Final Submission Narrative:

Some notes:

1. I said I would make a normalization program to go along with this, but I think you’ll find that what I did was PLENTY for a final project. I made a simple yet robust catalog search application from scratch. That is the extent of the final project
2. Running the file:

I never figured out how to open the file in my conda environment but it’s running on the other python 3 envionment on my machine, so I’m running it from the command line. The easiest way that I figured out how to do this is:

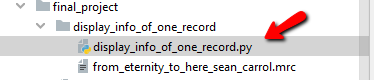
1. Open the py file in Visual Studio
2. Right-click on the script and select ‘Start without debugging’
3. This opens the file in the Windows command line. If you have a Mac, I don’t know how to run this.
4. Files that go along with this submission (obviously, the final version is final\_version.py):
   1. Every py file that I refer to in this narrative
      1. display\_info\_of\_one\_record.py
      2. display\_info\_of\_many\_records.py
      3. search\_info\_of\_many\_records.py
      4. search\_attempt\_02.py
      5. display\_found\_record.py
      6. perform\_another\_search.py
      7. final\_version.py
   2. Source data
      1. from\_eternity\_to\_here\_sean\_carrol.mrc (for practice with one record)
      2. test\_data\_7\_records.mrc (for practice with multiple records)
      3. data.mrc (for the final version of the project. Contains 50,000 records from the Boston Public Library)
   3. This narrative file davidhf2\_final\_project\_narrative.docx (48 pages)
5. Video demonstration: I’m not sure if the person grading this is going to be able to run the file. So I made a video demonstrating the functionality.

Link to video: <https://youtu.be/EZtlsTgv37c>

Displaying bib info from one marc record.

I ran pip install pycharm in my command line (forgot to take a screenshot of that)

Now I’m starting with this directory, which has my py file and a marc record:



The point of this new Py file will be to use the pymarc library to display as much bib info as I can display for one record.

I’m using the pymarc documentation available here: <https://readthedocs.org/projects/pymarc/downloads/pdf/latest/>

So to read the marc file, it looks like I need to do something like this (from the documentation):

**from pymarc import** MARCReader

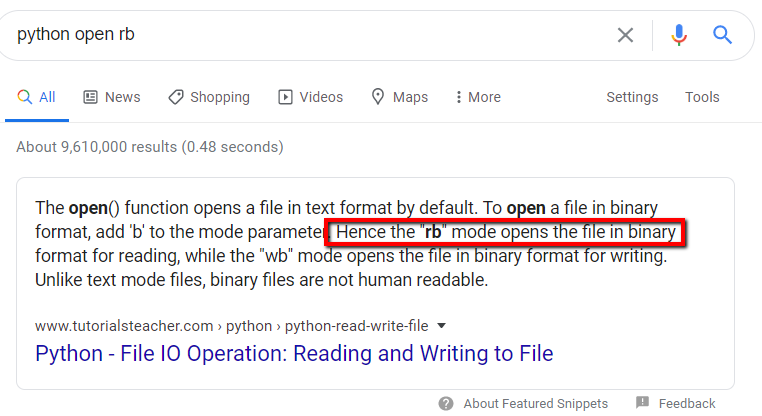
**with** open('test/marc.dat', 'rb') **as** fh:

reader = MARCReader(fh)

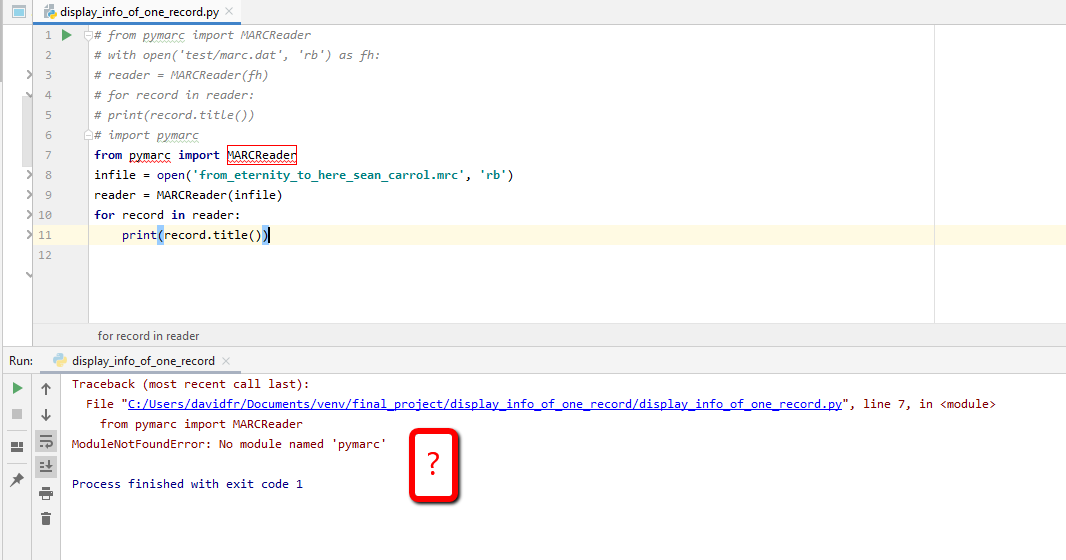
**for** record **in** reader:

**print**(record.title())

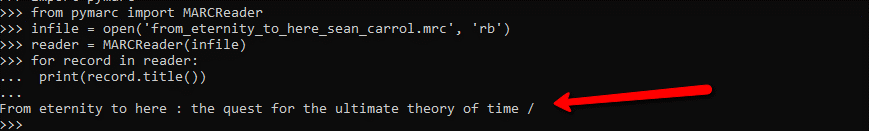
I’ll try that. But wait, what’s the rb mode?



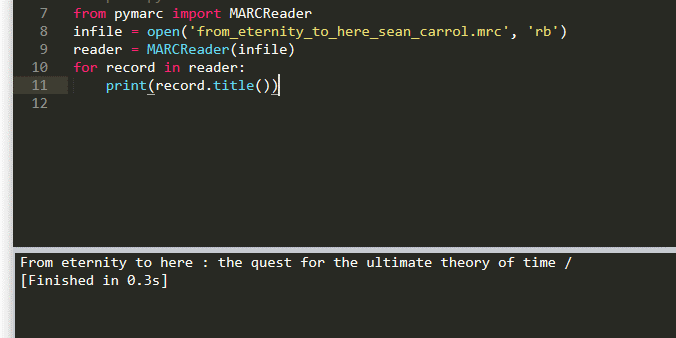
Oh, that makes sense. I’ll use that and see if it works.



I got a traceback, but I don’t understand why. It works in the console:



And the same file works in sublime:



I guess I’ll be using Sublime! No answer to why it didn’t work in PyCharm.

Moving on, let’s look at the marc file to see what we can display about this:

=LDR 01900cam a2200277 a 4500

=001 15783575

=005 20100609170620.0

=008 090622s2010\\\\nyuad\\\\b\\\\001\0\eng\\

=906 \\$a7$bcbc$corignew$d1$eecip$f20$gy-gencatlg

=925 0\$aacquire$b2 shelf copies$xpolicy default$eclaim1 2010-02-19

=955 \\$bxh00 2009-06-22$ixh07 2009-06-22 to Dewey$wrd11 2009-06-22$axe08 2010-04-20 1 copy rec'd., to CIP ver.$frg01 2010-04-22 Z-CipVer to CALM$aBarcode 00240384668 returned from bindery 2010-04-22$txg20 2010-06-09 copy 2 added

=010 \\$a 2009023828

=020 \\$a9780525951339 (hardcover)

=020 \\$a0525951334 (hardcover)

=035 \\$a(OCoLC)ocn318411645

=040 \\$aDLC$cDLC$dBTCTA$dUPZ$dJST$dC#P$dABG$dCDX$dBWX$dYDXCP$dMOF$dFER$dDLC

=050 00$aQC173.59.S65$bC37 2010

=082 00$a530.11$222

=100 1\$aCarroll, Sean M.,$d1966-

=245 10$aFrom eternity to here :$bthe quest for the ultimate theory of time /$cSean Carroll.

=260 \\$aNew York :$bDutton,$cc2010.

=300 \\$aix, 438 p. :$bill., charts ;$c24 cm.

=504 \\$aIncludes bibliographical references (p. [385]-420) and index.

=505 0\$apt. I. Time, experience, and the universe. The past is present memory ; The heavy hand of entropy ; The beginning and end of time -- pt. II. Time in Einstein's universe. Time is personal ; Time is flexible ; Looping through time -- pt. III. Entropy and time's arrow. Running time backward ; Entropy and disorder ; Information and life ; Recurrent nightmares ; Quantum time -- pt. IV. From the kitchen to the multiverse. Black holes : the ends of time ; The life of the universe ; Inflation and the multiverse ; The past through tomorrow ; Epilogue -- Appendix : Math.

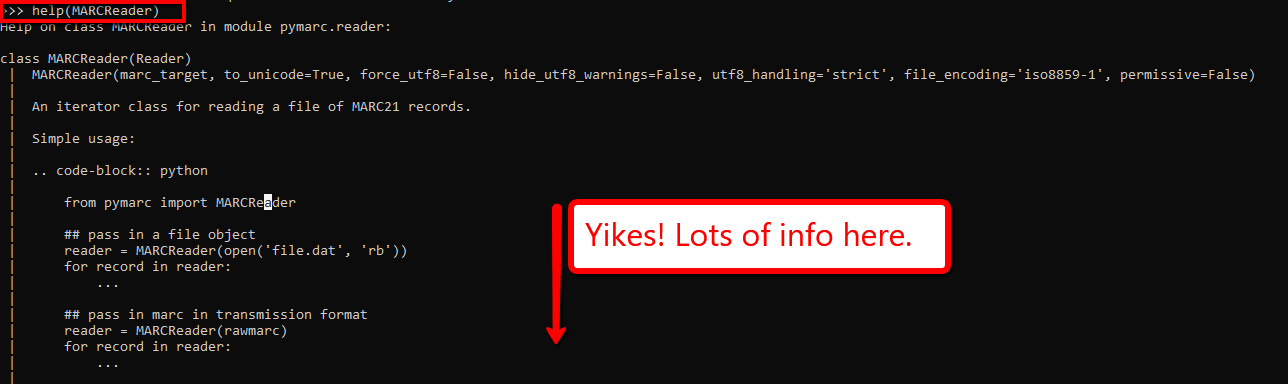
=520 \\$aA rising star in theoretical physics offers his awesome vision of our universe and beyond, all beginning with a simple question: Why does time move forward?

=650 \0$aSpace and time.

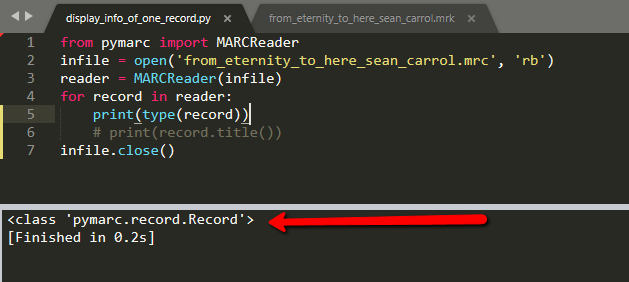
So, there’s a lot we can say! Let’s describe the record in terms of:

1. Record Number (001)
2. Main title
3. Subtitle
4. Author
5. ISBN
6. LCCN
7. OCLC number
8. LC class (I’ll just do the class and not the whole number)
9. Publication
10. Number of pages

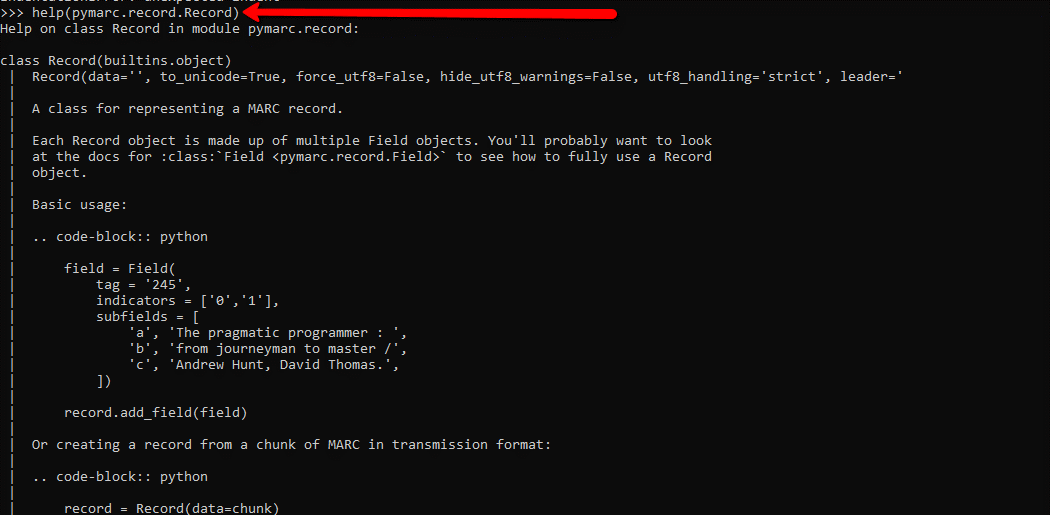
I’ll need to print stuff using the MARCReader methods. Need to find out what those are.



