# Project 2: Bad Data

Submission Due: October 28 by 11:59PM

## Program Description:

Data processing is one of the most common activities that software is designed to perform. One of the largest data producers and consumers in the world is the US National Oceanic and Atmospheric Administration (NOAA). They collect data from a myriad of sensors all over the globe measuring temperature, barometric pressure, wind speed, tidal currents, and more to predict future weather patterns. However, sometimes sensors malfunction and either provide incorrect information, as seen by the temperature sensor that contributed to the Artemis I launch being scrubbed, or they fail to collect any information at all. In cases where no data is collected, interpolation can be used by selecting data that is in close proximity to the gap, taking the average of that data, and using the resulting value to fill the gap. Additionally, in order to verify that such methods are effective, if no real data is available, synthetic data can be generated that mimics the real data.

For this project, you will be creating a program that can generate and then interpolate synthetic temperature data. Your program will provide the user with a menu of options that include the ability to create a new synthetic temperature dataset of a desired size, the ability to interpolate holes in that dataset using the surrounding data or using the entire dataset, and the ability to compare the original dataset to the interpolated dataset.

## Program Requirements

- As with all projects in this course, your program should have a comment at the top that includes your name, the date the program was created, and a brief description of the project, in your own words
- Import any required libraries
- All variables must be declared inside of functions, and no global variables are allowed
- Define a new function named `main` that takes in no arguments
  - Create a variable to store a `List` representing the synthetic data and initialize it to the values
    - -1,391,408,403,389,-1,439,397,429,410,-1,-1,-1,435,-1,351,415,401,415,-1
  - Create a variable to store a `List` representing the interpolated data and initialize it to an empty `List`
  - Output an appropriate welcome message to the user
  - Declare a variable to store the user's menu choice and initialize it to an appropriate value
  - Create a loop of your choice that allows the user to repeatedly choose menu options until the choose the option to quit the program
    - Call the function to output the menu and store the result in an appropriate variable
    - If the user enters 1, call the function to generate synthetic data and store the result in the appropriate `List` variable

    - If the user enters 2, 3, or 4, call the function to interpolate the data, passing in the appropriate number of minutes on either side of incorrect data to interpolate across and the synthetic data `List,` and store the result in the appropriate `List` variable
    - If the user enters 5, call the function to interpolate across all of the data, passing in the synthetic data `List,` and store the result in the appropriate `List` variable
    - If the user enters 6, call the function to compare the data, passing in the synthetic data `List` and the interpolated data `List`
    - If the user enters 7, output an appropriate exit message
    - If the user enters anything else, politely inform them of their mistake
- Define a new function named `menu` that takes in no arguments
  - Provide the user with the following numbered menu options
    - Generate a synthetic dataset
    - Interpolate with 2 minutes
    - Interpolate with 4 minutes
    - Interpolate with 6 minutes
    - Interpolate with all minutes
    - Output data comparison
    - Exit
  - Prompt the user for a choice
  - Return the user's input
- Define a new function named `outputValues` that takes in a `List` containing a series of values and a `String` representing the prefix for the line
  - Output the prefix
  - On the same line as the prefix, output each of the values
    - Note, this function can be used to output multiple types of values such as minutes and temperatures, however the data should line up vertically, so make sure you take that into account when designing the output
    - See example output for an example
  - Output an empty line
- Define a new function named `generateDataset` that takes in no arguments
  - Prompt the user for a number of minutes (10 or more) to generate data for
    - If the user enters a number less than 10, politely inform them of their mistake and repeatedly prompt them for a new value until they provide something 10 or greater
  - Declare an empty `List` to store the temperatures
  - Using the number of minutes the user entered, for each minute
    - Generate an integer between 1 and 5
      - If the integer is 5, add a -1 to the `List` indicating that no data was collected for that minute
      - Otherwise, add a random integer between 350 and 450, inclusively, to the `List` of temperatures
  - Using the appropriate function, output each of the minutes on the same line with the prefix "Min:"

