KEY_Lesson09_Packages

August 20, 2019

1 Packages

Remember: - Functions and methods are things that take an input, do something with that input, and then spit out an output. You've already been using functions and methods! - **Functions** can take lots of different object types as input. For example, `print` can take a variety of inputs including a variable, a string, a list, or numbers. - **Methods** can only be used on one specific type of object. For example, `append` can only be used on lists (and not strings or numbers). - The input to **functions** goes *between* parentheses after the name of function - Example: `print(print_input)` At least part of the input to **methods** comes *before* the name of the method and is followed by a period (`.`) - Example: `mylist.append(what_to_append)`

Python comes with certain built-in functions and methods like the ones we used in the previous section (such as `print` and `append`). But sometimes you might want to do things that are a bit more complicated. For this, you can import **packages**.

We're going to use our numbers list to play around with functions in the **`numpy`** package.

[1]:
```
# list of numbers
numbers = [10,2.3,-4,20,14,1,2,0,-3,1,-2,2,2,65.4,3,-23,123,43.1,32,57,32]

# print numbers
print(numbers)
```

```
[10, 2.3, -4, 20, 14, 1, 2, 0, -3, 1, -2, 2, 2, 65.4, 3, -23, 123, 43.1, 32, 57,
32]
```

Packages are groups of functions and methods that you can **`import`** and then use just like built-in functions and packages.

One example of a very powerful packages is **`numpy`**. To import numpy and load all of the functions in that packages into your environment (in this Jupyter Notebook), you do this:

[0]:
```
# import numpy package
import numpy
```

Then you can use the functions that are defined in this package. One function is the `mean` function to find the mean of a list of numbers. To do this, you have to use the prefix `numpy` so Python knows that the function comes from the numpy package:

[3]:
```
# get the mean of numbers using the numpy mean function
numpy.mean(numbers)
```

[3]:
```
17.99047619047619
```

Nice, it's the same as we got before when we calculated the mean ourselves!

Typing out numpy every time is a lot of work, so a lot of people like to shorten the prefix you have to use by giving the package a **nickname**. If you want the prefix to be `np` instead of `numpy`, you can import the package as follows:

[0]:
```
# import numpy package as np
import numpy as np
```

Now, if you want to find the mean of numbers using the numpy mean function, you just have to use the prefix `np`:

[5]:
```
# print the mean of numbers
np.mean(numbers)
```

[5]:
```
17.99047619047619
```

Let's explore some other functions in the `numpy` package.

Before, we learned how to find the absolute value of a number using the `abs` function. If we want to get the absolute value of everything in a list, we can use the numpy absolute value function by calling `np.abs`:

[6]:
```
# get absolute value of all numbers in numbers and save it to the variable␣
↪abs_nums
abs_nums = np.abs(numbers)

# print abs_nums
abs_nums
```

[6]:
```
array([ 10. ,   2.3,   4. ,  20. ,  14. ,   1. ,   2. ,   0. ,   3. ,
         1. ,   2. ,   2. ,   2. ,  65.4,   3. ,  23. , 123. ,  43.1,
        32. ,  57. ,  32. ])
```

Did you notice that this looks different than a list? Let's look at the type of `abs_nums`:

[7]:
```
# get the type of abs_nums
print(type(abs_nums))
```

```
<class 'numpy.ndarray'>
```

It's a numpy.ndarray. You'll learn more about what that means in a future lesson!

numpy also has a similar function for `round`.

Okay, we're going to play around with methods a bit more. One method that is useful is `sort`. You can use this to sort a list or a numpy array. It won't print anything out because it sorts the actual numbers list. Let's try this on the numbers list:

[8]:
```
# print numbers
print(numbers)
# sort numbers
numbers.sort()
# print numbers
print(numbers)
```

```
[10, 2.3, -4, 20, 14, 1, 2, 0, -3, 1, -2, 2, 2, 65.4, 3, -23, 123, 43.1, 32, 57,
32]
[-23, -4, -3, -2, 0, 1, 1, 2, 2, 2, 2.3, 3, 10, 14, 20, 32, 32, 43.1, 57, 65.4,
123]
```

As you can see, the `sort` method operated on the `numbers` list variable and changed the actual list to be sorted from the lowest value (-23) to the highest value (123). It even worked with our list of mixed float and integer values.

Now let's try it on the `abs_nums` list:

[9]:
```
# print abs_nums
print(abs_nums)
# sort abs_nums
abs_nums.sort()
# print abs_nums
print(abs_nums)
```

```
[ 10.    2.3   4.   20.   14.    1.    2.    0.    3.    1.    2.    2.
   2.   65.4   3.   23.  123.   43.1  32.   57.   32. ]
[  0.    1.    1.    2.    2.    2.    2.    2.3   3.    3.    4.   10.
  14.   20.   23.   32.   32.   43.1  57.   65.4 123. ]
```

Nice job! You just learned about packages in Python! You learned: - How to import new functions in packages in Python - More functions and methods

You'll learn more about numpy in a future lesson. There are also tons of other packages out there for different purposes. We'll learn about the `pandas` package in the next lesson, and the `matplotlib` package in a few days!