**Jeffery D. Winchester**

College of Technology, Davenport University

CISP 243: Python Programming

Samer Hanoudi

Final Project Assignment:

Weather Data

Tuesday, April 9th, 2024

**Overview**

**Program Requirements:**

* Filter the linked file and keep the following columns only: STATION, NAME/LOCATION, DATE, AWND, SNOW. Save the filtered file and name it as filteredData.csv
* For each NAME/LOCATION, calculate the average snow amount per month. Save the results in two separate .csv files; average2016.csv and average2017.csv
* For each NAME/LOCATION, calculate the total/sum of snow amount per month. Save the results in two separate .csv files; total2016.csv and total2017.csv
* Sort the data in the files; average2016.csv and average2017.csv. Store only the top 3 locations from each file. The top 3 locations , data will be stored in one file, named; top3.csv. Each column in the top3.csv file will store the results of each year, file will have the columns 2016 and 2017, included.
* From the 2 year data, find the top 10 readings/rows of AWND. Store the result in a file .csv file and name it top10AWND.csv. This new file will have all columns from filteredData.csv, but only the top 10 AWND

**Defined Scope:**

Export analyzed data into several CSV files and filter the data in CSV files to provide insights on average snowfall and wind speed readings.

**Intended Usage:**

A potential use of this program is to be used to analyze weather data in forecasting seasonal weather patterns.

**Processing Logic**

**Control & Data Flow:**

The program starts by importing the pandas library as ‘pd’. Next, the program reads the data from a CSV file located on my personal computer in the File Explorer and then, associates the CSV file as a DataFrame ‘OG\_data’.

On line 5, the program selects the columns (‘STATION’, ‘NAME’, ‘DATE’, ‘AWND’, and ‘SNOW’) from the DataFrame ‘OG\_Data’, creates a new DataFrame ‘filtered\_data’and then on line 7 the filtered data is saved into a new CSV file named “filteredData.csv.”

On line 10, the program convert the ‘DATE’ Column in ‘filtered\_data’ to DateTime Format using the ‘pd.to\_datetime( )’ function.

On lines 13 & 14, the program extracts the Year and Month from the ‘DATE’ Column in ‘filtered\_data’; which was previously converted, then adds the month/year to the new columns ‘MONTH’ and ‘YEAR’.

On lines 17 & 19, the program calculates the ‘average\_snow’ and ‘total\_snow’ per month for each location and year; using ‘.mean( )’ and ‘.sum( )’, then the program saves the DataFrames into new separate DataFrame for the years; 2016 and 2017.

Lines 22-32, splits ‘average\_snow’ (L.22&23) and ‘total\_snow’ (L.25&26) for 2016 and 2017 into their own new DataFrame for each year. Lines 28 & 29; saves the Average Snow and Total Snow to their own CSV file for each year.

Lines 35 & 36 sort the average snow data for 2016 and 2017 in Descending order. The program selects the Top 3 Locations with the Highest Snowfall for each year, and then stores the results in new DataFrames: ‘sorted\_average\_snow\_2016’ and ‘sorted\_average\_snow\_2017’.

Line 39, concatenates the Top 3 Locations for 2016 and 2017 into a single DataFrame named ‘top3\_data’. Then, Line 41 saves ‘top3\_data’ locations into a new CSV file named ‘top3.csv’.

Finally, Line 44, selects the top 10 rows from the ‘filtered\_data’ DataFrame and based on the Highest ‘AWND’ Average Wind Speed readings, stores the results in the DataFraem ‘top10\_AWND’ and Line 46, saves the top 10 AWND readings into a CSV file named ‘top10AWND.csv”.

