HW04: gRPC

Due: Mon Mar 16 at 11:59pm
Total: 55 points

In this homework you will make a gRPC service to help monitor the temperature of your Raspberry Pi CPU. I have already written the .proto file and the client for you. You will have to implement the service itself and deploy onto the Raspberry Pi.

In the main() function you will use `argparse` to accept either no arguments or the argument `--port` and single ***integer*** argument for the port number, defaulting to 50051. Use that port for your service. When running on your Raspberry Pi, both you and your Pi partner will be using the same computer (although you are working independently) and thus you must run on different ports. Whomever's first name comes first in the alphabet will run on port 50051. The other partner will run on port 50052.

Current Temperature & Temperature Stream - 15 pts

The first two functions should be fairly straight-forward to implement except for getting the CPU temperature. To get the current CPU temperature:

```
psutil.sensors_temperatures()['bcm2835_thermal'][0].current
```

You will need to install the psutil package on the Raspberry Pi.

Currently `psutil` only supports temperatures on Linux so this will ***NOT*** work on your Macs (on other Linux systems you would also need to change the key used). You will likely want to put this in a method of your class and maybe add an if statement like:

```
if not hasattr(psutil, 'sensors_temperatures'):
    return 40.0
```

Which will return the fixed value `40.0` for all systems on which `sensors_temperature()` is not supported.

The `Temperatures()` method will loop while `context.is_active()` which becomes `False` if the client cancels the stream. Remember to `time.sleep()`.

Min/Max Temperatures - 10 pts

This function returns the minimum and maximum temperatures ever recorded. Only when temperatures returned by `CurrentTemperature()` or `Temperatures()` need to be considered (i.e. you don't have to continually monitor the temperature yourself). If neither of those functions have returned since starting the service, then the behavior of this function is undefined (i.e. I don't care, it could be an exception, bogus values could be returned, etc). You will likely want to integrate this with the method you created for the first part.

Stress Test - 10 pts

This function simply has a while loop that keeps adding 1 to a number until the given amount of time has elapsed. Use time.monotonic() to get the “current time” in seconds.

Usability and Code Quality - 20 points

The quality of your code and the usability of the command line program will make up the plurality of the points in this assignment, broken down as follows:

- 5 pts - splitting up your code properly into functions/methods
- 5 pts - documenting your functions/methods/classes/modules
- 5 pts - using good variable names and commenting code as necessary
    - Note that commenting as necessary may mean very few or no comments as long as other things are done well
- 5 pts - appropriate `main()` and usage of `argparse`

If you go really off in something I don’t predict I will keep taking points off from this overall category, including going negative. In a 300-level class you should really be writing quality code.

NOTE: you should not commit any automatically generated files. You may update the .gitignore file if you wish these to be automatically ignored.