Functions

When programming, there are often times when you may want to repeat blocks of code over and over again in different parts of your program in a way that loops or if/else statements can't help with. In these cases, it is often useful to make functions. In fact, you have been using functions like `print()` and `range()` all week.

Think of a function like a recipe. When you look up a recipe, it gives you a list of instructions to follow in order to make a meal. Afterwards, when you want to make the same dish, you just need to recall the recipe and do so. Like a recipe, a function contains a list of instructions for a program to follow, and allows the program to access them over and over again when the function is called.

```
def spam():
    print("eggs")
    print("spam")
    print("eggs and spam")
```

Functions begin with the `def` keyword, followed by the function name, parenthesis and a colon. The code indented underneath it will execute as usual when the function is called.

```
>>> spam()
eggs
spam
eggs and spam
```

Like with loops and if/else statements, indentation is very important. The code in the function after the `def`, name, parenthesis, and colon must be indented, and any code not indented will be presumed to be the end of the function.

**Try it yourself:** write a function that prints your name.

### Parameters and Arguments

Functions can also have their abilities expanded greatly with parameter. Parameters are variables that the function accepts as input to be used within the function, introduced by putting variable names within the parenthesis after the function name, like so:

```
def spam(x):
    z = x+9
    print(z)
```
When the above function is called, a number has to be placed inside the parenthesis following it, unlike the previous function. This is because \texttt{spam(x)} has the parameter, \texttt{x}.

\begin{verbatim}
>>> spam(5)
14
\end{verbatim}

The input of a function is called the arguments. In the code above, 5 is the argument of \texttt{spam(5)}. These seem like similar concepts, but they are distinct: a parameter is placed within the definition of the function as required input, while an argument is the actual data input into a functions call. When speaking of a function call, the inputs are arguments. When speaking of the function itself, the inputs are parameters. In other words, a function has parameters and a function takes arguments.

If you think you've seen something similar to this before, you have – \texttt{print()} is a function that has a string parameter, and the string it prints is its argument!

\textbf{Try it yourself:} write a function that has a string as a parameter and tells you if it is your name.

\section*{The Return Statement}

Functions can also be ended with a return statement. The return statement sends data out to be used as the program sees fit, such as a string to be printed or a number to be assigned a variable:

\begin{verbatim}
def spam():
    x = 7
    return x
y = spam()
\end{verbatim}

\textbf{Try it yourself:} write a function that returns your name.

\section*{Variable Scoping}

Another important part of functions is the idea of \texttt{variable scope}. The scope of a variable is the area of the code in which it can be used. For example, in the above code for \texttt{spam(x)}, the scope of variable \texttt{z} is only within the function. If you were to attempt to use \texttt{z} in any other part of the code, you would get an error message, since for all intents and purposes, \texttt{z} only exists inside of \texttt{spam(x)}. Similarly, no variables created outside the function can be accessed directly by the function without being inserted as a parameter.

\textbf{Try it yourself:} write two functions that both take numbers as input, but return different values (\texttt{x * y} and \texttt{x / y}, for example). Use the same variable names for both functions.
Exercises:

1. Write a function that uses the pythagorean theorem \((a^2 + b^2 = c^2)\) to find the length of the hypotenuse when given the lengths of the other two sides of the triangle. (Hint: the square root of a number can also be found by raising the number to the \(\frac{1}{2}\) power.)

2. Write a function that finds the roots of a quadratic equation \(ax^2 + bx + c\) when given \(a, b,\) and \(c\).

   \[ x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \]

   Hint: Python will try to deal with imaginary numbers, but it may not get the exact answer if \(b^2 - 4ac\) is negative. You could check for \(b^2 - 4ac\) being negative and tell the user that there is no real answer, or you could just decide that inexact complex answers are okay.