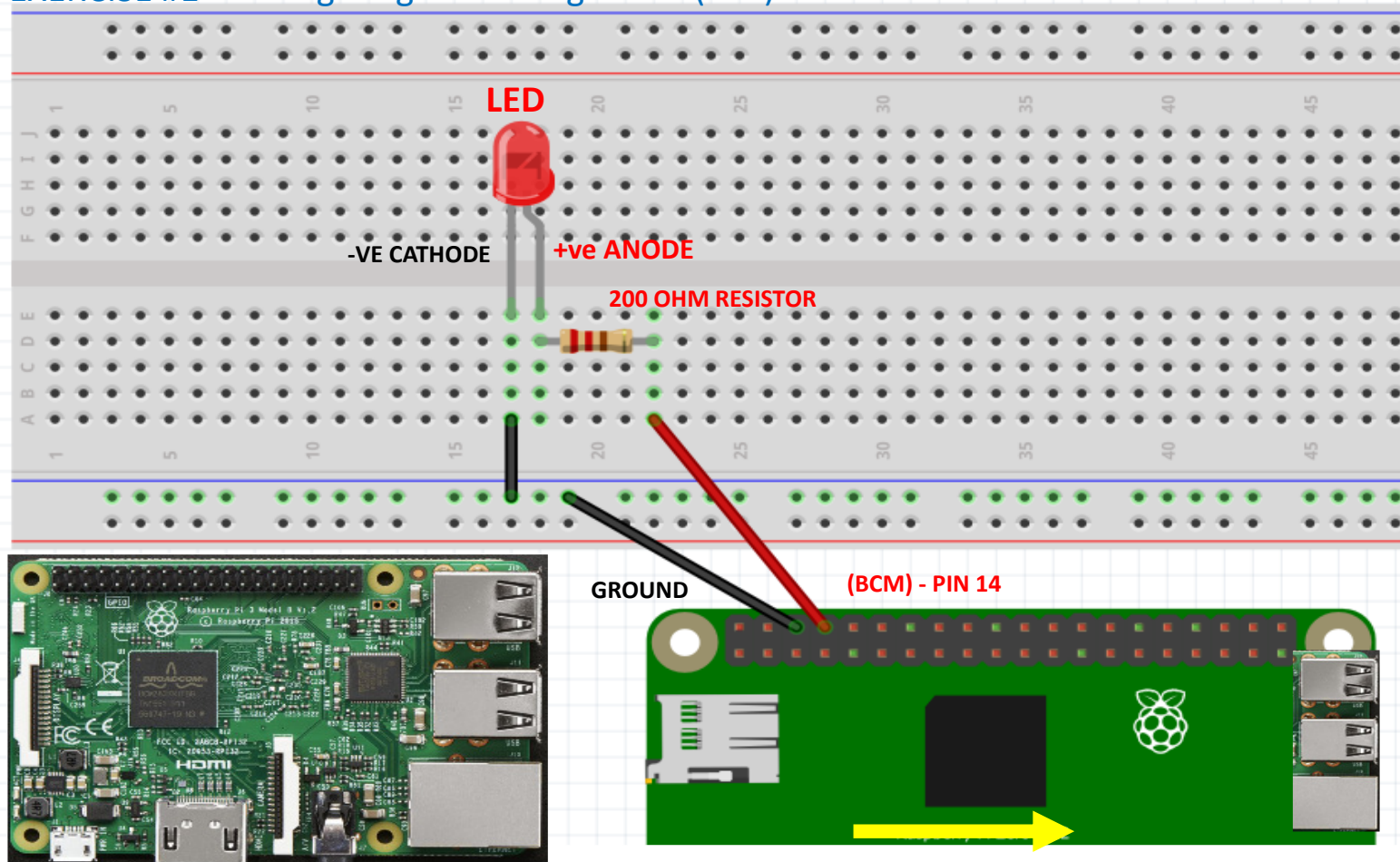


Tampines Regional Library  
LearnX Community  
Pi Python Introductory Course  
Course Material

## EXERCISE #1 Wiring a Light Emitting Diode (LED)



## TESTING OUR CIRCUIT USING THONNY SHELL

### Ex 1a. Turning our Red LED on

```
>>> from gpiozero import LED  
>>> red_led = LED(14)  
>>> red_led.on()
```

### Ex 1b. Turning it off

```
>>> red_led.off()
```

### Ex 1c. Make it blink (1 sec on 1 sec off)

```
>>> red_led.blink()
```

### Ex 1d. Make it pulsate a number of times

```
>>> red_led.blink(on_time=.5, off_time=.8, n=5)
```

## TESTING OUR CIRCUIT USING THE THONNY EDITOR

Ex 1e. Creating our 1<sup>st</sup> Raspberry Pi Python Program

Click on the green + icon to open a new page

And type these codes into the Editor

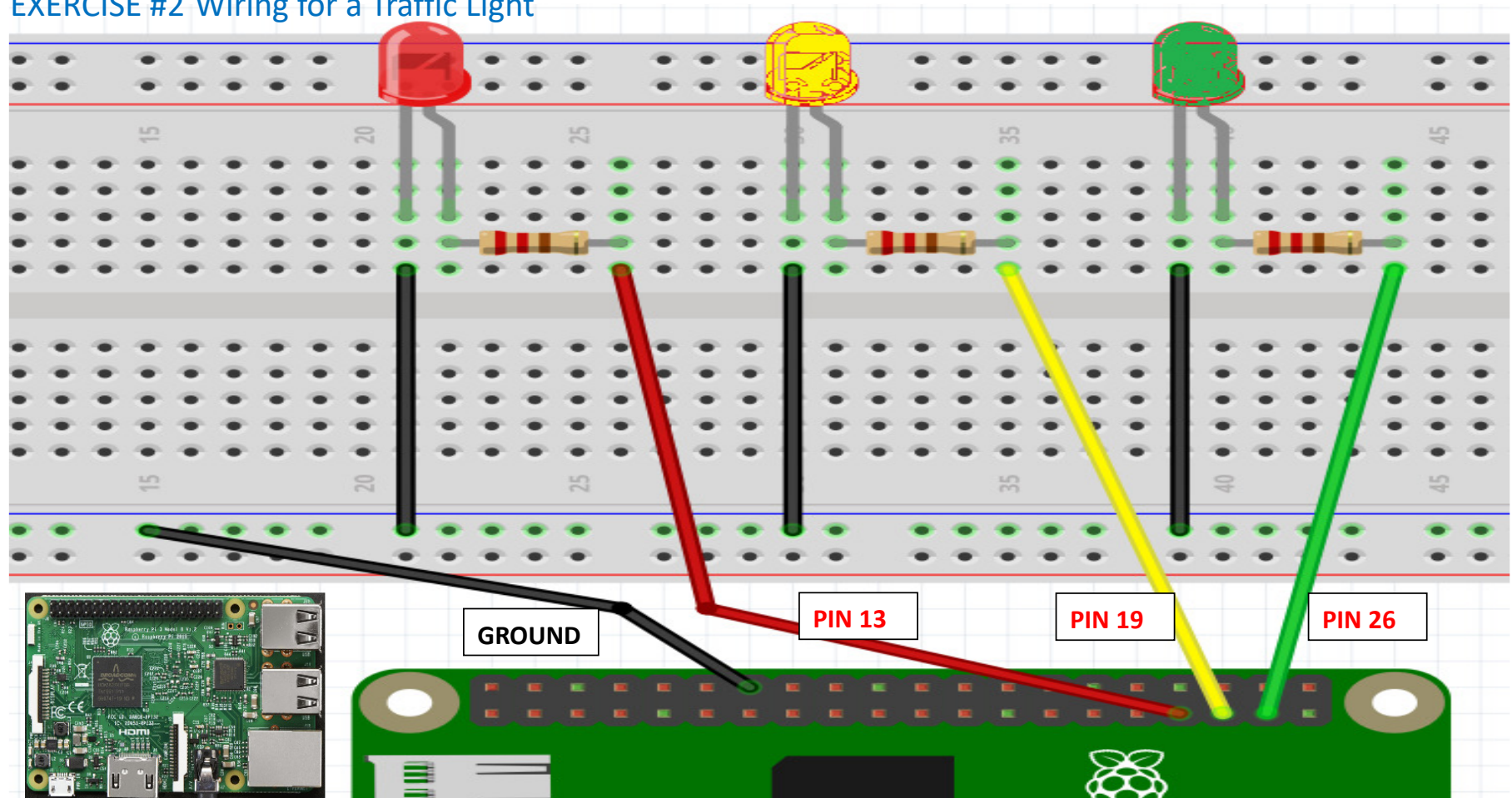
```
#Library # in python means – remarks for humans
from gpiozero import LED
from time import sleep
#Setup
tf_red_led = LED(14)
#Algorithm / Logic / Sequence
tf_red_led.on()
sleep(5)    #pause for 5 seconds
tf_red_led.off()
sleep(5)    #pause for 5 seconds
tf_red_led.blink(on_time=.5,off_time=.5,n=5)
sleep(5)    #allow 5 seconds for the blink to complete
```

