```
import os
import glob
import time
import RPi.GPIO as GPIO
import sys
import shutil
GPIO.setwarnings(False)

#sets up IO and temp readings
os.system('modprobe w1-gpio')
os.system('modprobe w1-therm')

base_dir = '/sys/bus/w1/devices/'
#assigns variable to each temperature sensor address
device_file1 = glob.glob(base_dir + '28-0000074b7f2d')[0] + '/w1_slave'
device_file2 = glob.glob(base_dir + '28-0000074b8662')[0] + '/w1_slave'
#variable definitions
oldtime = time.time() - 60
oldtime2 = time.time()
oldday = time.localtime()
otime = time.time()
ostate = 0
startup = 1
global  flow
flow = 47
energy = 0
totalen = 0
power = 0
dayen = 0

#flashes led on call
def flash(pin, state, wait):
    boo = not state
    GPIO.output(pin, state)
    time.sleep(wait)
    GPIO.output(pin, boo)

#opens temperature sensor when passed address
def read_temp_raw(d_file):
    f = open(d_file, 'r')
    lines = f.readlines()
    f.close()
    return lines

#reads and returns the temperature
def read_temp(d_file):
    lines = read_temp_raw(d_file)
    while lines[0].strip()[-3:] != 'YES':
        time.sleep(0.2)
        lines = read_temp_raw1()
    equals_pos = lines[1].find('t=')
    if equals_pos != -1:
        temp_string = lines[1][equals_pos+2:]
        temp_c = float(temp_string) / 1000.0
        return temp_c

#updates the vent LED and returns the vent state
def IRup():
    if GPIO.input(24):
        GPIO.output(17, 0)
        state = False
    else:
        GPIO.output(17, 1)
        state = True
    return state

#delays the program until specified time or until the measurement button has been
```

```
    pushed or the vent opens
68  def wait(itime, oldtime):
69      while(((time.time() - oldtime) < itime) and (IRup() == False) and (GPIO.input(23)
        == True)):
70          time.sleep(.1)
71          usave()
72
73  #tries to save data to a usb if there is one inserted
74  def usave():
75      folders = (os.popen("ls -l /media/pi/").readlines())
76      i = 0
77      flow = 0
78      while (i < len(folders)):
79          stringf = folders[i].split(" ")
80          if (len(folders[i]) > 15):
81              if (stringf[2] == "pi"):
82                  stringf = folders[i].split(":")
83                  if (len(stringf) == 1):
84                      stringf = stringf[0].split("196")
85                  stringf = stringf[1].split(" ")
86                  h = len(stringf)
87                  g = 1
88                  folders[i] = stringf[g]
89                  g = g + 1
90                  while (g < h):
91                      folders[i] = folders[i] + " " + stringf[g]
92                      g = g + 1
93                  folders[i] = folders[i].rstrip()
94                  print (folders[i] + " device has been inserted.\n")
95                  GPIO.output(6,1)
96                  usbfolders = os.listdir("/media/pi/" + folders[i])
97                  j = 0
98                  while (j < len(usbfolders)):
99                      if(usbfolders[j] == 'Data'):
100                         k = 0
101                         dirfiles = os.listdir("/media/pi/" + folders[i] + '/' +
                            usbfolders[j])
102                         while (k < len(dirfiles)):
103                             if(dirfiles[k] == '1-AIRSPEED.txt'):
104                                 if (lps('/media/pi/' + folders[i] +
                                    '/Data/1-AIRSPEED.txt')):
105                                     shutil.copyfile('/media/pi/' + folders[i] +
                                        '/Data/1-AIRSPEED.txt',
                                        '/home/pi/Desktop/Programs/Data/1-AIRSPEED.txt')
106                             k = k + 1
107                         shutil.rmtree('/media/pi/' + folders[i] + '/Data')
108                         time.sleep(0.1)
109                     j = j + 1
110                 shutil.copytree('/home/pi/Desktop/Programs/Data','/media/pi/' +
                    folders[i] + '/Data')
111                 time.sleep(0.1)
112                 os.system('sudo udisks --unmount /dev/sda1')
