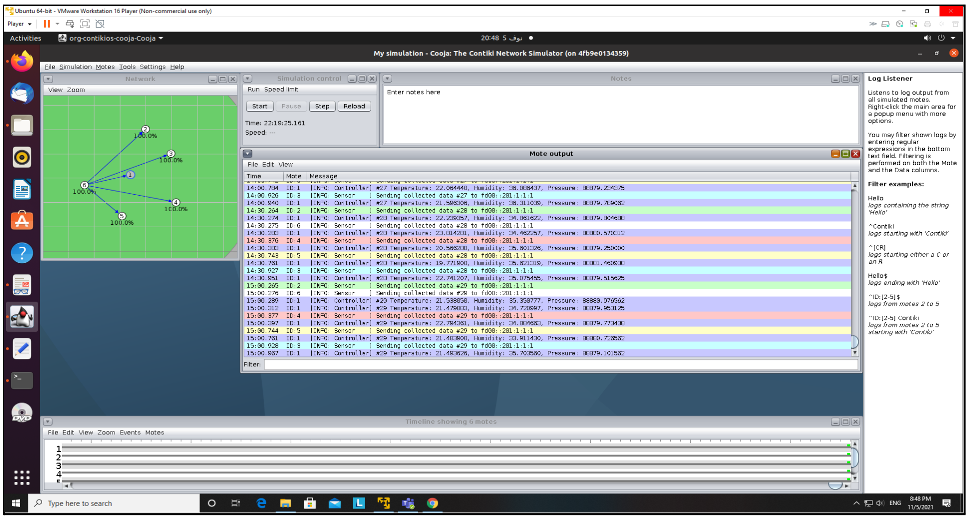
Assignment Three -- Contiki-NG Wireless Sensor Network

Group 3

AHMED SHEHATA MAHMOUD ABOMOUSTAFA
SARAH HOSSAM ABDELHAMEED ELMOWAFY
MOHAMED SAYED ABDELWAHAB HUSSIEN
ABDELMEGEED AHMED ABDELMEGEED
[UNIVERSITY OF OTTAWA]

1. How you implement this network. Give a screenshot of your network structure and output in the Cooja simulator.

- Scenario1


Ubuntu 64-bit - VMware Workstation 16 Player (Non-commercial use only)
Activities
org-contikios-cooja-Cooja
My simulation - Cooja: The Contiki Network Simulator (on 4fb9e0134359)
File Simulation Motes Tools Settings Help
Network
View Zoom
100.0%
100.0%
100.0%
100.0%
100.0%
Simulation control
Run Speed limit
Start
Pause
Step
Reload
Notes
Enter notes here
Mote output
File Edit View
Timeline showing 6 motes
File Edit View Zoom Events Motes
Type here to search

Time: 22:19:25.161
Speed: ---

```
Time       Mote  Message
14:00.784  ID:1  [INFO: Controller] #27 Temperature: 22.064440, Humidity: 36.086437, Pressure: 88879.234375
14:00.926  ID:3  [INFO: Sensor    ] Sending collected data #27 to fd00::201:1:1:1
14:00.940  ID:1  [INFO: Controller] #27 Temperature: 21.596306, Humidity: 36.311039, Pressure: 88879.789062
14:30.264  ID:2  [INFO: Sensor    ] Sending collected data #28 to fd00::201:1:1:1
14:30.274  ID:1  [INFO: Controller] #28 Temperature: 22.239357, Humidity: 34.861622, Pressure: 88879.804688
14:30.275  ID:6  [INFO: Sensor    ] Sending collected data #28 to fd00::201:1:1:1
14:30.283  ID:1  [INFO: Controller] #28 Temperature: 23.814281, Humidity: 34.462257, Pressure: 88880.570312
14:30.376  ID:4  [INFO: Sensor    ] Sending collected data #28 to fd00::201:1:1:1
14:30.383  ID:1  [INFO: Controller] #28 Temperature: 20.566288, Humidity: 35.601326, Pressure: 88879.250000
14:30.743  ID:5  [INFO: Sensor    ] Sending collected data #28 to fd00::201:1:1:1
14:30.761  ID:1  [INFO: Controller] #28 Temperature: 19.771900, Humidity: 35.621319, Pressure: 88881.460938
14:30.927  ID:3  [INFO: Sensor    ] Sending collected data #28 to fd00::201:1:1:1
14:30.951  ID:1  [INFO: Controller] #28 Temperature: 22.741207, Humidity: 35.075455, Pressure: 88879.515625
15:00.265  ID:2  [INFO: Sensor    ] Sending collected data #29 to fd00::201:1:1:1
15:00.276  ID:6  [INFO: Sensor    ] Sending collected data #29 to fd00::201:1:1:1
15:00.289  ID:1  [INFO: Controller] #29 Temperature: 21.538050, Humidity: 35.350777, Pressure: 88880.976562
15:00.312  ID:1  [INFO: Controller] #29 Temperature: 21.479883, Humidity: 34.720997, Pressure: 88879.953125
15:00.377  ID:4  [INFO: Sensor    ] Sending collected data #29 to fd00::201:1:1:1
15:00.397  ID:1  [INFO: Controller] #29 Temperature: 22.794361, Humidity: 34.884663, Pressure: 88879.773438
15:00.744  ID:5  [INFO: Sensor    ] Sending collected data #29 to fd00::201:1:1:1
15:00.761  ID:1  [INFO: Controller] #29 Temperature: 21.483900, Humidity: 33.911430, Pressure: 88880.726562
15:00.928  ID:3  [INFO: Sensor    ] Sending collected data #29 to fd00::201:1:1:1
15:00.967  ID:1  [INFO: Controller] #29 Temperature: 21.493626, Humidity: 35.703560, Pressure: 88879.101562
```

Filter:

Log Listener

Listens to log output from all simulated motes. Right-click the main area for a popup menu with more options.

You may filter shown logs by entering regular expressions in the bottom text field. Filtering is performed on both the Mote and the Data columns.