Save this program

File -> Save [provide a programme name. E.g. **Excercise1.py** then click OK

Click Run or the play icon to run the program

## EXERCISE #2 Wiring for a Traffic Light



Ex. 2a – Using the Logic from 1e, implement this.  
Save program as Exercise2a.py

### ALGORITHM FOR A TRAFFIC LIGHT SYSTEM

STARTS WITH GREEN  
AFTER 10 SECONDS  
GREEN GOES OFF  
AMBER COMES ON  
AFTER 5 SECONDS  
AMBER GOES OFF  
RED COMES ON  
AFTER 10 SECONDS  
GREEN COMES ON



```
Exercise2a.py ×
1 #Library
2 from gpiozero import LED
3 from time import sleep
4
5 #Setup Components / Variables
6 tf_red_led = LED(13)
7 tf_amber_led=LED(19)
8 tf_green_led = LED(26)
9
10 #Algorithm
11
12 #initialize the LEDs
13 tf_red_led.off()
14 tf_amber_led.off()
15 tf_green_led.off()
16
17
18 tf_green_led.on()
19 sleep(10)
20 tf_green_led.off()
21 tf_amber_led.on()
22 sleep(5)
23 tf_amber_led.off()
24 tf_red_led.on()
25 sleep(10)
26 tf_red_led.off()
27 tf_green_led.on()
28
```

### Ex. 2b Traffic Light in a Loop (Exercise2b.py)

```
Exercise2b.py x
1 #Library
2 from gpiozero import LED
3 from time import sleep
4
5 #Setup Components / Variables
6 tf_red_led = LED(13)
7 tf_amber_led=LED(19)
8 tf_green_led = LED(26)
9
10 #Algorithm
11
12 #initialize the LEDs
13 tf_red_led.off()
14 tf_amber_led.off()
15 tf_green_led.off()
16
17
18 while True:
19     tf_green_led.on()
20     sleep(10)
21     tf_green_led.off()
22     tf_amber_led.on()
23     sleep(5)
24     tf_amber_led.off()
25     tf_red_led.on()
26     sleep(10)
27     tf_red_led.off()
```

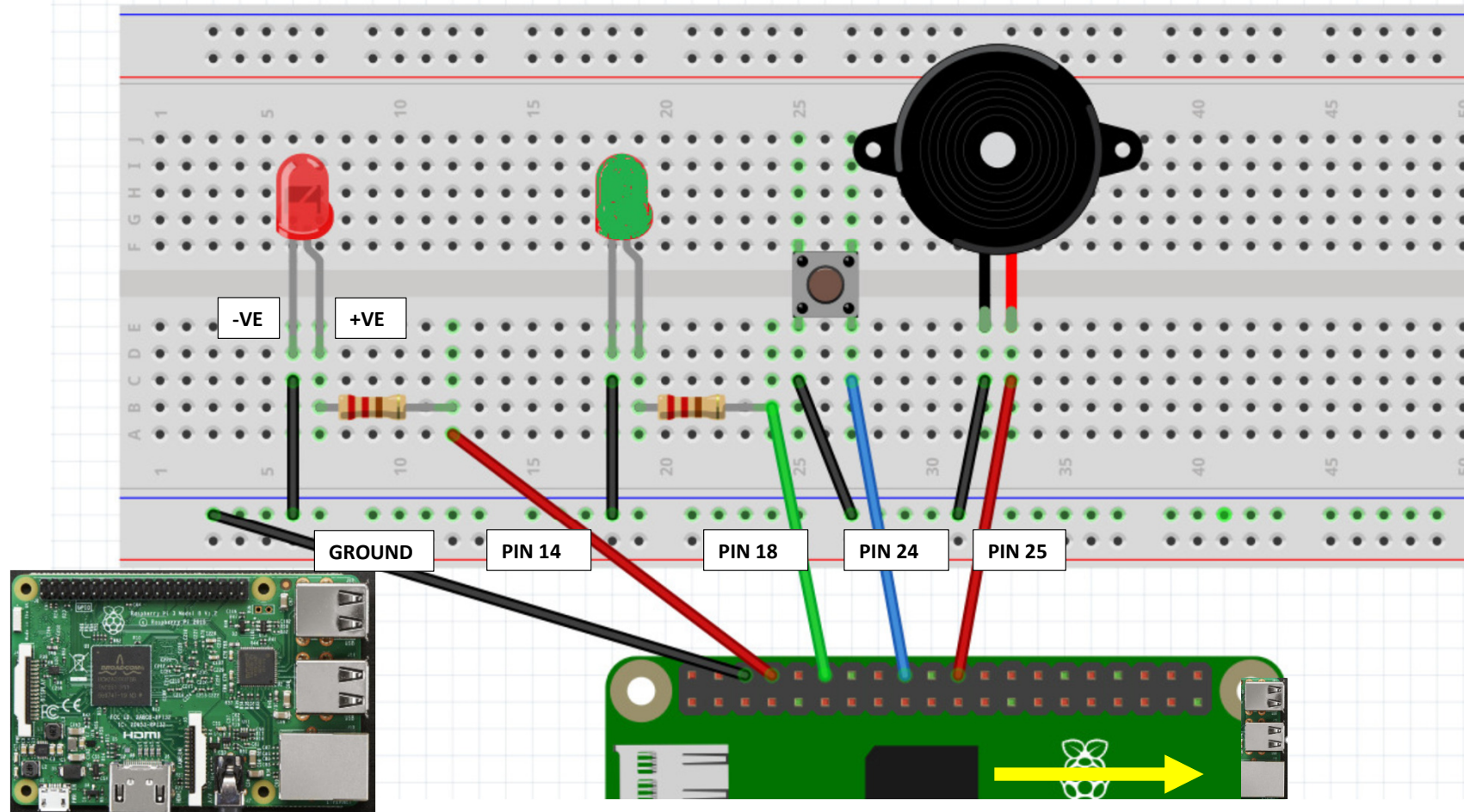
Functionalize

### Ex. 2c Traffic Light as a Function (Exercise2c.py)

```
Exercise2c.py x
1 #Library
2 from gpiozero import LED
3 from time import sleep
4
5 #Setup Components / Variables
6 tf_red_led = LED(13)
7 tf_amber_led=LED(19)
8 tf_green_led = LED(26)
9
10 #Algorithm
11 #initialize the LEDs
12 tf_red_led.off()
13 tf_amber_led.off()
14 tf_green_led.off()
15
16 #Function
17 def trafficLight():
18     tf_green_led.on()
19     sleep(10)
20     tf_green_led.off()
21     tf_amber_led.on()
22     sleep(5)
23     tf_amber_led.off()
24     tf_red_led.on()
25     sleep(10)
26     tf_red_led.off()
27
28 while True:
29     trafficLight()
```



