1 Introduction

In Problem 2 we use Python in order to create a length converter. The user is asked to provide two input elements: a length (any chosen number) and a unit (from the list: meter, mile, inch, foot, yard). That number is then converted to all the other units present in the list, and the output is printed on the screen. The user can also decide whether or not to extend this list of units to include several S.I. multiples of the meter (nanometer, micrometer, millimeter, centimeter and kilometer). If they are included, the conversion is applied again, and all converted values of the input number are printed on the screen, along with the appropriate units. In the following sections we describe in some detail the different elements of the code, as well as several examples to test it.

2 Description

Definitions

We begin by initializing the two input variables, and defining a Python dictionary containing the possible units. In particular, each key of this dictionary is one of these units, and the associated values are conversion factors (we shall detail these later on).

The User is then required to provide the two input variables (if the specified unit usr unit is not a key in the dictionary units dict, the User is prompted to enter a valid unit). We also note that the lower() method is called, so that entering capital letters in the name of the unit does not generate an error.

Function definition

The next step is to define the function converter, which takes as input a number num, a dictionary un dict and a string unit. When this function is called, its input is, respectively: number, units dict, usr unit. The idea is the following. In order to avoid implementing explicitly all conversion factors between every possible pair of unit systems, we can use one of the units as a reference. In this case we choose the meter: this is why all the conversion factors defined in units dict simply give the relation between each unit and the meter.

The function converter proceeds to convert the input number to its equivalent in meters: input_nbr. After recalling the input values, the function then loops over all the keys contained in the dictionary, and for all those which are different than the input unit, it calculates the corresponding conversion and prints it to the screen (along with the corresponding key, which is the unit).

Calling the function - output

In the last part we call the function once, with the User-defined variables and the initially defined units dict as input. This prints to the screen the equivalent values in all other units.

The User also has the possibility of choosing whether or not to extend the dictionary to include several other units: some S.I. multiples of the meter. If the User confirms, then the update() method is used to add these new keys (the new units) and their corresponding values (again, their conversion factors as in the previous case). Finally, the converter function is called again, but this time using the newly updated dictionary as input. The resulting converted values of the input number are printed to the screen.
3 Examples

Here we consider several examples to test the code. To run it, simply type in the terminal:

$ python length_converter.py

**Example 1** In this first example, we consider the input: 25.32 inch. We can also decide to extend the dictionary after the first call of the *converter* function. The output is given in Figure 1.

![Figure 1: Example 1 with input 25.32 inch](image)

**Example 2** In this second example, we consider the input: 0.23 mile. We can also decide to extend the dictionary after the first call of the *converter* function. The output is given in Figure 2.

**Example 3** Finally, let us consider one of the examples given in the problem, namely 159 yard, to check whether our output is correct. Again, extending the dictionary to include the second set of units, the output can be seen in Figure 3. We can easily see that this input matches the one given in the problem.
Figure 2: Example 2 with input 0.23 mile

Figure 3: Example 3 with input 159 yard