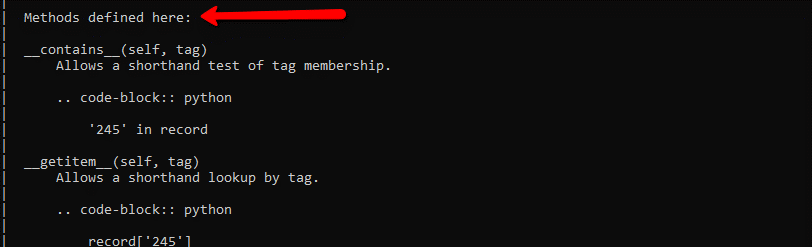
That actually didn’t help me. I didn’t find the .title() method that I saw in the example. Need to figure out the class of my variable ‘record’:



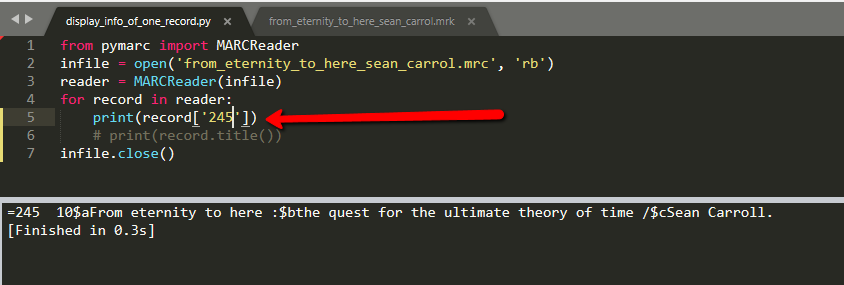
That looks weird to me. But…



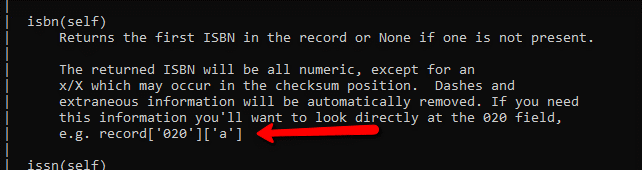
Here is where I’m going to find my methods!



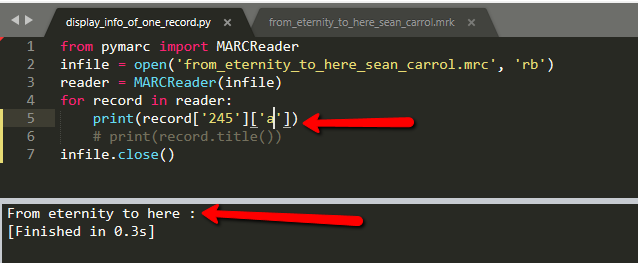
I don’t really know what \_\_getitem\_\_ means but it looks like I can use square brackets to call a marc tag as if I’m calling an index position.



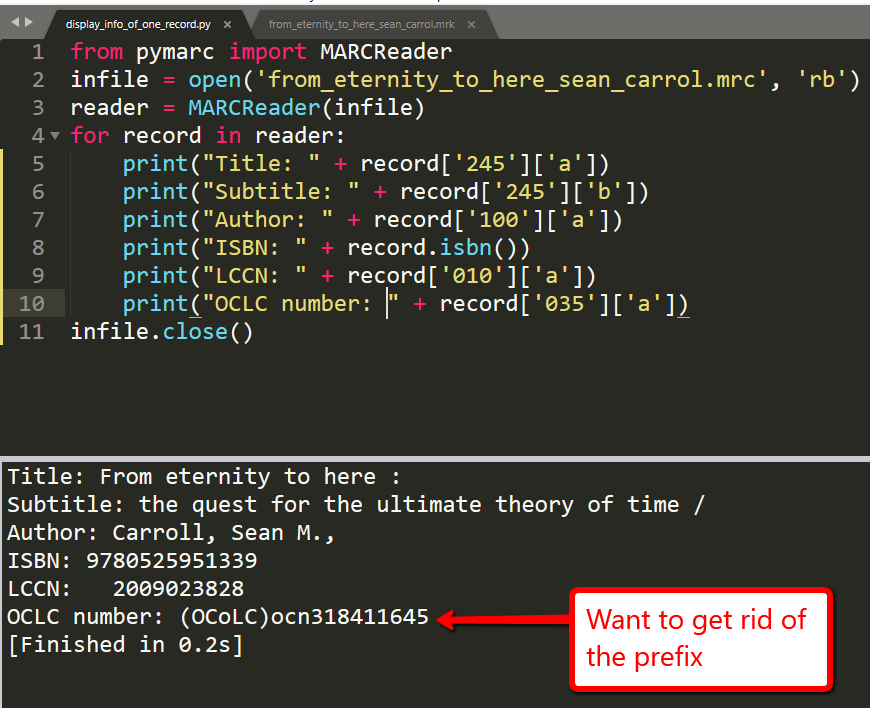
That’s handy but it’s not very clean. But I found a hint:



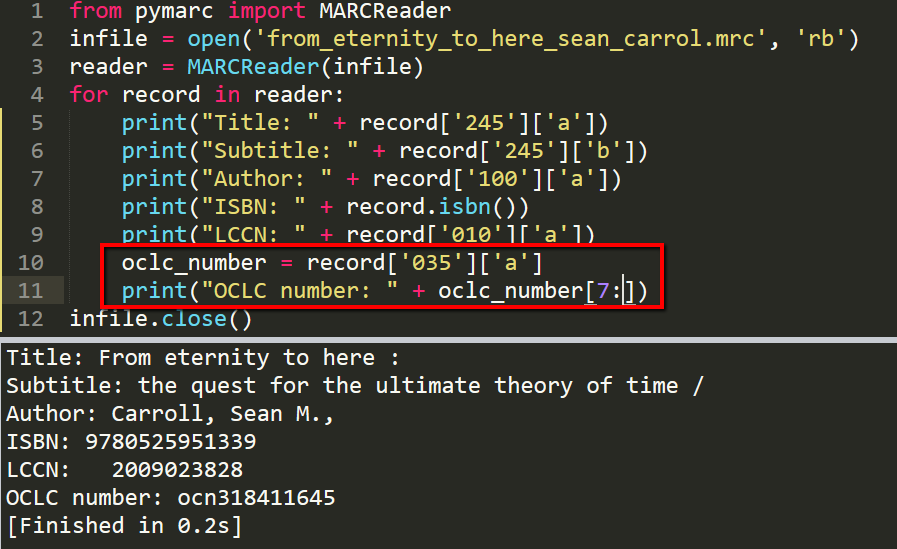
It looks like I can call the subfield by putting the subfield indicator in square brackets after the tag in brackets.



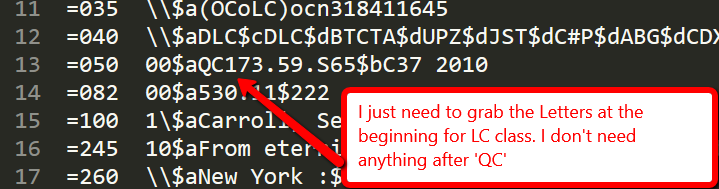
Much prettier! Now I’m going to plug everything into my script so that I can output the 10 things on my list.

I got all the way to OCLC number and I don’t like the output: 

Fixed:

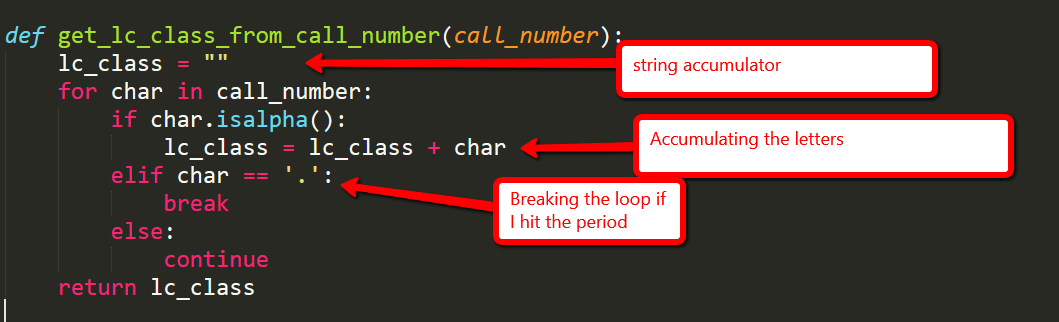


LC class is going to be a challenge:

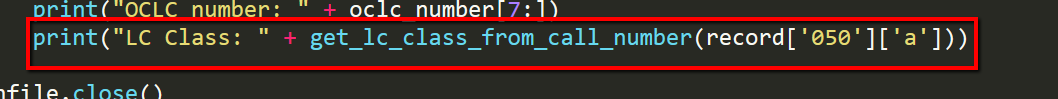


I’m going to solve this by writing a function called ‘get\_lc\_class\_from\_call\_number(call\_number)’

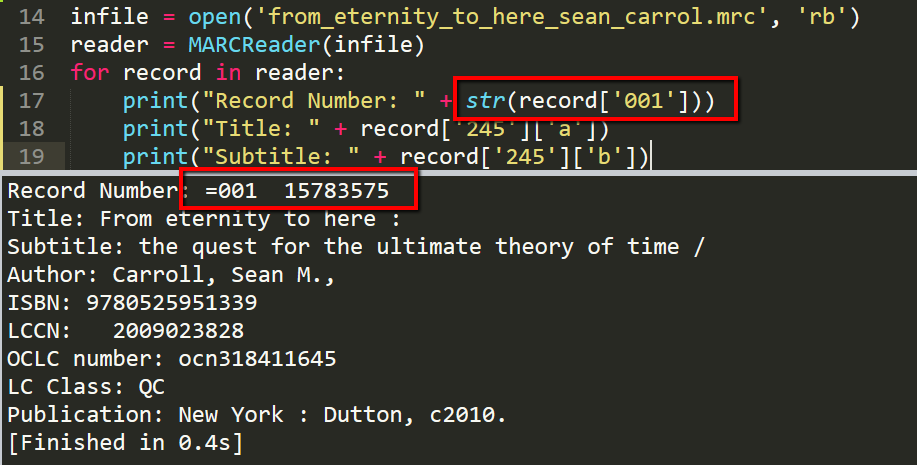
That worked!