Filter examples:

Hello
logs containing the string 'Hello'

^Contiki
logs starting with 'Contiki'

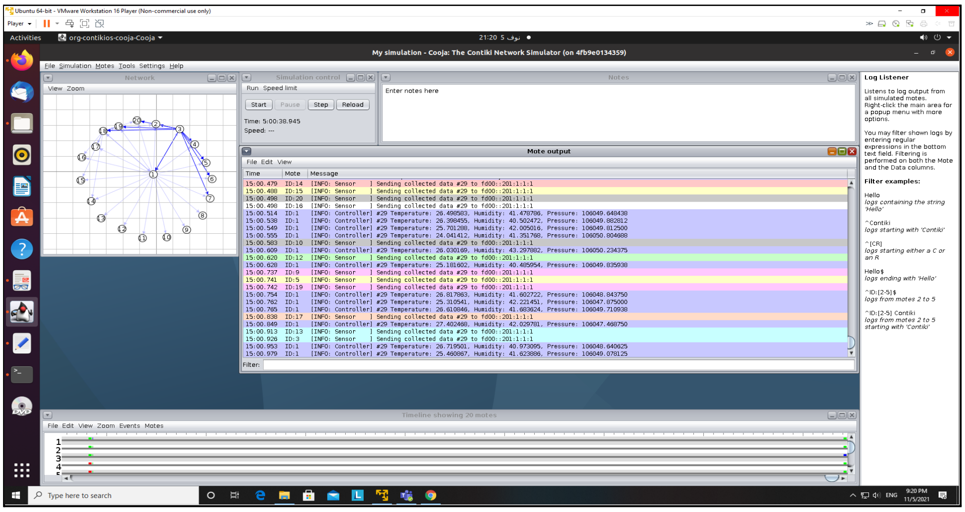
^[CR]
logs starting either a C or an R

Hello$
logs ending with 'Hello'

^ID:[2-5]$
logs from motes 2 to 5

^ID:[2-5] Contiki
logs from motes 2 to 5 starting with 'Contiki'

- Scenario2


Ubuntu 64-bit - VMware Workstation 16 Player (Non-commercial use only)
Activities
org-contikios-cooja-Cooja
My simulation - Cooja: The Contiki Network Simulator (on 4fb9e0134359)
File Simulation Motes Tools Settings Help
Network
View Zoom
Simulation control
Run Speed limit
Start
Pause
Step
Reload
Notes
Enter notes here
Mote output
File Edit View
Timeline showing 20 motes
File Edit View Zoom Events Motes
Type here to search

Time: 5:00:38.945
Speed: ---

```
Time       Mote   Message
15:00.479  ID:14  [INFO: Sensor    ] Sending collected data #29 to fd00::201:1:1:1
15:00.488  ID:15  [INFO: Sensor    ] Sending collected data #29 to fd00::201:1:1:1
15:00.498  ID:20  [INFO: Sensor    ] Sending collected data #29 to fd00::201:1:1:1
15:00.498  ID:16  [INFO: Sensor    ] Sending collected data #29 to fd00::201:1:1:1
15:00.514  ID:1   [INFO: Controller] #29 Temperature: 26.498583, Humidity: 41.478786, Pressure: 106049.648438
15:00.538  ID:1   [INFO: Controller] #29 Temperature: 26.398455, Humidity: 40.502472, Pressure: 106049.882812
15:00.549  ID:1   [INFO: Controller] #29 Temperature: 25.701288, Humidity: 42.005016, Pressure: 106049.812500
15:00.555  ID:1   [INFO: Controller] #29 Temperature: 24.041412, Humidity: 41.351768, Pressure: 106050.804688
15:00.583  ID:10  [INFO: Sensor    ] Sending collected data #29 to fd00::201:1:1:1
15:00.609  ID:1   [INFO: Controller] #29 Temperature: 26.030169, Humidity: 43.297882, Pressure: 106050.234375
15:00.620  ID:12  [INFO: Sensor    ] Sending collected data #29 to fd00::201:1:1:1
15:00.628  ID:1   [INFO: Controller] #29 Temperature: 25.181602, Humidity: 40.485954, Pressure: 106049.835938
15:00.737  ID:9   [INFO: Sensor    ] Sending collected data #29 to fd00::201:1:1:1
15:00.741  ID:5   [INFO: Sensor    ] Sending collected data #29 to fd00::201:1:1:1
15:00.742  ID:19  [INFO: Sensor    ] Sending collected data #29 to fd00::201:1:1:1
15:00.754  ID:1   [INFO: Controller] #29 Temperature: 26.817863, Humidity: 41.602722, Pressure: 106048.843750
15:00.762  ID:1   [INFO: Controller] #29 Temperature: 25.310541, Humidity: 42.221451, Pressure: 106047.875000
15:00.765  ID:1   [INFO: Controller] #29 Temperature: 26.610846, Humidity: 41.683624, Pressure: 106049.710938
15:00.838  ID:17  [INFO: Sensor    ] Sending collected data #29 to fd00::201:1:1:1
15:00.849  ID:1   [INFO: Controller] #29 Temperature: 27.402468, Humidity: 42.029781, Pressure: 106047.468750
15:00.913  ID:13  [INFO: Sensor    ] Sending collected data #29 to fd00::201:1:1:1
15:00.926  ID:3   [INFO: Sensor    ] Sending collected data #29 to fd00::201:1:1:1
15:00.953  ID:1   [INFO: Controller] #29 Temperature: 26.719501, Humidity: 40.973095, Pressure: 106048.640625
15:00.979  ID:1   [INFO: Controller] #29 Temperature: 25.460867, Humidity: 41.623886, Pressure: 106049.078125
```

Filter:

Log Listener

Listens to log output from all simulated motes. Right-click the main area for a popup menu with more options.

You may filter shown logs by entering regular expressions in the bottom text field. Filtering is performed on both the Mote and the Data columns.

Filter examples:

Hello
logs containing the string 'Hello'

^Contiki
logs starting with 'Contiki'

^[CR]
logs starting either a C or an R

Hello$
logs ending with 'Hello'

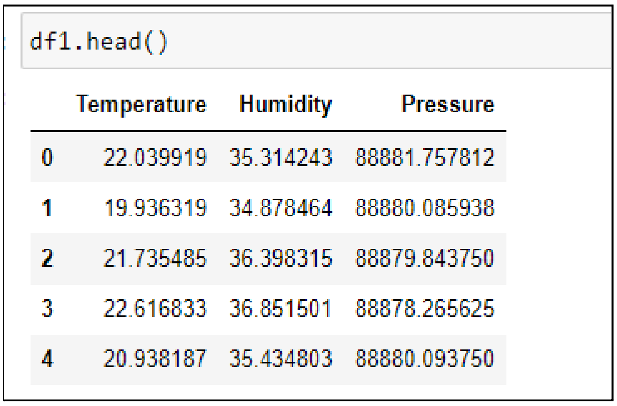
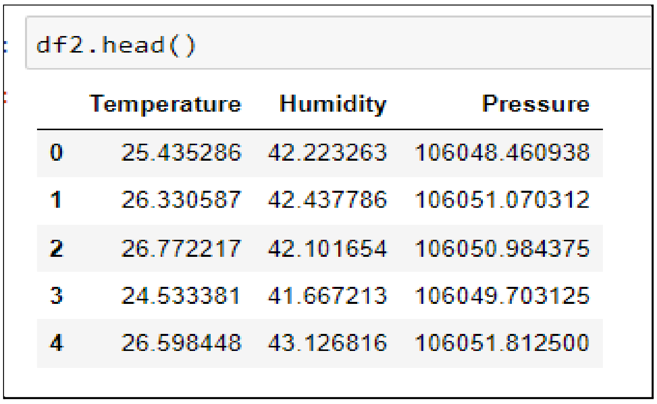
^ID:[2-5]$
logs from motes 2 to 5