### Exercise 3 – Wiring a Pedestrian Crossing





## TESTING OUR CIRCUIT USING THONNY SHELL

### Ex 3a. Turning our Buzzer on

```
>>> from gpiozero import Buzzer  
>>> buzz= Buzzer(25)  
>>> buzz.on()
```

### Ex 3b. Turning it off

```
>>> buzz.off()
```

### Ex 3c. Make it blink – 1 sec on 1 sec off

```
>>> buzz.blink()
```

### Ex 3d. Make it pulsate a number of times

```
>>> buzz.blink(on_time=.5, off_time=.8, n=5)
```

## TESTING OUR CIRCUIT USING THONNY SHELL

### Ex 3e. Button Activating/Deactivating Buzzer

```
Python 3.9.2 (/usr/bin/python3)
>>> from gpiozero import Buzzer, LED, Button
>>> from signal import pause
>>> buzz = Buzzer(25)
>>> button=Button(24)
>>> button.when_pressed = buzz.on
>>> button.when_released = buzz.off
>>> pause()
|
```

### Ex 3f. Button Activating / Deactivating Red LED (Pin 14)



Ex. 3g Pedestrian Crossing Program – Version 1  
Save program as Exercise3g.py

### ALGORITHM FOR A PEDESTRIAN CROSSING

STARTS WITH RED  
AFTER 10 SECONDS  
RED GOES OFF  
GREEN COMES ON  
AFTER 10 SECONDS  
GREEN BLINKS FIVE TIMES  
AFTER 5 SECONDS  
GREEN GOES OFF  
RED COMES ON



```
Exercise3g.py *%
1 #Library
2 from gpiozero import LED
3 from time import sleep
4 #Component Setup
5 green_led = LED(18)
6 red_led=LED(14)
7 |
8 #Algorithm
9 red_led.on()
10 sleep(10)
11 red_led.off()
12 green_led.on()
13 sleep(10)
14 green_led.blink(on_time=.5,off_time=.5,n=5)
15 sleep(5)
16 green_led.off()
17 red_led.on()
18
```

### Ex. 3h Pedestrian Crossing Program – Version 2 – With Buzzer

Save program as Exercise3h.py

#### ALGORITHM FOR A PEDESTRIAN CROSSING

STARTS WITH RED  
AFTER 10 SECONDS  
RED GOES OFF  
GREEN COMES ON  
AFTER 10 SECONDS  
GREEN BLINKS FIVE TIMES  
**BUZZER BLINKS FIVE TIMES**  
AFTER 5 SECONDS  
GREEN GOES OFF  
RED COMES ON



```
Exercise3h.py x
1 #Library
2 from gpiozero import LED, Buzzer
3 from time import sleep
4 #Component Setup
5 green_led = LED(18)
6 red_led=LED(14)
7 buzz=Buzzer(25)
8
9 #Algorithm
10 red_led.on()
11 sleep(10)
12 red_led.off()
13 green_led.on()
14 sleep(10)
15 green_led.blink(on_time=.5,off_time=.5,n=5)
16 buzz.blink(on_time=.5, off_time=.5, n=5)
17 sleep(5)
18 green_led.off()
19 red_led.on()
20
21
```

## Ex. 3j Pedestrian Crossing Program – Version 3 – Introducing a Function

Save program as Exercise3j.py

### ALGORITHM FOR A PEDESTRIAN CROSSING

STARTS WITH RED  
AFTER 10 SECONDS  
RED GOES OFF  
GREEN COMES ON  
AFTER 10 SECONDS  
GREEN BLINKS FIVE TIMES  
**BUZZER BLINKS FIVE TIMES**  
AFTER 5 SECONDS  
GREEN GOES OFF  
RED COMES ON



```
Exercise3j.py ✕
1 #Library
2 from gpiozero import LED, Buzzer
3 from time import sleep
4 #Component Setup
5 green_led = LED(18)
6 red_led=LED(14)
7 buzz=Buzzer(25)
8
9 #Function
10 def greenman():
11     red_led.on()
12     sleep(10)
13     red_led.off()
14     green_led.on()
15     sleep(10)
16     green_led.blink(on_time=.5,off_time=.5,n=5)
17     buzz.blink(on_time=.5, off_time=.5, n=5)
18     sleep(5)
19     green_led.off()
20     red_led.on()
21
22 #Algorithm
23 while True:
24     greenman()
```

## Ex. 3k Pedestrian Crossing Program – Version 4 – Add a Button

Save program as Exercise3k.py

### ALGORITHM FOR A PEDESTRIAN CROSSING

STARTS WITH RED ON

**WHEN BUTTON IS PRESSED**

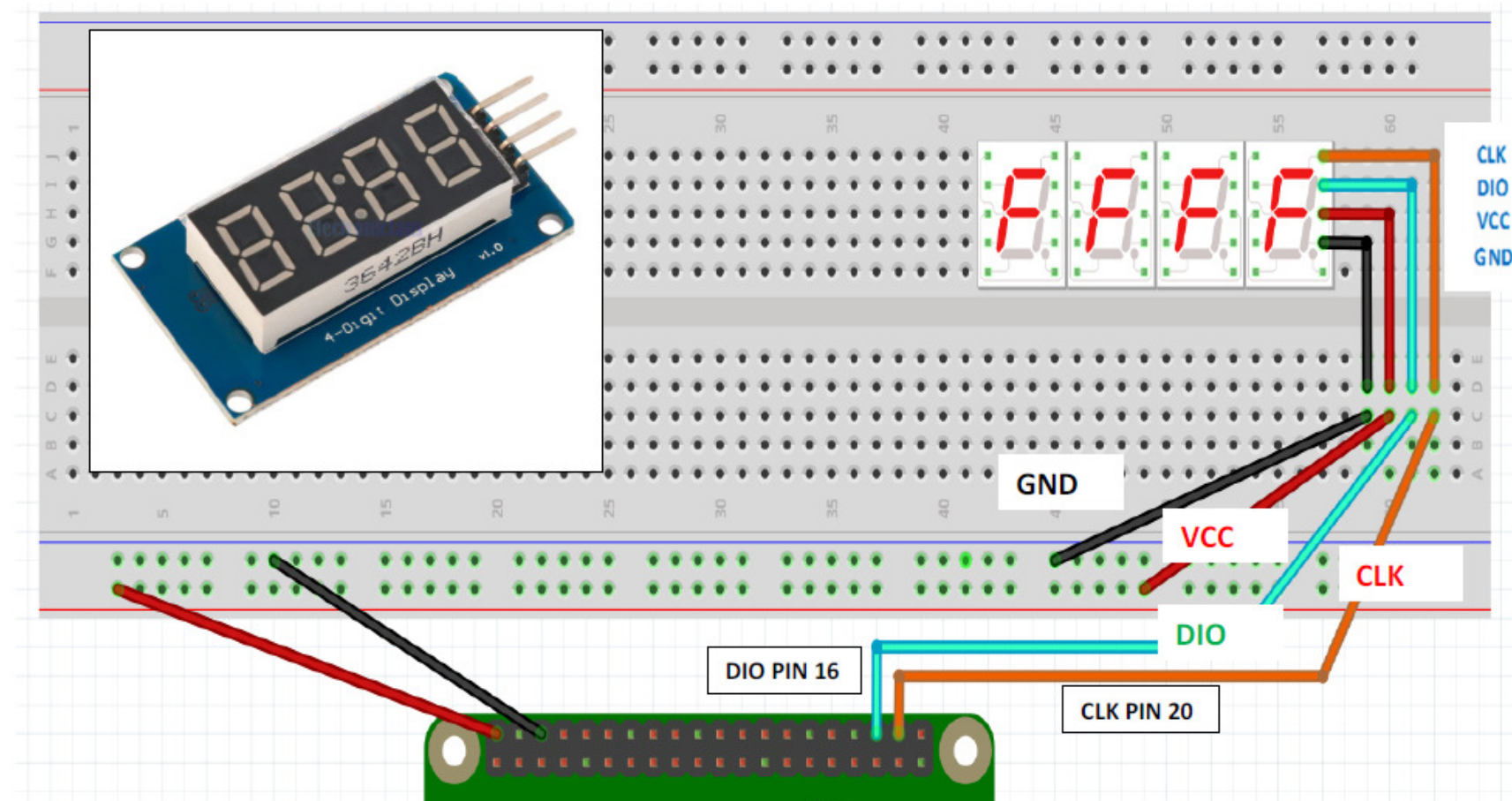
ACTIVATE Function greenman()



```
Exercise3k.py *  
1 #Library  
2 from gpiozero import LED, Buzzer, Button  
3 from time import sleep  
4 from signal import pause  
5 #Component Setup  
6 green_led = LED(18)  
7 red_led=LED(14)  
8 buzz=Buzzer(25)  
9 button=Button(24)  
10  
11 #Function  
12 def greenman():  
13     print('Button was pressed')  
14     sleep(10)  
15     red_led.off()  
16     green_led.on()  
17     sleep(10)  
18     green_led.blink(on_time=.5,off_time=.5,n=5)|  
19     buzz.blink(on_time=.5, off_time=.5, n=5)  
20     sleep(5)  
21     green_led.off()  
22     red_led.on()  
23     print('waiting for Next Button Press')  
24  
25 #Algorithm  
26 red_led.on()  
27 print('Program Started')  
28 print('waiting for Button to be Pressed')  
29 button.when_pressed = greenman  
30 pause()
```

