5. Python Modules

5.1. Modules and Getting Help

5.2. More About Using Modules

5.3. The math module

5.4. The random module

5.6. Glossary

5.7. Exercises

5.1. Modules and Getting Help

A module is a file containing Python definitions and statements intended for use in other Python programs. There are many Python modules that come with Python as part of the standard library. We have already used one of these quite extensively, the turtle module. Recall that once we import the module, we can use things that are defined inside.

1 import turtle # allows us to use the turtles library

2

​3 wn = turtle.Screen() # creates a graphics window

4 alex = turtle.Turtle() # create a turtle named alex

5

​6 alex.forward(150) # tell alex to move forward by 150 units

7 alex.left(90) # turn by 90 degrees

8 alex.forward(75) # complete the second side of a rectangle

9 wn.exitonclick()

10

​

Here we are using Screen and Turtle, both of which are defined inside the turtle module.

But what if no one had told us about turtle? How would we know that it exists. How would we know what it can do for us? The answer is to ask for help and the best place to get help about the Python programming environment is to consult with the Python Documentation.

The Python Documentation site for Python version 3 (the home page is shown below) is an extremely useful reference for all aspects of Python. The site contains a listing of all the standard modules that are available with Python (see Global Module Index). You will also see that there is a Language Reference and a Tutorial (mostly aimed at people who are already familiar with another programming language), as well as installation instructions, how-tos, and frequently asked questions. We encourage you to become familiar with this site and to use it often.

Check your understanding

modules-1-3: In Python a module is:

A. A file containing Python definitions and statements intended for use in other Python programs.

B. A separate block of code within a program.

C. One line of code in a program.

D. A file that contains documentation about functions in Python.

modules-1-4: To find out information on the standard modules available with Python you should:

A. Go to the Python Documentation site.

B. Look at the import statements of the program you are working with or writing.

C. Ask the professor

D. Look in this textbook.

modules-1-5: True / False: All standard Python modules will work in activecode.

A. True

B. False

5.2. More About Using Modules

Before we move on to exploring other modules, we should say a bit more about what modules are and how we typically use them. One of the most important things to realize about modules is the fact that they are data objects, just like any other data in Python. Module objects simply contain other Python elements.

The first thing we need to do when we wish to use a module is perform an import. In the example above, the statement import turtle creates a new name, turtle, and makes it refer to a module object. This looks very much like the reference diagrams we saw earlier for simple variables.

In order to use something contained in a module, we use the dot notation, providing the module name and the specific item joined together with a “dot”. For example, to use the Turtle class, we say turtle.Turtle. You should read this as: “In the module turtle, access the Python element called Turtle”.

We will now turn our attention to a few other modules that you might find useful.

5.3. The math module

The math module contains the kinds of mathematical functions you would typically find on your calculator and some mathematical constants like pi and e. As we noted above, when we import math, we create a reference to a module object that contains these elements.

Here are some items from the math module in action. If you want more information, you can check out the Math Module Python Documentation.

1 import math

2

​3 print(math.pi)

4 print(math.e)

5

​6 print(math.sqrt(2.0))

7

​8 print(math.sin(math.radians(90))) # sin of 90 degrees

9

Notice another difference between this module and our use of turtle. In turtle we create objects (either Turtle or Screen) and call methods on those objects. Remember that a turtle is a data object (recall alex and tess). We need to create one in order to use it. When we say alex = turtle.Turtle(), we are calling the constructor for the Turtle class which returns a single turtle object.

Mathematical functions do not need to be constructed. They simply perform a task. They are all housed together in a module called math. Once we have imported the math module, anything defined there can be used in our program. Notice that we always use the name of the module followed by a dot followed by the specific item from the module (math.sqrt). You can think of this as lastname.firstname where the lastname is the module family and the firstname is the individual entry in the module.

If you have not done so already, take a look at the documentation for the math module.

Check your understanding

modules-3-2: Which statement allows you to use the math module in your program?

A. import math

B. include math

C. use math

D. You don't need a statement. You can always use the math module

5.4. The random module

We often want to use random numbers in programs. Here are a few typical uses:

To play a game of chance where the computer needs to throw some dice, pick a number, or flip a coin,

To shuffle a deck of playing cards randomly,

To randomly allow a new enemy spaceship to appear and shoot at you,

To simulate possible rainfall when we make a computerized model for estimating the environmental impact of building a dam,

For encrypting your banking session on the Internet.

Python provides a module random that helps with tasks like this. You can take a look at it in the documentation. Here are the key things we can do with it.

1 import random

2

​3 prob = random.random()

4 print(prob)

5

​6 diceThrow = random.randrange(1, 7) # return an int, one of 1,2,3,4,5,6

7 print(diceThrow)

8

Press the run button a number of times. Note that the values change each time. These are random numbers.

The randrange function generates an integer between its lower and upper argument, using the same semantics as range — so the lower bound is included, but the upper bound is excluded. All the values have an equal probability of occurring (i.e. the results are uniformly distributed).

