PIN to a low logic level (turn off the light)
    GPIO.output(LED_PIN, GPIO.LOW)
    # lasting 0.1 seconds
    time.sleep(0.1)
  time.sleep(0.7)
```

先將 LED 燈照講義的接法給接好，這邊要注意的是 LED 的正負極要看清楚，不然永遠都不會亮，然後再根據摩斯密碼來設定 S 跟 O 應該要是如何的亮法，再將此內容寫入 python code 執行後就能得到想要的結果。

Q2

```
#!/usr/bin/python

# import package
import sys
import Adafruit_DHT
import RPi.GPIO as GPIO
import time

# do not display warning messages
GPIO.setwarnings(False)
# numbered in order according to GPIO pin
GPIO.setmode(GPIO.BOARD)
```

```
# LED pin is 12 (GPIO18)
LED_PIN = 12
GPIO.setup(LED_PIN, GPIO.OUT)

sensor_args = { '11': Adafruit_DHT.DHT11,
                '22': Adafruit_DHT.DHT22,
                '2302': Adafruit_DHT.AM2302 }
# the sensor is the second parameter with a value of 11
# the pin is the third parameter with a value of 4
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)
# enter a temperature threshold value
threshold = input()

while True:
  # get humidity and temperature
  humidity, temperature = Adafruit_DHT.read_retry(sensor, pin)
  if humidity is not None and temperature is not None:
      # output temperature and humidity values
      print('Temp={0:0.3f}*  Humidity={1:0.3f}%'.format(temperature, humidity))
      # temperature is above the threshold
      if temperature >= threshold:
        # set the LED_PIN to a high logic level (turn on the light)
        GPIO.output(LED_PIN, GPIO.HIGH)
      # temperature is below the threshold
      else:
        # set the LED_PIN to a low logic level (turn off the light)
        GPIO.output(LED_PIN, GPIO.LOW)
  else:
      print('Failed to get reading. Try again!')
      sys.exit(1)
  time.sleep(1)
```

LED 燈的接法不用改變，然後將溫溼度感測器照講義的接法給接好，再根據 AdafruitDHT.py 來修改 python code，加上一行 threshold = input()讓門檻值能夠用輸入的方式自定義門檻值，然後再加上一個 if else 來讓溫度超過門檻值時 LED 燈亮起；反之則關閉。執行後就能得到想要的結果。

Q3

```
# import package
import RPi.GPIO as GPIO
import time

# do not display warning messages
GPIO.setwarnings(False)
# set v = 343m/s
v = 343
# set pin
TRIG = 16
E = 18
LED_PIN = 12

print('1')
# numbered in order according to GPIO pin
GPIO.setmode(GPIO.BOARD)
# set TRIG to GPIO.OUT
GPIO.setup(TRIG, GPIO.OUT)
# set E to GPIO.IN
GPIO.setup(E, GPIO.IN)
# set TRIG to a low logic level
GPIO.setup(TRIG, GPIO.LOW)
# set LED_PIN to GPIO.OUT
GPIO.setup(LED_PIN, GPIO.OUT)

def measure():
  # trigger a low logic level
  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:
    # start time
    pulse_start = time.time()
  while GPIO.input(E) == GPIO.HIGH:
    # time received from the echo
    pulse_end = time.time()
  # duration from emission to receiving the echo
  t = pulse_end - pulse_start
  # distance = time * velocity
  d = t * v
  # one-way distance
  d = d / 2
  # m to cm
  d = d * 100

  # LED exhibits different reactions based on varying distances
  if(d < 10):
    GPIO.output(LED_PIN, GPIO.HIGH)
  elif(d < 20):
    GPIO.output(LED_PIN, GPIO.HIGH)
    time.sleep(0.1)
    GPIO.output(LED_PIN, GPIO.LOW)
    time.sleep(0.1)
  else:
    GPIO.output(LED_PIN, GPIO.LOW)

  return d

while(1):
  # output distance(cm)
  print(measure())
  time.sleep(1)

GPIO.cleanup()
```

一樣 LED 燈的接法不用改變，但因為超音波感測器會用到 1K 電阻，所以要換一顆電阻給 LED 用，然後將溫溼度感測器先不用拆下來，下一題會用到。超音波感測器照講義的接法給接好，要注意要接 5V 而不是 3.3V，再根據講義上的範例 code 來修改 python code，其實要改的地方就只有根據距離長度來決定 LED 燈要怎麼亮或是要不要亮。

Q4

