Name: Abdullah Shahzad
Course: CS 152
Section: B
Date: 9/24/2021

Project 3 Report:

Title: Calculating Thermoclines in the Great Pond with Programs and Modules that Reuse Code

Abstract:
A function is a block of code that performs a particular operation. For this project I had to code a library of functions because I knew I'd have to perform similar operations with data again and again. Instead of having to rewrite the code every time, I had a dedicated .py file with functions such as sum and average that I could import (borrow) and call (execute) in different files whenever I wanted to.
The rest of the project revolved around calculating the depth of the thermocline (the transition layer between deep and surface water) for the month of July. The data only had the temperature of the water in Great Pond at different depths and I had to, using a function, use the temperature to calculate the density of the water at a particular depth. From there, I had to write a function that could check at what depth the rate of change (the derivative) of the density was greatest—that was the depth of the thermocline. After figuring out the depth at which surface and deep water mixed for each day in July, I had to print the answers to a .csv file and make a graph.

Equations:
There were two specialized equations used in my program

1) Variance: $$\sigma^2 = \frac{\Sigma_i (x_i - \bar{x})^2}{N - 1}$$

Where σ^2 is the variance and it is equal to the squared difference between the list item and the mean of all items in that list divided by the number of items in that list minus 1. A list is a way of structuring data in python. For calculations in this report, a list stores data inside square brackets '[ ]'.

2) Converting Temperature to Density:

```
rho = 1000 * (1 - (t + 288.9414) * (t - 3.9863)**2 / (508929.2*(t + 68.12963)))
```

Where rho means density and t means temperature.

Output: The output of the thermoclines for the month of July will be at the end of this report in the form of a graph with the day number on the x-axis and the thermocline depth on the y-axis.

***Key for Required Output 1**: **sum** refers to the sum of all the values in the list, **mean** refers to the mean of all the values in the list, **max** and **min** are the highest and lowest values in that list and **var** represents the variance of the values in the list calculated using the formula in the Abstract of this report.*

Results:

Required Output 1:

```
PS C:\Users\HP\OneDrive\Desktop\Project_03> py stats.py
sum: 10.0
mean: 2.5
min: 1.0
max: 4.0
var: 1.67
PS C:\Users\HP\OneDrive\Desktop\Project_03>
```

Fig. 1.1

I tested the functions that I wrote in stats.py by making a test function. The test function called the function it was testing within itself.

All the calculating functions required an argument that would be a list so that they could compute it. The test function used a simple list: [1, 2, 3, 4]. The results of the calculations are in **Fig. 1.1**. The test function for calculating the mean of the values in the list can be seen in **Fig. 1.2**. All the other test functions were similar except that they called different functions in stats.py and hence got their different results.

```
def test():
    '''Testing my mean function'''
    list1 = [1, 2, 3, 4] #initializing the list to calculate the mean of
    print('mean:', mean(list1)) #calling the mean function inside a print statement with list1 as an argument
if __name__ == "__main__": #prevents the test function from running when imported on another file
    test() #test function call
```

Fig. 1.2

Required Output 2:

The function from Required Output 1 can also be used to compute a column of data. In **Fig. 1.3**, the function is working with data from Gate Pond during the month of July in 2019.

```
PS C:\Users\HP\OneDrive\Desktop\Project_03> py analyze.py GoldieJuly2019.csv  10
sum: 74318.86000000007
mean: 24.97
min: 20.80
max: 28.61
var: 1.91
PS C:\Users\HP\OneDrive\Desktop\Project_03>
```

Fig. 1.3

I can confirm whether the calculated values in **Fig. 1.3** are correct by going to excel and using a built-in feature to calculate the mean of the column in excel that this data is computing. If the answers are the same, then the functions in stats.py are working correctly. In **Fig. 1.4**, I have

used the built-in function to calculate the average of the data in the column and it is outputted in the box with a green outline.

K
27.84
27.85
27.8
27.73
27.75
27.76
24.9727

Fig. 1.4

The mean calculated by stats.mean() is 24.97 and the mean calculated by Excel is 24.9727. The data in the command line in **Fig 1.3** is formatted to two decimal places to make it more readable but the answers are the same, meaning that stats.mean() is working correctly.