    - Using the appropriate function, output the generated temperature for each of the minutes in the `List`, on the same line, with the prefix "Temp:"
    - Return the `List` of temperatures
- Define a new function named `calcMean` that takes in a `List` containing a series of temperatures
    - Calculate the mean across all of the temperatures
        - Note if the temperature is -1, ignore it and do not include it in the calculation
    - If all of the temperatures in the `List` were -1, return a -1
    - Otherwise, return the mean
- Define a new function named `calcVariance` that takes in a `List` containing a series of temperatures, and a number representing the mean temperature
    - Calculate the variance across all of the temperatures
        - Note if the temperature is -1, ignore it and do not include it in the calculation
        - Remember the equation to calculate variance is $\frac{\sum(x - \bar{x})^2}{n-1}$ where:
            - $x$ is a temperature
            - $\bar{x}$ is the mean temperature
            - $n$ is the number of temperatures
    - Return the variance
- Define a new function named `interpolateMinutes`, that takes in an integer representing the number of minutes to use in interpolate and a `List` containing a series of temperatures
    - Output how many minutes are being used to interpolate with
    - Declare an empty `List` to store the interpolated data
    - For each temperature in the passed in `List`
        - If the temperature is -1
            - Extract the values to the left and right of the temperature, based on the number of minutes specified, and create a new single `List` containing all of those values
            - Using the appropriate function, calculate the mean of those temperatures
            - Append the mean to the `List` of interpolated data
            - Note, if the -1 temperature is at the beginning or end of the List, only use the available temperatures to calculate the mean
        - Otherwise, append the temperature to the `List` of interpolated data
    - Return the `List` of interpolated data
- Define a new function named `interpolateGlobal`, that takes in a `List` containing a series of temperatures
    - Output that all minutes are being used for interpolation
    - Declare an empty `List` to store the interpolated data
    - Using the appropriate function calculate the mean of the `List` of temperatures and store it in an appropriate variable
    - For each temperature in the passed in `List`
        - If the temperature is -1
            - Append the mean to the `List` of interpolated data
        - Otherwise, append the temperature to the `List` of interpolated data

  - Return the `List` of interpolated data
- Define a new function named `compareData`, that takes in a `List` containing the synthetic temperatures and a `List` containing the interpolated temperatures
  - If interpolation has not been performed yet, inform the user that they must use one of the interpolation functions first, and then return from this function
  - Using the appropriate function, output each of the minutes on the same line with the prefix “Min:”
  - Using the appropriate function, output the temperature for each of the minutes in the synthetic `List`, on the same line, with the prefix "Temp:”
  - Using the appropriate function, output the temperature for each of the minutes in the interpolated `List`, on the same line, with the prefix "Temp:”
  - Using the appropriate function, calculate the mean of the synthetic temperatures and then output it with two decimal points of precision and an appropriate message
  - Using the appropriate function, calculate the variance of the synthetic temperatures and then output it with two decimal points of precision and an appropriate message
  - Calculate the standard deviation of the synthetic temperatures and then output it with two decimal points of precision and an appropriate message
    - Remember that standard deviation is the square root of the variance
  - Using the appropriate function, calculate the mean of the interpolated temperatures and then output it with two decimal points of precision and an appropriate message
  - Using the appropriate function, calculate the variance of the interpolated temperatures and then output it with two decimal points of precision and an appropriate message
  - Calculate the standard deviation of the interpolated temperatures and then output it with two decimal points of precision and an appropriate message
    - Remember that standard deviation is the square root of the variance
  - Using matplotlib:
    - Plot the synthetic temperatures in red using the marker o
    - Plot the interpolated temperatures in blue using the marker o
    - Set the title of the plot to “Synthetic Data vs Interpolated Data”, with a font size of 14
    - Set the x label of the plot to “Minutes”, with a font size of 14
    - Set the y label of the plot to “Temperature”, with a font size of 14
    - Set the grid to True
    - Show the plot
    - Note that plots will appear in the Plots pane in Spyder
- Call the `main` function
- Your code should be well documented in terms of comments. For example, good comments in general consist of a header (with your name, date, and brief description), comments for each variable, docstrings describing each function, and commented blocks of code
- Your program source code should be named **TempSense.py**

## Programming Suggestions

- It is recommended that you do not try to build the entire program first and then test it. Make sure you are testing functions as you build them to make sure you are not including errors
- Function stubs are a great way to have the function header present without the body! Just remember to add `pass` to the body so that nothing happens when the function is called
- You will need to do a bit of research yourself on how to plot a line chart in Python using matplotlib. This means you will need to use your favorite search engine and start looking at various websites that offer suggestions and tutorials on how to provide the requested functionality. Remember, you will need to include both `Lists` in a single line chart.
- Do not forget to call your `main` function at the end!