## Exercise 4 – Wiring and Testing Count Down Display

### TM1637 7-SEGMENT DISPLAY





## TESTING OUR CIRCUIT USING THONNY SHELL

### Ex 4a

```
>>> import tm1637
>>> display = tm1637.TM1637(20, 16)
>>> #20=CLK 16=DIO
>>> display.clear()
>>> display.set_values([' ', ' ', ' ', '7'])
```

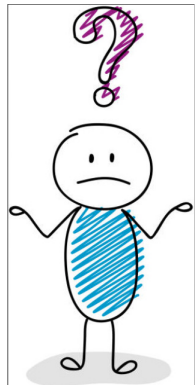


### Ex 4b

```
>>> display.clear()
>>> display.set_values([' ', ' ', '8 ', '7'])
```



### Ex 4c



How do you display 1234

The 7 Segment Display Accepts Data as a List

A List is a collection of Data

Like this -> ['Apple','Orange','Raspberry','Banana']

We can assign as variable name to it

```
FruitBasket= ['Apple','Orange','Raspberry','Banana']
```

```
>>> FruitBasket= ['Apple','Orange','Raspberry','Banana']
```

Each it in the List has an index position, starting from 0

```
>>> print(FruitBasket[0] )
```

We can change the 'value' of an item in the List

e.g. we want to replace 'Orange' with 'Grapes'

```
>>> FruitBasket[1]='Grapes'
```

```
>>> print(FruitBasket)
```

Making the 7 Segment Display Library takes the display  
Data as a List

[ ' ', ' ', ' ', '9']  
[ ' ', ' ', ' ', '8']  
[ ' ', ' ', ' ', '7']  
[ ' ', ' ', ' ', '6']  
[ ' ', ' ', ' ', '5']  
[ ' ', ' ', ' ', '4']  
[ ' ', ' ', ' ', '3']  
[ ' ', ' ', ' ', '2']  
[ ' ', ' ', ' ', '1']  
[ ' ', ' ', ' ', '0']



```
counter.py * ×  
1 import tm1637  
2 from time import sleep  
3 display = tm1637.TM1637(20, 16)  
4 display.set_values([' ', ' ', ' ', '9'])  
5 sleep(1)  
6 display.set_values([' ', ' ', ' ', '8'])  
7 sleep(1)  
8 display.set_values([' ', ' ', ' ', '7'])  
9 sleep(1)  
10 display.set_values([' ', ' ', ' ', '6'])  
11 sleep(1)  
12 display.set_values([' ', ' ', ' ', '5'])  
13 sleep(1)  
14 display.set_values([' ', ' ', ' ', '4'])  
15 sleep(1)  
16 display.set_values([' ', ' ', ' ', '3'])  
17 sleep(1)  
18 display.set_values([' ', ' ', ' ', '2'])  
19 sleep(1)  
20 display.set_values([' ', ' ', ' ', '1'])  
21 sleep(1)  
22 display.set_values([' ', ' ', ' ', '0'])  
23 sleep(1)  
24
```

The data to be  
displayed is sent to the  
function in the library  
`display.set_values`

At 1 seconds intervals  
in this case

## Introduction to the Python For Loop

counter.py \* x

```
1 import tm1637
2 from time import sleep
3 display = tm1637.TM1637(20, 16)
4 display.set_values([' ', ' ', ' ', '9'])
5 sleep(1)
6 display.set_values([' ', ' ', ' ', '8'])
7 sleep(1)
8 display.set_values([' ', ' ', ' ', '7'])
9 sleep(1)
10 display.set_values([' ', ' ', ' ', '6'])
11 sleep(1)
12 display.set_values([' ', ' ', ' ', '5'])
13 sleep(1)
14 display.set_values([' ', ' ', ' ', '4'])
15 sleep(1)
16 display.set_values([' ', ' ', ' ', '3'])
17 sleep(1)
18 display.set_values([' ', ' ', ' ', '2'])
19 sleep(1)
20 display.set_values([' ', ' ', ' ', '1'])
21 sleep(1)
22 display.set_values([' ', ' ', ' ', '0'])
23 sleep(1)
24
```

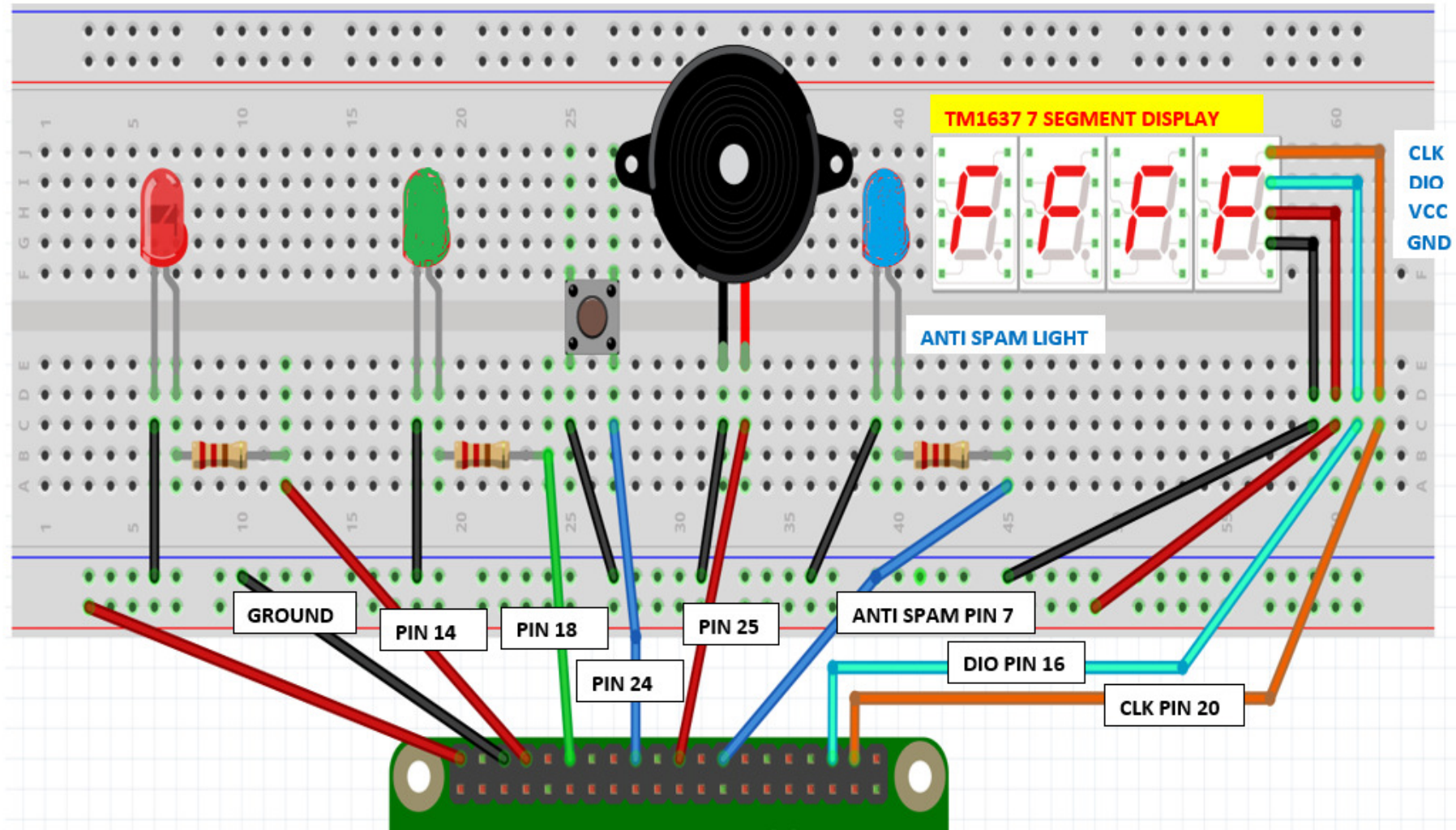
This value is changing from 9 to 0  
Create a variable  
to represent this value