^ID:[2-5] Contiki
logs from motes 2 to 5 starting with 'Contiki'

- In scenario 1 we create 1 controller with 5 sensors and scenario 2 with 20 sensors and the controller reads the data gram from sensors at the same time in 30 rounds (we add addtive_noise to each sensor)

2. **After the controller receives data from sensors, you can save the output messages in the "Mote output" window to a .txt file. Write a program (Choose the programming language you like, Python, Java, etc.) that can extract the data controller received, calculate the average value for each round, and add the controller channel additive noise to calculate the estimated value and estimate errors.**

❖ We save the output in a file and use python language to extract the readings:

Scenario1	Scenario2
	

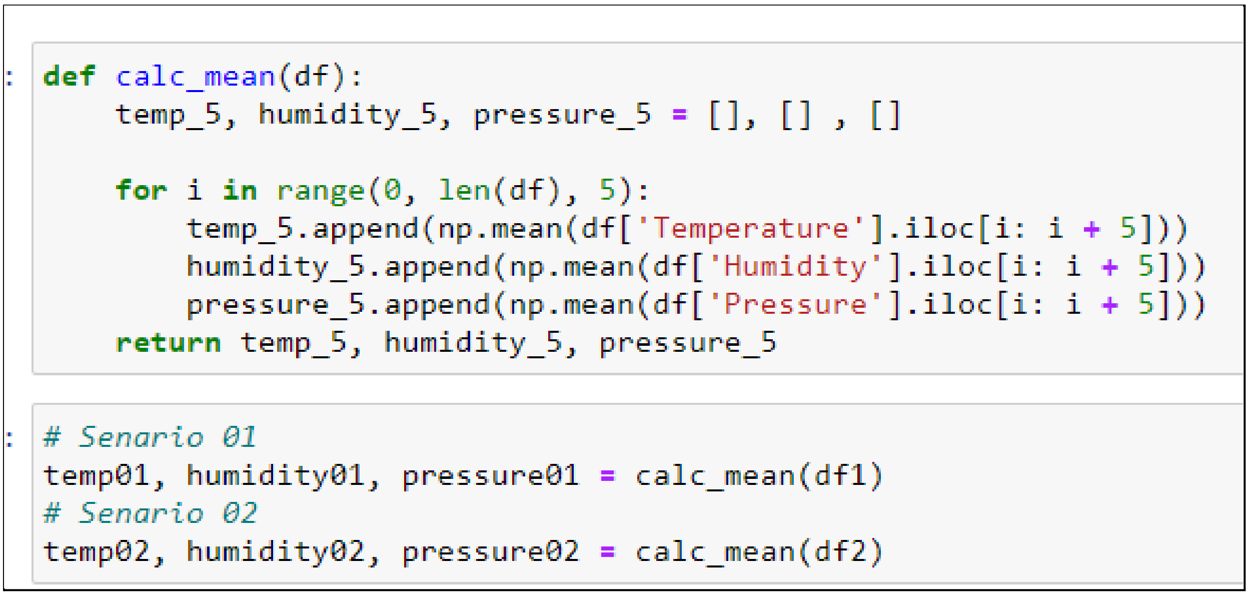
```
df1.head()
```

	Temperature	Humidity	Pressure
0	22.039919	35.314243	88881.757812
1	19.936319	34.878464	88880.085938
2	21.735485	36.398315	88879.843750
3	22.616833	36.851501	88878.265625
4	20.938187	35.434803	88880.093750

```
df2.head()
```

	Temperature	Humidity	Pressure
0	25.435286	42.223263	106048.460938
1	26.330587	42.437786	106051.070312
2	26.772217	42.101654	106050.984375
3	24.533381	41.667213	106049.703125
4	26.598448	43.126816	106051.812500

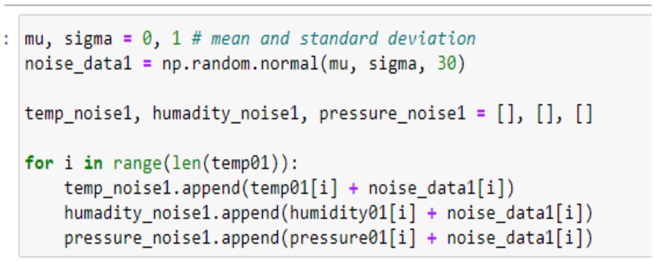
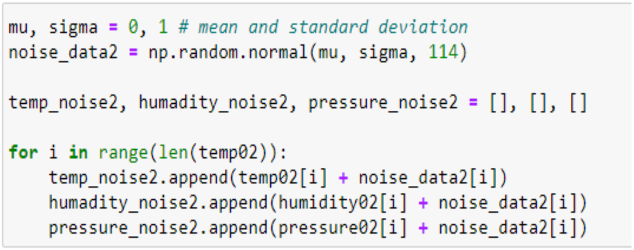
❖ Then we calculate the mean value for each round:



```
def calc_mean(df):
    temp_5, humidity_5, pressure_5 = [], [] , []

    for i in range(0, len(df), 5):
        temp_5.append(np.mean(df['Temperature'].iloc[i: i + 5]))
        humidity_5.append(np.mean(df['Humidity'].iloc[i: i + 5]))
        pressure_5.append(np.mean(df['Pressure'].iloc[i: i + 5]))
    return temp_5, humidity_5, pressure_5
```

```
# Senario 01
temp01, humidity01, pressure01 = calc_mean(df1)
# Senario 02
temp02, humidity02, pressure02 = calc_mean(df2)
```