**Process Speceficiation (PDL, Pseudo Code, Flow Chart):**

*Program Design Language (PDL)*

* Import the pandas library as pd
* Read data from a CSV file located at “/Users/winch/OneDrive/[Davenport.edu/Python](http://davenport.edu/Python) Programming/Final Weather data project/HistoricalData.csv” into a DataFrame named ‘OG\_data’.
* Filter columns ‘STATION’, ‘NAME’, ‘DATE’, ‘AWND’, ‘SNOW’ from ‘OG\_data’ DataFrame and create a new DataFrame named ‘filtered\_data’.
* Save ‘filtered\_data’ DataFrame into a new CSV file named “filteredData.csv”
* Convert the ‘DATE’ column in ‘filtered\_data’ DataFrame to DateTime formatting using pd.to\_datetime( ) function
* Extract year and month from the ‘DATE’ column and add them as new columns ‘YEAR’ and ‘MONTH’ to ‘filtered\_data’ DataFrame.
* Calculate the average snowfall per month for each location and year, storing the results in the ‘average\_snow’ DataFrame using the groupby( ) and mean ( ) functions.
* Calculate the total snowfall per month for each location and year, storing the results in the ‘total\_snow’ DataFrame using the groupby( ) and sum( ) functions.
* Split ‘average\_snow’ and ‘total\_snow’ DataFrames into separate DataFrames for the years 2016 and 2017
* Save the average snowfall data for 2016 and 2017 into separate CSV files named “average2016.csv” and “average2017.csv”
* Save the total snowfall data for 2016 and 2017 into separate CSV files named “total2016.csv” and “total2017.csv”
* Sort the average snowfall data for 2016 and 2017 in descending order and select the top 3 locations with the highest snowfall for each year, storing the results in ‘sorted\_average\_snow\_2016’ and ‘sorted\_average\_snow\_2017’ DataFrames.
* Concatenate the top 3 locations for 2016 and 2017 into a single DataFrame named ‘top3\_data’
* Save ‘top3\_data’ DataFrame into a CSV file named “top3.csv”.
* Select the top 10 rows from ‘filtered\_data’ DataFrame based on the highest ‘AWND’ (Average Wind Speed) readings and store the results in the ‘top10\_AWND’ DataFrame.
* Save ‘top10\_AWND’ DataFrame into a CSV file named “top10AWND.csv”

*Pseudo Code*

* Import the pandas library as pd
* Read data from a CSV file into a DataFrame named ‘OG\_data’
* Filter columns from ‘OG\_data’ DataFrame and create a new DataFrame named ‘filtered\_data’
* Save ‘filtered\_data’ into a new CSV file
* Convert the ‘DATE’ column in ‘filtered\_data’ to datetime format
* Extract the year and month from the ‘DATE’ column and add them as new columns ‘YEAR’ and ‘MONTH’
* Calculate the average and total snowfall per month for each location and year, storing the results in ‘average\_snow’ and ‘total\_snow’ DataFrames
* Split ‘average\_snow’ and ‘total\_snow’ DataFrames into separate DataFrames for the years 2016 and 2017
* Save the average and total snow data for the years 2016 and 2017 into separate CSV files
* Sort the average snow data fro 2016 and 2017 in descending order and select the top 3 locations with the highest snowfall for each year
* Concatenate the top 3 locations for 2016 and 2017 into a single DataFrame
* Save the top 3 data locations into a CSV file
* Select the top 10 rows from ‘filtered\_data’ based on the highest ‘AWND’ (Average Wind Speed) readings and store the result in the ‘top10\_AWND’ DataFrame
* Save the top 10 AWND readings into a CSV file

*Flowchart*

**START** → Read CSV → Filter Columns → Save Filtered Data

Convert Date → Extract Year & Month → Calculate Average Snow & Total Snow → Split Data for 2016 & 2017 → Save Data for 2016 & 2017

Sort Data for 2016 & 2017 → Select Top 3 Locations → Concatenate Top 3 Locations → Save top 3 Locations

Select Top 10 AWND Readings → Save Top 10 AWND Readings → **END PROGRAM**

**Flow of Subroutines/Methods/Functions and Specific Algorithms:**

* Subroutines
  + Selecting specific columns from DataFrames
  + Splitting data for 2016 and 2017
* Methods
  + ‘pd.read\_csv( )’ method from Pandas library to read data fromm CSV file
  + ‘To\_csv’ method used to save data to a CSV file
  + ‘pd.to\_datetime( )’ method to convert the ‘DATE’ column to datetime format
  + ‘Dt.month’ and ‘dt.year’ dataframe used to extract month/year from the ‘DATE’ column
  + ‘To\_csv’ method used to save data to a CSV file
  + Saving top 3 locations data to a CSV file
  + ‘nlargest( )’ method for finding top values
  + Saving top 10 AWND data readings to a CSV file
* Functions
  + Sorting average snow data
* Algorithms
  + Grouping and calculating averages of snowfall
  + Grouping and and calculating total snow

**Data (INPUT/OUTPUT)**

ELEMENT Descriptions, Relationships, Initial Values, & Range:

* ‘OG\_data’
  + Type: Pandas DataFrame
  + Dimension: Rows (CSV file entires) and Columns (diff. attributes)
  + Initial Values: reads the CSV file specified in the path
  + Range of Possible Values: depends on the content request of the CSV file
  + Reads the entire CSV file in “/Users/winch/OneDrive/[Davenport.edu/Python](http://davenport.edu/Python) Programming/Final Weather data project/HistoricalData.csv”
* ‘Filtered\_data’
  + Type: Pandas DataFrame
  + Dimension: Rows and 5-columns
  + Initial Values: empty; until ‘OG\_data’ is filtered
  + Range of Possible Values: dependent on the range of values for the requested columns in ‘OG\_data’
  + Element is Derived from ‘OG\_data’ and filters 5 specific columns.
* ‘Average\_snow’
  + Type: Pandas DataFrame
  + Dimension: Rows and 4-columns
  + Initial Values: empty; until DataFrame calculations are calculated on ‘filtered\_data’
  + Range of Possible Values: Average snow calclations
  + Derived from ‘filtered\_data’
* ‘Total\_snow’
  + Type: Pandas DataFrame
  + Dimension: Rows & 4-columns
  + Initial Values: empty; until DataFrame calculations are calculated on ‘filtered\_data’
  + Range of Possible Values: total snow calculations
  + Derived from ‘filterred\_data’
* ‘Average\_snow\_2016’, ‘average\_snow\_2017’, ‘total\_snow\_2016’, ‘total\_snow\_2017’
  + Type: Pandas DataFrame
  + Dimension: Rows & 4-columns
  + Initial Values: empty; until DataFrames are filtered from ‘average\_snow’ and ‘total\_snow’
  + Range of Possible Values: ‘SNOW’ column contains representative numeral for snowfall in inches
  + ‘Average\_snow\_[2016/2017] are subsets of ‘average\_snow’ and ‘total\_snow\_[2016/2017]’ are subsets of ‘total\_snow’ each filtered by year.
* ‘Sorted\_average\_snow\_2016’, ‘sorted\_average\_snow\_2017’
  + Type: Pandas DataFrame
  + Dimension: 3-rows & 4-columns
  + Initial Values: empty; until DataFrames are sorted from ‘average\_snow\_2016’ and ‘average\_snow\_2017’
  + Range of Possible Values: ‘SNOW’ column contains representative numeral for snowfall in inches
  + Sorted subsets of ‘average\_snow\_[2016,2017]’
* ‘Top3\_data’
  + Type: Pandas DataFrame
  + Dimension: 3-rows & 8-columns
  + Initial Values: empty; until concatenated
  + Range of Possible Values: ‘SNOW’ column contains representative numeral for snowfall in inches
  + Derived by concatenating top 3 locations for 2016 and 2017 by way of sorted subsets of ‘average\_snow\_[2016,2017]’
* ‘top10\_AWND’
  + Type: Pandas DataFrame
  + Dimension: 10-rows & 7-columns
  + Initial Values: empty; until filtered from ‘filtered\_data’
  + Range of Possible Values: contains ‘AWND’ values representing wind speed
  + Derived from ‘filtered\_data’ by selecting top 10 records based on the ‘AWND’ Average Wind Speed.

**Components (Source code names, Classes, Methods)**

**Software Components**

* Data Reader
  + Purpose: for reading data from the CSV file
* Data Processor
  + Handles filtering, manipulation, and aggregation of the CSV file
* Data Analyzer
  + Analyzes processed data from CSV file and extracts specific columns
* Data Writer
  + Manages the output and saves data to CSV files

**UML Class Diagram**

* Weather Data
* Data: DataFrame
  + Read\_data (string)
  + Filter\_columns (string)
  + Convert\_to\_datetime ( )
  + Extract\_year\_month ( )
  + Calculate\_average\_snow ( ): DataFrame
  + Calculate\_total\_snow ( ): DataFrame
  + Split\_data\_by\_year (integer): DataFrame
  + Save\_data\_to\_csv (string)
  + Sort\_by\_snow (dataframe)
  + Find\_top\_locations ( ): DataFrame
  + Find\_top\_wind\_speeds ( ): DataFrame

**Properties and Constraints**

* ‘OG\_data’: holds all the weather data in its original format
  + Contraints: Must have the valid DataFrame object for our requested weather data: ‘STATION’, ‘NAME’, ‘DATE’, ‘AWND’, and ‘SNOW’

**Algorithms defined in Processing Logic**

* ‘calculate\_average\_snow( )’: method calculates the average snowfall for each location and year.
* ‘calculate\_total\_snow( )’: method calculates the total snowfall for each location and year.
* ‘sort\_by\_snow( )’: method sorts DataFrame by snowfall values in descending order.
* ‘find\_top\_locations( )’: method identifies the top locations based on snowfall.
* ‘find\_top\_wind\_speeds( )’: method finds the top wind speeds recorded in dataset.

**Testing**

**Input Data**

* Any CSV file that contains data. An organization could use any of the columns; this program is not limited to the columns that were chosen.

**Expected Output Data**

* If searching for averages, expected output should be the ‘Correct’ averages that were requested.

**Success Criteria**

* Output CSV files should contain the correct number of rows/columns as requested.
* Output CSV file should be formatted correctly
* Output CSV file should not contain anything unexpected or any errors