Give it a name **counter** and run it in a for loop

counter.py \* x

```
1 #Library
2 import tm1637
3 from time import sleep
4 #Setup
5 display = tm1637.TM1637(20, 16)
6 #Algorithm
7 for counter in range(9,-1,-1):
8     display.set_values([' ', ' ', ' ', counter])
9     sleep(1)
10
```

## Exercise 5 – Wiring Pedestrian Crossing with Anti Spam LED and Count Down Display



Putting everything we have learned together into a full fledged pedestrian crossing

Recap our Algorithm and Code for earlier version

## ALGORITHM FOR A PEDESTRIAN CROSSING

STARTS WITH RED ON

**WHEN BUTTON IS PRESSED**

ACTIVATE Function greenman()

```
Exercise3k.py *  
1 #Library  
2 from gpiozero import LED, Buzzer, Button  
3 from time import sleep  
4 from signal import pause  
5 #Component Setup  
6 green_led = LED(18)  
7 red_led=LED(14)  
8 buzz=Buzzer(25)  
9 button=Button(24)  
10  
11 #Function  
12 def greenman():  
13     print('Button was pressed')  
14     sleep(10)  
15     red_led.off()  
16     green_led.on()  
17     sleep(10)  
18     green_led.blink(on_time=.5, off_time=.5, n=5)  
19     buzz.blink(on_time=.5, off_time=.5, n=5)  
20     sleep(5)  
21     green_led.off()  
22     red_led.on()  
23     print('waiting for Next Button Press')  
24  
25 #Algorithm  
26 red_led.on()  
27 print('Program Started')  
28 print('waiting for Button to be Pressed')  
29 button.when_pressed = greenman  
30 pause()
```

Now we want  
to add the  
Countdown  
here as well



## Injecting codes for countdown display

```
Exercise3k.py *  
1 #Library  
2 from gpiozero import LED, Buzzer, Button  
3 from time import sleep  
4 from signal import pause  
5 #Component Setup  
6 green_led = LED(18)  
7 red_led=LED(14)  
8 buzz=Buzzer(25)  
9 button=Button(24)  
10  
11 #Function  
12 def greenman():  
13     print('Button was pressed')  
14     sleep(10)  
15     red_led.off()  
16     green_led.on()  
17     sleep(10)  
18     green_led.blink(on_time=.5,off_time=.5,n=5)  
19     buzz.blink(on_time=.5, off_time=.5, n=5)  
20     sleep(5)  
21     green_led.off()  
22     red_led.on()  
23     print('waiting for Next Button Press')  
24  
25 #Algorithm  
26 red_led.on()  
27 print('Program Started')  
28 print('waiting for Button to be Pressed')  
29 button.when_pressed = greenman  
30 pause()
```

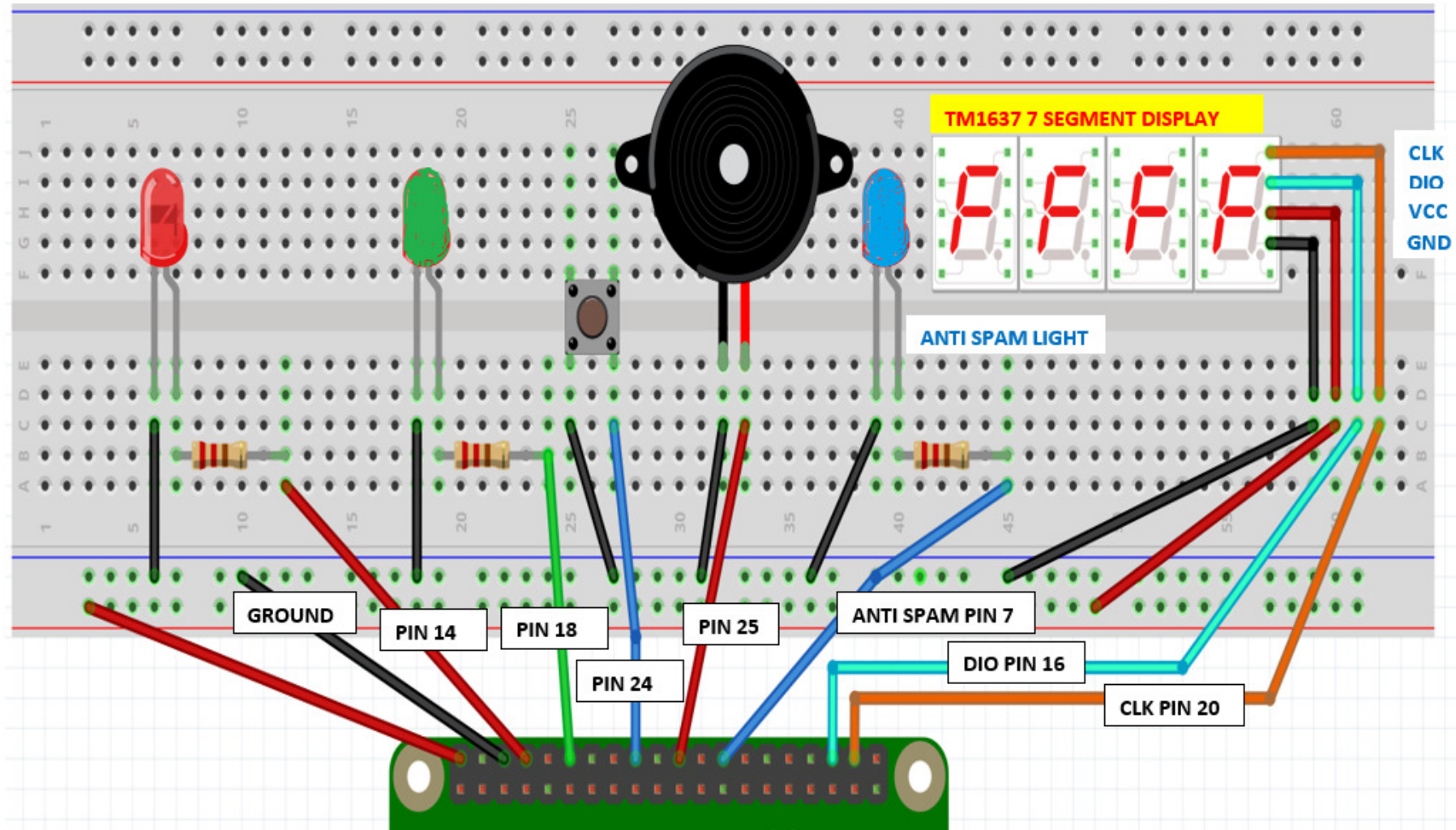
Now we want  
to add the  
Countdown  
here as well



```
Exercise3l.py x  
1 #Library  
2 from gpiozero import LED, Buzzer, Button  
3 from time import sleep  
4 from signal import pause  
5 import tm1637  
6 #Component Setup  
7 green_led = LED(18)  
8 red_led=LED(14)  
9 buzz=Buzzer(25)  
10 button=Button(24)  
11 display=tm1637.TM1637(20,16)  
12 #Function  
13 def greenman():  
14     print('Button was pressed')  
15     sleep(10)  
16     red_led.off()  
17     green_led.on()  
18     sleep(10)  
19     for counter in range(5,-1,-1):  
20         green_led.blink(on_time=.5,off_time=.5,n=1)  
21         buzz.blink(on_time=.5, off_time=.5, n=1)  
22         display.set_values([' ', ' ', ' ', counter])  
23         sleep(1)  
24     green_led.off()  
25     red_led.on()  
26     display.clear()  
27     print('waiting for Button to be Pressed')  
28 #Algorithm  
29 red_led.on()  
30 display.clear()  
31 print('waiting for Button to be Pressed')  
32 button.when_pressed = greenman  
33 pause()
```




