

Introduction to Engineering Methods (C++)
File: N125P2F

Programming Assignment #2: Unit Conversion Tables

This programming project will allow the user to create a table of unit conversions, where the user specifies:

- the dimension (example: length)
- the starting unit (example: ft)
- ending unit (example: m)
- the starting value (example: 0)
- the ending value (example: 10)
- the increment (1)
- the number of digits after the decimal point for the starting unit (example: 1)
- the number of digits after the decimal point for the ending unit (example: 3)

For the example above, the output might be similar to the table shown below:

Length (ft)	Length (m)
0.0	0.000
1.0	0.305
2.0	0.610
3.0	0.914
4.0	1.219
5.0	1.524
6.0	1.829
7.0	2.134
8.0	2.438
9.0	2.743
10.0	3.048

Program Requirements:

1. The program should provide an initial description.
2. The user should be prompted to enter the dimension (length, mass, or time).
3. The user should be given a choice of initial units and final units for each dimension as follows:

- Length: m, ft, cm or in

- Mass: kg, g or slug

- Time: h, min or s

Note that the initial unit cannot be the same as the final unit, but there are still many possible conversions. For example, with length there are 12 possible conversions

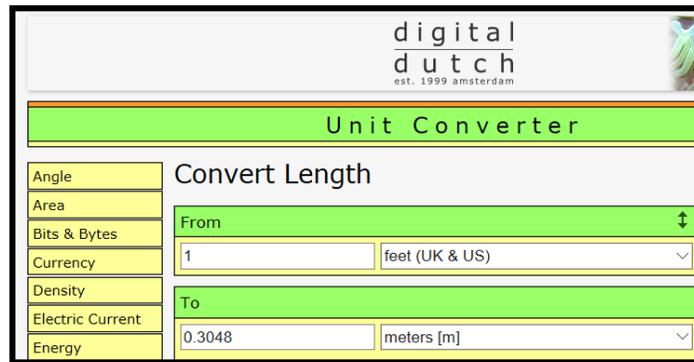
m to ft	ft to m	cm to m	in to m
m to cm	ft to cm	cm to ft	in to ft
m to in	ft to in	cm to in	in to cm

Similarly, there are multiple conversion possibilities for mass and time.

4. The units should be read as strings and any combination of uppercase and lowercase letters can be used. The user should be able to use the full unit name (plural or singular) or the

symbol (for example, meters, meter or m). Use a function to convert the input unit to all lowercase letters. Only the unit symbol should be used in the table.

- There are many sources for conversion factors. Use the conversion factors in the unit conversion program Digital Dutch (<https://www.digitaldutch.com/unitconverter/length.htm>). An example is shown below. If a value of 1 ft is entered and converted to m, we see that the conversion factor is 0.3048.



The screenshot shows the 'digital dutch' logo at the top, with 'est. 1999 amsterdam' below it. The main heading is 'Unit Converter'. On the left, there is a vertical menu with categories: Angle, Area, Bits & Bytes, Currency, Density, Electric Current, and Energy. The main content area is titled 'Convert Length'. It features a 'From' section with a text input field containing '1' and a dropdown menu set to 'feet (UK & US)'. Below that is a 'To' section with a text input field containing '0.3048' and a dropdown menu set to 'meters [m]'. A small image of a hand holding a pen is visible in the top right corner of the website interface.

- The table should include a heading with the dimension name and the units. The number of digits displayed must match the user's specifications. The table should be right justified and all decimal points should be aligned. See the sample table on the previous page.
- The program should calculate the number of lines in the table based on the user inputs specifying starting value, ending value and increment. The number of lines in the table must be at least 3, but not more than 25. If the ending value is not a multiple of the increment, the table should end with the last possible value before the ending value specified. For example, if the user specifies the following value:
Starting value: 10
Ending value: 100
Increment: 20
The program should determine that 5 lines are needed and that the last value in the table should be 90 (not 110).
- Use loops allow the user to correct any bad inputs (and provide a suitable error message). Bad inputs should include:
 - Choice of dimension (length, mass or time)
 - Choice of units (with correct spelling) for each dimension
 - Starting and ending unit cannot be the same
 - Number of digits ≥ 0
 - Starting value, increment and ending value must all be positive.
 - Number of lines in the table must be from 3 to 25. If this condition fails, the user should re-enter the starting value, increment and ending value.
- Give the user the option of re-running the program.
- Efficiency:** The program must be efficient or else it could become exceeding long and significant grade penalties will apply. For example, a poorly written program might repeat the code for generating a table dozens of times for the many possible conversions. An efficient program would only have one set of instructions to generate a table and the table would use the assigned values for dimension, units, digits, starting value, etc.
- Give the user the option of re-running the program (use a **do while loop**).
- The program should use at least 3 functions, including:
 - Function to convert a string to all lower case.
 - Function that returns the appropriate conversion factor based on the starting unit and ending unit.
 - Function of your choice (it must be useful and well-designed).

Test Cases:

Run the program for the following test cases:

Dimension	Starting Unit	Ending Unit	Starting Value	Ending Value	Increment	# digits (starting unit)	# digits (ending unit)
length	ft	m	0	10	1	1	3
length	METER	feet	100	300	30	0	1
time	H	S	0	4	0.2	1	0
time	second	MIN	5000	10000	400	0	0
mass	kilogram	Slug	0	1	0.05	2	4
mass	KG	Gram	1	5	0.25	2	0

Note: The first test case is the same as the example shown on page 1

Report:

Follow the guidelines listed in Format for Programming Projects (available on the instructor's web page).

Extra Credit Suggestions: (for a maximum of 10 additional points on the program grade)

1. Add additional dimensions and/or units.
2. Add lines to the table.
3. Use additional well-designed functions.
4. Add an option to have a table with either 2 or 3 columns. If 3 columns is selected, prompt the user to enter a 3rd unit.
5. Use your imagination!