```python
import RPi.GPIO as GPIO


def turn_off(led):
    if GPIO.input(led):
        print("Turn off: " + str(led))
        GPIO.output(led, False)


def turn_on(led):
    if not GPIO.input(led):
        print("Turn on: " + str(led))
        GPIO.output(led, True)


class Button:

    def __init__(self, name, button_id, led1, led2, led3):
        self.name = name
        self.buttonId = button_id
        self.led1 = led1
        self.led2 = led2
        self.led3 = led3
        self.isRunning = False
        self.didPress = False
        self.holdLength = 0
        self.state = 0
        self.HOLD_THRESHOLD = 20

    def setup(self):
        if self.isRunning:
            return

        self.isRunning = True

        GPIO.setup(self.led1, GPIO.OUT)
        GPIO.setup(self.led2, GPIO.OUT)
        GPIO.setup(self.led3, GPIO.OUT)

        GPIO.setup(self.buttonId, GPIO.IN,
```

```
40 pull_up_down=GPIO.PUD_DOWN)
41
42         turn_off(self.led1)
43         turn_off(self.led2)
44         turn_off(self.led3)
45
46     def check_state(self):
47
48         if GPIO.input(self.buttonId):
49             if not self.didPress:
50                 self.state += 1
51                 self.didPress = True
52                 self.holdLength = 0
53                 print(self.name + " pressed, moving
   to state: " + str(self.state))
54         elif self.didPress:
55             self.didPress = False
56             self.holdLength = 0
57             print(self.name + " released")
58
59         #  Increment the holdLength if the button is
   held
60         if self.didPress:
61             self.holdLength += 1
62
63         #  Check if we have been holding longer than
   the threshold
64         if self.holdLength >= self.HOLD_THRESHOLD:
65             #  Reset the hold variables and set state
    to 0
66             self.holdLength = 0
67             self.state = 0
68             print(self.name + " reset by holding")
69
70         #  Turn on/off leds based on state
71         if self.state >= 3:
72             turn_on(self.led3)
73             self.state = 3
74
75         if self.state >= 2:
76             turn_on(self.led2)
```

```
77
78          if self.state >= 1:
79              turn_on(self.led1)
80
81          if self.state == 0:
82              turn_off(self.led1)
83              turn_off(self.led2)
84              turn_off(self.led3)
85
86      def get_name(self):
87          return self.name
88
```

```python
from datetime import date
from datetime import datetime

output_file = 'output.log'


# saves a pill counters value to file
def save_to_file(light,val):
    file = open(output_file, "a")  # open file in append

    out = str(datetime.now().strftime("%H:%M:%S")) + ' ' + str(date.today()) + ' :: ' + str(light) + ' ' + str(val) + '\n'

    file.write(str(out))  # write out to file
    file.close()  # close file


# appends a star message to the output file
def announce_start():
    file = open(output_file, "a")  # open file in append

    out = 'Program Started ' + str(datetime.now().strftime("%H:%M:%S")) + ' ' + str(date.today()) + '\n'

    file.write(str(out))  # write out to file
    file.close()  # close file
```

```python
import RPi.GPIO as GPIO
from Button import Button
import time

GPIO.setmode(GPIO.BOARD)

blueButton = Button("BlueButton", 40, 8, 19, 10)
# greenButton = Button("GreenButton", 6, 24, 17, 27)
# yellowButton = Button("YellowButton", 23, 15, 14, 4
)
# redButton = Button("RedButton", 16, 8, 25, 11)

blueButton.setup()
# greenButton.run()
# yellowButton.run()
# redButton.run()

while (True):
    blueButton.check_state()
    # greenButton.check_state()
    # yellowButton.check_state()
    # redButton.check_state()
    time.sleep(.125)

```