Exercise 19: Functions and Variables

Functions may have been a mind-blowing amount of information, but do not worry. Just keep doing these exercises and going through your checklist from the last exercise and you will eventually get it.

There is one tiny point though that you might not have realized, which we'll reinforce right now. The variables in your function are not connected to the variables in your script. Here's an exercise to get you thinking about this:

```python
def cheese_and_crackers(cheese_count, boxes_of_crackers):
    print "You have %d cheeses!" % cheese_count
    print "You have %d boxes of crackers!" % boxes_of_crackers
    print "Man that's enough for a party!"
    print "Get a blanket.\n"


print "We can just give the function numbers directly:"
cheese_and_crackers(20, 30)


print "OR, we can use variables from our script:"
amount_of_cheese = 10
amount_of_crackers = 50

cheese_and_crackers(amount_of_cheese, amount_of_crackers)


print "We can even do math inside too:"
cheese_and_crackers(10 + 20, 5 + 6)


print "And we can combine the two, variables and math:"
cheese_and_crackers(amount_of_cheese + 100, amount_of_crackers + 1000)
```

This shows all the different ways we're able to give our function `cheese_and_crackers` the values it needs to print them. We can give it straight numbers. We can give it variables. We can give it math. We can even combine math and variables.

In a way, the arguments to a function are kind of like our `=` character when we make a variable. In fact, if you can use `=` to name something, you can usually pass it to a function as an argument.

What You Should See

You should study the output of this script and compare it with what you think you should get for each of the examples in the script.

```
$ python ex19.py
We can just give the function numbers directly:
You have 20 cheeses!
You have 30 boxes of crackers!
Man that's enough for a party!
Get a blanket.

OR, we can use variables from our script:
You have 10 cheeses!
You have 50 boxes of crackers!
Man that's enough for a party!
Get a blanket.

We can even do math inside too:
You have 30 cheeses!
You have 11 boxes of crackers!
Man that's enough for a party!
Get a blanket.

And we can combine the two, variables and math:
You have 110 cheeses!
You have 1050 boxes of crackers!
Man that's enough for a party!
Get a blanket.
```

Study Drills

1. Go back through the script and type a comment above each line

explaining in English what it does.
2. Start at the bottom and read each line backward, saying all the important characters.
3. Write at least one more function of your own design, and run it 10 different ways.

Common Student Questions

How can there possibly be 10 different ways to run a function?

Believe it or not, there's a theoretically infinite number of ways to call any function. In this case, do it like I've got with lines 8-12 and be creative.

Is there a way to analyze what this function is doing so I can understand it better?

There's many different ways, but try putting an English comment above each line describing what the line does. Another trick is to read the code out loud. Yet another is to print the code out and draw on the paper with pictures and comments showing what's going on.

What if I want to ask the user for the numbers of cheese and crackers?

Remember you just need to use `int()` to convert what you get from `raw_input()`.

Does making the variables on lines 13 and 14 change the variables in the function?

Nope, those variables are separate and live outside the function. They are then passed to the function and temporary versions are made just for the function's run. When the function exits these temporary variables go away and everything keeps working. Keep going in the book and this should become clearer.

Is it bad to have global variables (like on lines 13 and 14) with the same name as function variables?

Yes, since then you're not quite sure which one you're talking about. But sometimes necessity means you have to use the same name, or you might do it on accident. Just avoid it whenever you can.

Are lines 12-19 overwriting the function `cheese_and_crackers`?

No, not at all. It's calling them, which is basically a temporary jump to the first line of the function, then a jump back after the last line of the function has ended. It's not replacing the function with anything.

Is there a limit to the number of arguments a function can have?

It depends on the version of Python and the computer you're on, but it is fairly large. The practical limit though is about five arguments before the function becomes annoying to use.

Can you call a function within a function?

Yes, you'll make a game that does this later in the book.

Buying is easy. Just fill out the form below and we'll get started.

Full Name

Email Address

[✓] Pay With Credit Card (by Stripe™)

[ ] Use your PayPal™ account.

Buy Learn Python The Hard Way, 3rd Edition