**Name:** Thomas Caetano

**Date:** 05-12-2021

**Course:** PYTHON FUNDAMENTALS – UW (IT FDN 110 B)

GitHub Link: <https://github.com/tcaetano1970/Assignment_08>

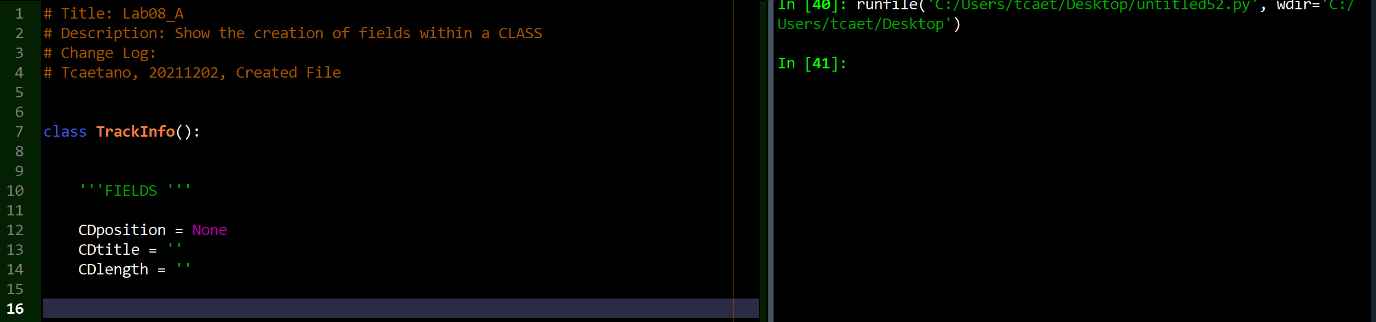
**INTRODUCTION**

Labs A through E increased in complexity in a way that is was somewhat easy for me to follow the creation of fields, contructor, attributes, properties and methods although I am still struggling with much of this. The concept of constructors makes sense to me, the assembly of code dealing with properties was more complicated but the logic is graspable.

The methods used to display data ( I wasn’t able to work the bugs) but the logic is also understandable. This particular webpage <https://vegibit.com/python-class-examples/> was helpful in how it explained the creation of objects, classes, python variables aka attributes. This page actually made me understand the code much better. The code here seemed to accomplish all the things we talked about without the use of the @staticmethods, without getters or setters – after reading this page <https://www.programiz.com/python-programming/property> everything made more sense nevertheless.

As far as the main code I tried to ensure that the parts requested by the docstrings were filled in but I am not able to tie things together yet.

LAB 08-A: Working with Classes:  
In this Lab, we’ll work with classes.  
The task / data we use: Create a class to hold information about music tracks / songs on a CD / music album. (position, title, length)  
• Create a class file, save it as Lab08\_A.py in the Mod\_08 folder.  
• Add code to create a class TrackInfo.  
• Add code to create fields for position, title, length with data types int, string, string, respectively.  
• Test the class and write down how the code works.

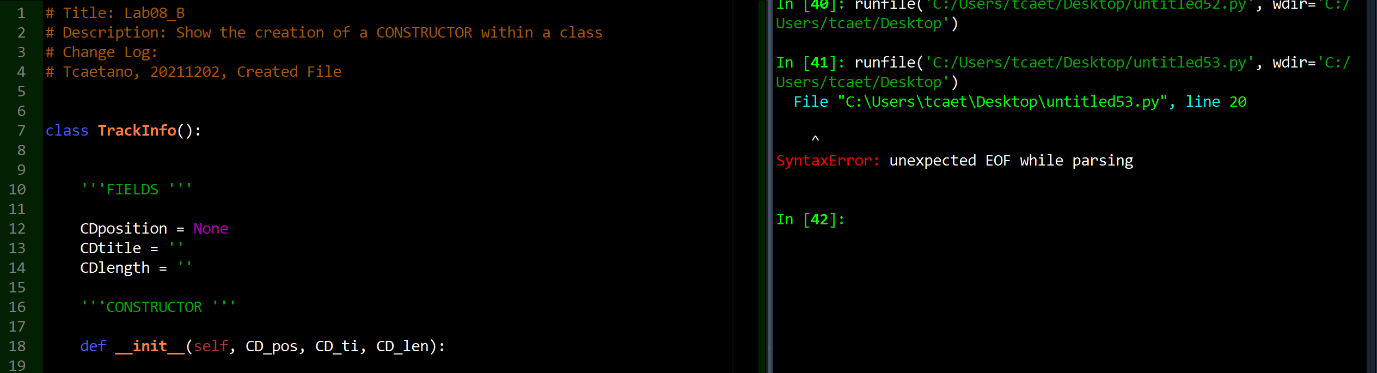


This initial code shows the creation of a class entitled TrackInfo followed by the creation of three fields: position, title and length (for a CD Inventory programme).