Required Output 3

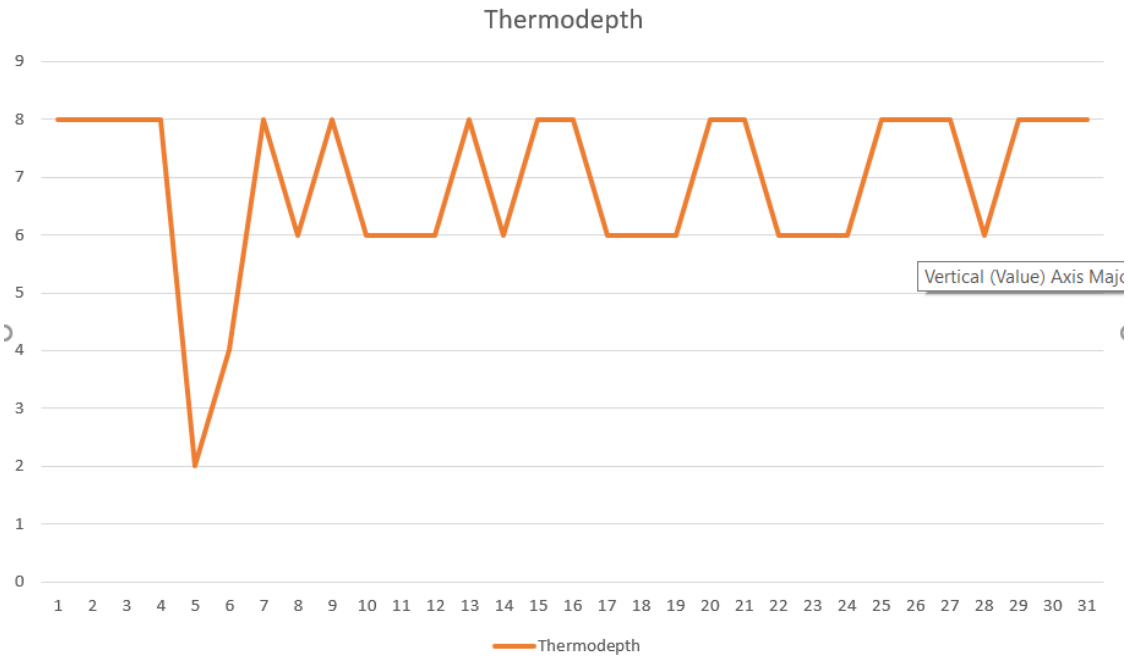


Fig. 1.5 Note: Required Output 3 code is in JulyThermocline1.py

Fig. 1.5 Shows a plot of the results for the depth of the thermocline (thermodepth) for each day (31 total days) in the month of July

Required Element 4: Follow-up Questions:

1. Explain what is meant by the term “code reuse”?

It means utilizing code that's already been written instead of reinventing the wheel by importing a program onto your .py file.

2. Explain what is meant by the term "modular design"?

It is a technique for programming where you divide the code into different functions or parts that can be run separately so that someone doesn't have to run the entire program and can access a specific function.

3. Perform a Google search for a woman statistician and give a one sentence description of a contribution they made.

Name: Gertrude Mary Cox

Contribution: Dubbed the "First Lady of Statistics", her efforts were fundamental to the development of the vibrant statistics community in the Research Triangle of North Carolina.

Source: https://www.worldofstatistics.org/famous-statisticians-from-history/

Reflection:

Lecture concepts utilized in this project include, creating and using a library of functions to program, using lists while coding functions, using formulas in functions to make data, and fetching, copying and pasting data (more for extensions). Real world problems that this could connect to would be mapping scenarios using equations. Things like monitoring and calculating the changes in the sea level at the coast, computing data about the population, calculating costs and gross income at businesses, all of this is made possible and efficient using code that would probably look very similar to the one in this project. At the end, when I was calculating the derivative of the thermocline with respect to the depth of the water, I also realized that python or other coding languages may be very powerful math tools. After coding a function, I could keep it, change the variables and use it to calculate different things.

You must also address the following questions in your response:

1. What does it mean to import a module, package, or library into a Python program?

It means that the functions that the module, package, or library had can be run on the python program they have been imported to.

2. What are the key similarities and differences between string and list data types?

A string is a sequence of characters, while a list is a collection of elements. Both can be accessed via indices, which is a smiliarty. Differences include how lists can contain elements of different types while a string is a string, and that lists are mutable while strings are immutable.

3. Discuss different ways we have learned to get input from the user into our programs?

We have learned two ways of getting data from users: using the input command and through command line control options.