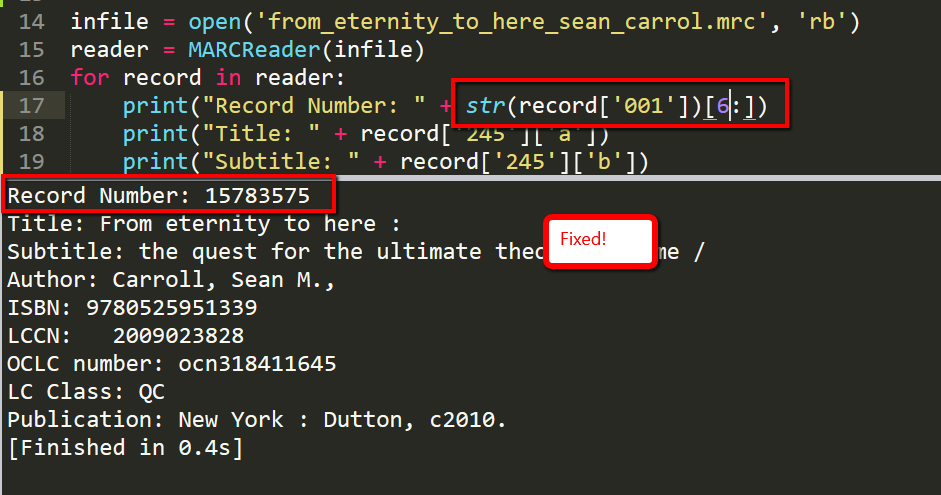
Here’s where I call it:



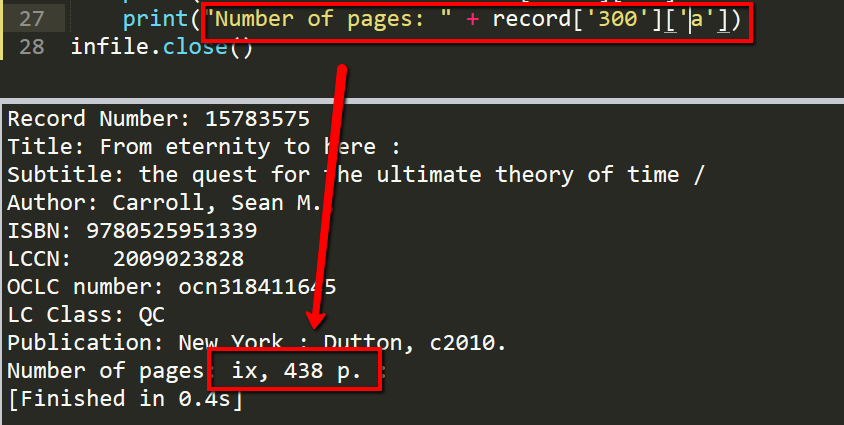
I just decided I want to add ‘Record Number’ at the top. If I use record['001'], I get this:



So I need to figure out how I’m going to get rid of the =001, but I think I know how. I’m just going to slice it.

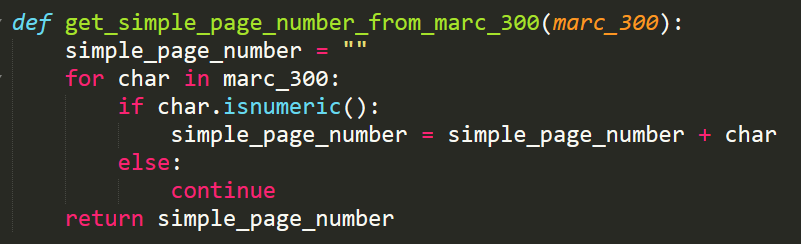


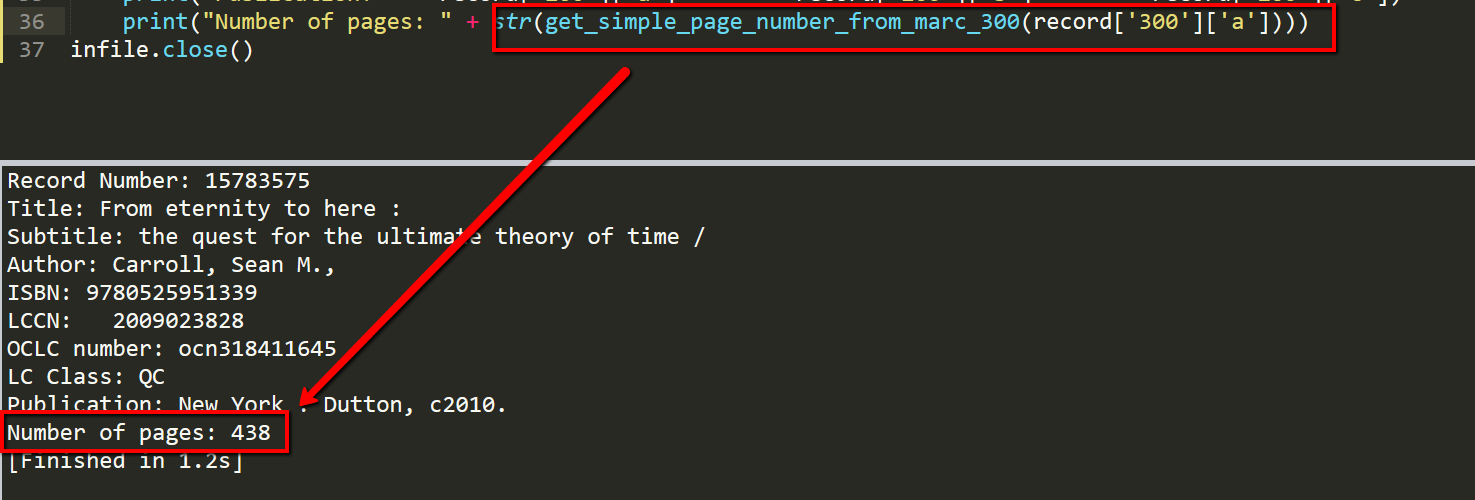
I don’t like how my “number of pages” turned out:



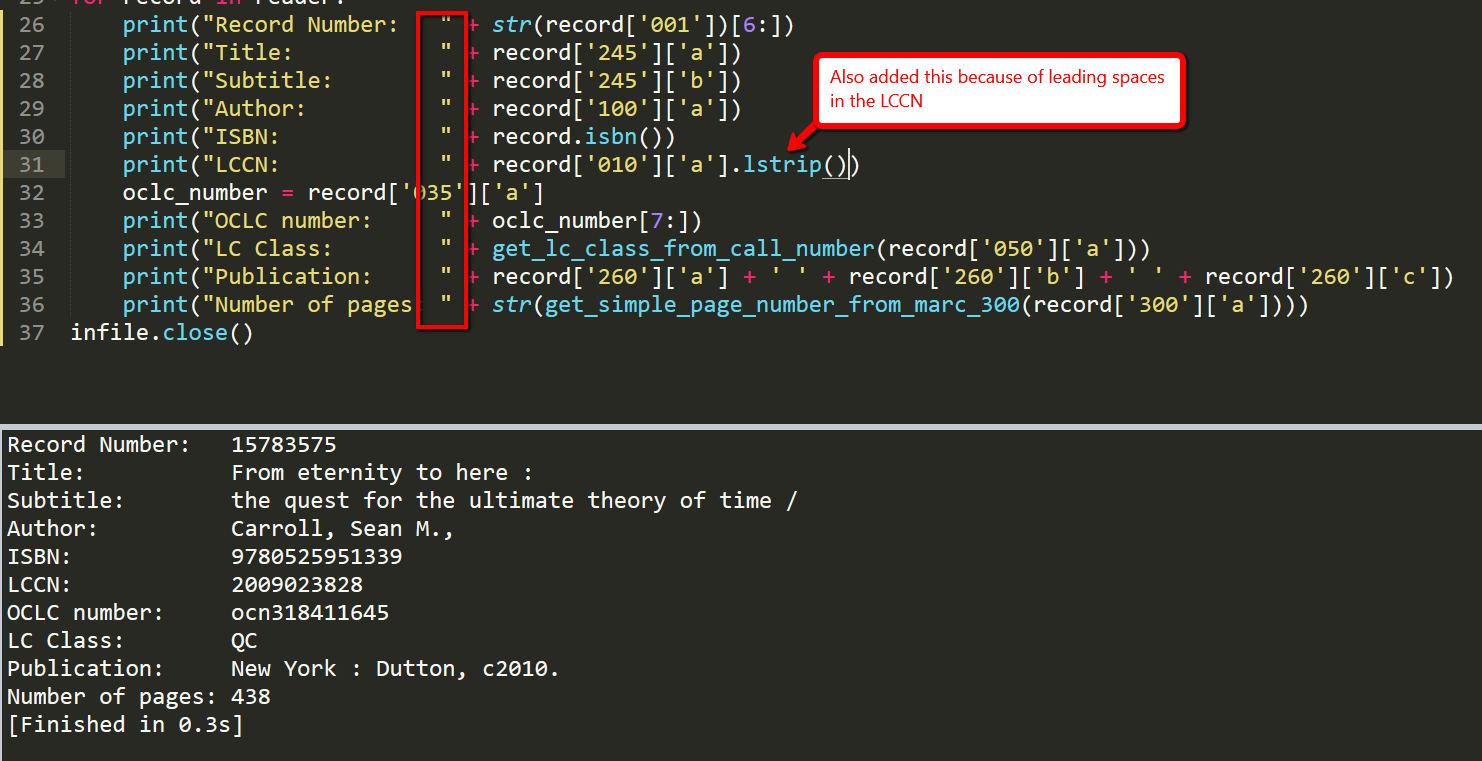
I know that the book starts with ‘ix’ pages in the first pagination and 438 in the second pagination, but I want a simpler value here. I just want 438. I’m going to write a function called get\_simple\_page\_number\_from\_marc\_300(marc\_300).

That worked!





Now I have a problem with the output. I want to indent everything, so it looks nicer.



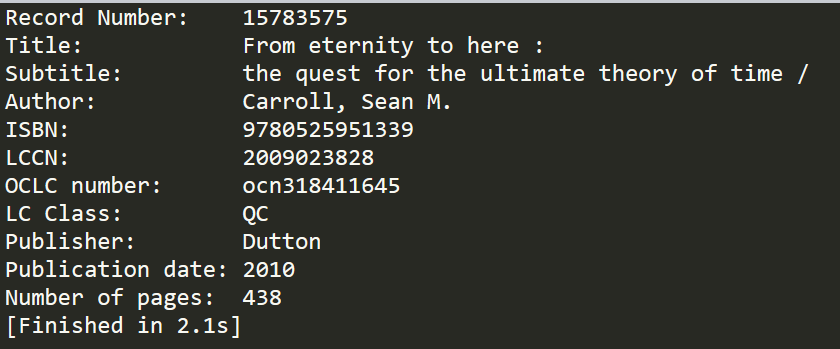
Next Steps

I want this to be scalable and flexible. My next step is to create an application that can search a large file containing many records. I will stick with monographs only, though. I’ll need to:

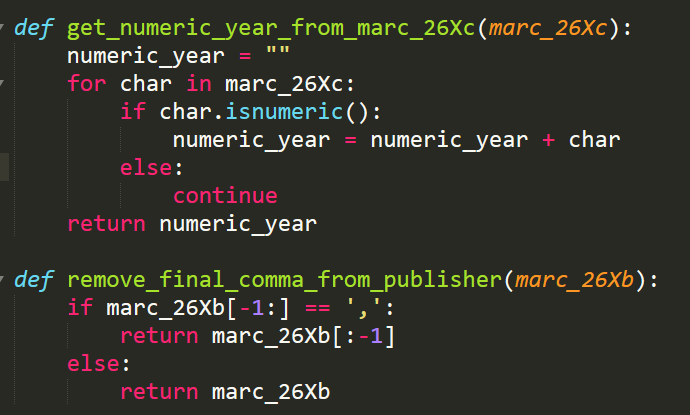
1. Convert each output into a dictionary of key value pairs instead of the display I produced in the script ouput:
2. Add some exception handling if the records lack the data that I’m trying to grab from them
3. Do something fancy with the Publication info so that the program will grab it from the 264 if it exists there instead of or in addition to the 260.
4. Create the keyword search functionality from scratch!
5. Create the normalization program from scratch! It will be a very simple normalization. I think I’m going to write something where it just adds a local field to each record. Maybe I’ll also do one that makes an update to an author’s name or a simple update like that.

I have already made a few changes to the script display\_info\_of\_one\_record.py.