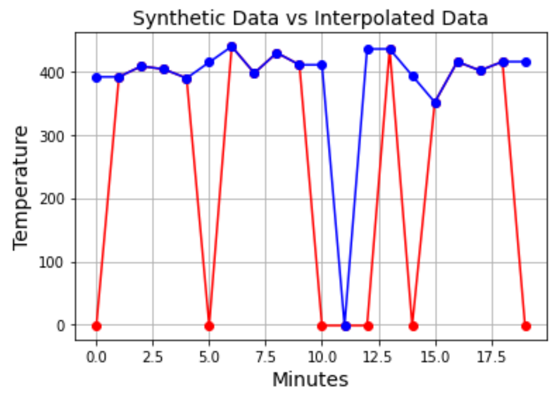
## Submission

- Your program will be graded largely upon whether it works correctly
- Your program will also be graded based upon your program style. This means that you should use comments (as directed) and meaningful variable names
- You must submit the **TempSense.py** file to Canvas by the due date and time
- Projects are meant to be individual coding assignments, so no collaboration is allowed. This includes downloading code off of the internet. Any discovered instances of this will be considered cheating and appropriate actions will the taken according to the course syllabus
- Be sure that you have tested the version of the program you wish to submit to make sure it works correctly. You will not be allowed to resubmit work after the deadline that does not have a third-party timestamp such as Dropbox or an email. Timestamps generated by your operating system will not be acceptable.

## Sample Output:

```
Welcome to the sensor simulator!

Menu:
1. Generate a synthetic dataset
2. Interpolate with 2 minutes
3. Interpolate with 4 minutes
4. Interpolate with 6 minutes
5. Interpolate with all minutes
6. Output data comparison
7. Exit
Please enter a choice: 2

Interpolating with 2 minutes

Menu:
1. Generate a synthetic dataset
2. Interpolate with 2 minutes
3. Interpolate with 4 minutes
4. Interpolate with 6 minutes
```

```
5. Interpolate with all minutes
6. Output data comparison
7. Exit
Please enter a choice: 6

Min:      0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16
17  18  19
Temp:   -1 391 408 403 389  -1 439 397 429 410  -1  -1  -1 435  -1 351 415
401 415  -1
Temp:  391 391 408 403 389 414 439 397 429 410 410  -1 435 435 393 351 415
401 415 415

Original Data Set
Mean: 406.38
Variance: 524.42
Standard Deviation: 22.90

Interpolated Data Set
Mean: 407.42
Variance: 425.15
Standard Deviation: 20.62
```



```
Menu:
1. Generate a synthetic dataset
2. Interpolate with 2 minutes
3. Interpolate with 4 minutes
4. Interpolate with 6 minutes
5. Interpolate with all minutes
6. Output data comparison
7. Exit
Please enter a choice: 8

8 is not a valid choice

Menu:
1. Generate a synthetic dataset
2. Interpolate with 2 minutes
3. Interpolate with 4 minutes
4. Interpolate with 6 minutes
5. Interpolate with all minutes
6. Output data comparison
```

```
7. Exit
Please enter a choice: 1

How many minutes would you like to generate data for (10 or more minutes):
8
8 minutes is too brief! Please enter a time that is 10 minutes or more.
How many minutes would you like to generate data for (10 or more minutes):
10
Min:      0   1   2   3   4   5   6   7   8   9
Temp:  441 417 383 426  -1 418  -1 399 432 435

Menu:
1. Generate a synthetic dataset
2. Interpolate with 2 minutes
3. Interpolate with 4 minutes
4. Interpolate with 6 minutes
5. Interpolate with all minutes
6. Output data comparison
7. Exit
Please enter a choice: 7

Thank you for using the simulator!


Welcome to the sensor simulator!

Menu:
1. Generate a synthetic dataset
2. Interpolate with 2 minutes
3. Interpolate with 4 minutes
4. Interpolate with 6 minutes
5. Interpolate with all minutes
6. Output data comparison
7. Exit
Please enter a choice: 6

Please use one of the interpolation strategies before outputting data!

Menu:
1. Generate a synthetic dataset
2. Interpolate with 2 minutes
3. Interpolate with 4 minutes
4. Interpolate with 6 minutes
5. Interpolate with all minutes
6. Output data comparison
7. Exit
Please enter a choice: 7

Thank you for using the simulator!
```

## Bonus:

TBA