NUNAM ASSIGNMENT

A step by step Explanation

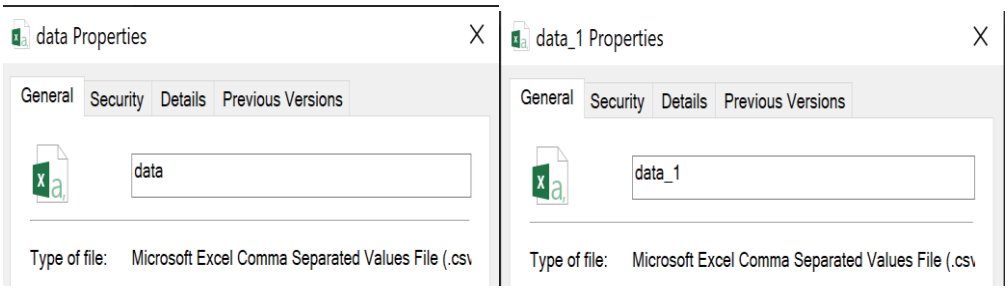
NAME: PARTHIBAN S

REGNO: 17MIS0307

Task 1:

- The given two data files are:
  1. data.xlsx
  2. data_1.xlsx

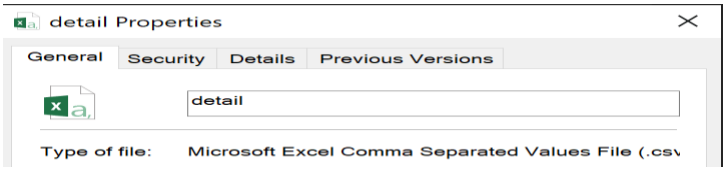
- Converting two given data files to .csv files
  1. data.csv
  2. data_1.csv



- Combining two csv files to detail.csv file using python

```python
#importing os
import os
#importing pandas
import pandas as pd
#creating dataframe
nunam_assignment = pd.DataFrame()
#file directory walk in
for file in os.listdir(os.getcwd()):
    #ifcase for searching file ends with .csv
    if file.endswith('.csv'):
        #reading file and appending
        nunam_assignment = nunam_assignment.append(pd.read_csv(file))
#merging into a file
nunam_assignment.to_csv('detail.csv')
```

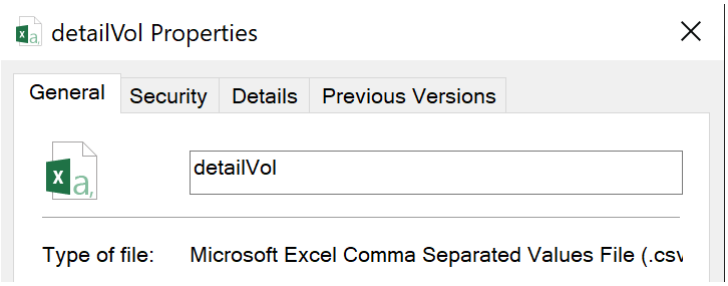
Result:



- Combining two csv files to detailVol.csv file using python

```
#importing os
import os
#importing pandas
import pandas as pd
#creating dataframe
nunam_assignment = pd.DataFrame()
#file directory walk in
for file in os.listdir(os.getcwd()):
    #ifcase for searching file ends with .csv
    if file.endswith('.csv'):
        #reading file and appending
        nunam_assignment = nunam_assignment.append(pd.read_csv(file))
#merging into a file
nunam_assignment.to_csv('detailVol.csv')
```

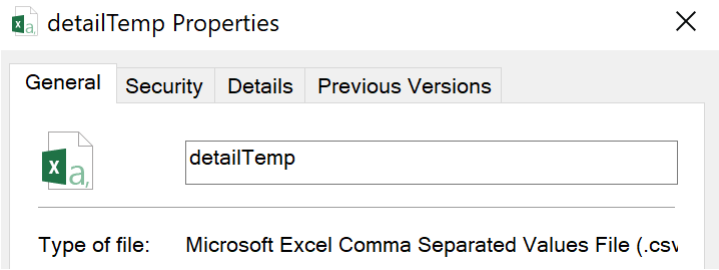
Result:



- Combining two csv files to detailTemp.csv file using python

```python
#importing os
import os
#importing pandas
import pandas as pd
#creating dataframe
nunam_assignment = pd.DataFrame()
#file directory walk in
for file in os.listdir(os.getcwd()):
    #ifcase for searching file ends with .csv
    if file.endswith('.csv'):
        #reading file and appending
        nunam_assignment = nunam_assignment.append(pd.read_csv(file))
#merging into a file
nunam_assignment.to_csv('detailTemp.csv')
```

Result:



Task 2:

- Apply down-sampling method to reduce the sampling rate to 1 sample/minute

1. Importing pandas as pd
2. Importing datetime as datetime
3. By setting the low_memory argument to *False*
4. Reading detail.csv
5. Using down sampling fetching value of 04:23:15 in Record ID

Python Code:

```
from datetime import datetime

import pandas as pd

date_str = '04:23:15'
nunam_assignment = pd.read_csv(
    'detail.csv', low_memory=False,
     date = datetime.strptime(date_str, '%H:%M:%S'),
     index_col=['Record ID']
)
nunam_assignment
```

Task 3:

- Low pass filter technique for noise removal on the data set for 'detailVolDownsampled.csv'.

