Introduction to Python, ISA, ULisboa

Assignment P4: “haversine”

To show in class 8: November 10, 2023

Instructor: Manuel Campagnolo

In class, we discussed the function **haversine** from the package haversine that calculates the distance (in km by default) between two points on Earth using their latitude and longitude.

The goal of the assignment is to test the function **myhaversine** and the auxiliary functions **mysqrt, mycos, myarcsin, mysin** which are defined in files **myfunctions.py** and **myhaversine.py**. In particular, the groups need to test if function **myhaversine** computes distances between any pair of locations on Earth within 100 m of the “correct” distance returned by the function **haversine**, and slightly adapt the code if necessary so the test is successful.

To understand the code that defines **myhaversine**, you can look at this link on Wikipedia. The code is correct, but it relies on approximations of functions sine, cosine, sqrt (square root) and arcsin (inverse sine function) which might not be precise enough.

Besides **myhaversine**, the following auxiliary functions to be tested individually. All of those functions have some parameter that allow to improve their precision (either n or tol).

- **mysqrt**. The input is a number and the output is 0 if the input is less than 0 or the approximate value of the square root of the input otherwise
- **myarcsin**. The input is a number (x) and the output is -90 if the input is less than -1, 90 if the input is larger than 1, or the approximate value of the inverse sine angle (in degrees), i.e. y =arcsin(x)
- **mysin**. The input is a number (angle in degrees) and the output is an approximation of the sine of that angle
- **mycos**. The input is a number (angle in degrees) and the output is an approximation of the cosine of that angle

Tasks:

1. Create file **test_haversine.py** that will be executed in the terminal with **pytest test_haversine.py** or with **python -m pytest test_haversine.py** to test all five functions **mysqrt, mycos, myarcsin, mysin** and **myhaversine** individually;
2. Adjust parameters in the auxiliary functions if needed so that your test on **myhaversine** (it should compute distances between any locations on Earth within 100 m of the distance returned by the function **haversine** from package haversine) is successful;

3. The group should collaborate on GitHub. All interactions from group members should be available on the file's history (file **test_haversine.py)**. Commit messages should be very brief but explanatory.

Criteria for evaluation:

1. Did groups create and execute a correct **test_haversine.py** with **pytest**?
2. Are "corner cases" to be tested for each function meaningful and representative?
3. Was the code for auxiliary functions reasonably adapted, so test_haversine.py is executed successfully?
4. Did team members collaborate by committing their code and changes they made on the GitHub repository and are their commit messages meaningful? The instructor will look at the file history on GitHub for the group.

Extra provided code: file **some_distances.py** that can be executed to get some examples of the output of **myhaversine**.