## Exercise 5 – Wiring Pedestrian Crossing with Anti Spam LED and Count Down Display



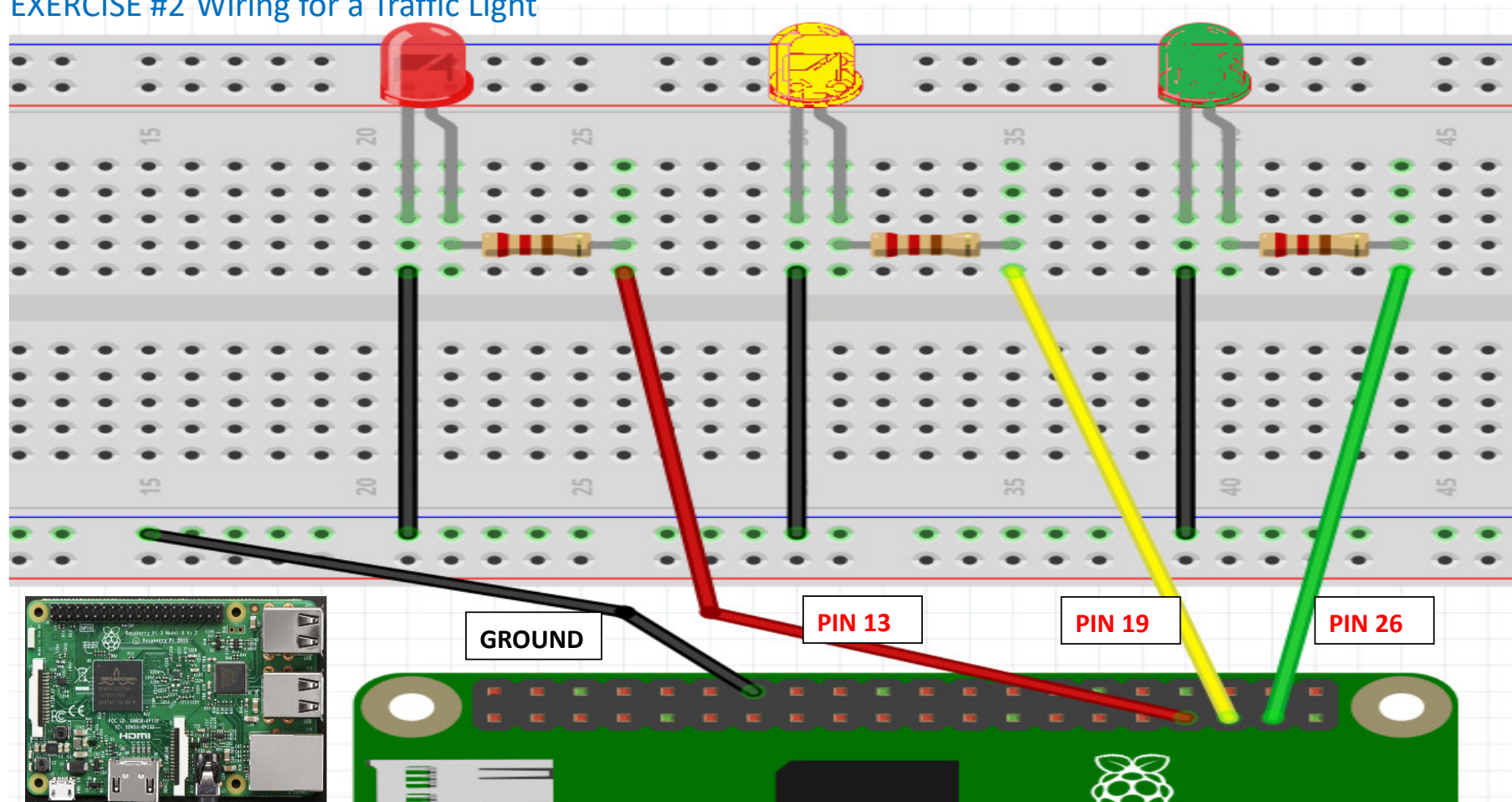
## Adding Check for Anti Spamming Exercise3m.py

```
#Library
from gpiozero import LED, Buzzer, Button
from time import sleep
from signal import pause
import tm1637
#Component Setup
green_led = LED(18)
red_led=LED(14)
anti_spam_led = LED(7)
buzz=Buzzer(25)
button=Button(24)
display=tm1637.TM1637(20,16)
#Function
def activate():
    if anti_spam_led.is_lit==True:
        pass
    else:
        anti_spam_button.on()
        greenman()
```



