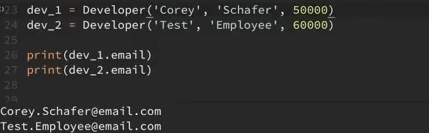
**Inheritance – Creating Subclasses**

Classes created from another class (base class) have all the attributes and the methods of the base class. In the following example we create a subclass called Developer:



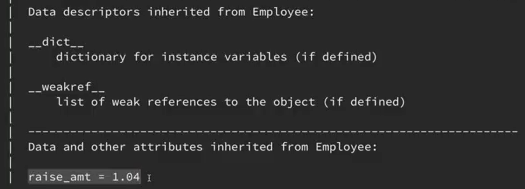
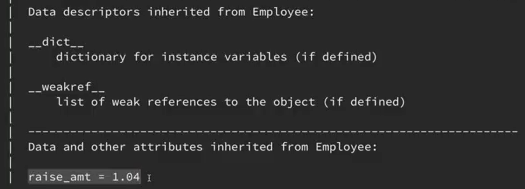
Since nothing is passes inside the class specifically, python goes up the chain of inheritance called the method resolution order, and instantiates an object based on the parent class Employee:



Let us look at the help() function:

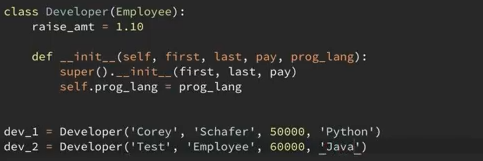


It returns the method resolution order, the chain in which python looks at the inheritance. Then it mentions the methods used and the attributes accessed also. It returns the following:



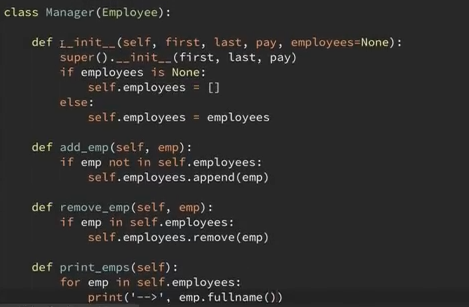
Now let consider a situation where you want to overwrite a certain attribute of the child class, create new attributes and use attributes from the parent class as well.

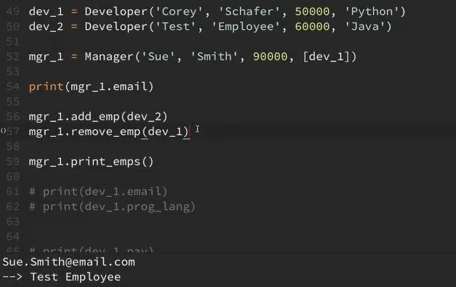
The function super() is used to allow the attributes of the parent class to be passed. Only prog\_lang is the new attribute being passed here:





Let us create another subclass Manager. Here there are a few changes, where it has methods to add and remove employees as shown:





**Built-in function: *isinstance()***

It checks whether an object is an instance of a class.



Both the above return True

But for the following:



It returns False, because class Manager is not related to class Developer.

**Built-in function: *issubclass()***

It checks whether a class is a subclass of another class:



Both the above return True.



The above line returns False.