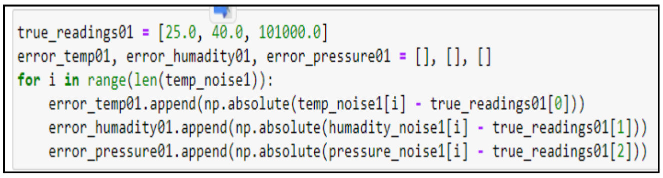
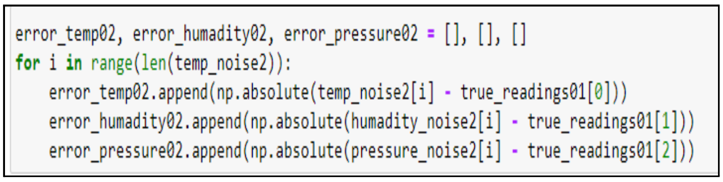
- ❖ Add noise to data according to a normal Gaussian distribution

Scenario1	Scenario2
	

```
mu, sigma = 0, 1 # mean and standard deviation
noise_data1 = np.random.normal(mu, sigma, 30)

temp_noise1, humadity_noise1, pressure_noise1 = [], [], []

for i in range(len(temp01)):
    temp_noise1.append(temp01[i] + noise_data1[i])
    humadity_noise1.append(humidity01[i] + noise_data1[i])
    pressure_noise1.append(pressure01[i] + noise_data1[i])
```

```
mu, sigma = 0, 1 # mean and standard deviation
noise_data2 = np.random.normal(mu, sigma, 114)

temp_noise2, humadity_noise2, pressure_noise2 = [], [], []

for i in range(len(temp02)):
    temp_noise2.append(temp02[i] + noise_data2[i])
    humadity_noise2.append(humidity02[i] + noise_data2[i])
    pressure_noise2.append(pressure02[i] + noise_data2[i])
```

- ❖ Estimated Error scenario

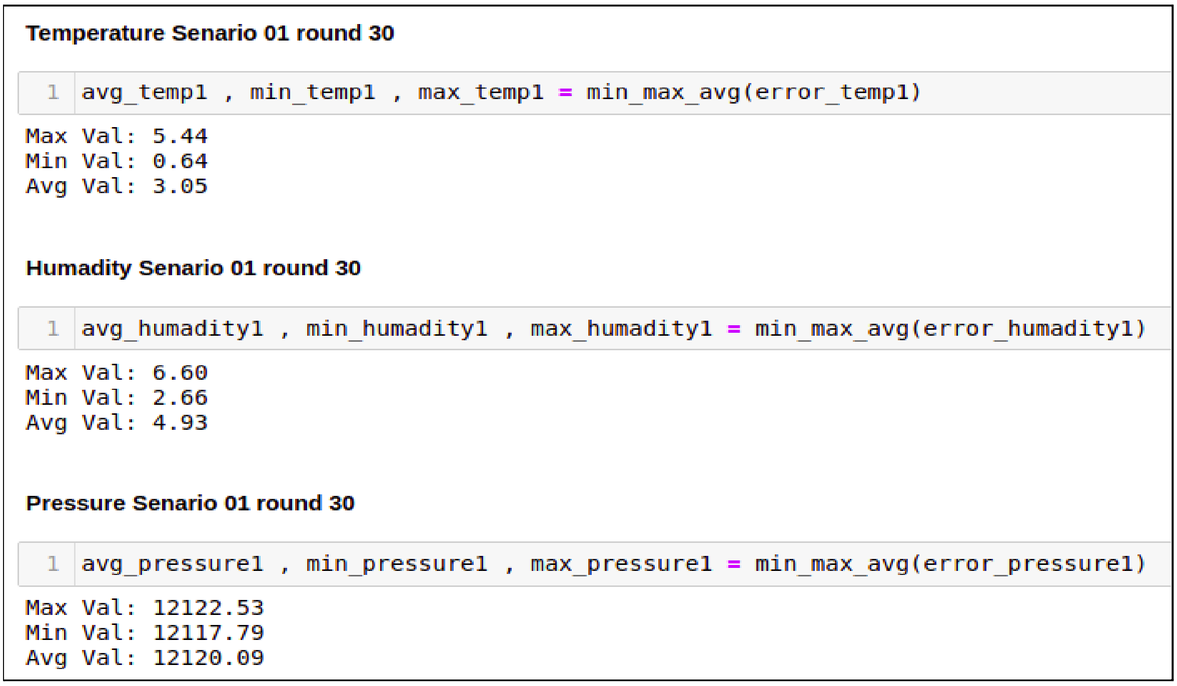
Scenario1	Scenario2
	

```
true_readings01 = [25.0, 40.0, 101000.0]
error_temp01, error_humadity01, error_pressure01 = [], [], []
for i in range(len(temp_noise1)):
    error_temp01.append(np.absolute(temp_noise1[i] - true_readings01[0]))
    error_humadity01.append(np.absolute(humadity_noise1[i] - true_readings01[1]))
    error_pressure01.append(np.absolute(pressure_noise1[i] - true_readings01[2]))
```

```
error_temp02, error_humadity02, error_pressure02 = [], [], []
for i in range(len(temp_noise2)):
    error_temp02.append(np.absolute(temp_noise2[i] - true_readings01[0]))
    error_humadity02.append(np.absolute(humadity_noise2[i] - true_readings01[1]))
    error_pressure02.append(np.absolute(pressure_noise2[i] - true_readings01[2]))
```

3. Record the smallest, largest, and average estimate errors for Scenario-1 and Scenario-2.

- ❖ **Scenario 1 :**



Temperature Senario 01 round 30

```
avg_temp1 , min_temp1 , max_temp1 = min_max_avg(error_temp1)
```

```
Max Val: 5.44
Min Val: 0.64
Avg Val: 3.05
```

Humadity Senario 01 round 30

```
avg_humadity1 , min_humadity1 , max_humadity1 = min_max_avg(error_humadity1)
```

