Week 1 Assignment: Numeric Data Processing

Overview

In this assignment, you will begin working with a public data set from the National Oceanic and Atmospheric Administration and compute some basic statistics over the data.

Source Data

Download the Boulder, CO, daily station data for 2018, which is part of the data collected in the U.S. Climate Reference Network (USCRN). Save this file as **Data.txt** using the "Save As" option in your browser and move it to the directory you are using for this assignment.

Step 1: Data Preparation

The description of the data file format contains a lot of useful information. Please read through it. Not everything is important, but definitely *Sections 4 and 5* deserve some attention. We are interested in Field 9, T_DAILY_AVG.

The program being developed (see Step 2 below) expects its data to come in sorted numerically ascending, one value per line, so we need to use some command-line tools to take care of this.

Step 1.a: Extract the column of data

Using the command-line program **cut**, extract just Field 9 from the data file and save it in a file called **T_DAILY_AVG.txt**. You will need to read the documentation of the **cut** program (e.g., **man cut**) to figure out exactly what options are needed, but it will be something like this:

```
cut [cut-options] Data.txt > T_DAILY_AVG.txt
```

Step 1.b: Sort the column of data

The next step is to sort the data numerically using the **sort** program. The sorted data should be saved in a file named **T_DAILY_AVG_sorted.txt**. Please read the documentation of **sort** (e.g., **man sort**) to figure out how to do this. The general form of the command will be:

```
sort [sort-options] T_DAILY_AVG.txt > T_DAILY_AVG_sorted.txt
```

Step 1.c: Do it in a pipeline

As shown in lecture, commands can be combined in a pipeline without saving the intermediate data into files, so we could do all of the above in one shot so that the numerically sorted, ascending data is piped directly to a Python program something like this:

```
<cut command> | <sort command> | python compute_stats.py
```

Part 1 Deliverable: Create and upload a document called **Week1_Part1.txt** and include the three command lines for Parts 1.a, 1.b, and 1.c.

Part 2: compute_stats.py

In this part of the assignment, you will implement a Python script that reads data from standard input, computes some basic statistics over the values that are read in, and writes those statistics to standard output. This program will be implemented in a file called **compute_stats.py**.

This program will assume that the data is read in a stream of numbers, one per line. The data coming in will have already been sorted numerically in ascending order (i.e., what you did in Part 1). There might be "missing-data" values (read the data file documentation) that must be ignored when computing the statistics.

The statistics that must be computed are

1) Average value
2) Minimum value
3) Maximum value
4) Median value

Before your program terminates, it should print the computed values formatted similar to the following:

```
min: -10.3, max: 24.1, average: 1.6471354835154, median: 2.23
```

Part 2 Deliverable: upload **compute_stats.py**

Upload

Please combine **Week1_Part1.txt** and **compute_stats.py** into a single ZIP file.