The random() function returns a floating point number in the range [0.0, 1.0) — the square bracket means “closed interval on the left” and the round parenthesis means “open interval on the right”. In other words, 0.0 is possible, but all returned numbers will be strictly less than 1.0. It is usual to scale the results after calling this method, to get them into a range suitable for your application.

In the case shown here, we’ve converted the result of the method call to a number in the range [0.0, 5.0). Once more, these are uniformly distributed numbers — numbers close to 0 are just as likely to occur as numbers close to 0.5, or numbers close to 1.0. If you continue to press the run button you will see random values between 0.0 and up to but not including 5.0.

1 import random

2

​3 prob = random.random()

4 result = prob \* 5

5 print(result)

6

​

It is important to note that random number generators are based on a deterministic algorithm — repeatable and predictable. So they’re called pseudo-random generators — they are not genuinely random. They start with a seed value. Each time you ask for another random number, you’ll get one based on the current seed attribute, and the state of the seed (which is one of the attributes of the generator) will be updated. The good news is that each time you run your program, the seed value is likely to be different meaning that even though the random numbers are being created algorithmically, you will likely get random behavior each time you execute.

Check your understanding

modules-4-3: Which of the following is the correct way to reference the value pi within the math module. Assume you have already imported the math module.

A. math.pi

B. math(pi)

C. pi.math

D. math->pi

modules-4-4: Which module would you most likely use if you were writing a function to simulate rolling dice?

A. the math module

B. the random module

C. the turtle module

D. the game module

modules-4-5: The correct code to generate a random number between 1 and 100 (inclusive) is:

A. prob = random.randrange(1, 101)

B. prob = random.randrange(1, 100)

C. prob = random.randrange(0, 101)

D. prob = random.randrange(0, 100)

modules-4-6: One reason that lotteries don’t use computers to generate random numbers is:

A. There is no computer on the stage for the drawing.

B. Because computers don't really generate random numbers, they generate pseudo-random numbers.

C. They would just generate the same numbers over and over again.

D. The computer can't tell what values were already selected, so it might generate all 5's instead of 5 unique numbers.

5.5. Creating Modules

You’ve seen how to use modules like random, math, and turtle, but how would you create a module?

Every time you’ve written a Python script you’ve created a module!

A Python module is just a Python source code file. Let’s consider the Python file shown below.

coffee\_shop.py

"""

The coffee shop module contains functions and contains variables

important to implementing a coffee shop.

"""

# Set some variables

shop\_name = "Runestone Brew House"

coffee\_sizes = ["small", "medium", "large"]

coffee\_roasts = ["hot chocolate", "light", "medium", "dark", "espresso"]

This is a Python script named coffee\_shop.py that contains three variables: shop\_name, coffee\_sizes, and coffee\_roasts. The shop\_name is a string, coffee\_sizes is a list containing strings, and coffee\_roasts is also a list containing strings.

modules-5-1: A module is another name for:

A. the code inside a function

B. a file containing Python code

C. the comments before a function

D. a small block of Python code

✔️ Python modules are just Python source code files.

Activity: 5.5.1 Multiple Choice (question4\_5\_1)

That’s so great! We’ve got the basics of a coffee shop. All you need is some roasted coffee and cups. You’re good to go.

If you try to run that code though, it doesn’t do much that’s visible to a user…

How can we use the coffee\_shop module? We can import it and use it in other Python source code files. Let’s consider the Python file shown below.

coffee\_customer.py

import coffee\_shop

# Output the information we know from the module

print("Welcome to", coffee\_shop.shop\_name)

print("Available sizes:", coffee\_shop.coffee\_sizes)

print("Available roasts:", coffee\_shop.coffee\_roasts)

This is a Python script named coffee\_customer.py that imports our coffee\_shop module, then prints out the information from that module.

Note

The module files must be in the same directory on your computer for Python to know how to import them automatically

5.6. Glossary

deterministic

A process that is repeatable and predictable.

documentation

A place where you can go to get detailed information about aspects of your programming language.

module

A file containing Python definitions and statements intended for use in other Python programs. The contents of a module are made available to the other program by using the import statement.

pseudo-random number

A number that is not genuinely random but is instead created algorithmically.

random number

A number that is generated in such a way as to exhibit statistical randomness.

random number generator

A function that will provide you with random numbers, usually between 0 and 1.

standard library

A collection of modules that are part of the normal installation of Python.

5.7. Exercises

Use a for statement to print 10 random numbers.

Repeat the above exercise but this time print 10 random numbers between 25 and 35, inclusive.

The Pythagorean Theorem tells us that the length of the hypotenuse of a right triangle is related to the lengths of the other two sides. Look through the math module and see if you can find a function that will compute this relationship for you. Once you find it, write a short program to try it out.

Search on the internet for a way to calculate an approximation for pi. There are many that use simple arithmetic. Write a program to compute the approximation and then print that value as well as the value of math.pi from the math module.