```
# -*- coding: utf-8 -*-
"""
File Name: process_data.py
Purpose: data analysis exercise
Author: Samuel Wong
"""
import numpy as np

scale=np.array([2,2,1,1,2,1,2,2,2,1,1,1]) #exact
#all units below are in cm
#sigma are the uncertainty (1 std deviation)
#y measurements and measurement uncertainty
y_meas=np.array([2.65,3.55,2.00,3.50,3.60,3.30,2.60,3.35,3.70,1.40,0.08,0.25])
sigma_y_meas=np.array([0.05,0.05,0.05,0.05,0.05,0.05,0.05,0.05,0.05,0.05,0.02,0.05])
#x measurements and measurement uncertainty
x_meas=np.array([0.60,0.90,2.70,2.55,1.00,1.25,1.65,1.10,0.80,1.00,0.90,2.90])
sigma_x_meas=np.array([0.05,0.05,0.05,0.05,0.05,0.05,0.05,0.05,0.05,0.05,0.02,0.02])
#calibration and its uncertainty
calibration_x=2.00 #25 unit
sigma_cal_x = 0.02
calibration_y=1.90 #25 unit
sigma_cal_y=0.02

#convert to (x,y) unit using the formula:
# unit = measured*scale*(calibration units/calibration)
x = x_meas*scale*(25/calibration_x)
y = y_meas*scale*(25/calibration_y)
#calculate total uncertainty using quadrature
dx = scale*25*np.sqrt( (1/calibration_x**2)*(sigma_x_meas**2) +
                       (x_meas**2/calibration_x**4)*sigma_cal_x**2)
dy = scale*25*np.sqrt( (1/calibration_y**2)*(sigma_y_meas**2) +
                       (y_meas**2/calibration_y**4)*sigma_cal_y**2)

for i in range(x.size):
    print("{} \t {} \t {} \t {}".format(round(x[i],2),round(dx[i],1),
          round(y[i],2),round(dy[i],1)))
```