I have a Publisher and Publication date:

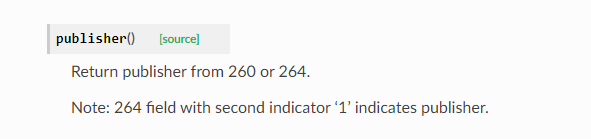


I had to write a few more functions to clean up the 26X data.

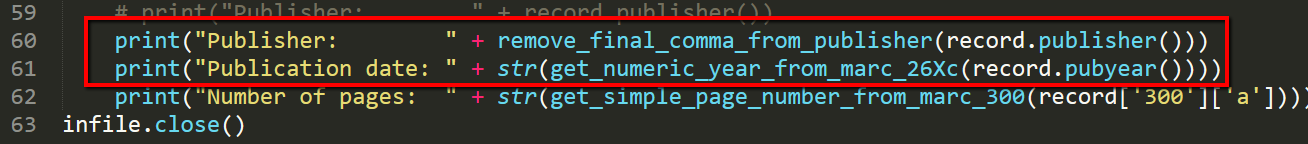


I also figured out how to grab publisher data and have the program dynamically take it from either the 260 or the 264 field depending on what the record has. There’s a pymarc method that does that.

From the web-version of the pymarc documentation:



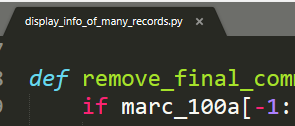
I used it in my code here:



Next I need to:

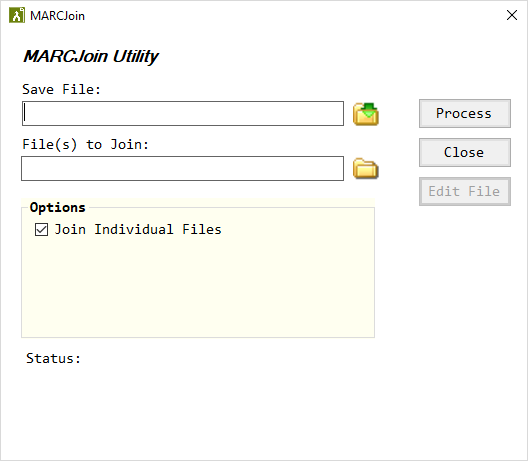
1. Add a bunch more records to my initial file.
2. Make sure that everything works (Need to add some exception handling)
3. Forward engineer the current script into something that stores the outputs as a list of dictionaries instead of just the pretty display.

I’m going to copy my script and rename it ‘display\_info\_of\_many\_records.py’. The previous one ‘display\_info\_of\_one\_record.py’ is done for now.



Now to add records to my file.

I’m going to attempt to do this in Marc Edit using the MARCjoin utility:

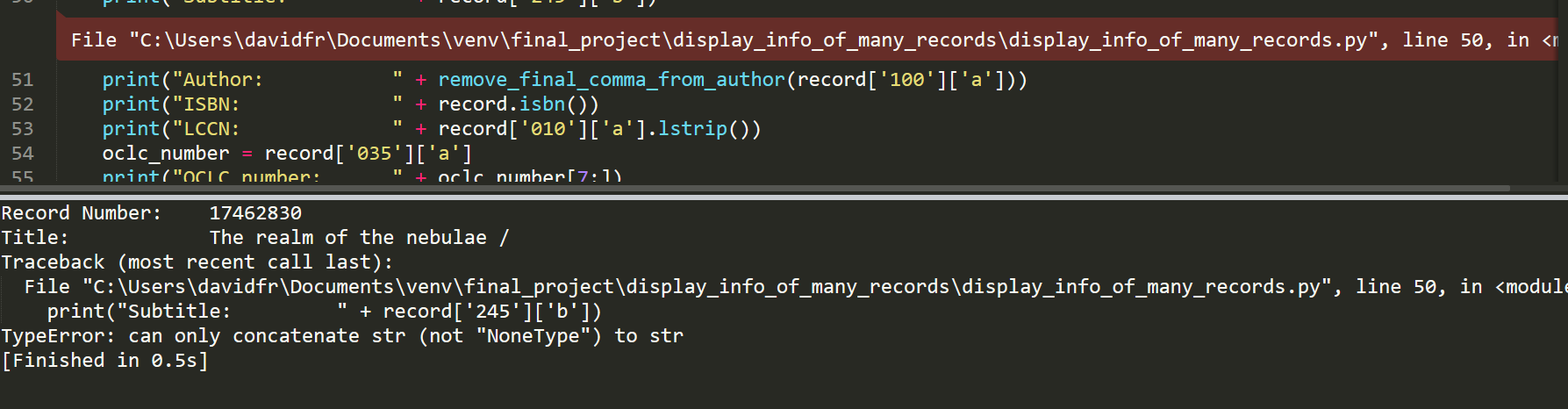


That worked! Now I have a file called data.mrc. Here’s my record count from marcedit for that new file with seven records:

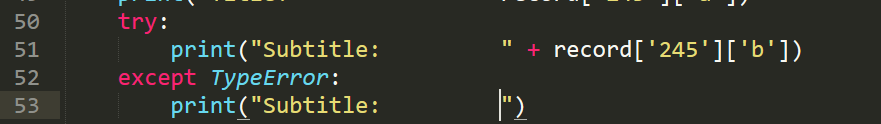


I’m just going to rename my infile to data.mrc and see what happens. I’m expecting errors.

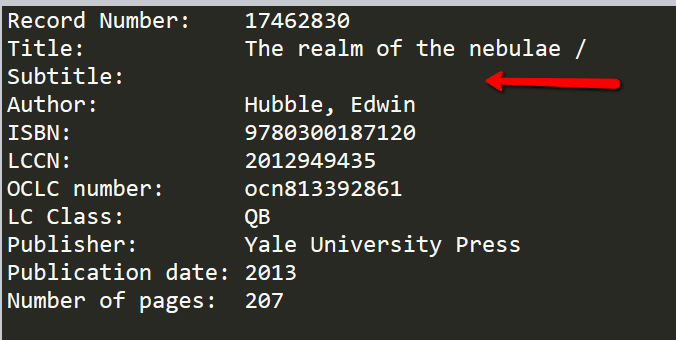
Here’s my first one:



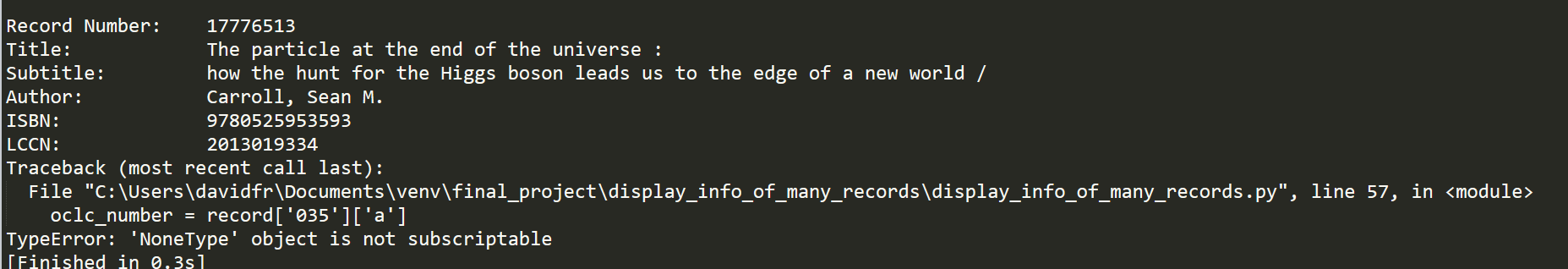
One of my records didn’t have a subtitle. I’m going to see if I can get past this first error by doing an exception handling on that field.



That actually got me pretty far! I was able to get the first 4 records. Here’s that first one that initially threw a traceback:



Of course, I’m getting another TypeError on the OCLC number for record number 17776513:



I think that instead of doing this one-by-one, I’m just going to enter the exception handling for each field.

Okay so I have the exception handling done for each field. Now I’m basically going to copy/paste the whole for loop but instead of printing everything I’m going to be adding dictionaries to a list. So it’s the same work but instead of using print() I’m going to be:

1. Defining an empty list
2. Starting a for loop
3. Defining an empty dictionary inside the loop
4. Assigning key value pairs to go into the dictionary

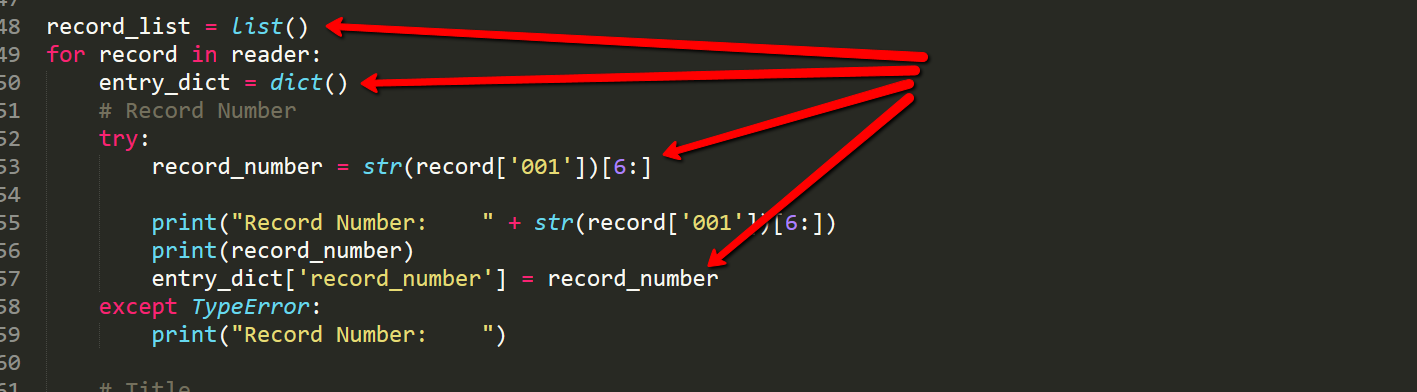
The end result will be a list of dictionaries that I can reference with searches.

Here’s how I’m starting that:

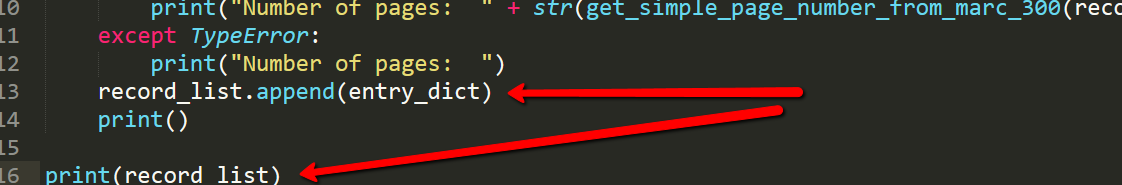


That didn’t work. I found out that I can only loop through my reader once, which is sort of a drag but I got it working anyway. So instead of copy/pasting my loop, I’m actually just keying everything into the same loop.

