Physics Without Frontiers

Practical Programming
in Python

Inspired by 'Practical Programming' by Paul Gries, Jennifer Campbell, Jason Montojo

Lecture 6: Summary & Exercises
Program Organization

Modules, Name Spaces, Main Programs, Libraries

"You can't write perfect software."
– Andrew Hunt, The Pragmatic Programmer

Lecture 6: Summary

In this lecture you learned the following:

- A module is a collection of functions and variables (in general objects) grouped together in a file. To use a module, you must first import it using `import module_name`. After it has been imported, you refer to its contents using `module_name.function_name` or `module_name.variable`.
- Variable `__name__` is created by Python and can be used to specify that some code should only run when the module is run directly and not when the module is imported.
- Programs have to do more than just run to be useful; they have to run correctly. One way to ensure that they do is to test them, which you can do in Python using module `doctest`

Lecture 6: Exercises

When writing code, only use Python concepts that have been introduced in the lectures already.

Exercise 1:

Import module math, and use its functions to complete the following exercises. (You can call `dir(math)` to get a listing of the items in math.)

a. Write an expression that produces the floor of -2.8.

b. Write an expression that rounds the value of -4.3 and then produces the absolute value of that result.

c. Write an expression that produces the ceiling of the sine of 34.5.

Exercise 2:

In the following exercises, you will work with Python's `calendar` module:

a. Visit the Python documentation website at https://docs.python.org/release/3.6.6/py-modindex.html, and look at the documentation on module `calendar`.

b. Import module `calendar`.

c. Using function `help`, read the description of function `isleap`.

d. Use `isleap` to determine the next leap year.

e. Use `dir` to get a list of what `calendar` contains.

f. Find and use a function in module `calendar` to determine how many leap years there will be between the years 2000 and 2050, inclusive.

g. Find and use a function in module `calendar` to determine which day of the week July 29, 2019, will be.

Exercise 3:

Create a file named exercise.py with this code inside it:

```
def average(num1, num2):
    """
    Return the average of num1 and num2.

    Examples:

        >>> average(10,20)
        15.0
        >>> average(2.5, 3.0)
        2.75
    """
    return num1 + num2 / 2
```

a. Run `exercise.py`. Import `doctest` and run `doctest.testmod()`.

b. Both of the tests in function `average` docstring fail. Fix the code and rerun the tests. Repeat this procedure until the tests pass.