Mercedes AMG Petronas F1 Team Technical Interview Solution

Author: Robert Pringle (pringle@live.co.uk)

This is my solution for the Mercedes AMG Petronas F1 team technical interview which was implemented using Python 3.9.7. This README provides a brief file structure outline and then a report of the program, including any assumptions I made. **A user manual (MANUAL.md) is also provided which describes how to run the system and test suite.**

File Structure

- README.pdf - provides a pdf version of this file
- MANUAL.md - describes how to run the system and unit tests
- MANUAL.pdf - provides a pdf version of how to run the system and unit tests
- Channel Processing System.docx - The task specification provided
- **src/ - contains all the code for the project**
- src/main.py - the program which implements the task
- src/test.py - test cases which test the functions within main.py
- src/inputs - contains all the input files which are read into main.py
- src/inputs/channels.txt - the original channels.txt provided
- src/inputs/parameters.txt - the original parameters.txt provided
- src/outputs - stores the outputs of the main.py program

Report

Python

This program was implemented using Python 3.9.7. See the user manual (MANUAL.md) on how to run it.

Findings

From the data provided I calculated the value of b to equal 6.2698521667770072 and this metric data and the data in the channel arrays is exported to src/outputs.

Floating Point Error Handling

The floating point error causes calculations with normal floating point numbers to become inaccurate, due to the way they are handled internally within the system. I identified this was going to be an issue that needed to be resolved, as the program needed to perform calculations on numbers with large decimal places. I therefore imported the python Decimal library to handle all the floating point numbers. This basically means each decimal number is of type Decimal INSTEAD of float within the program. This allowed for accurate calculations as the library (which comes default with Python) handled all of the floating point representation for me.

Floating Point Assumption

As the decimal numbers within the channels.txt had values with the maximum of 16 decimal places, I assumed that I should round all the calculations I performed to 16 decimal places. However, if you did want the program to not round to 16 decimal places you could edit the integer of line 12 in src/main.py to make it higher or lower.

Input and Output Format Assumption

I made the assumption that the parameters.txt file and the channel.txt could not be changed, therefore the program performs string manipulation to extract the required data from these files. Function readParameters() handles reading in parameters.txt and readChannel() handles reading in channels.txt. Additionally, I assumed the output files had to be exported in the same format as the provided files and this is handled by writeOutput(). Therefore, to test any other data with the program the .txt file imported needs to be the same format as the original files. Each output folder contains three files: channels_data.txt (outputs ALL the channels calculated on seperate lines), metric_data.txt (outputs metric b) and parameters_used.txt (outputs the parameters used).