In python everything is an object and classes define objects.

Entering TrackInfo() in the console produces an output showing that the CLASS was successfully created by being referenced to a memory address. The fields although created only contain a None value for integers and string values for the other two.

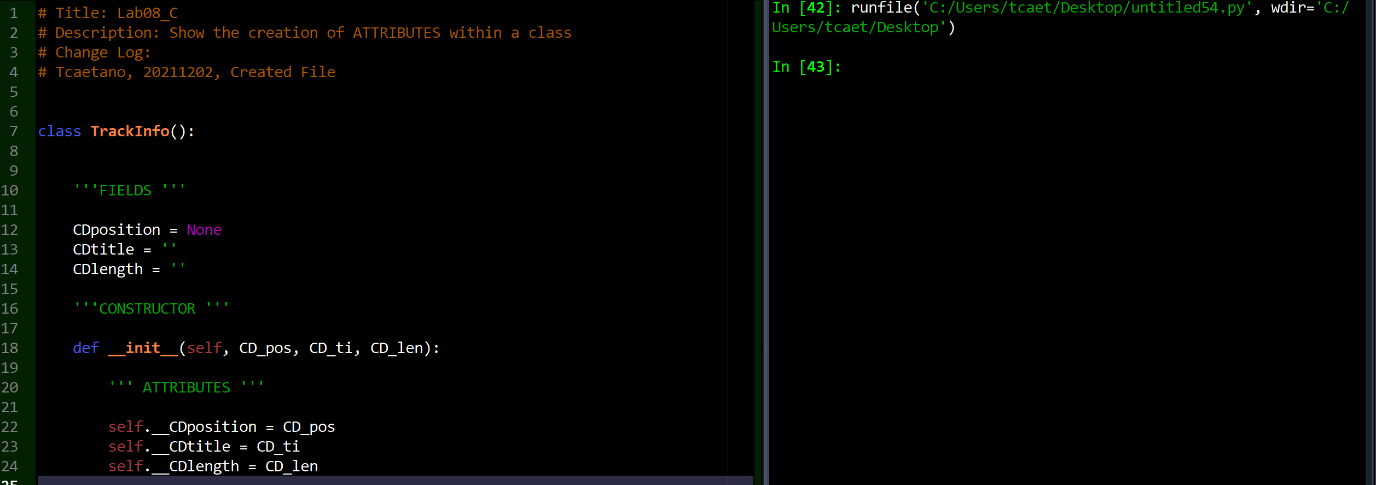
LAB 08-B: Working with Constructors:  
In this Lab, we’ll add a constructor to the class we created in Lab 08-A.  
• Make a copy of Lab08\_A.py and save it as Lab08\_B.py  
• Add code to create a constructor.  
• Add code to populate the class fields.  
• Test the class and write down how the code works.



In this example we add a constructor to line 17 using the \_\_init\_\_ method. The syntax is: \_\_init\_\_(self, …, …):

