#Libraries

import RPi.GPIO as GPIO #library for Raspberry Pi GPIOs

import time #library to use sleep function

import board

import digitalio

import adafruit\_character\_lcd.character\_lcd as characterlcd

#GPIO Mode (BOARD / BCM)

GPIO.setmode(GPIO.BCM)

# Modify this if you have a different sized character LCD

lcd\_columns = 16

lcd\_rows = 2

# Raspberry Pi Pin Config:

lcd\_rs = digitalio.DigitalInOut(board.D5)

lcd\_en = digitalio.DigitalInOut(board.D6)

lcd\_d4 = digitalio.DigitalInOut(board.D12)

lcd\_d5 = digitalio.DigitalInOut(board.D13)

lcd\_d6 = digitalio.DigitalInOut(board.D16)

lcd\_d7 = digitalio.DigitalInOut(board.D17)

# Initialise the lcd class

lcd = characterlcd.Character\_LCD\_Mono(

lcd\_rs, lcd\_en, lcd\_d4, lcd\_d5, lcd\_d6, lcd\_d7, lcd\_columns, lcd\_rows)

#set GPIO Pins

TRIGGER = 19 # board pin as trigger

ECHO = 20 # board pin as echo

#set GPIO direction (IN / OUT)

GPIO.setup(TRIGGER, GPIO.OUT)

GPIO.setup(ECHO, GPIO.IN)

lcd.clear()

#function distance will use 2 GPIOs to trigger and echo to calculate distance using the distance formula

def distance():

# set Trigger to HIGH

GPIO.output(TRIGGER, True)

# set Trigger after 0.01ms to LOW

time.sleep(0.00001)

GPIO.output(TRIGGER, False)

StartTime = time.time()

StopTime = time.time()

# save StartTime

while GPIO.input(ECHO) == 0:

StartTime = time.time()

# save time of arrival

while GPIO.input(ECHO) == 1:

StopTime = time.time()

# time difference between start and arrival

TimeElapsed = StopTime - StartTime

# multiply with the sonic speed (34300 cm/s)

# and divide by 2, because there and back

distance = (TimeElapsed \* 34300) / 2

return distance

#simple if statement

if \_\_name\_\_ == '\_\_main\_\_':

#simple try exception programming

try:

while True:

dist = distance()#we accept the value in a variable dist

print ("Measured Distance = %.1f cm" % dist)#display dist

lcd.clear()

lcd.message = ("Dist.:%.1f cm" % dist)

time.sleep(2)

# Reset by pressing CTRL + C

except KeyboardInterrupt:

print("Measurement stopped by User")

GPIO.cleanup()#finally GPIO cleanup to flush all the buffers of the GPIOs used in this code