Extension (optional):

Key: Windspeed_Thermodepth.csv has the data for wind speed and thermodepth that I used to make the graph in my last extension.
The **MLRC files** are files that were created because I had to do a lot of cutting and pasting to get to the point where I just had 1 Windspeed_Thermodeth.csv file.
JulyThermoclineBoth.csv contains the data for the thermocline at two different times.

.Ext 1: Write functions in your stats.py file to compute more types of statistics.

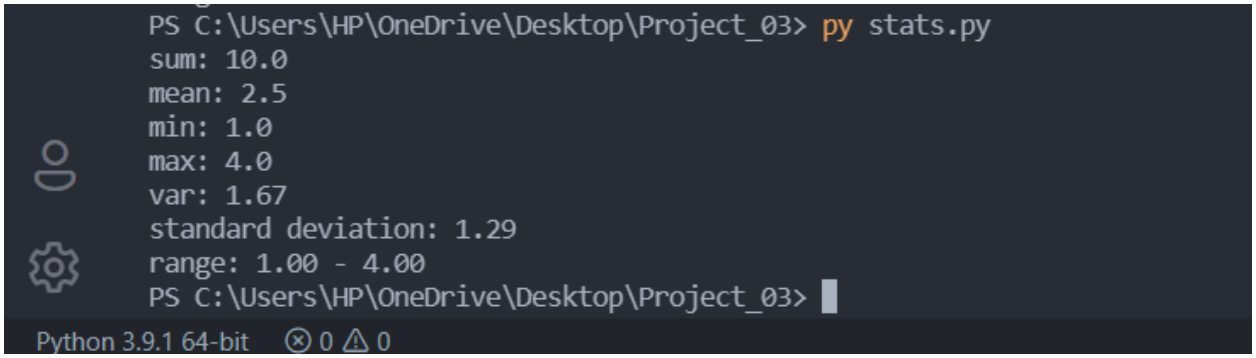


Fig 1.6

I added the calculations for standard deviation and range in my stats.py file. These calculations use the same list: [1,2,3,4] as required output 1.

Ext 2: Compare Different Times in the Goldie Data

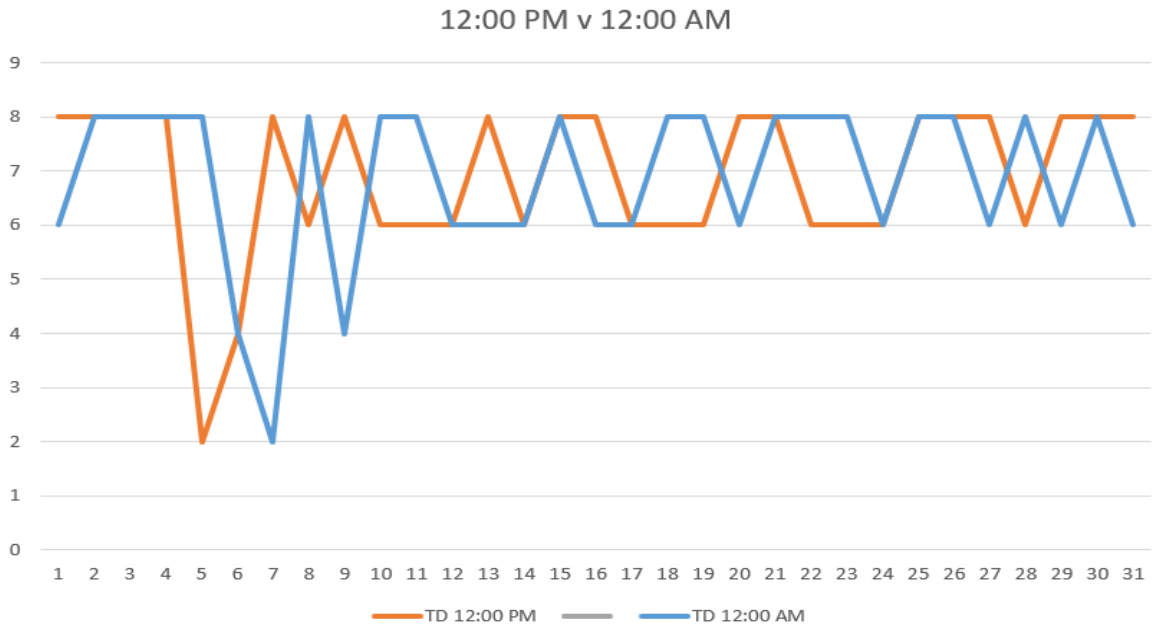


Fig 1.7

Here, I copied the code from required output 3 and made a new .py file. They both printed to different .csv files the data for different times in the day (12:00 PM and 12:00 AM). I then used the paste command to combine the two. The paste command takes

two columns from .csv files and puts them in 1 .csv file. I then made it into a graph and viola.