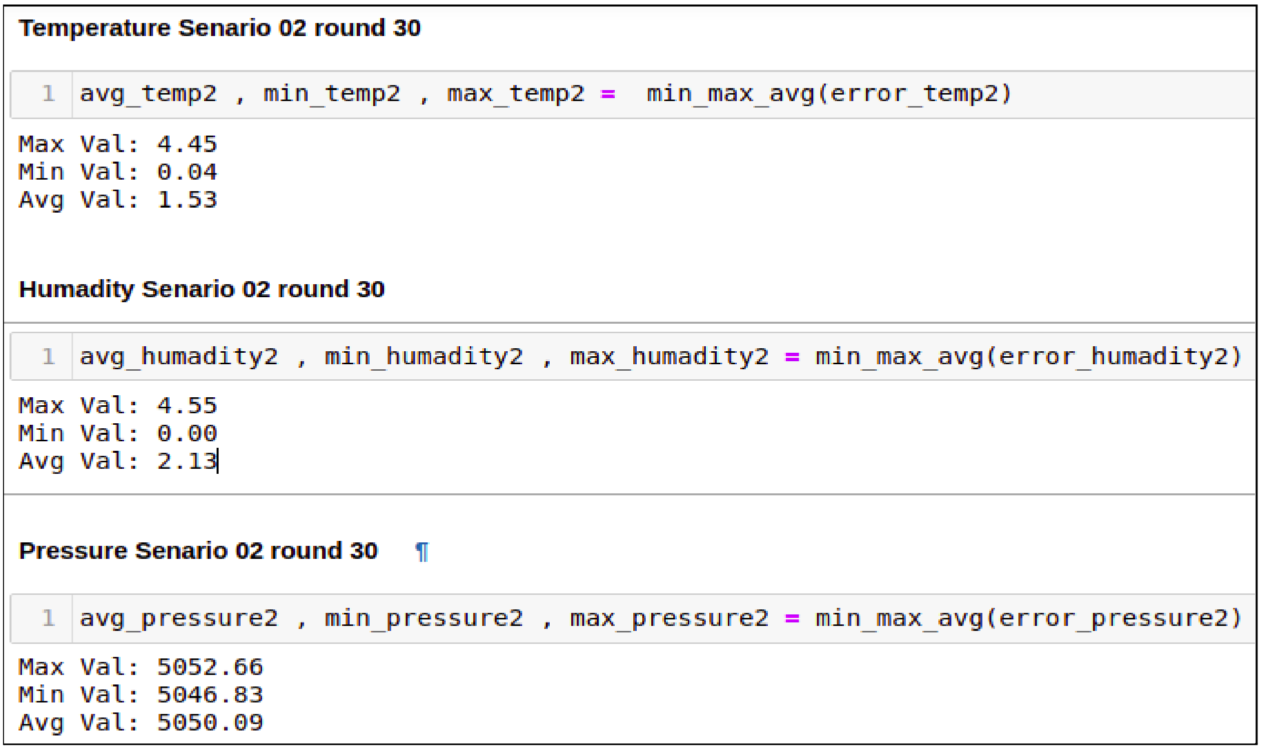
```
Max Val: 6.60
Min Val: 2.66
Avg Val: 4.93
```

Pressure Senario 01 round 30

```
avg_pressure1 , min_pressure1 , max_pressure1 = min_max_avg(error_pressure1)
```

```
Max Val: 12122.53
Min Val: 12117.79
Avg Val: 12120.09
```

❖ Scenario 2 :



Temperature Senario 02 round 30

```
avg_temp2 , min_temp2 , max_temp2 =  min_max_avg(error_temp2)
```

```
Max Val: 4.45
Min Val: 0.04
Avg Val: 1.53
```

Humadity Senario 02 round 30

```
avg_humadity2 , min_humadity2 , max_humadity2 = min_max_avg(error_humadity2)
```

```
Max Val: 4.55
Min Val: 0.00
Avg Val: 2.13
```

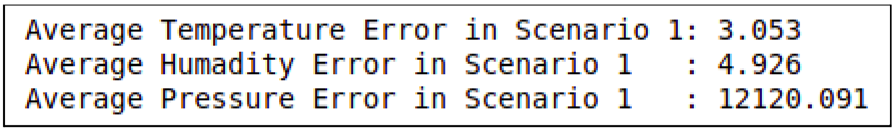
Pressure Senario 02 round 30 ¶

```
avg_pressure2 , min_pressure2 , max_pressure2 = min_max_avg(error_pressure2)
```

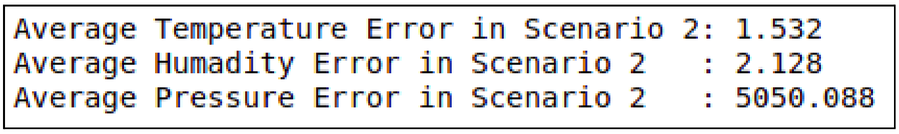
```
Max Val: 5052.66
Min Val: 5046.83
Avg Val: 5050.09
```

4. What are your conclusions upon comparison of the results of Scenario-1 and Scenario-2?

As we see from average estimated error from readings of scenario 1 and scenario 2, the mean error of scenario 2 is lower than the error of scenario 1, and this because, the number of sensors in scenario 2 higher than number of sensors in scenario 1 and this lead to more readings and this make the controller more accurate as the number of reading increases



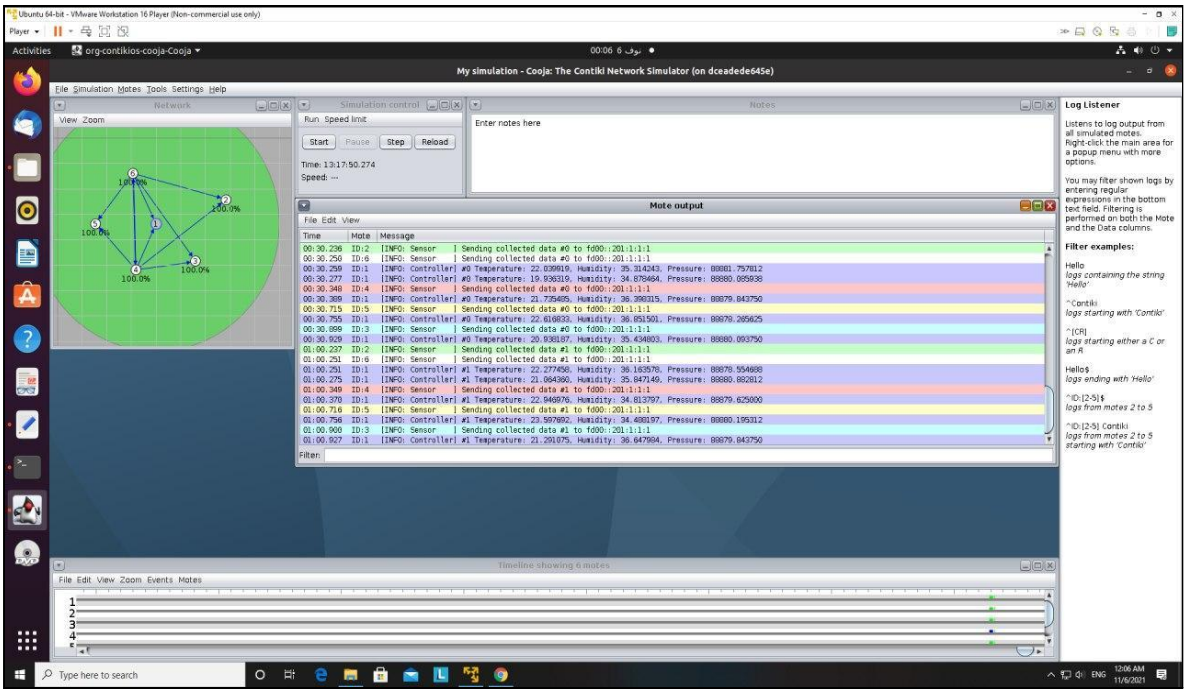
```
Average Temperature Error in Scenario 1: 3.053
Average Humadity Error in Scenario 1   : 4.926
Average Pressure Error in Scenario 1   : 12120.091
```



```
Average Temperature Error in Scenario 2: 1.532
Average Humadity Error in Scenario 2   : 2.128
Average Pressure Error in Scenario 2   : 5050.088
```