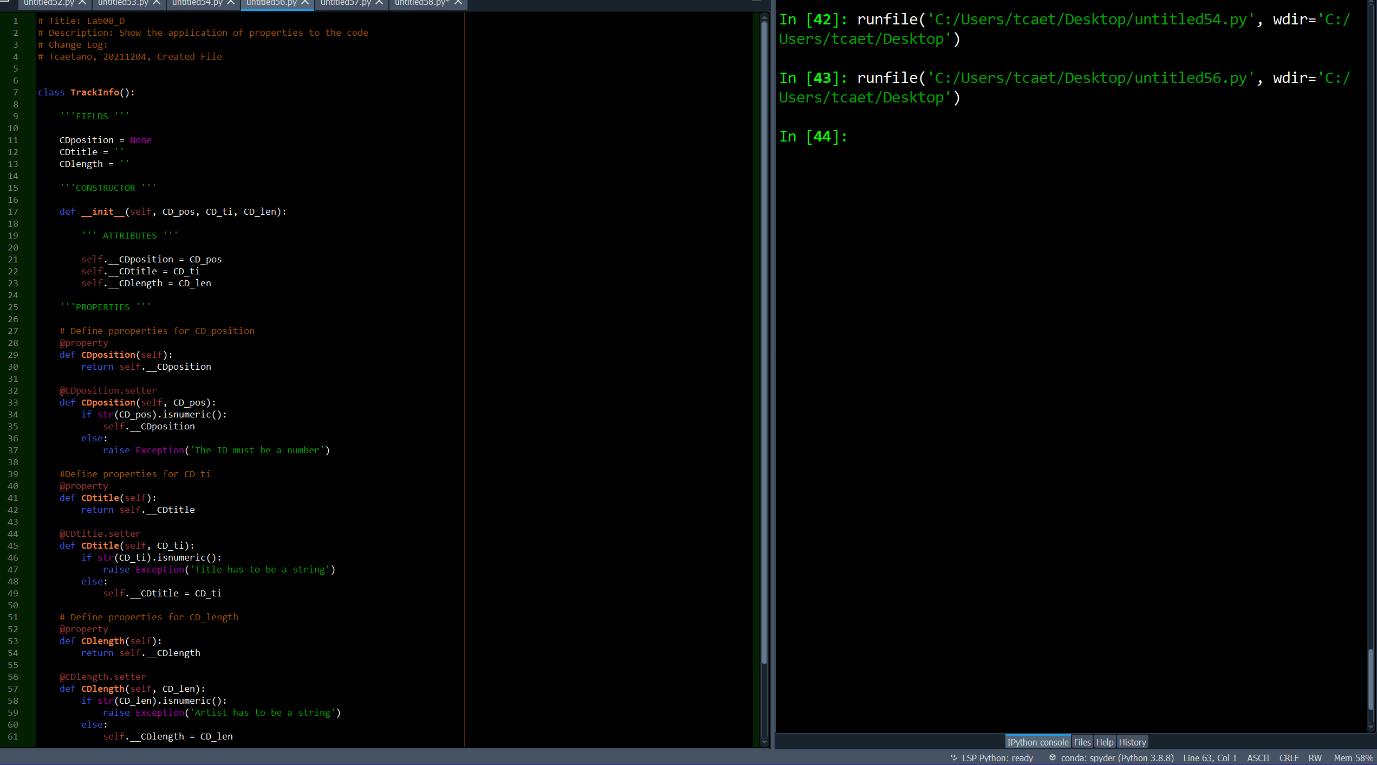
Constructors are used to instantiate an object thus they assign values to portions of the data inside the class when the object is created. The \_\_init\_\_ method is the constructor and is always called upon the creation of the object.

Lab 08-C: Working with Attributes  
In this Lab, we’ll add attributes to the class we created in Lab 08-B.  
• Make a copy of Lab08\_B.py and save it as Lab08\_C.py  
• Add code to create attributes.  
• Add code to populate the class attributes.  
• Test the class and write down how the code works.



Attributes are added to the class in this code. The variables: position, title and length will store information – these variables with the capability to store information within a class are called attributes and, in this case, given that all three were created inside the constructor not outside, will be assigned as properties of the object not the class.

LAB 08-D: Working with properties  
In this Lab, we’ll add properties to the class we created in Lab 08-C.  
• Make a copy of Lab08\_C.py and save it as Lab08\_D.py  
• Add code to create properties (setters and getters) for all three attributes.  
• Add code to verify the validity of the values.  
• Test the class and write down how the code works.