1. Install packages scipy, csv, pandas, numpy, matplotlib
2. And define a function plot
3. Read csv file for detailVolDownsampled.csv
4. And data frequency data for 'Voltage'
5. And define a function bandpassFilter(signal)

Python code:

```
from scipy.signal import filtfilt
from scipy import stats
import csv
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import scipy

def plot():
    data = pd.read_csv('.detailVolDownsampled.csv')
    sensor_data = data[['Voltage']]

    sensor_data = np.array(sensor_data)

    time = np.linespace(0, 0.0002, 4000)

    plt.plot(time, sensor_data)
    plt.show()

    filtered_signal = bandPassFilter(sensor_data)
```

```
    plt.plot(time, filtered_signal)
    plot.show()

def bandPassFilter(signal):
    fs = 4000.0
    lowcut = 20.0
    highcut = 50.0

    nyq = 0.5 * fs
    low = lowcut / nyq
    high = highcut / nyq

    order = 2

    b, a = scipy.signal.butter(order, [low, high], 'bandpass', analog=False)
    y = scipy.signal.filtfilt(b, a, signal, axis = 0)

    return(y)
```

Task 4:

- Run profile for all the functions; use cProfile for Python for profiling of individual functions.

```
import cProfile
import re
cProfile.run('re.compile("foo|bar")')
```

1. Importing the packages cProfile, re
2. And running re.compile

Result:

```
Run: mycode
C:\Users\NCR\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/NCR/Desktop/PYTHON CODE/mycode.py"
         244 function calls (237 primitive calls) in 0.003 seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        1    0.000    0.000    0.002    0.002 <string>:1(<module>)
        2    0.000    0.000    0.000    0.000 enum.py:283(__call__)
        2    0.000    0.000    0.000    0.000 enum.py:562(__new__)
        9    0.000    0.000    0.000    0.000 enum.py:659(name)
        1    0.000    0.000    0.000    0.000 enum.py:790(_missing_)
        1    0.000    0.000    0.000    0.000 enum.py:797(_create_pseudo_member_)
        1    0.000    0.000    0.000    0.000 enum.py:833(__and__)
        1    0.000    0.000    0.000    0.000 enum.py:869(_decompose)
        1    0.000    0.000    0.000    0.000 enum.py:886(<listcomp>)
        1    0.000    0.000    0.002    0.002 re.py:250(compile)
        1    0.000    0.000    0.002    0.002 re.py:289(_compile)
        1    0.000    0.000    0.000    0.000 sre_compile.py:249(_compile_charset)
        1    0.000    0.000    0.000    0.000 sre_compile.py:276(_optimize_charset)
        2    0.000    0.000    0.000    0.000 sre_compile.py:453(_get_iscased)
Run  TODO  Problems  Terminal  Python Packages  Python Console  Event Log
PEP 8: W391 blank line at end of file  64:1  CRLF  UTF-8  4 spaces  Python 3.8
```

```
        1    0.000    0.000    0.000    0.000 sre_parse.py:76(__init__)
        2    0.000    0.000    0.000    0.000 sre_parse.py:81(groups)
        1    0.000    0.000    0.000    0.000 sre_parse.py:921(fix_flags)
        1    0.000    0.000    0.001    0.001 sre_parse.py:937(parse)
        9    0.000    0.000    0.000    0.000 types.py:171(__get__)
        1    0.000    0.000    0.000    0.000 {built-in method __new__ of type object at 0x00007FFE2E55B810}
        1    0.000    0.000    0.000    0.000 {built-in method _sre.compile}
        1    0.002    0.002    0.003    0.003 {built-in method builtins.exec}
       27    0.000    0.000    0.000    0.000 {built-in method builtins.isinstance}
    30/27    0.000    0.000    0.000    0.000 {built-in method builtins.len}
        2    0.000    0.000    0.000    0.000 {built-in method builtins.max}
        9    0.000    0.000    0.000    0.000 {built-in method builtins.min}
        6    0.000    0.000    0.000    0.000 {built-in method builtins.ord}
       48    0.000    0.000    0.000    0.000 {method 'append' of 'list' objects}
        1    0.000    0.000    0.000    0.000 {method 'disable' of '_lsprof.Profiler' objects}
        5    0.000    0.000    0.000    0.000 {method 'find' of 'bytearray' objects}
        1    0.000    0.000    0.000    0.000 {method 'get' of 'dict' objects}
        2    0.000    0.000    0.000    0.000 {method 'items' of 'dict' objects}
        1    0.000    0.000    0.000    0.000 {method 'setdefault' of 'dict' objects}
        1    0.000    0.000    0.000    0.000 {method 'sort' of 'list' objects}
```

Task 5:

- Run unit test on each function
  1. Import package unittest
  2. Creating class teststringmethods
  3. Defining function test_upper, test_isupper, test_split with self parameter
  4. special __name__ variable to have a value "__main__".

Python code:

```
import unittest
class TestStringMethods(unittest.TestCase):
    def test_upper(self):
        self.assertEqual('foo'.upper(), 'FOO')
    def test_isupper(self):
        self.assertTrue('FOO'.isupper())
        self.assertFalse('Foo'.isupper())
    def test_split(self):
        s = 'hello world'
        self.assertEqual(s.split(), ['hello', 'world'])
        # check that s.split fails when the separator is not a string
        with self.assertRaises(TypeError):
            s.split(2)
if __name__ == '__main__':
    unittest.main()
```

Result:

```
Run:  mycode
C:\Users\NCR\AppData\Local\Programs\Python\Python38\python.exe "C:/Users/NCR/Desktop/PYTHON CODE/mycode.py"
...
----------------------------------------------------------------------
Ran 3 tests in 0.000s

OK

Process finished with exit code 0
```

****THANKYOU****