```
#!/usr/bin/python

# import package
import sys
import Adafruit_DHT
import RPi.GPIO as GPIO
import time

# do not display warning messages
GPIO.setwarnings(False)
# numbered in order according to GPIO pin
GPIO.setmode(GPIO.BOARD)

sensor_args = { '11': Adafruit_DHT.DHT11,
                '22': Adafruit_DHT.DHT22,
                '2302': Adafruit_DHT.AM2302 }

# the sensor is the second parameter with a value of 11
# the pin is the third parameter with a value of 4
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)

while True:
  # get humidity and temperature
  humidity, temperature = Adafruit_DHT.read_retry(sensor, pin)
  # set v = 331 + 0.6 * temperature
  v = 331 + 0.6 * temperature
  # set pin
  TRIG = 16
  E = 18
  LED_PIN = 12
```

```
print('1')

# numbered in order according to GPIO pin
GPIO.setmode(GPIO.BOARD)
# set TRIG to GPIO.OUT
GPIO.setup(TRIG, GPIO.OUT)
# set E to GPIO.IN
GPIO.setup(E, GPIO.IN)
# set TRIG to a low logic level
GPIO.setup(TRIG, GPIO.LOW)
# set LED_PIN to GPIO.OUT
GPIO.setup(LED_PIN, GPIO.OUT)

def measure():
  # trigger a low logic level
  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:
    # start time
    pulse_start = time.time()
  while GPIO.input(E) == GPIO.HIGH:
    # time received from the echo
    pulse_end = time.time()
  # duration from emission to receiving the echo
  t = pulse_end - pulse_start
  # distance = time * velocity
  d = t * v
  # one-way distance
  d = d / 2
  # m to cm
  d = d * 100

  # LED exhibits different reactions based on varying distances
  if(d < 10):
```

```
    GPIO.output(LED_PIN, GPIO.HIGH)
  elif(d < 20):
    GPIO.output(LED_PIN, GPIO.HIGH)
    time.sleep(0.1)
    GPIO.output(LED_PIN, GPIO.LOW)
    time.sleep(0.1)
  else:
    GPIO.output(LED_PIN, GPIO.LOW)

  return d

while(1):
  print('============================')
  # output temperature
  print('Temp:{0:0.1f}*C'.format(temperature))
  # output velocity
  print('V=331+0.6*{0:0.1f}'.format(temperature))
  print(' =' + str(v))
  # output distance(cm)
  print('Distance:' + str(measure()) + 'cm')

  time.sleep(1)

GPIO.cleanup()
```

電路接線部分不用更動，只要把程式碼中的 v 改成 331 + 0.6 * temperature 就好，其中 temperature 是由溫溼度感測器量到的。然後一樣根據距離來決定 LED 燈要怎麼亮或是要不要亮。

3. 問題與解法

在我要用 SSH 無線連線時，不管怎樣都無法連上，後來詢問助教後依舊找不到問題，最後改用自己的網路才解決。我推測可能是連 WiFi 分享器的人太多了？

在我做第三題時，發現超音波感測器量出來的距離都超級小，甚至還有負的，後來發現是我的超音波感測器要接 5V 而不是 3.3V，所以把 VCC 接到 5V 後就能夠正常解決了。

4. 心得

之前的 lab 就有用過 Raspberry Pi 了，Raspberry Pi 是由英國樹莓派基金會開發的單板電腦，使用 ARM 架構的 CPU，基本上只要加上一片 SD 卡作為儲存空間，就可以使用了，可以將它視為一張信用卡大小的微電腦。Raspberry Pi 最初的發展理念跟 Arduino 很像，它們都希望用便宜的硬體來促進 EE/CS 領域教育的發展。而 Raspberry Pi 能用來做機器人、無人機、掌機、平板等等應用，可以說是「麻雀雖小，五臟俱全」。

在做實驗時，要先透過 USB 轉 TTL 序列傳輸線將 Raspberry Pi 與電腦連接。連接完成後，要調整 Speed (bps)，上網查維基百科後，這指的是「有效數據訊號調變載波的速率」，就是單位時間內載波調變狀態變化的次數。1 鮑代表每秒傳輸 1 個符號，典型的鮑率是 300、1200、2400、9600、115200、19200 等 bps，以這次實驗為例則表示 1 秒可以傳輸 115200 個符號，值得注意的是，「鮑」(Baud)本身已是速率，不需要寫成「鮑率」(Baud Rate)，但在中文口語化的溝通上還是常以「鮑率」來描述「鮑」。

這次實驗並不難，因為之前就有用過樹莓派了，且之前的實驗也都有用到 LED、溫溼度感測器以及超音波感測器，所以接線方面並不難，code 方面也是大概修改講義給的 code 就好，我這次花最多時間的是解決我不能用無線的問題，光是處理這個問題大概就花了我一個小時左右。