5. Will your answers change if you run each program only twice instead of 30 times? Explain why?

❖ Scenario 1


Ubuntu 64-bit - VMware Workstation 16 Player (Non-commercial use only)
Activities
org-contikios-cooja-Cooja
My simulation - Cooja: The Contiki Network Simulator (on dceadede645e)
File Simulation Motes Tools Settings Help
Network
View Zoom
Simulation control
Run Speed limit
Start
Pause
Step
Reload
Notes
Enter notes here
Mote output
File Edit View
Timeline showing 6 motes
File Edit View Zoom Events Motes
Type here to search

Time: 13:17:50.274
Speed: ---

```
Time        Mote   Message
00:30.236   ID:2   [INFO: Sensor    ] Sending collected data #0 to fd00::201:1:1:1
00:30.250   ID:6   [INFO: Sensor    ] Sending collected data #0 to fd00::201:1:1:1
00:30.259   ID:1   [INFO: Controller] #0 Temperature: 22.039919, Humidity: 35.314243, Pressure: 88881.757812
00:30.277   ID:1   [INFO: Controller] #0 Temperature: 19.936319, Humidity: 34.878464, Pressure: 88880.085938
00:30.348   ID:4   [INFO: Sensor    ] Sending collected data #0 to fd00::201:1:1:1
00:30.389   ID:1   [INFO: Controller] #0 Temperature: 21.735485, Humidity: 36.398315, Pressure: 88879.843750
00:30.715   ID:5   [INFO: Sensor    ] Sending collected data #0 to fd00::201:1:1:1
00:30.755   ID:1   [INFO: Controller] #0 Temperature: 22.616833, Humidity: 36.851501, Pressure: 88878.265625
00:30.899   ID:3   [INFO: Sensor    ] Sending collected data #0 to fd00::201:1:1:1
00:30.929   ID:1   [INFO: Controller] #0 Temperature: 20.938187, Humidity: 35.434803, Pressure: 88880.093750
01:00.237   ID:2   [INFO: Sensor    ] Sending collected data #1 to fd00::201:1:1:1
01:00.251   ID:6   [INFO: Sensor    ] Sending collected data #1 to fd00::201:1:1:1
01:00.251   ID:1   [INFO: Controller] #1 Temperature: 22.277458, Humidity: 36.163578, Pressure: 88878.554688
01:00.275   ID:1   [INFO: Controller] #1 Temperature: 21.064360, Humidity: 35.847149, Pressure: 88880.882812
01:00.349   ID:4   [INFO: Sensor    ] Sending collected data #1 to fd00::201:1:1:1
01:00.370   ID:1   [INFO: Controller] #1 Temperature: 22.946976, Humidity: 34.813797, Pressure: 88879.625000
01:00.716   ID:5   [INFO: Sensor    ] Sending collected data #1 to fd00::201:1:1:1
01:00.756   ID:1   [INFO: Controller] #1 Temperature: 23.597692, Humidity: 34.488197, Pressure: 88880.195312
01:00.900   ID:3   [INFO: Sensor    ] Sending collected data #1 to fd00::201:1:1:1
01:00.927   ID:1   [INFO: Controller] #1 Temperature: 21.291075, Humidity: 36.647984, Pressure: 88879.843750
```

Filter:

Log Listener

Listens to log output from all simulated motes. Right-click the main area for a popup menu with more options.

You may filter shown logs by entering regular expressions in the bottom text field. Filtering is performed on both the Mote and the Data columns.

Filter examples:

Hello
logs containing the string 'Hello'

^Contiki
logs starting with 'Contiki'

