Pseudocode:

In the main function:

Let line be an array of size 20. (n)

Open file named “biodata.dat” to read from, this is outFile.

Create and open a file named “filtered_biodata.dat” to write to, this is newFile.

Write “BIODATA Formatted Output” to newFile and add a space under that line.

Read the first line of outFile and assign that number to the variable numReadings. (1)

Use the getline function to mark the first line of outFile as read.

FOR (numbers starting at 0 < numReadings, increasing the number each time) (3n)

Use getline to read the next line.

Using the tempChange function, add the return value to the variable avgTemp.

Pass the line that was read to the dateChange function.

Add a line space after the formatted output.

Write “Average Temp --- “ to newFile.

Using setprecision to keep the decimal places at 2, write the answer to avgTemp divided by numReadings to newFile, followed by “ C” for the degree unit.

Close outFile and newFile.

Return 0. (1)

End of main function.

In the function dateChange:

Write “ recorded on “ to newFile.

FOR (numbers starting at 4 and up to 6, increasing each time) (2n)

Take the number from that numbered spot in the line passed to the function and write that to newFile. This is the month.

Write "/" to newFile.

FOR (numbers starting at 6 and up to 8, increasing each time) (2n)

Take the number from that numbered spot in the line passed to the function and write that to newFile. This is the day.

Write "/" to newFile.

FOR (numbers starting at 0 and up to 4, increasing each time) (4n)

Take the number from that numbered spot in the line passed to the function and write that to newFile. This is the year.

Write " at " to newFile.

FOR (numbers starting at 8 and up to 12, increasing each time) (4n)

Take the number from that numbered spot in the line passed to the function and write that to newFile. This is the time, it is in 24hr mode so there is no more formatting that needs to be done.

Write a line space to newFile.

End of dateChange function.

In the tempChange function:

FOR (numbers starting at 14 and up to 19, increasing each time) (5n)

Take the number from that numbered spot in the line passed to the function and store that in tempArr. This is the temperature reading.

Take the character string tempArr and store it as a float in temp using strtof. (1)

SWITCH (based on the 14th spot in the full line read from outFile, which should be the degree unit) (there are 3 comparisons each time this function is called, 3)

In the case that the letter is "F", we have to go from Fahrenheit to Celsius.

Subtract 32 from temp and then divide by 1.8.

IF (the new temp < 0) (1)

This program does not accept negative temperature readings, an error message is printed and the program is ended immediately.

Using setprecision to keep the decimal places at 2, write the new temp to newFile, followed by " C --- " for the degree unit and formatting.

In the case that the letter is "C", there are no modifications we have to make.

IF (the temp < 0) (1)

This program does not accept negative temperature readings, an error message is printed and the program is ended immediately.

Using setprecision to keep the decimal places at 2, write the new temp to newFile, followed by " C --- " for the degree unit and formatting.

In the case the letter is not "F" or "C", we have our default case with an error message.

This program only allows "F" and "C", so the user should edit the folder contents accordingly. The program ends immediately.

This function returns temp. (1)

End of tempChange function.

21n+5

This program is O(n)