Documenting Functions

It is good practice to document your functions. Include a brief description of what the function does, the parameters it requires and the return value if any. In Python, documentation for a function is provided as a docstring. A docstring is a multiline comment. The line begins with a capital letter and ends with a period. The first sentence is a short description of the function. The second line is blank followed by a more detailed description of the function. Note that the documentation for a function should follow the rules of indentation.

You can print out the docstring associated with each function using the __doc__ attribute.

In [1]:

```
def print_triangle(s):
    '''
        This function print out a right triangle with *\'s.

        Parameters: It accepts one parameter which is an integer value.
        Return Value: There is no return value.
    '''
    for i in range(1, s+1):
        for j in range(1,i+1):
            print('*', end='')
        print()
```

In [2]:

```
print(print_triangle.__doc__)    #This will print out the docstring associated with the function.
```

```
        This function print out a right triangle with *'s.

        Parameters: It accepts one parameter which is an integer value.
        Return Value: There is no return value.
```

In [3]:

```
def sum1(n):
    '''
        This function prints out the sum of the first n integers.

        Parameters:  It has one parameter which is the value of n.
        Return value:  The return value is the sum of the first n integers.
    '''
    s = 0
    for i in range(1, n+1):
        s += i
    return s
```

In [4]:

```
print(sum1.__doc__)
```

```
        This function prints out the sum of the first n integers.

        Parameters:  It has one parameter which is the value of n.
        Return value:  The return value is the sum of the first n integers.
```

In [5]:

```
'''
You can use the __doc__ attribute to learn more about the functions in a module after first import
ing the function.
'''
from math import sqrt
print(sqrt.__doc__)
```

```
print(sqrt.__doc__)
```

```
Return the square root of x.
```

In [6]:

```
'''
You can use the help() function to learn more about the contents of the entire module after first
importing the module.
'''
import math
help(math)
```

```
Help on built-in module math:

NAME
    math

DESCRIPTION
    This module provides access to the mathematical functions
    defined by the C standard.

FUNCTIONS
    acos(x, /)
        Return the arc cosine (measured in radians) of x.

    acosh(x, /)
        Return the inverse hyperbolic cosine of x.

    asin(x, /)
        Return the arc sine (measured in radians) of x.

    asinh(x, /)
        Return the inverse hyperbolic sine of x.

    atan(x, /)
        Return the arc tangent (measured in radians) of x.

    atan2(y, x, /)
        Return the arc tangent (measured in radians) of y/x.

        Unlike atan(y/x), the signs of both x and y are considered.

    atanh(x, /)
        Return the inverse hyperbolic tangent of x.

    ceil(x, /)
        Return the ceiling of x as an Integral.

        This is the smallest integer >= x.

    copysign(x, y, /)
        Return a float with the magnitude (absolute value) of x but the sign of y.

        On platforms that support signed zeros, copysign(1.0, -0.0)
        returns -1.0.

    cos(x, /)
        Return the cosine of x (measured in radians).

    cosh(x, /)
        Return the hyperbolic cosine of x.

    degrees(x, /)
        Convert angle x from radians to degrees.

    erf(x, /)
        Error function at x.

    erfc(x, /)
        Complementary error function at x.

    exp(x, /)
        Return e raised to the power of x.
```

```
    expm1(x, /)
        Return exp(x)-1.

        This function avoids the loss of precision involved in the direct evaluation of exp(x)-1
for small x.

    fabs(x, /)
        Return the absolute value of the float x.

    factorial(x, /)
        Find x!.

        Raise a ValueError if x is negative or non-integral.

    floor(x, /)
        Return the floor of x as an Integral.

        This is the largest integer <= x.

    fmod(x, y, /)
        Return fmod(x, y), according to platform C.

        x % y may differ.

    frexp(x, /)
        Return the mantissa and exponent of x, as pair (m, e).

        m is a float and e is an int, such that x = m * 2.**e.
        If x is 0, m and e are both 0.  Else 0.5 <= abs(m) < 1.0.

    fsum(seq, /)
        Return an accurate floating point sum of values in the iterable seq.

        Assumes IEEE-754 floating point arithmetic.

    gamma(x, /)
        Gamma function at x.

    gcd(x, y, /)
        greatest common divisor of x and y

    hypot(x, y, /)
        Return the Euclidean distance, sqrt(x*x + y*y).

    isclose(a, b, *, rel_tol=1e-09, abs_tol=0.0)
        Determine whether two floating point numbers are close in value.

          rel_tol
            maximum difference for being considered "close", relative to the
            magnitude of the input values
          abs_tol
            maximum difference for being considered "close", regardless of the
            magnitude of the input values

        Return True if a is close in value to b, and False otherwise.

        For the values to be considered close, the difference between them
        must be smaller than at least one of the tolerances.

        -inf, inf and NaN behave similarly to the IEEE 754 Standard.  That
        is, NaN is not close to anything, even itself.  inf and -inf are
        only close to themselves.

    isfinite(x, /)
        Return True if x is neither an infinity nor a NaN, and False otherwise.

    isinf(x, /)
        Return True if x is a positive or negative infinity, and False otherwise.

    isnan(x, /)
        Return True if x is a NaN (not a number), and False otherwise.

    ldexp(x, i, /)
        Return x * (2**i).

        This is essentially the inverse of frexp().
```

```
    lgamma(x, /)
        Natural logarithm of absolute value of Gamma function at x.

    log(...)
        log(x, [base=math.e])
        Return the logarithm of x to the given base.

        If the base not specified, returns the natural logarithm (base e) of x.

    log10(x, /)
        Return the base 10 logarithm of x.

    log1p(x, /)
        Return the natural logarithm of 1+x (base e).

        The result is computed in a way which is accurate for x near zero.

    log2(x, /)
        Return the base 2 logarithm of x.

    modf(x, /)
        Return the fractional and integer parts of x.

        Both results carry the sign of x and are floats.

    pow(x, y, /)
        Return x**y (x to the power of y).

    radians(x, /)
        Convert angle x from degrees to radians.

    remainder(x, y, /)
        Difference between x and the closest integer multiple of y.

        Return x - n*y where n*y is the closest integer multiple of y.
        In the case where x is exactly halfway between two multiples of
        y, the nearest even value of n is used. The result is always exact.

    sin(x, /)
        Return the sine of x (measured in radians).

    sinh(x, /)
        Return the hyperbolic sine of x.

    sqrt(x, /)
        Return the square root of x.

    tan(x, /)
        Return the tangent of x (measured in radians).

    tanh(x, /)
        Return the hyperbolic tangent of x.

    trunc(x, /)
        Truncates the Real x to the nearest Integral toward 0.

        Uses the __trunc__ magic method.

DATA
    e = 2.718281828459045
    inf = inf
    nan = nan
    pi = 3.141592653589793
    tau = 6.283185307179586

FILE
    (built-in)
```

In [7]:

```
'''
You can use the help() function on modules you created.
'''
import my_arithmetic
```

```
help(my_arithmetic)
```

```
Help on module my_arithmetic:

NAME
    my_arithmetic

FUNCTIONS
    add_two(i1, i2)
        This function adds two integers.

        This function accepts two integers as input and returns their sum.

    divide_two(i1, i2)
        This function divides one integer by another.

        This function accepts two integers as input and returns the quotient.

    minus_two(i1, i2)
        This function subtracts one integer from the other.

        This function accepts two integers as input and returns their difference.

    multiply_two(i1, i2)
        This function multiplies two integers.

        This function accepts two integers as input and returns their product.

FILE
    c:\users\owner\desktop\utd\summer2020\mis 6382\material\rabih\lecture2\my_arithmetic.py
```