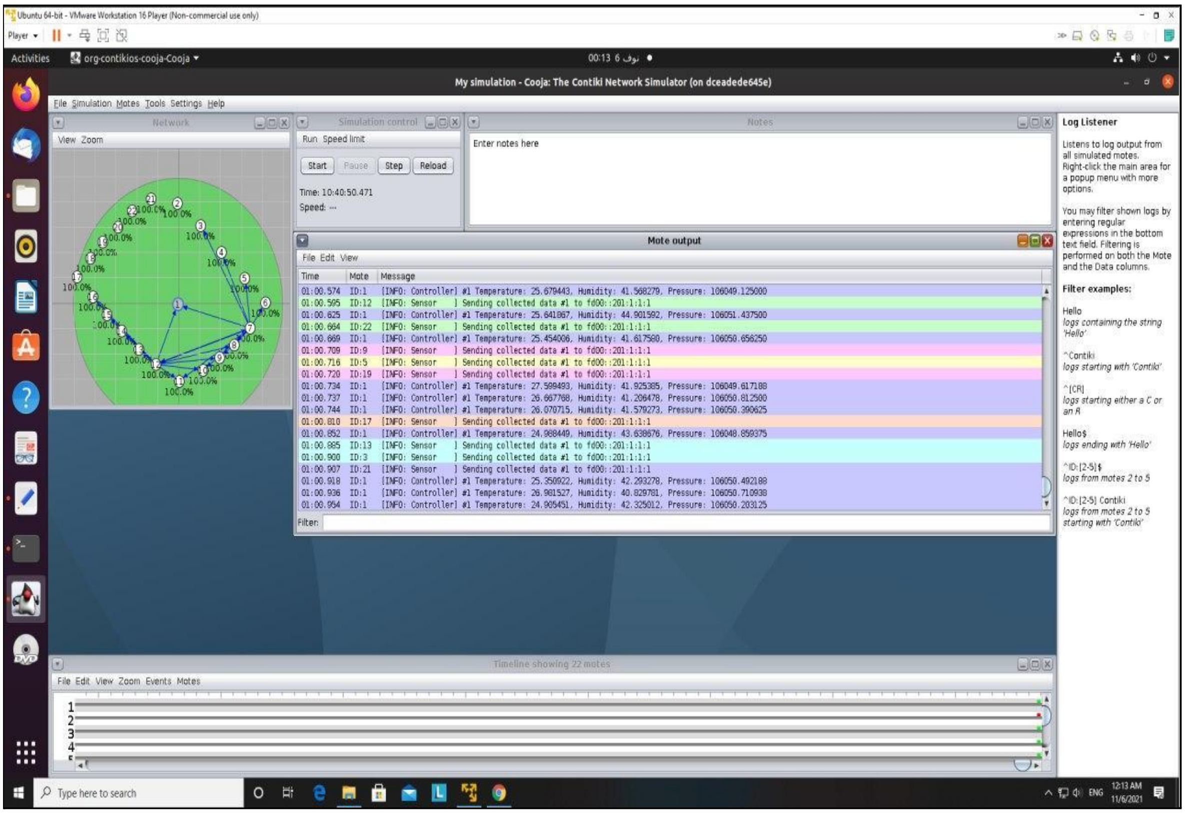
^[CR]
logs starting either a C or an R

Hello$
logs ending with 'Hello'

^ID:[2-5]$
logs from motes 2 to 5

^ID:[2-5] Contiki
logs from motes 2 to 5 starting with 'Contiki'

❖ Scenario 2


Ubuntu 64-bit - VMware Workstation 16 Player (Non-commercial use only)
Activities
org-contikios-cooja-Cooja
My simulation - Cooja: The Contiki Network Simulator (on dceadede645e)
File Simulation Motes Tools Settings Help
Network
View Zoom
Simulation control
Run Speed limit
Start
Pause
Step
Reload
Notes
Enter notes here
Mote output
File Edit View
Timeline showing 22 motes
File Edit View Zoom Events Motes
Type here to search

Time: 10:40:50.471
Speed: ---

```
Time        Mote    Message
01:00.574   ID:1    [INFO: Controller] #1 Temperature: 25.679443, Humidity: 41.568279, Pressure: 106049.125000
01:00.595   ID:12   [INFO: Sensor    ] Sending collected data #1 to fd00::201:1:1:1
01:00.625   ID:1    [INFO: Controller] #1 Temperature: 25.641867, Humidity: 44.901592, Pressure: 106051.437500
01:00.664   ID:22   [INFO: Sensor    ] Sending collected data #1 to fd00::201:1:1:1
01:00.669   ID:1    [INFO: Controller] #1 Temperature: 25.454006, Humidity: 41.617580, Pressure: 106050.656250
01:00.709   ID:9    [INFO: Sensor    ] Sending collected data #1 to fd00::201:1:1:1
01:00.716   ID:5    [INFO: Sensor    ] Sending collected data #1 to fd00::201:1:1:1
01:00.720   ID:19   [INFO: Sensor    ] Sending collected data #1 to fd00::201:1:1:1
01:00.734   ID:1    [INFO: Controller] #1 Temperature: 27.599493, Humidity: 41.925385, Pressure: 106049.617188
01:00.737   ID:1    [INFO: Controller] #1 Temperature: 26.667768, Humidity: 41.206478, Pressure: 106050.812500
01:00.744   ID:1    [INFO: Controller] #1 Temperature: 26.070715, Humidity: 41.579273, Pressure: 106050.390625
01:00.810   ID:17   [INFO: Sensor    ] Sending collected data #1 to fd00::201:1:1:1
01:00.852   ID:1    [INFO: Controller] #1 Temperature: 24.988449, Humidity: 43.638676, Pressure: 106048.859375
01:00.885   ID:13   [INFO: Sensor    ] Sending collected data #1 to fd00::201:1:1:1
01:00.900   ID:3    [INFO: Sensor    ] Sending collected data #1 to fd00::201:1:1:1
01:00.907   ID:21   [INFO: Sensor    ] Sending collected data #1 to fd00::201:1:1:1
01:00.918   ID:1    [INFO: Controller] #1 Temperature: 25.350922, Humidity: 42.293278, Pressure: 106050.492188
01:00.936   ID:1    [INFO: Controller] #1 Temperature: 26.981527, Humidity: 40.829781, Pressure: 106050.710938
01:00.954   ID:1    [INFO: Controller] #1 Temperature: 24.905451, Humidity: 42.325012, Pressure: 106050.203125
```

Filter:

Log Listener

Listens to log output from all simulated motes. Right-click the main area for a popup menu with more options.

You may filter shown logs by entering regular expressions in the bottom text field. Filtering is performed on both the Mote and the Data columns.

Filter examples:

Hello
logs containing the string 'Hello'

^Contiki
logs starting with 'Contiki'

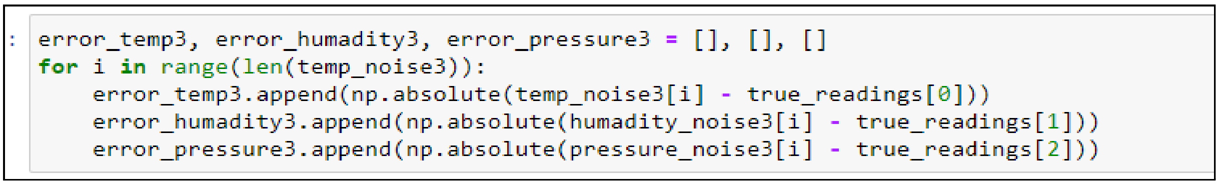
^[CR]
logs starting either a C or an R

Hello$
logs ending with 'Hello'

^ID:[2-5]$
logs from motes 2 to 5

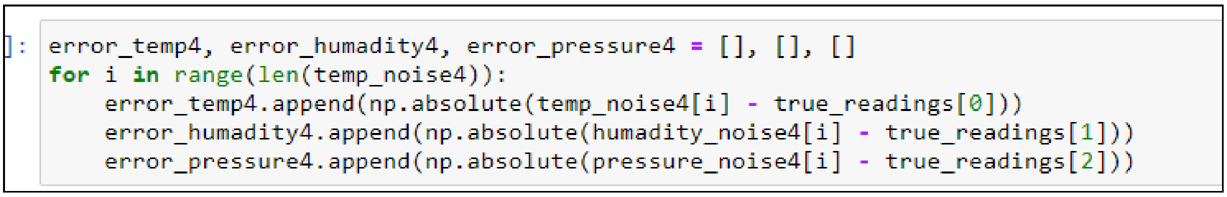
^ID:[2-5] Contiki
logs from motes 2 to 5 starting with 'Contiki'