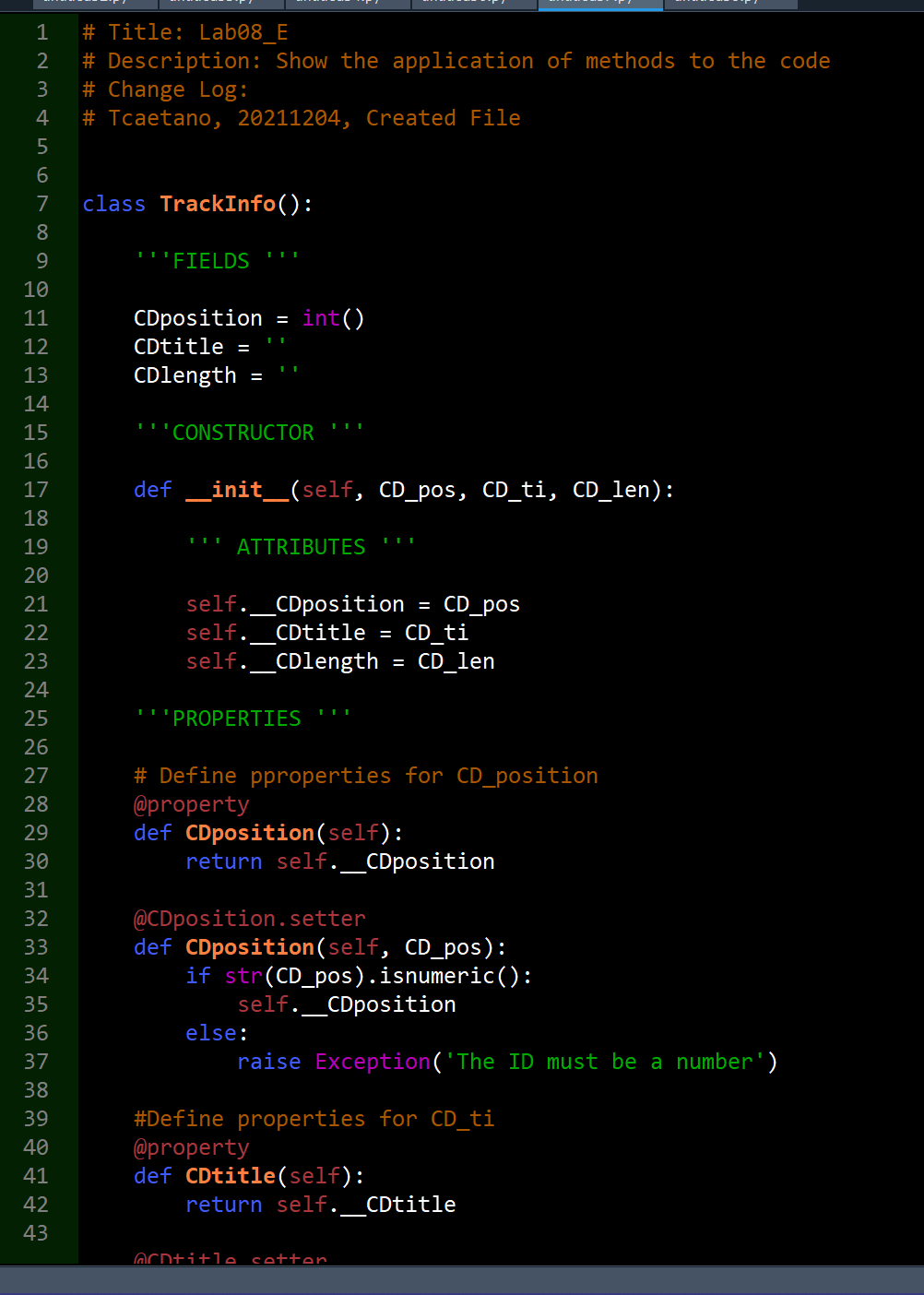


The properties added here ensure that the class has more usability. Attributes are easier to manipulate. The code provided on <https://www.programiz.com/python-programming/property> did an excellent job at explaining this as it started with very basic concepts.

LAB 08-E: Working with Methods:  
In this Lab, we’ll add methods to the class we created in Lab 08-D.

• Make a copy of Lab08\_D.py and save it as Lab08\_E.py  
• Add code to create a \_\_str\_\_ method to return a formatted content of the attributes.  
• Add code to verify the proper functioning of the method.  
• Test the class and write down how the code works.

Text

Description automatically generated

I am not sure what, Text

Description automatically generated

…actually accomplishes.

The methods below this code seemed to handle all the strings without any need for it. \_\_str\_\_ represents class objects as strings, but aren’t they all strings to begin with? The page <https://www.educative.io/edpresso/what-is-the-str-method-in-python> provided some information but I still find it hard to understand.