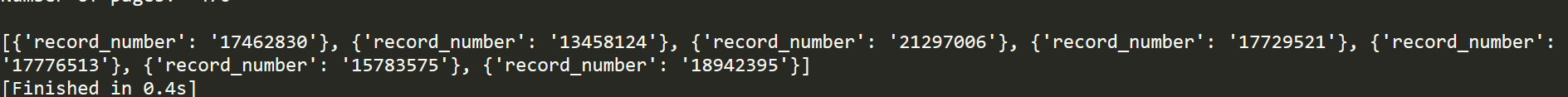
I tested with record\_number:



Then at the bottom of the loop…

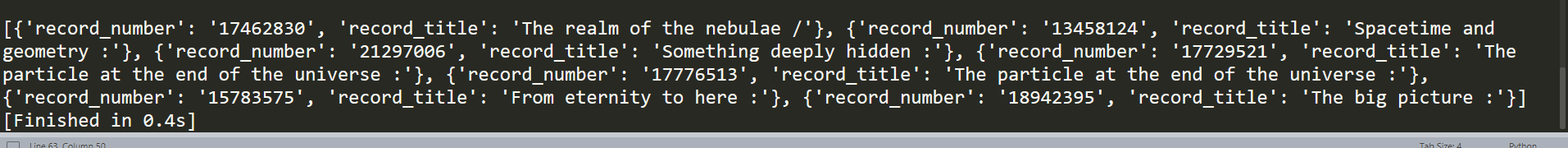


So it prints out all of the records as before but it also shows me my dictionary list in the output:



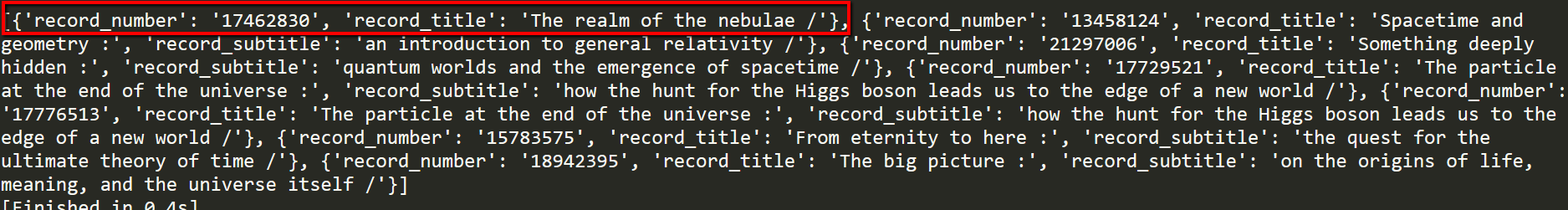
Now I just need to repeat that process for each field.

It worked when I added title!



Not every record has a subtitle. Let’s see what happens when I do the same thing with subtitle.

Success! The dict\_entry was created with only record\_number and record\_title!



Hurray! Here’s the end result of my dictionary list (I ran it through an online json formatter):

{

   "record\_number":"17462830",  
   "record\_title":"The realm of the nebulae /",  
   "record\_author":"Hubble, Edwin",  
   "record\_isbn":"9780300187120",  
   "record\_lccn":"2012949435",  
   "record\_oclc\_number":"ocn813392861",  
   "record\_lc\_class":"QB",  
   "record\_publisher":"Yale University Press",  
   "record\_publication\_date":"2013",  
   "record\_number\_of\_pages":"207"

},  
{

   "record\_number":"13458124",  
   "record\_title":"Spacetime and geometry :",  
   "record\_subtitle":"an introduction to general relativity /",  
   "record\_author":"Carroll, Sean M.",  
   "record\_isbn":"0805387323",  
   "record\_lccn":"2004296148",  
   "record\_oclc\_number":"ocm53245141",  
   "record\_lc\_class":"QC",  
   "record\_publisher":"Addison Wesley",  
   "record\_publication\_date":"2004",  
   "record\_number\_of\_pages":"513"

},  
{

   "record\_number":"21297006",  
   "record\_title":"Something deeply hidden :",  
   "record\_subtitle":"quantum worlds and the emergence of spacetime /",  
   "record\_author":"Carroll, Sean M.",  
   "record\_isbn":"9781524743017",  
   "record\_lccn":"2019286931",  
   "record\_oclc\_number":"on1086570568",  
   "record\_lc\_class":"QC",  
   "record\_publisher":"Dutton, an imprint of Penguin Random House",  
   "record\_publication\_date":"2019",  
   "record\_number\_of\_pages":"347"

},  
{

   "record\_number":"17729521",  
   "record\_title":"The particle at the end of the universe :",  
   "record\_subtitle":"how the hunt for the Higgs boson leads us to the edge of a new world /",  
   "record\_author":"Carroll, Sean M.",  
   "record\_isbn":"9780525953593",  
   "record\_lccn":"2012285624",  
   "record\_oclc\_number":"ocn795168227",  
   "record\_lc\_class":"QC",  
   "record\_publisher":"Dutton",  
   "record\_publication\_date":"2012",  
   "record\_number\_of\_pages":"34116"

},  
{

   "record\_number":"17776513",  
   "record\_title":"The particle at the end of the universe :",  
   "record\_subtitle":"how the hunt for the Higgs boson leads us to the edge of a new world /",  
   "record\_author":"Carroll, Sean M.",  
   "record\_isbn":"9780525953593",  
   "record\_lccn":"2013019334",  
   "record\_lc\_class":"QC",  
   "record\_publisher":"Plume",  
   "record\_publication\_date":"2013",  
   "record\_number\_of\_pages":""

},  
{

   "record\_number":"15783575",  
   "record\_title":"From eternity to here :",  
   "record\_subtitle":"the quest for the ultimate theory of time /",  
   "record\_author":"Carroll, Sean M.",  
   "record\_isbn":"9780525951339",  
   "record\_lccn":"2009023828",  
   "record\_oclc\_number":"ocn318411645",  
   "record\_lc\_class":"QC",  
   "record\_publisher":"Dutton",  
   "record\_publication\_date":"2010",  
   "record\_number\_of\_pages":"438"

},  
{

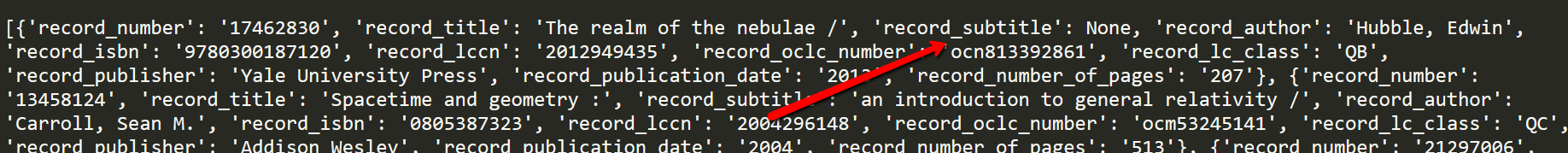
   "record\_number":"18942395",  
   "record\_title":"The big picture :",  
   "record\_subtitle":"on the origins of life, meaning, and the universe itself /",  
   "record\_author":"Carroll, Sean M.",  
   "record\_isbn":"9780525954828",  
   "record\_lccn":"2015050590",  
   "record\_lc\_class":"QH",  
   "record\_publisher":"Dutton est. 1852, an imprint of Penguin Random House LLC",  
   "record\_publication\_date":"2016",  
   "record\_number\_of\_pages":"470"

}

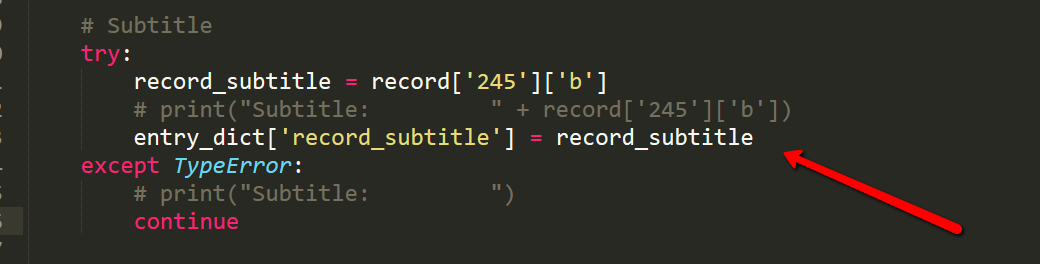
There’s still a lot to do. I don’t think I actually want to print out every record each time I run the program. The point of the program is to search. I’m going to start a new file called search\_info\_of\_many\_records.py and copy/paste what I have done so far into that file. The goal will be to create the same dictionary without printing out every record at the top of the output.

Okay so I’m in my new file now. Goal is to remove all of the print statements (except for the one that prints out my dictionary list). I also need to do something with the exception handlers because the only line of code I have in the except blocks is a print, so I think I’ll need to use the continue keyword. Let’s see if that works!

Drats! When I got to record\_subtitle for the first record (the one that lacks a subtitle), the dictionary stored None. That wasn’t happening before:

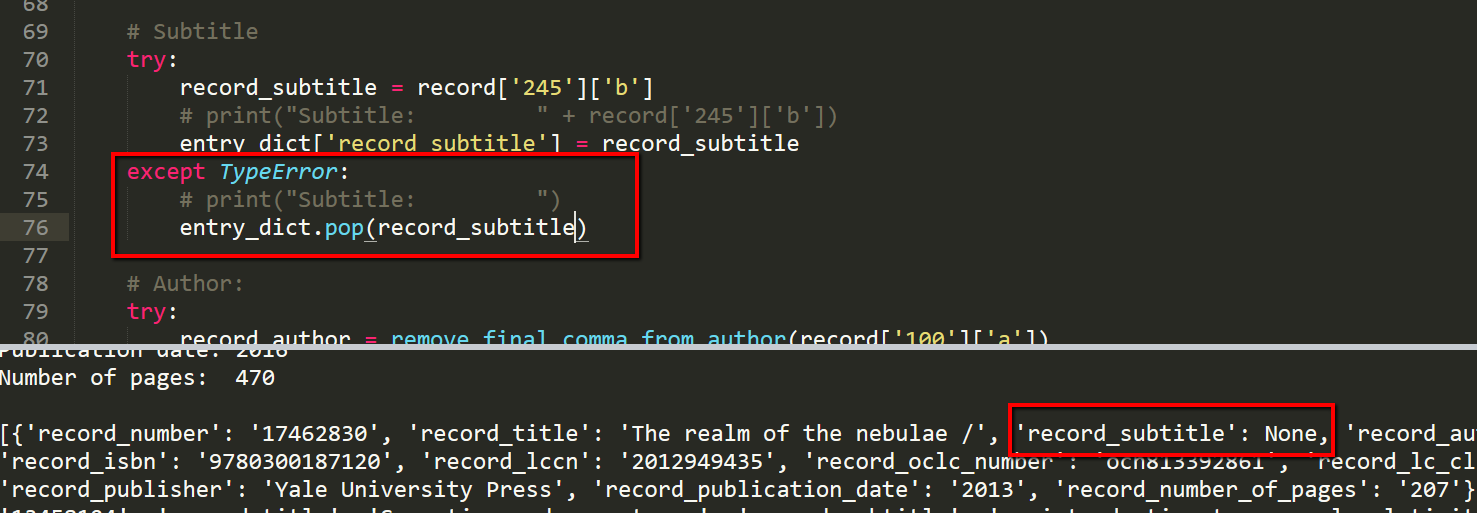


Here’s the code that lead to that outcome:



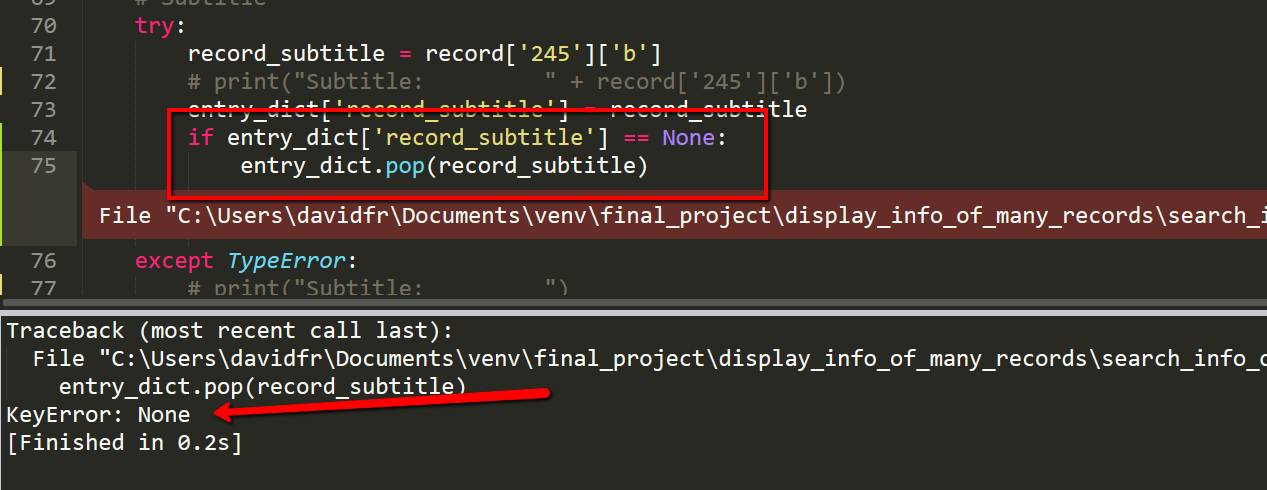
What if instead of using the continue keyword in the except block, I instead remove the key from the dictionary? Let’s try that! I don’t want to store None here.

