Use parameters in functions

Previously, we defined and called our first function.
It didn't require any information from outside the function, but other functions might.
This means we need to talk about using parameters in functions.
In Python, a parameter is an object that is included in a function definition for use in that function.
Parameters are accepted into a function through the parentheses after a function name.
The function that we created in the last video isn't taking in any parameters.
Now, let's revisit another function called range() that does use parameters.
If you recall, the range() function generates a sequence of numbers from a start point to the value before the stop point.
Therefore,range() does include parameters for the start and stop indices that each accept an integer value.
For instance, it could accept integers 3 and 7.
This means the sequence it generates will run from 3 to 6.
In our previous example, we wrote a function that displayed a welcome message when someone logged in.
It would be even more welcoming if we included the employee's name with the message.
Let's define a function with a parameter so we can greet employees by name!
When we define our function, we'll include the name of the parameter that our function depends on.
We place this parameter, the name variable, inside the parentheses.
The rest of the syntax stays the same.
Now, let's go to the next line and indent so we can tell Python what we want this function to do.
We want it to print a message that welcomes the employee using the name that's passed into the function.
Bringing this variable into our print statement requires a few considerations.
Like before, we start with the welcome message we want to print.
In this case, though, we're not stopping our message after we tell them they're logged in.
We want to continue and add the employee's name to the message.
That's why we're placing a comma after "You're logged in" and then adding the name variable.
Since this is a variable and not a specific string, we don't place it in quotation marks.
Now that our function is set up, we're ready to call it with the specific argument that we want to pass in.
In Python, an argument is the data brought into a function when it is called.
For example, earlier, when we passed 3 and 7 into the range() function, these were arguments.
In our case, let's imagine we want to greet an employee named Charley Patel.
We'll call our greet_employee() function with this argument.
And when we run this, Charley Patel gets a personalized welcome message!
In this example, we only have one parameter in our function.
But we can have more.
Let's explore an example of this.
Maybe instead of a single name parameter, we have a parameter for first name and a second

parameter for last name.
If so, we would need to adjust the code like this.
First, when we define the function, we include both parameters and separate them with a comma.
Then, when we call it, we also include two arguments.
This time we're greeting someone with the first name of Kiara and with the last name of Carter.
These are also separated by a comma.
Let's run this and welcome Kiara Carter!
As we just explored, using more than one parameter just requires a few adjustments.
Great work in this video!
We learned a lot about working with parameters in a function.
This understanding is something you'll need as you continue to write Python scripts.

Revision #1
Created 2023-12-19 06:32:01 UTC by naruzkurai
Updated 2023-12-27 11:35:24 UTC by naruzkurai