Lab 2 Functions

Learning Goals

1. Define your own function.

2. Call your function.

3. Define a function that returns a value.

4. Call that function.

5. Add a '''docstring''' to a function definition.

Define a function

Lab2-1.py

Using turtle graphics, define a function to draw a triangle of a given size.

```
# by Jane Student (your name here)
# Lab2-1.py

import turtle

def draw_triangle(size):
    # your block of statement go here


# Some ways to use the your function

draw_triangle(30)

x = 40
draw_triangle(x)

draw_triangle(x + 20)

turtle.forward(60)

# draw a series of larger triangles
length = 70
for i in range(8):
    draw_triangle(length)
    length = length + 15
```

Lab2-2.py

Define a function celsius_to_fahrenheit.

Given a temperature in fahrenheit, it returns the temperature in degrees celsius.

Information:

boiling	212 F	is	100 C
freezing	32 F	is	0 C
boiling - freezing	212-32 is180 F,	or	100 C

```
# by Jane Student (your name here)
# Lab2-2.py

def celsius_to_fahrenheit(celsius):
    # your block of statement go here:
    fahrenheit = ...  #
    # last statement is a return
    return fahrenheit

# Use your function
print(celsius_to_fahrenheit(100))
f = celsius_to_fahrenheit(34)
print(f)
print(celsius_to_fahrenheit(-40.0))
```

Lab 2-3.py

Add a **docstring** to your turtle graphic draw_triangle(size) function definition, then Save As Lab 2-3.py.

When the first indented statement after the def line in a function definition is a string, we call it a "docstring" – it documents what the function does. **Triple quoted** strings span multiple lines, so use them for all docstrings.

Add a docstring to your draw_triangle() function. It must be the 1st indented item following the def line.

Content:
Briefly tell what the function does.
Tell what input arguments the function takes.
Tell how the turtle ends up (pointing in the same direction, or in another direction?).

Run your program.
In the shell, type **`help(draw_triangle)`**

You see the function name and its parameters -- see the example with `draw_zigzag(distance)` to the right.

You also see your **docstring**, just as you typed it in the function.

Lab 2-4.py

Add a docstring to your celsius_to_fahrenheit(celsius) function, then Save As Lab 2-4.py.

In this case, working out what a couple of temperatures should be provides a way to make sure you are getting the correct results. Include them in the docstring.

Example of a docstring

```
import turtle

def draw_zigzag(distance):
    ''' Draws zagged lines,
        just for fun;
        each line is distance units long
        Turtle ends up heading in same
        direction as it orginally had.
    '''
    for item in range(6):
        turtle.left(20)
        turtle.forward(distance)
        turtle.right(40)
        turtle.forward(distance)
        turtle.left(20)

# call the function
draw_zigzag(60)
```

In the python shell

```
>>> help(draw_zigzag)
Help on function draw_zigzag in module _

draw_zigzag(distance)
    Draws zagged lines,
    just for fun;
    each line is distance units long
    Turtle ends up heading in same
    direction as it orginally had.

>>>
```

Grading

10 points for labs 2-1, 2-2
- 1 # student name in a comment line near start of program
- 3 program runs correctly
- 3 function defined
- 3 function called several times

10 points for labs 2-3, 2-4
- 2 docstring formed with triple quotes
- 3 docstring 1st statement after the def line in the function definition
- 2 docstring states purpose
- 3 help(`functionname`) shows docstring

XC 2 points each
- define draw_polygon(size, n_sides)
- define kelvin_to_fahrenheit(temp)