Lab 1 – Temperature Table (temp_table.c)

Learning Goals

1. Develop your ability to write text files, compile them with gcc, and run them in the command line.
2. Develop your ability to write programs using for loops, functions, basic input/output, and define statements in C.

The Task

You have likely written a program like this in Java or Python. Let's go for one more!

Write a C program called **temp_table.c** that generates a Fahrenheit to Celsius conversion table, then prompts the user to enter a temperature in Fahrenheit and converts it to Celsius. To do the prompt, you'll need to use `scanf()` (see below)

The table should go from freezing (32 F) to boiling (212 F), counting by 10's; the Fahrenheit column should be rounded to 0 decimal places, and the Celsius rounded to 1 decimal place.

The prompt should accept numbers with decimal places, and report the Celsius conversion with 2 decimal places.

More Specific Requirements:

0) Name of file MUST be "temp_table.c"
1) The values for freezing and boiling must be specified using #define statements, and used wherever appropriate.
2) The calculation of the conversion from Fahrenheit to Celsius must be implemented ***within a function;*** perhaps named `f_to_c`, but the name is up to you. It should take a Fahrenheit temperature as a parameter, and return the Celsius conversion. (Recall the formula: $(F - 32) \times 5/9 = C$)
3) All variables used should be declared at the start of a function (or parameters).
4) The columns must right align.

See below for a sample output.

scanf()

In class, we have introduced 3 functions from the stdio library for reading user input – `getchar()`, `gets()`, and `fgets()`.

To read input that isn't just single characters or pure strings, we can utilize the `scanf()` method, which works like the inverse of the `printf()`.

Here's an example:

```
int x;
double y;
char[30] str;
scanf("%d %lf %s", &x, &y, str);
```

If the user inputs:

```
5 23.7 hello world
```

`scanf` will assign 5 to x, 23.7 to y, and “hello” to str. (%s will only read up to a whitespace character, unless a specified width is given)

Note:

1) The `%lf` reads a double (“long float”), whereas `%f` only reads a float
2) The amperstand “&” before the primitive variable names, but NOT the str.

TL;DR

For this assignment, you only need to read a ***double****. So the command is merely:*

```
scanf(“%lf”, &<doublevar>);
```

Sample Output

```
$ gcc -o temp_table temp_table.c
$ ./temp_table
Fahrenheit    Celsius
        32        0.0
        42        5.6
        52       11.1
        62       16.7
        72       22.2
        82       27.8
        92       33.3
       102       38.9
       112       44.4
       122       50.0
       132       55.6
       142       61.1
       152       66.7
       162       72.2
       172       77.8
       182       83.3
       192       88.9
       202       94.4
       212      100.0

Enter a temperature in Fahrenheit: 98.7
Celsius:      37.06
```