That didn’t work:



That sort of leads me to believe that the exception handler isn’t actually doing anything here. It was really handy when I was printing strings but it looks like I don’t actually hit the TypeError at all when I’m assigning values to keys using the entry\_dict[‘key’] = value paradigm. I’ll test that theory by popping the key inside the try block.

That threw a key error:



Okay so I’m going to have to continue on this path and just accept the fact that the dictionary is going to store None if the field is blank. We’ll see how that goes when I start searching. I also still don’t know if the exception handlers are doing anything but I’m going to keep them in there as long as the program doesn’t get mad about them being there.

Okay so goal number 1 complete. I’m not thrilled about the fact that the dictionary is storing None for empty fields, but I’m going to try to work through that. Here’s my new dictionary list:

{

"record\_number":"17462830",

"record\_title":"The realm of the nebulae /",

"record\_subtitle":"None",

"record\_author":"Hubble, Edwin",

"record\_isbn":"9780300187120",

"record\_lccn":"2012949435",

"record\_oclc\_number":"ocn813392861",

"record\_lc\_class":"QB",

"record\_publisher":"Yale University Press",

"record\_publication\_date":"2013",

"record\_number\_of\_pages":"207"

},

{

"record\_number":"13458124",

"record\_title":"Spacetime and geometry :",

"record\_subtitle":"an introduction to general relativity /",

"record\_author":"Carroll, Sean M.",

"record\_isbn":"0805387323",

"record\_lccn":"2004296148",

"record\_oclc\_number":"ocm53245141",

"record\_lc\_class":"QC",

"record\_publisher":"Addison Wesley",

"record\_publication\_date":"2004",

"record\_number\_of\_pages":"513"

},

{

"record\_number":"21297006",

"record\_title":"Something deeply hidden :",

"record\_subtitle":"quantum worlds and the emergence of spacetime /",

"record\_author":"Carroll, Sean M.",

"record\_isbn":"9781524743017",

"record\_lccn":"2019286931",

"record\_oclc\_number":"on1086570568",

"record\_lc\_class":"QC",

"record\_publisher":"Dutton, an imprint of Penguin Random House",

"record\_publication\_date":"2019",

"record\_number\_of\_pages":"347"

},

{

"record\_number":"17729521",

"record\_title":"The particle at the end of the universe :",

"record\_subtitle":"how the hunt for the Higgs boson leads us to the edge of a new world /",

"record\_author":"Carroll, Sean M.",

"record\_isbn":"9780525953593",

"record\_lccn":"2012285624",

"record\_oclc\_number":"ocn795168227",

"record\_lc\_class":"QC",

"record\_publisher":"Dutton",

"record\_publication\_date":"2012",

"record\_number\_of\_pages":"34116"

},

{

"record\_number":"15783575",

"record\_title":"From eternity to here :",

"record\_subtitle":"the quest for the ultimate theory of time /",

"record\_author":"Carroll, Sean M.",

"record\_isbn":"9780525951339",

"record\_lccn":"2009023828",

"record\_oclc\_number":"ocn318411645",

"record\_lc\_class":"QC",

"record\_publisher":"Dutton",

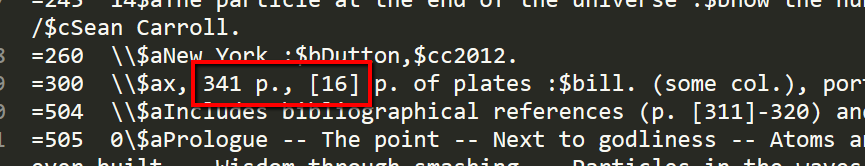
"record\_publication\_date":"2010",

"record\_number\_of\_pages":"438"

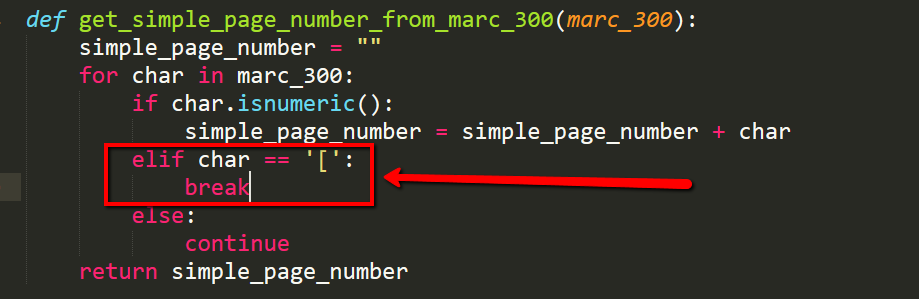
}

Ah! I found a bug. Record number 17729521 shouldn’t have 34116 pages.

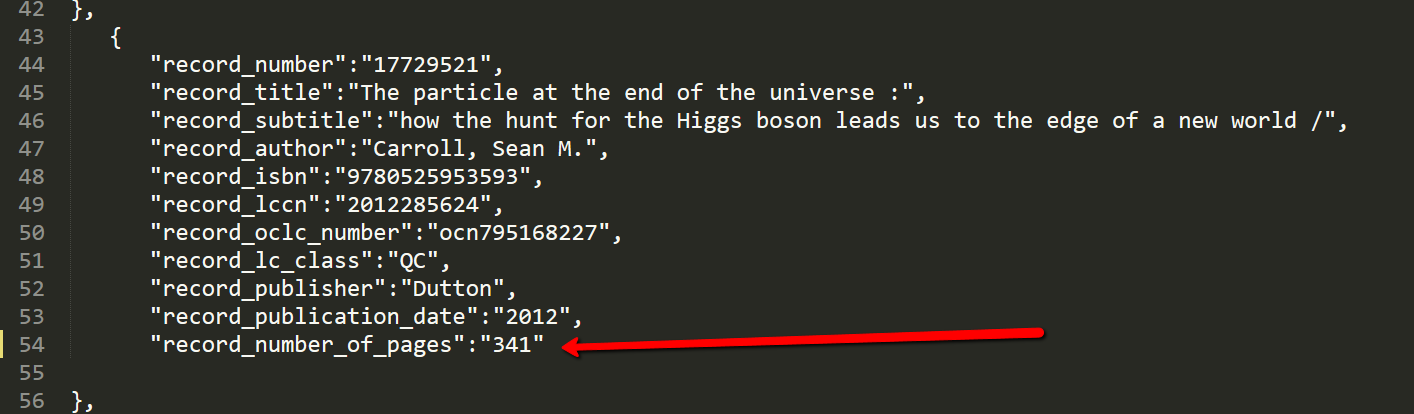
The problem is why the function get\_simple\_page\_number\_from\_marc\_300(marc\_300). Let’s see the marc record:



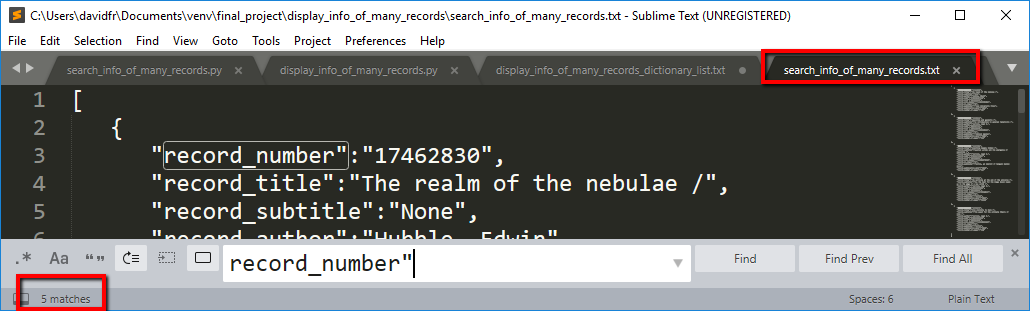
I’m going to add this elif block to that function to see if it does the trick:

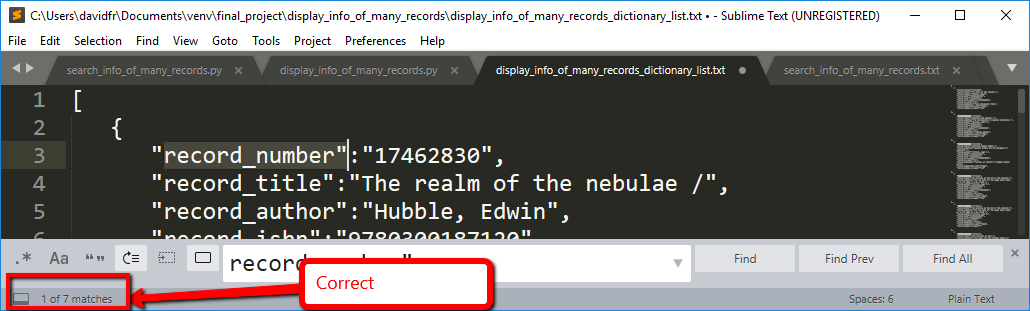


That did the trick!



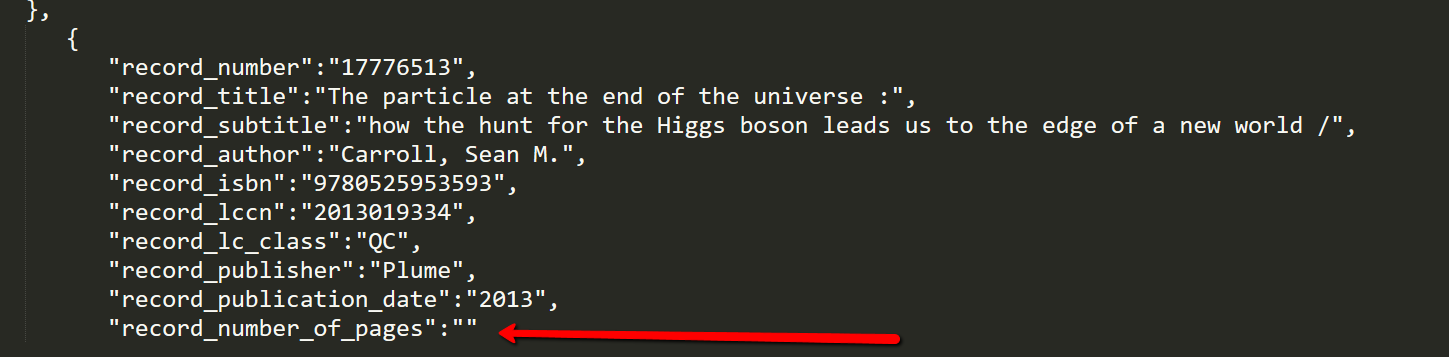
But…I have a worse bug. Where the first script added all 7 records, for some reason the second script only spit out 5 and I’m not sure why:



