```
def main():
    """Determine the address of your
temperature data on the RPi, which
will always begin
    with a "28." In my case, my
`device_folder` is
    '/sys/bus/w1/devices/
28-83185055b9ff/w1-slave`. That is
the location on my RPi of
    where the raw temperature data is
being saved. The numbers following
the 28 prefix
    will be different for you.
    """


    base_dir = '/sys/bus/w1/devices/'
    device_folder = glob.glob(base_dir
+ '28*')[0]
    device_file = device_folder + '/
w1_slave'

    #Define your Twilio credentials
    account_sid = 'TWILIO_ACCOUNT'
    auth_token = 'TWILIO_TOKEN'
```

```
    client = Client(account_sid,
auth_token)

    #GPIO Setup. The code needs to
tell the RPi which GPIO pins to read
data from.
    temp_channel = 4
    temp = GPIO.setmode(GPIO.BCM)
    temp = GPIO.setup(temp_channel,
GPIO.IN)

    #Function to open the device file
and read the raw temperature data
    def read_temp_raw():
        f = open(device_file, 'r')
        lines = f.readlines()
        f.close()
        return lines

    #Function to extract and parse the
raw temp data, and convert Celsius to
Fahrenheit.
    def read_temp():
        lines = read_temp_raw()
        while lines[0].strip()[-3:] != 'YES':
            time.sleep(0.2)
            lines = read_temp_raw()
        equals_pos = lines[1].find('t=')
        if equals_pos != -1:
            temp_string = lines[1]
[equals_pos+2:]
```

```
        temp_c = float(temp_string) /
1000.0
        temp_f = temp_c * 9.0 / 5.0 +
32.0
        temp_f = round(temp_f)
        return temp_f

    #Function to create a text message
string if the temperature is too warm.
    def warm_message():
        client.messages.create(
          to='ALERT_PHONE',
          from_='TWILIO_PHONE',
          body="It's currently " +
str(read_temp()) + " degrees in my
crib, how about " \
          "turning up the air conditioning
or opening a window?")

    #Function to create a text message
string if the temperature is too cold.
    def cold_message():
        client.messages.create(
```

```
        client.messages.create(
          to='ALERT_PHONE',
          from_='TWILIO_PHONE',
          body="It's currently " +
str(read_temp()) + " degrees in my
crib, how about " \
          "turning up the air conditioning
or opening a window?")

    #Function to create a text message
string if the temperature is too cold.
    def cold_message():
        client.messages.create(
          to='ALERT_PHONE',
          from_='TWILIO_PHONE',
          body="It's currently " +
str(read_temp()) + " degrees in my
crib, how about " \
          "turning the heat up a little
bit?")

    #Run perpetually. Send the
message based on the temperature.
    while True:
      if read_temp() > 82:
        warm_message()
      if read_temp() < 60:
        cold_message()
      time.sleep(300)
```

```
#include <Adafruit_Nepixel.>

int ledpin=3;
int ledNo=12;

Adafruit_Neopixel strip=Adafruit_Neo
pixel(ledNo,ledpin,NEO_RGB);


int buzzerpin=2;
int echopin=6;
int trigpin=5;
int mindistance=100;
int maxdistance=300;

void setup()
{
  PinMode(buzzerpin,OUTPUT);
PinMode(trigpin,OUTPUT);
pinMode(echopin,INPUT);
serial.begin(9600);
strip.begin();
for(int i=0,i=ledNo;i++)
{
strip.setpixe1color(i,strip.color(0,0,0);
}
strip.show();
```

```
}
void loop()
{
int distance =calcdistance();
Serial.println(distance);
int ledsTOGlow=map (distance,mindi
stance,maxdistance,ledNo);
Serial.println(ledsTOGlow);
if(ledsTOGlow==12)
{
digitalWrite(buzzerpin,HIGH);
}
else
{
digitalWrite (buzzerpin,LOW);
}
```

```
if(i<4)
{
strip.setpixe1color(i,strip.color(50,0,0
));
}
else if (i>=4 && i<=8)
strip.setpixe1color(i,strip.color(50,50,
0));
}
strip.setpixe1color(i,strip.color(0,50,0
));
}
}
for(int i=ledsTOGlow;i<ledNo;i++)
{
strip.setpixe1color(i,strip.color(0,0,0))
;
}
strip.show();
delay(50);
}
int calcdistance ()
{
```

```
int calcdistance ()
{
long distance, duration ;
digital1Write(trigpin,LOW);
delayMicroseconds(2);
digital1Write (trigpin,HIGH);
delayMicroseconds (10);
digitalWrite (trigpin,LOW);
duration=pulseIn(echopin,HIGH);
distance=duration /29/2;
if(distance >=maxDistance)
{
distance=maxDistance;
}
if(distance<=minDistance)
{
distance=minDistance;
}
return distance;
}
```