Ext 3: Add more command line control options

```
65    if __name__ == '__main__': #doesn't automatically ca
66        main(sys.argv[1], sys.argv[2])
67
```

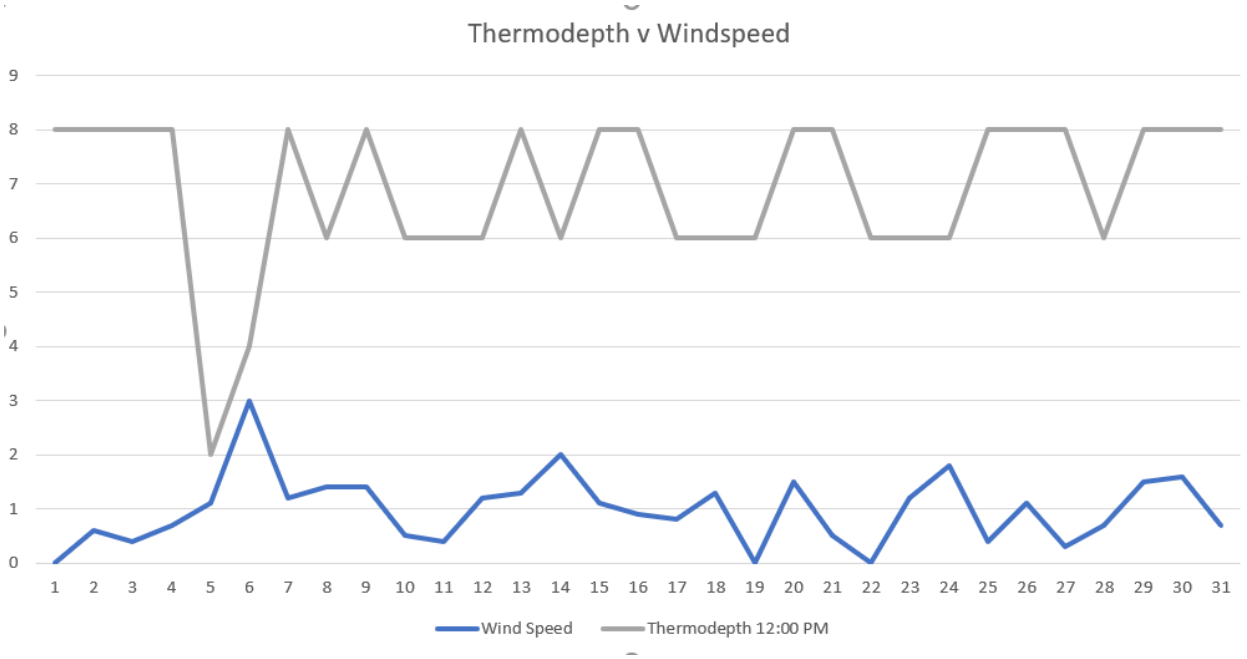
Fig 1.8

```
PS C:\Users\HP\OneDrive\Desktop\Project_03> py .\JulyThermocline.py "12:03:00 PM" "12:03:00 AM"
1 , 8.0
2 , 8.0
3 , 8.0
4 , 8.0
5 , 2.0
6 , 4.0
7 , 8.0
8 , 6.0
Python 3.9.1 64-bit  ⊗ 0 △ 0
```

Fig 1.9

I coded the thermocline file to take input in from the command line to decide which time the data should be recorded from. I imported the sys function, which is just 'import sys' at the top of the file and put in the arguments like **Fig 1.8** demonstrates. It can be seen to work in **Fig 1.9**

Ext 4: What if you graph wind direction and thermocline together, is there a relationship?



I got the data for the wind speed from a file in project 2. I used grep, which grabs the data from one file and puts the data into another file, to isolate the data for wind speed at 12:00 PM, I then pasted the data from the windspeed.csv and the JulyThermocline.csv file together. After getting all the data into one file, I could make a graph out of it. As for a relationship between the two things, I do see one. The data for the wind speed has a lot more activity, but it does mimic the increase and decrease of the thermodepth at that time.

Acknowledgements: Professor Allen Harper, Zihui Zhang