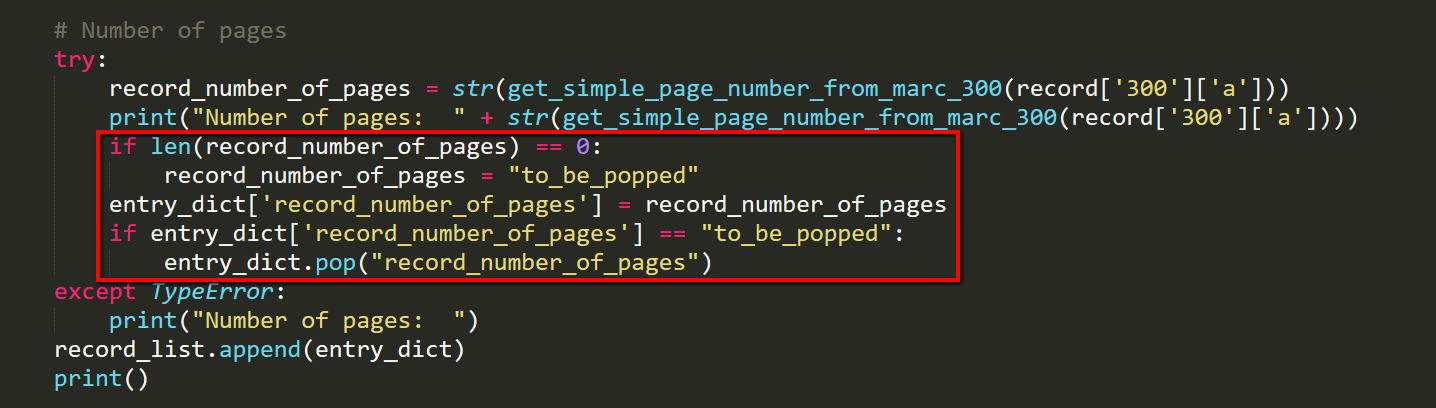


So I think I need to abandon my second script for now. It had two problems. Didn’t add every record to the dictionary and applied a None value for a blank field. Going back to ‘display\_info\_of\_many\_records.py’

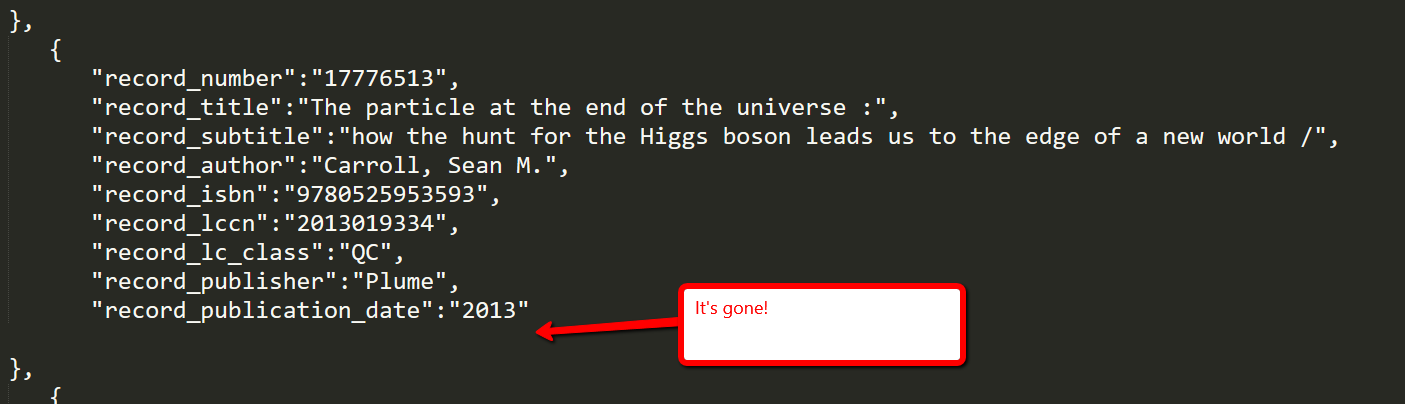
So I went back into ‘display\_info\_of\_many\_records.py’ and found another problem. Where the number of pages is blank, the script is adding an empty string for record\_number\_of\_pages, which I don’t want.:



Found a fun way to fix that problem! I set the empty string to a new string value ‘to\_be\_popped’ and then I popped it.



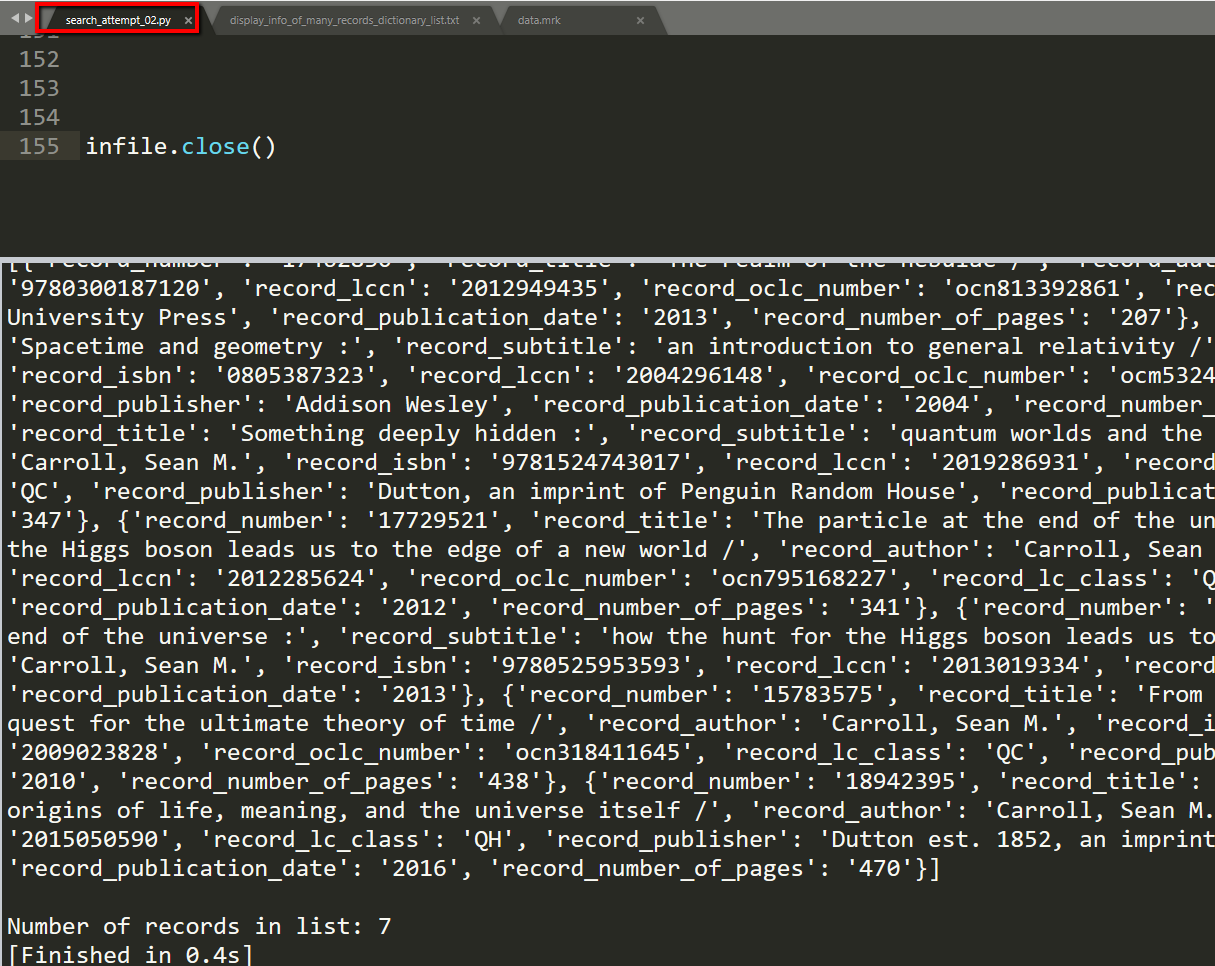
Result:



Next step

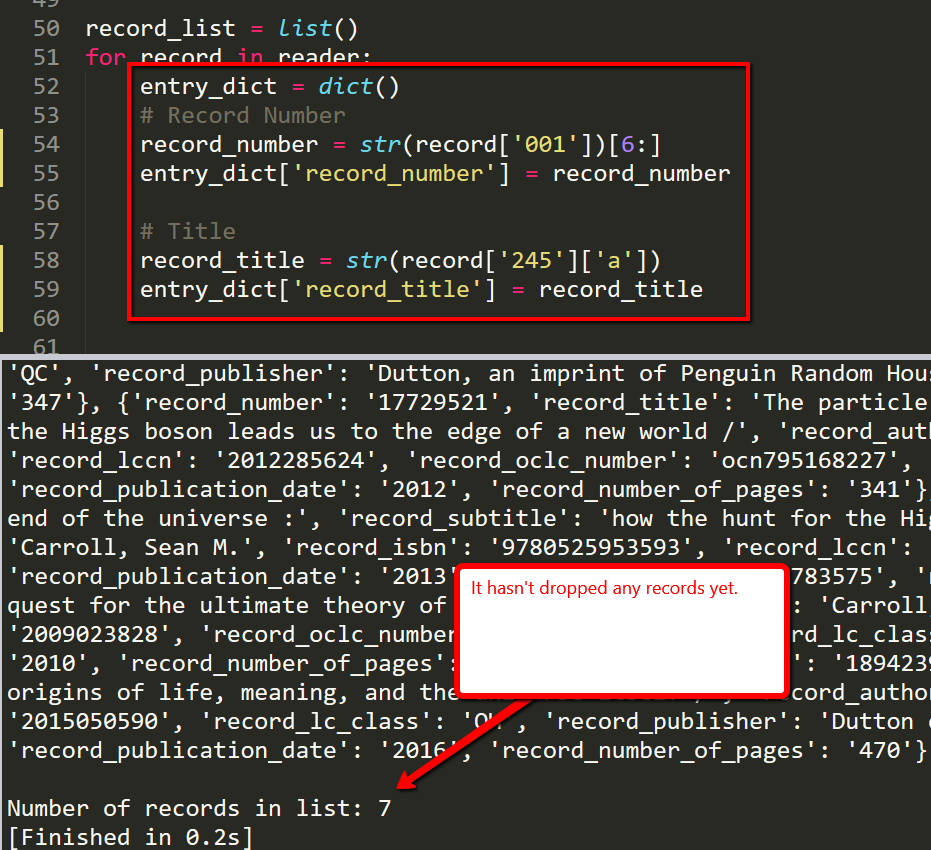
I want to go back and see if I can’t get the same dictionary by removing all of the exception handlers. I’ll use a new file for this. Really want to get this working without having to print each record at the top of the output. Alternative: Import the dictionary from display\_info\_of\_many\_records.py into a new script.

Okay so now I’m starting over with the file ‘search\_attempt\_02.py’:



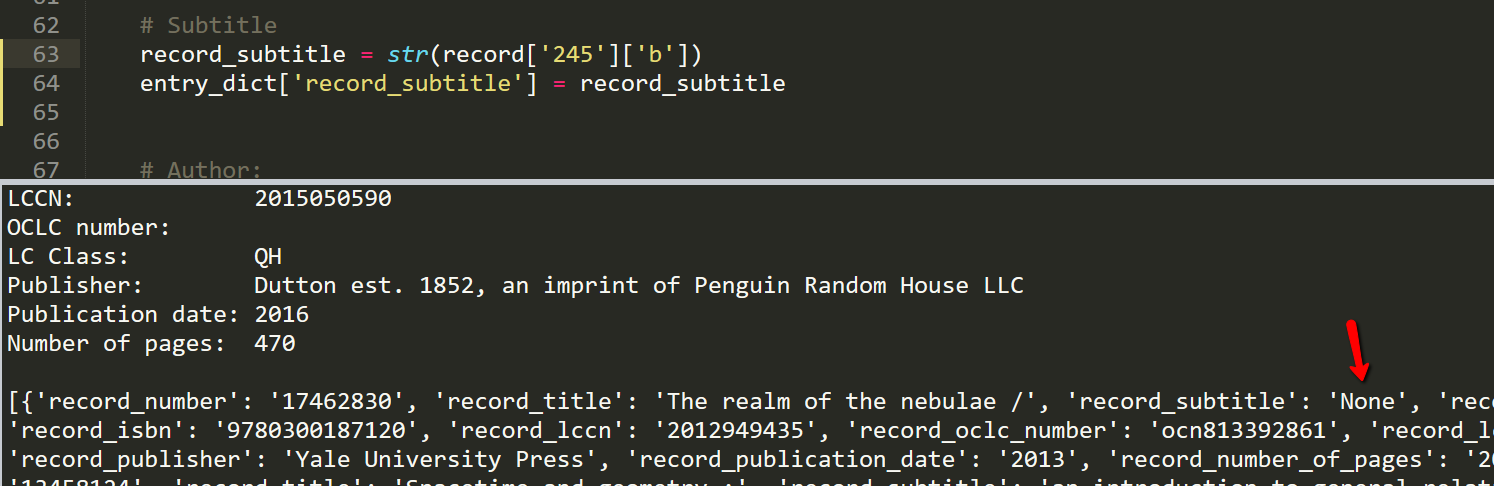
I’m going to try to write the dictionary without the exception handlers.