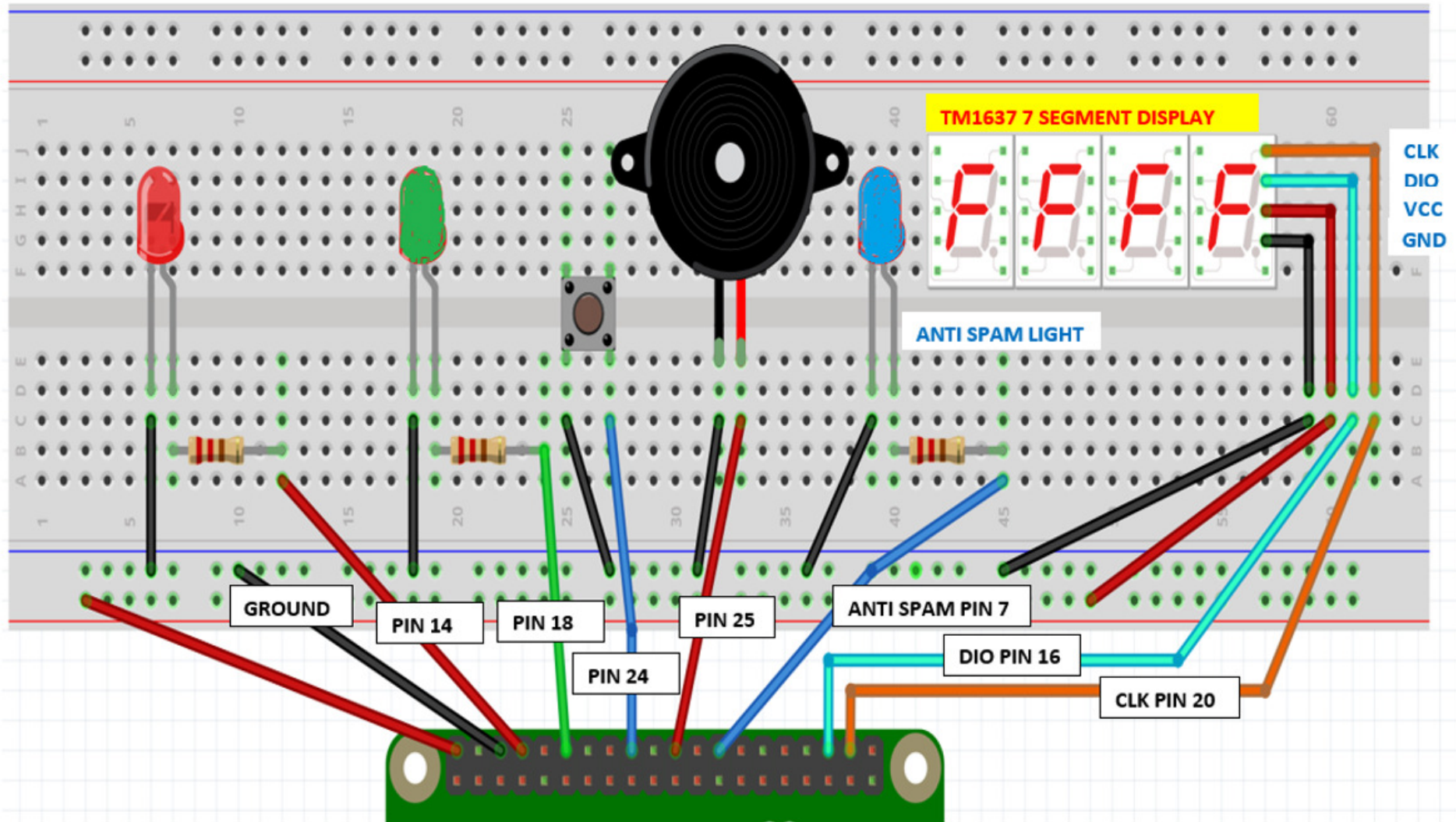
```
def greenman():
    print('Button was pressed')
    sleep(10)
    red_led.off()
    green_led.on()
    sleep(10)
    for counter in range(5,-1,-1):
        green_led.blink(on_time=.5,off_time=.5,n=1)
        buzz.blink(on_time=.5, off_time=.5, n=1)
        display.set_values([' ', ' ', ' ', counter])
        sleep(1)
    green_led.off()
    red_led.on()
    display.clear()
    print('waiting for Button to be Pressed')
#Algorithm
red_led.on()
anti_spam_led.off()
display.clear()
print('waiting for Button to be Pressed')
button.when_pressed=activate
pause()
```

## EXERCISE #2 Wiring for a Traffic Light



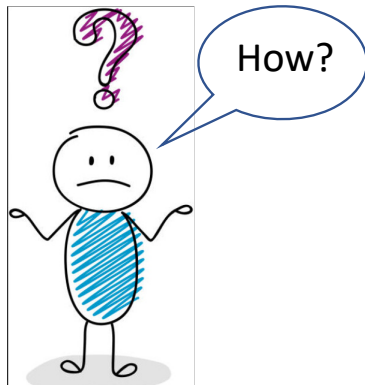


## Exercise 5 – Wiring Pedestrian Crossing with Anti Spam LED and Count Down Display



## Pedestrian Crossing and Traffic Light System

Co-ordinating the two systems is important



1		P.Crossing	Traffic Light
2	Start Point	Red	Green
3	When Button is Pressed		
4	Wait 5 Seconds ( <b>Sleep</b> )		
5			Amber Blinks
6	Wait 5 Seconds		
7		Green	Red
8	Wait 10 Seconds		
9		Green Blinks	
10		Buzzer Beeps	
11		Countdown Starts	
12	Wait 5 Seconds		
13		Red	Green

Suggestion for how this can be done

We take the code from Exercise3l.py

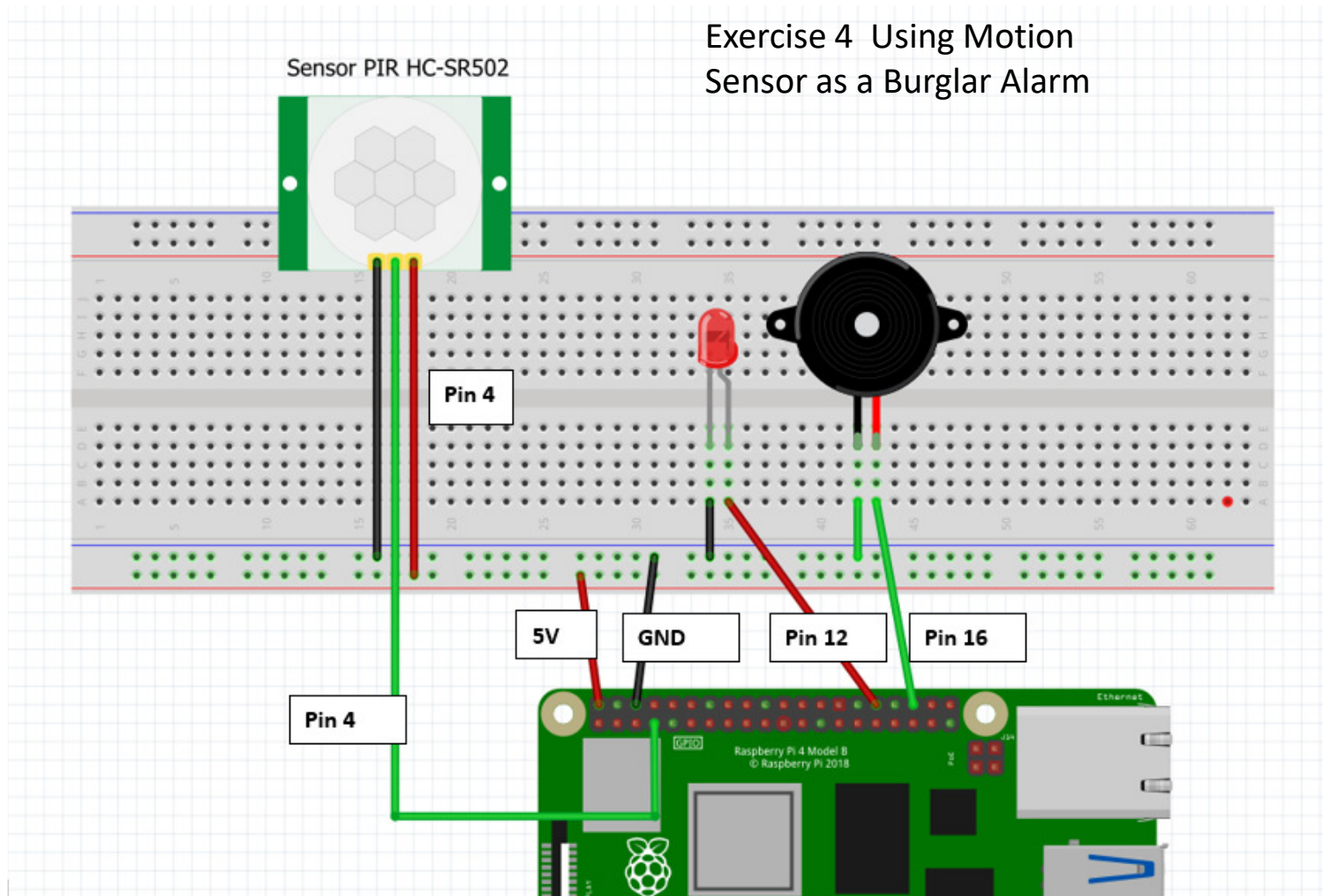
```
Exercise3l.py
1 #Library
2 from gpiozero import LED, Buzzer, Button
3 from time import sleep
4 from signal import pause
5 import tm1637
6 #Component Setup
7 green_led = LED(18)
8 red_led=LED(14)
9 buzz=Buzzer(25)
10 button=Button(24)
11 display=tm1637.TM1637(20,16)
12 #Function
13 def greenman():
14     print('Button was pressed')
15     sleep(10)
16     red_led.off()
17     green_led.on()
18     sleep(10)
19     for counter in range(5,-1,-1):
20         green_led.blink(on_time=.5,off_time=.5,n=1)
21         buzz.blink(on_time=.5, off_time=.5, n=1)
22         display.set_values([' ', ' ', ' ', ' ', counter])
23         sleep(1)
24     green_led.off()
25     red_led.on()
26     display.clear()
27     print('waiting for Button to be Pressed')
28 #Algorithm
29 red_led.on()
30 display.clear()
31 print('waiting for Button to be Pressed')
32 button.when_pressed = greenman
33 pause()
```

```
#Setup Components / Variables
tf_red_led = LED(13)
tf_amber_led=LED(19)
tf_green_led = LED(26)
```