113                 GPIO.output(6, 0)
114                 time.sleep(0.4)
115                 flash(6,1,0.1)
116                 flash(6,0,0.1)
117                 flash(6,1,0.1)
118                 print (folders[i] + " device has been removed.\n")
119         i = i + 1
120     GPIO.output(6, 0)
121
122 #reads the LPS file drive and assigns a new LPS amount
123 def lps(directory):
124     global flow
125     with open(directory, "r") as ff:
126         fstring = ff.read()
127         if(len(fstring) > 3):
```

```
            if (fstring[1] == 'l'):
                start = 1;
            elif (fstring[2] == 'l'):
                start = 2
            elif (fstring[3] == 'l'):
                start = 3
            elif (fstring[4] == 'l'):
                start = 4
            if ((fstring[start] == 'l') and (fstring[start + 1] == 'p') and
            (fstring[start + 2] == 's')):
                print (fstring)
                k = 0
                lstring = ''
                while (k < start):
                    lstring = lstring + fstring[k]
                    k = k + 1
                flow = int(lstring)
                return True
        return False

#START OF MAIN PROGRAM

lps("/home/pi/Desktop/Programs/Data/1-AIRSPEED.txt")
                                    #sets LPS
while 1:                            #loops indefinitley
    GPIO.setmode(GPIO.BCM)          #set up GPIO using BCM numbering
    GPIO.setup(5, GPIO.OUT)
    GPIO.setup(17, GPIO.OUT)
    GPIO.setup(6, GPIO.OUT)
    GPIO.setup(24, GPIO.IN)
    GPIO.setup(23, GPIO.IN, pull_up_down = GPIO.PUD_UP)
    usave()                         #tries to copy data to usb
    if IRup():                      #checks the vent position and waits a diferent
    amount
        otime = time.time()         #of time depending on the last time the vent
        was open
        wait(1, oldtime)            #if the vent is open now
    elif ((time.time() - otime) <= 300):
        wait(10, oldtime)           #if the vent was last open under 300 secs ago
    elif ((time.time() - otime) <= 900):
        wait(30, oldtime)           #if the vent was last open under 900 secs ago
    else:
        wait(60, oldtime)           #else
    oldtime = time.time()           #saves the measurement start time
    GPIO.output(5, 1)               #turns on the measurement light
    temp1 = read_temp(device_file1) #reads the first temperature
    IRup()
    flash(5, 0, 0.1)                #flashes LED and updates vent
    IRup()
    temp2 = read_temp(device_file2) #reads the second temperature
    IRup()
    flash(5, 0, 0.1)                #flashes LED and updates vent
    IRup()                          #creates date string
    date2 = '/home/pi/Desktop/Programs/Data/' + str((time.localtime()).tm_year) + "-" +
    str((time.localtime()).tm_mon) + "-" + str((time.localtime()).tm_mday) + '.csv'
    with open(date2, 'a') as f:     #opens file to save data
        if (f.tell() == 0):         #adds headers if it's a new file
            f.write("Day of month,Date,Vent Temperature,Ambient Temperature,Difference
            in Temperature,Vent Open/Closed,Power(W),Energy(kWh),Total Energy(kWh)\n")
        timedata = str(time.ctime()).split(" ")
        if (timedata[2] == ''):     #splits up date and time to reformat it
            timedata[2] = timedata[3]
            timedata[3] = timedata[4]
            timedata[4] = timedata[5]   #reformats date string
        savetime = "\"" + timedata[1] + " " + timedata[2] + ", " + timedata[4] + " " +
        timedata[3] + "\""
        tempdata = str((time.localtime()).tm_mday) + "," + savetime + ',' +
```

```
            str(round(temp1, 3)) + ',' + str(round(temp2, 3)) + ',' + str(round(abs(temp1 -
            temp2), 3)) + ',' + str(IRup())
189         if IRup():                      #calculates power if the vent is open
190             if(totalen == 0):
191                 oldtime2 = time.time() - 2.7
192             power = 1.125 * flow * (temp1 - temp2)
193             energy = (power * (time.time() - oldtime2)) / 3600 / 1000
