Python.

First, Python is a high level language, which means it's optimized for reading by people instead of machines. It's also an interpreted language which means it isn't compiled directly to machine code and importantly is commonly used in an interactive fashion. This might be quite different if you're used to programming in something like Java or C, where you write your code, compile it, run it, and watch the output. In Python, you can start the interactive interpreter and begin writing code, line by line, with the interpreter evaluating each statement as you write it. ­­­­­­-­­­­This turns out to be very useful for tasks that require a lot of investigation, versus those that require a lot of design. Shell scripting is one example of this. And data cleaning is another. Or you can write scripts that execute programs like you might be used to another languages. We're going to use Python throw specialization in this interactive fashion.

Python is a dynamically typed language, similar to languages like JavaScript. This means that when you declare a variable, you can assign it to be an integer on one line, and a string on the next line.

We can run this cell by hitting shift+enter or by clicking on the play head icon in the tool bar. The output from the statement is immediately printed. If you were using Python in a non interactive mode, nothing would print. But since we're using it in interactive mode, we get the value immediately.

What's happening underneath is that the browser is sending your Python code across to a machine in the cloud, which executes the code in a Python three interpreter, and sends the results back.

So it's important to know that the Python interpreter is stateful. That is, that your variables exist between cells.

Okay, functions are great but they're a bit different than you might find in other languages and here are some of subtleties involved. First, since there's no typing, you don't have to set your return time. Second, you don't have to set user return statement at all actually. There's a special value called none that's returned. None a similar to null in Java and represents the absence of value.Third, in Python, you can have default values for parameters.

Here's an example. In this example, we can rewrite the add numbers function to take three parameters, but we could set the last parameter to be none by default. This means that you can call add numbers with just two values or with three, and you don't have to rewrite the function signature to overload it.

All of the optional parameters, the ones that you got default values for, need to come at the end of the function declaration. We're going to dive into strings in more detail, but print will take an item, try and convert it to a string and print the output.

Python does have classes which can have attached methods, and be instantiated as objects. Classes in Python are generally named using camel case, which means the first character of each word is capitalized.

To define a method, you just write it as you would have a function. The one change, is that to have access to the instance which a method is being invoked upon, you must include self, in the method signature.

There are a couple of implications of object-oriented programming in Python, that you should take away from this very brief example. First, objects in Python do not have private or protected members. If you instantiate an object, you have full access to any of the methods or attributes of that object. Second, there's no need for an explicit constructor when creating objects in Python. You can add a constructor if you want to by declaring the \_\_init\_\_ method.

The map function is one of the basis for functional programming in Python.Functional programming is a programming paradigm in which you explicitly declare all parameters which could change through execution of a given function. But functional programming causes one to think more heavily while chaining operations together. And this really is a sort of underlying theme in much of data science and date cleaning in particular. So, functional programming methods are often used in Python, and it's not uncommon to see a parameter for a function, be a function itself.

The map built-in function is one example of a functional programming feature of Python, that I think ties together a number of aspects of the language. The map function signature looks like this. The first parameters of function that you want executed, and the second parameter, and every following parameter, is something which can be iterated upon. But when we go to print out the map, we see that we get an odd reference value instead of a list of items that we're expecting. This is called lazy evaluation. In Python, the map function returns to you a map object. It doesn't actually try and run the function min on two items,

until you look inside for a value.

This is an interesting design pattern of the language, and

it's commonly used when dealing with big data.

This allows us to have very efficient memory management,

even though something might be computationally complex.

Lambda's are Python's way of creating anonymous functions.

These are the same as other functions, but they have no name.

The intent is that they're simple or

short lived and it's easier just to write out the function in one line

instead of going to the trouble of creating a named function.

You declare a lambda function with the word lambda followed by a list of

arguments, followed by a colon and then a single expression and this is key.

There's only one expression to be evaluated in a lambda.

The expression value is returned on execution of the lambda.

Note that you can't have default values for lambda parameters and

you can't have complex logic inside of the lambda itself

because you're limited to a single expression.

Sequences are structures that we can iterate over, and

often we create these through loops or by reading in data from a file. Python has built in support for creating these collections using a more abbreviated

syntax called list comprehensions.

Just like with lambdas, list comprehensions are a condensed format

which may offer readability and performance benefits and

you'll often find them being used in data science tutorials or on stack overflow.

Numpy is the fundamental package for numeric computing with Python.

It provides powerful ways to create store and manipulate data,

which makes it able to seamlessly and

speedily integrate with a wide variety of databases and data formats.

This is also the foundation that Pandas is built on which is a high performance

data-centric package