Modify the codes to match the Algorithm

	P.Crossing	Traffic Light
1 Start Point	Red	Green
2 When Button is Pressed		
3 Wait 5 Seconds (Sleep)		
4		Amber Blinks
5 Wait 5 Seconds		
6	Green	Red
7 Wait 10 Seconds		
8	Green Blinks	
9	Buzzer Beeps	
10	Countdown Starts	
11 Wait 5 Seconds		
12	Red	Green
13		

## Exercise 4 Using Motion Sensor as a Burglar Alarm





#### **Ex 4a. Motion Sensor – LED Test**

Save this as Exercise4a.py

```
#Libraries
from gpiozero import MotionSensor, LED,
Buzzer
from signal import pause
from time import sleep
```

```
#Setup Variables for Components
motion_detector = MotionSensor(4)
red_led = LED(12)
buzz = Buzzer(16)
buzz.off()
```

```
#Algorithm
motion_detector.when_motion=
red_led.on
motion_detector.when_no_motion =
red_led.off
pause
```

#### **Ex 4b. Motion Sensor – Buzzer Test**

Save this as Exercise4b.py

```
#Libraries
from gpiozero import MotionSensor, LED, Buzzer
from signal import pause
from time import sleep
```

```
#Setup Variables for Componentst
motion_detector = MotionSensor(4)
red_led = LED(12)
buzz = Buzzer(16)
```

```
#Algorithm
buzz.off()
motion_detector.when_motion= buzz.on
motion_detector.when_no_motion = buzz.off
pause
```

#### **Ex4c Motion Sensor – FLASHING LED AND BUZZER**

Save program as Exercise4c.py

#Libraries

from gpiozero import MotionSensor, LED, Buzzer

from signal import pause

from time import sleep

#Setup Components / Variables

motion\_detector = MotionSensor(4)

red\_led = LED(12)

buzz = Buzzer(16)

buzz.off()

#Functions

**def alarm():**

    buzz.blink(on\_time=.5,off\_time=.5,n=5)

    red\_led.blink(on\_time=.5,off\_time=.5,n=5)

    sleep(5)

**def alarmOff():**

    buzz.off()

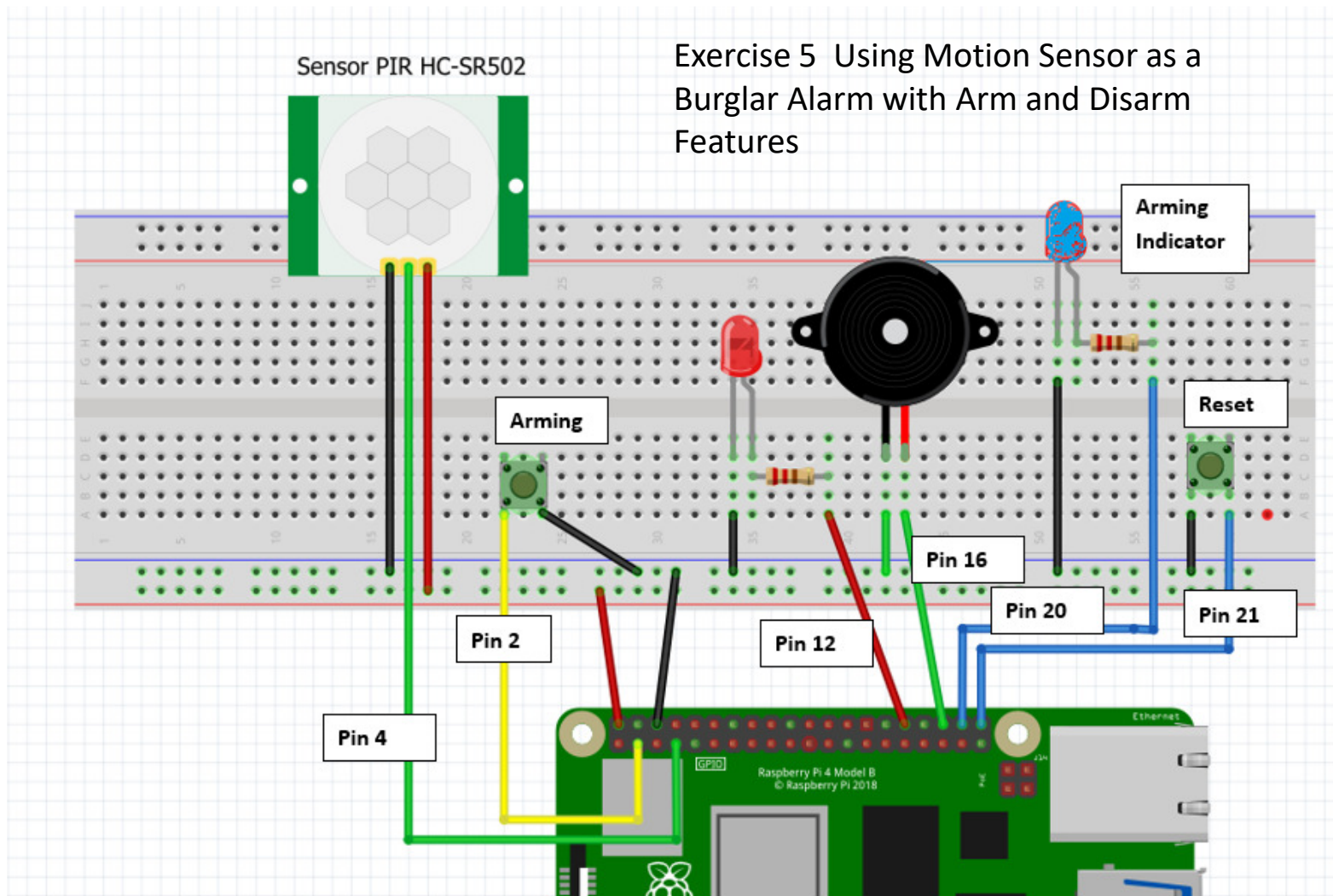
    red\_led.off()

#Algorithm

motion\_detector.when\_motion= alarm

motion\_detector.when\_no\_motion = alarmOff

pause



### Ex 5. BURGLAR ALARM WITH ARM AND RESET BUTTONS

#save this program as Exercise5.py

#Library

```
from gpiozero import MotionSensor, LED, Buzzer, Button
from signal import pause
from time import sleep
```

#Variables for Components

```
password='12345'
motion_detector = MotionSensor(4)
red_led = LED(12)
armed_led=LED(20)
armed_led.off()
arm_button=Button(2)
reset_button=Button(21)
buzz = Buzzer(16)
buzz.off()
```

#Functions

```
def alarm():
    red_led.blink(on_time=.5,off_time=.5,n=20)
    buzz.blink(on_time=.5,off_time=.5,n=20)
    sleep(20)
```

```
def armAlarm():
    armed_led.on()
    motion_detector.when_motion = alarm
```

```
def disarmAlarm():
    pwd=input('Enter Password')
    if pwd != password:
        pass
    else:
        armed_led.off()
        buzz.off()
        red_led.off()
        motion_detector.when_motion = None
```

DO Nothing

#Algorithm

```
arm_button.when_pressed = armAlarm
reset_button.when_pressed = disarmAlarm
```

```
while True:
    pause()
```