194             totalen = totalen + energy
195             tempdata = tempdata + ',' + str(round(power, 3)) + ',' + str(round(totalen,
                3)) + ",0"
196         else:                           #adds zeroes if the vent is closed
197             energy = 0
198             power = 0
199             if (not(totalen == 0)):
200               tempdata = tempdata + ",0,0," + str(round(totalen, 3))
201               dayen = dayen + totalen   #adds the total energy to the end
202               totalen = 0
203             else:
204                tempdata = tempdata + ",0,0,0"
205         print (str(round(time.time() - oldtime2, 2)) + " seconds")
206         oldtime2 = time.time()          #saves the last time power was calculated
207         if (startup):                   #adds a line to show the start of the program
208             tempdata = tempdata + ",Program has started\n"
209         else:                           #or adds the line terminator
210             tempdata = tempdata + '\n'
211         f.write(tempdata)               #saves the data
212     if ((time.localtime()).tm_mday != oldday.tm_mday):
213         with open('/home/pi/Desktop/Programs/Data/2-DAILYDATA.csv', 'a') as f:
214             f.write(str(oldday.tm_year) + '/' + str(oldday.tm_mon) + '/' +
                str(oldday.tm_mday) + ',' + str(round(dayen,6)) + '\n')
215         oldday = time.localtime()       #if the day has changed the program will save
            total kWh
216         dayen = 0                       #reset total
217
218
219     os.system("clear")
220     print(savetime.split("\"")[1] + '\n')
221     print(str(flow) + "lps")
222     print("\nAmbient Temperature")
223     print("Degrees C  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30
        31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50")
224     print("           |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
        |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |")
225     temp2a = (str(format(temp2, '.3f'))) + "  -  |"
226     temp2 = temp2 - 0.5
227     while (temp2 >= 15):
228         temp2a = temp2a + "="
229         temp2 = temp2 - 0.25
230     temp2a = temp2a + "|"
231     print(temp2a)
232     print("\nVent Temperature")
233     print("Degrees C  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30
        31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50")
234     print("           |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
        |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |")
235     temp1a = (str(format(temp1, '.3f'))) + "  -  |"
236     temp1 = temp1 - 0.5
237     while (temp1 >= 15):
238         temp1a = temp1a + "="
239         temp1 = temp1 - 0.25
240     temp1a = temp1a + "|"
241     print(temp1a)
242
243     if (IRup()):
244         print("\nPower")
245         print("Power kW   0.0     0.1     0.2     0.3     0.4     0.5     0.6
            0.7     0.8     0.9     1.0     1.1     1.2     1.3     1.4     1.5     1.6
```

```
            1.7")
246         print("           |       |       |       |       |       |       |
            |       |       |       |       |       |       |       |       |
            |")
247         powera = (str(format(power / 1000, '.4f'))) + "  -  |"
248         power = power - 25
249         while (power >= 0):
250             powera = powera + "="
251             power = power - 12.5
252         powera = powera + "|"
253         print(powera)
254         print("\nEnergy " + (str(format(totalen, '.4f'))) + "kWh\n")
255
256     print("\n\n\n")
257
258
259     IRup()                                  #updates vent
260     GPIO.output(5, 0)                       #flashes LED to show end of measurement
261     flash(5, 1, 0.1)
262     flash(5, 0, 0.1)
263     flash(5, 1, 0.1)
264     time.sleep(0.5)                          #delay to next read
265     IRup()                                   #updates vent
266     startup = 0
267
268
269 #END OF MAIN PROGRAM
270
271
272
273
274
275
276
```