AMS 209 Foundations of Scientific Computing

Bhishek Manek

November 14, 2016

Homework 5 Question 2-3

The routine can be basically divided into 3 parts:

- Counting of index of input unit system in a defined dictionary
- Calculating ratios with which other unit systems scale in terms of input system
- Calculating values in other unit system and presenting as a list

Defining dictionaries

In this section of my program, I specify two dictionaries which define the conversion factors in different unit systems taking meter as reference. So value to key 'meter' is defined as 1.

m2o dictionary defines conversion between meter, foot, mile, yard and inches whereas m2oo defines conversion factors for nm, µm, mm, cm and km.

Initializing lists and count

This part will initialize lists required for the python routine. The required lists are \textbf{x}, \textbf{y}, \textbf{final} and \textbf{final1}. \textbf{x} and \textbf{y} are basically the ratios of scaling in different unit systems with respect to the input unit system.
So if the input is in meters, then

\[ x = [0.000621, 39.370079, 3.28084, 1.093613] \]

which basically corresponds to the ratios \( \frac{1 \text{ meter}}{1 \text{ mile}} \) and so on for other units.

This section of code also initializes `count` which basically finds the index of input unit system in the `m2o` dictionary so that the input system can be avoided while printing, thus displaying the final values in all unit systems except the input system.

**Counting of index of input unit system in m2o**

This is the first major part of routine where it determines the index (stored in `count`) of input system in the dictionary `m2o`.

**Calculating scaling ratios**

Here I calculate the lists `x` and `y` which contain the scaling ratios as explained in the section *Initializing lists and count*.

**Final result**

In this part of routine, the input numerical value is calculated in different units using the lists `x` and `y` stored as `ratio` and `ratio1` respectively. After getting the numerical value in the unit systems, I add the name of the unit system as a string and store it in lists `final` and `final1`.

**Input**

This part will ask for input of numerical value of length (`n`) and its corresponding unit system (`a`).

At the end, I call the subroutine `converting_units(n,q)`.
Some examples

- $n = 20$ and unit system is in meter

```
[bmanek@aether:q25]
```
```
python q2.py
Please input a length (number only): 20
Please type a unit system (meter, mile, inch, foot, yard): meter
['65.6168 foot', '787.40158 inch', '0.01242 mile', '21.87226 yard']
['20000000.0 um', '20000.0 nm', '0.02 km', '2000000000.0 nm', '2000.0 cm']
```

- $n = 37$ and unit system is in foot

```
[bmanek@aether:q25]
```
```
python q2.py
Please input a length (number only): 37
Please type a unit system (meter, mile, inch, foot, yard): foot
['443.999988722 inch', '0.00780338937589 mile', '12.3333295741 yard', '11.2775996391 meter']
['1127.75996391 nm', '1127.75996391 mm', '0.011275996391 km', '1127.75996391 cm']
```

- $n = 189$ and unit system is in yard

```
[bmanek@aether:q25]
```
```
python q2.py
Please input a length (number only): 189
Please type a unit system (meter, mile, inch, foot, yard): yard
['567.000172822 foot', '6084.00190104 inch', '0.10732242877 mile', '172.821647146 meter']
['172821647.146 um', '172821647.146 mm', '0.172821647146 km', '1.72821647146e+11 nm', '17282.1647146 cm']
```