❖ Estimated Error scenario 1 round 2



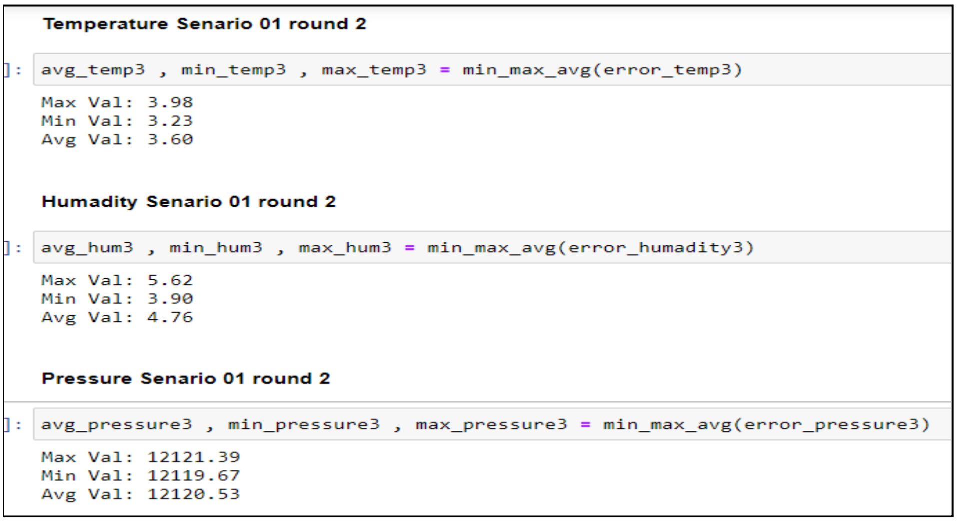
```
error_temp3, error_humadity3, error_pressure3 = [], [], []
for i in range(len(temp_noise3)):
    error_temp3.append(np.absolute(temp_noise3[i] - true_readings[0]))
    error_humadity3.append(np.absolute(humadity_noise3[i] - true_readings[1]))
    error_pressure3.append(np.absolute(pressure_noise3[i] - true_readings[2]))
```

❖ Estimated Error scenario 2 round 2



```
error_temp4, error_humadity4, error_pressure4 = [], [], []
for i in range(len(temp_noise4)):
    error_temp4.append(np.absolute(temp_noise4[i] - true_readings[0]))
    error_humadity4.append(np.absolute(humadity_noise4[i] - true_readings[1]))
    error_pressure4.append(np.absolute(pressure_noise4[i] - true_readings[2]))
```

❖ Min, avg, max in Scenario 1 :



Temperature Senario 01 round 2

```
avg_temp3 , min_temp3 , max_temp3 = min_max_avg(error_temp3)
```

```
Max Val: 3.98
Min Val: 3.23
Avg Val: 3.60
```

Humadity Senario 01 round 2

```
avg_hum3 , min_hum3 , max_hum3 = min_max_avg(error_humadity3)
```

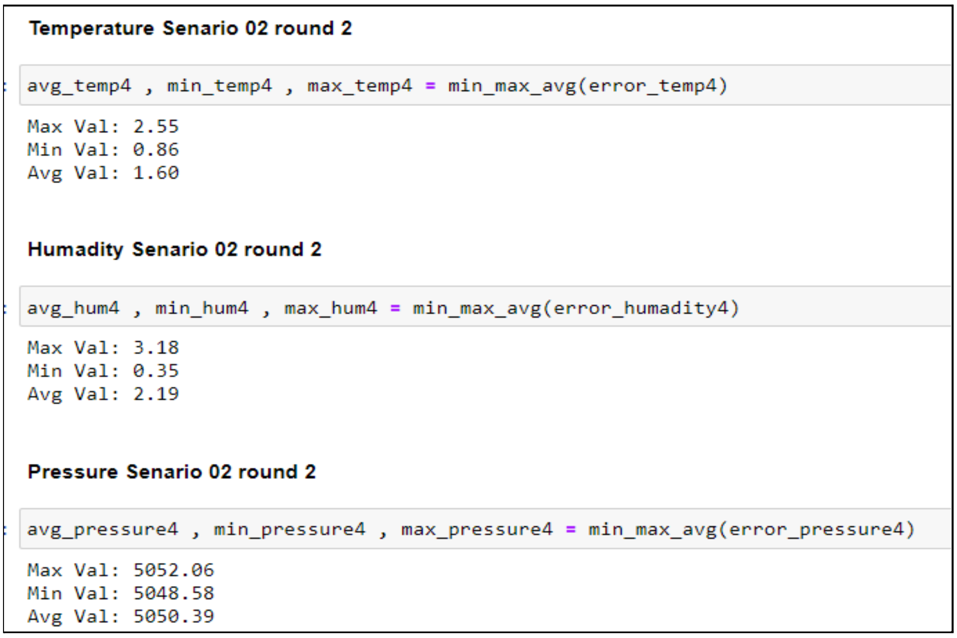
```
Max Val: 5.62
Min Val: 3.90
Avg Val: 4.76
```

Pressure Senario 01 round 2

```
avg_pressure3 , min_pressure3 , max_pressure3 = min_max_avg(error_pressure3)
```

```
Max Val: 12121.39
Min Val: 12119.67
Avg Val: 12120.53
```

❖ Min, avg, max in Scenario 2:



Temperature Senario 02 round 2

```
avg_temp4 , min_temp4 , max_temp4 = min_max_avg(error_temp4)
```

```
Max Val: 2.55
Min Val: 0.86
Avg Val: 1.60
```

Humadity Senario 02 round 2

```
avg_hum4 , min_hum4 , max_hum4 = min_max_avg(error_humadity4)
```

```
Max Val: 3.18
Min Val: 0.35
Avg Val: 2.19
```

Pressure Senario 02 round 2

```
avg_pressure4 , min_pressure4 , max_pressure4 = min_max_avg(error_pressure4)
```

```
Max Val: 5052.06
Min Val: 5048.58
Avg Val: 5050.39
```

- ❖ From These readings, we can conclude that the results become different after we make just 2 rounds, as the readings become little.