So far so good with record\_number and record\_title:



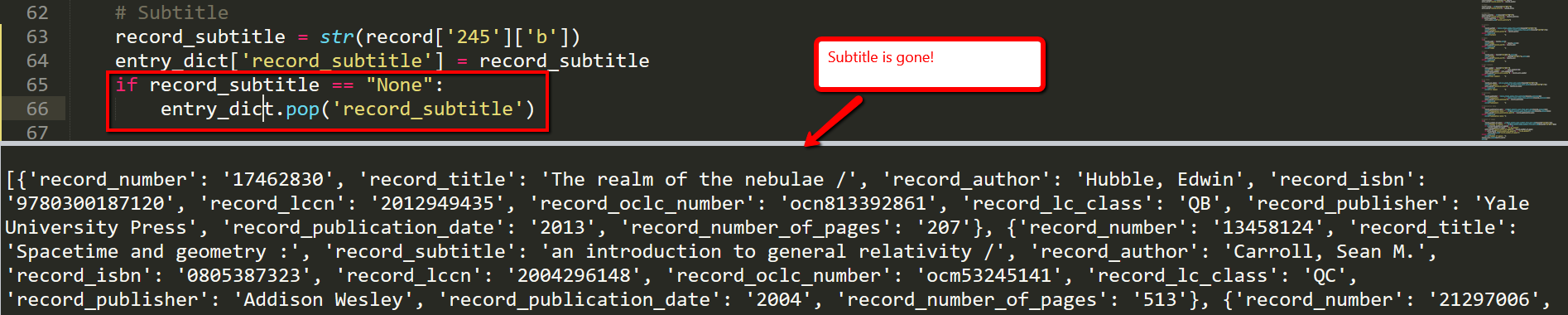
Now I need to do subtitle, which is the tricky one because the first record doesn’t have one.

Okay so it returned None. Noooooo!:



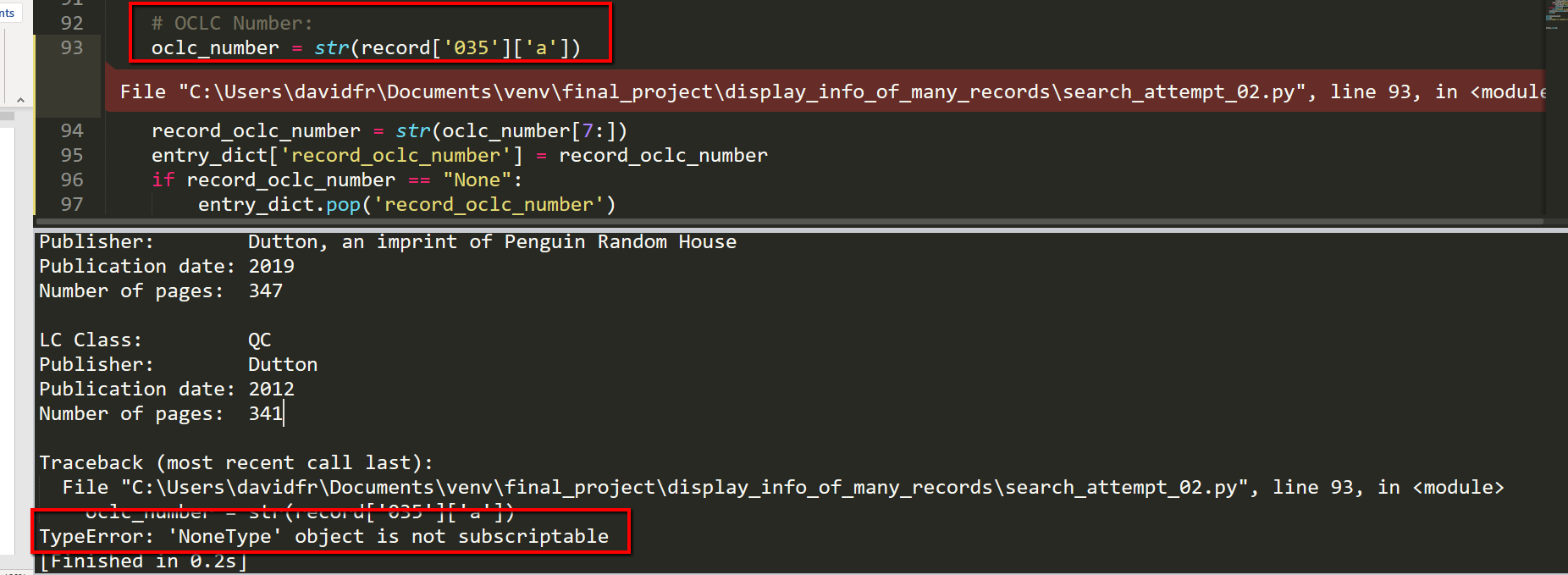
Maybe I can do the same pop trick here.

Success!



I think I’ll just add this if block to each field. It won’t hurt if the condition isn’t met.

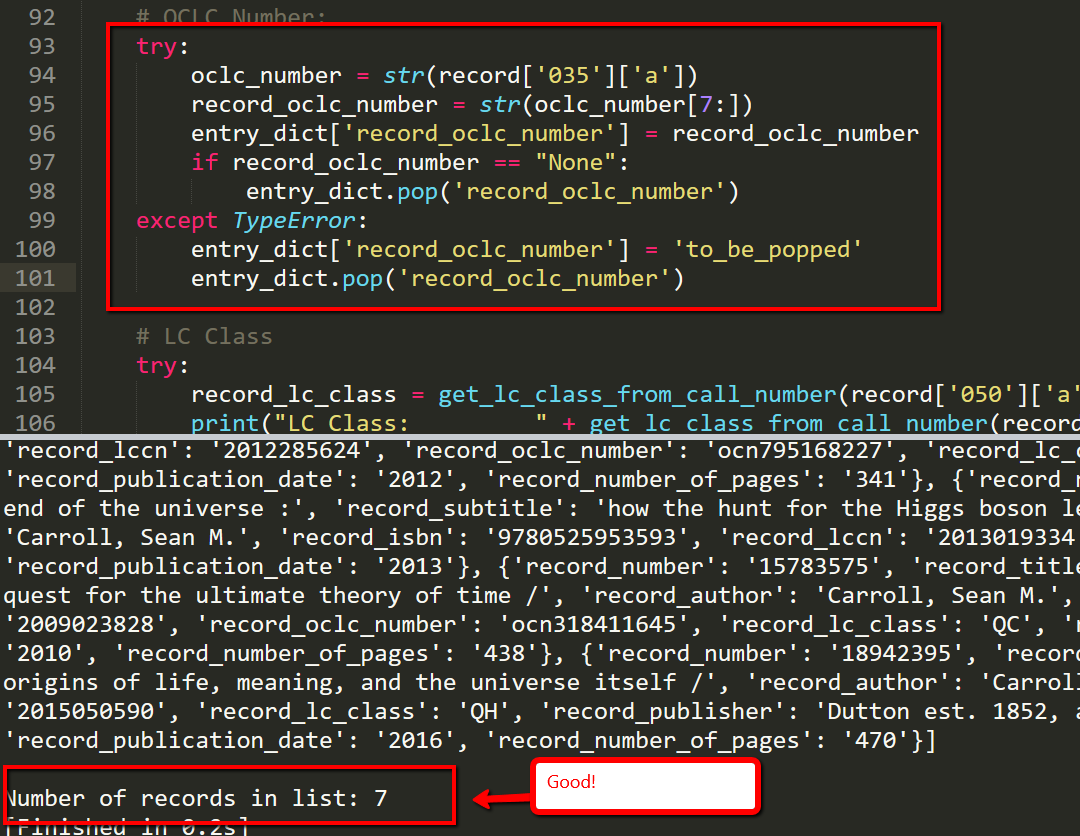
I ran into a TypeError for the OCLC number:



Record number 17776513 doesn’t have an 035a:



Okay I fixed it and I still have 7 records in the list:



I’m going to keep this one exception handler in here. I think I need it because I try to perform the slice. I should have done this in a function, but I have it working now so I’m just going to keep going. I don’t think I need to do this exception for each field.

Okay fantastic! So that worked. The file ‘search\_attempt\_02.py’ created the dictionary with all 7 records and no keys storing empty strings. Here’s the list of dictionaries:

[

{

"record\_number":"17462830",

"record\_title":"The realm of the nebulae /",

"record\_author":"Hubble, Edwin",

"record\_isbn":"9780300187120",

"record\_lccn":"2012949435",

"record\_oclc\_number":"ocn813392861",

"record\_lc\_class":"QB",

"record\_publisher":"Yale University Press",

"record\_publication\_date":"2013",

"record\_number\_of\_pages":"207"

},

{

"record\_number":"13458124",

"record\_title":"Spacetime and geometry :",

"record\_subtitle":"an introduction to general relativity /",

"record\_author":"Carroll, Sean M.",

"record\_isbn":"0805387323",

"record\_lccn":"2004296148",

"record\_oclc\_number":"ocm53245141",

"record\_lc\_class":"QC",

"record\_publisher":"Addison Wesley",

"record\_publication\_date":"2004",

"record\_number\_of\_pages":"513"

},

{

"record\_number":"21297006",

"record\_title":"Something deeply hidden :",

"record\_subtitle":"quantum worlds and the emergence of spacetime /",

"record\_author":"Carroll, Sean M.",

"record\_isbn":"9781524743017",

"record\_lccn":"2019286931",

"record\_oclc\_number":"on1086570568",

"record\_lc\_class":"QC",

"record\_publisher":"Dutton, an imprint of Penguin Random House",

"record\_publication\_date":"2019",

"record\_number\_of\_pages":"347"

},

{

"record\_number":"17729521",

"record\_title":"The particle at the end of the universe :",

"record\_subtitle":"how the hunt for the Higgs boson leads us to the edge of a new world /",

"record\_author":"Carroll, Sean M.",

"record\_isbn":"9780525953593",

"record\_lccn":"2012285624",

"record\_oclc\_number":"ocn795168227",

"record\_lc\_class":"QC",

"record\_publisher":"Dutton",

"record\_publication\_date":"2012",

"record\_number\_of\_pages":"341"

},

{

"record\_number":"17776513",

"record\_title":"The particle at the end of the universe :",

"record\_subtitle":"how the hunt for the Higgs boson leads us to the edge of a new world /",

"record\_author":"Carroll, Sean M.",

"record\_isbn":"9780525953593",

"record\_lccn":"2013019334",

"record\_lc\_class":"QC",

"record\_publisher":"Plume",

"record\_publication\_date":"2013"

},

{

"record\_number":"15783575",

"record\_title":"From eternity to here :",

"record\_subtitle":"the quest for the ultimate theory of time /",

"record\_author":"Carroll, Sean M.",

"record\_isbn":"9780525951339",

"record\_lccn":"2009023828",

"record\_oclc\_number":"ocn318411645",

"record\_lc\_class":"QC",

"record\_publisher":"Dutton",

"record\_publication\_date":"2010",

"record\_number\_of\_pages":"438"

},

{

"record\_number":"18942395",

"record\_title":"The big picture :",

"record\_subtitle":"on the origins of life, meaning, and the universe itself /",

"record\_author":"Carroll, Sean M.",

"record\_isbn":"9780525954828",

"record\_lccn":"2015050590",

"record\_lc\_class":"QH",

"record\_publisher":"Dutton est. 1852, an imprint of Penguin Random House LLC",

"record\_publication\_date":"2016",

"record\_number\_of\_pages":"470"

}

]

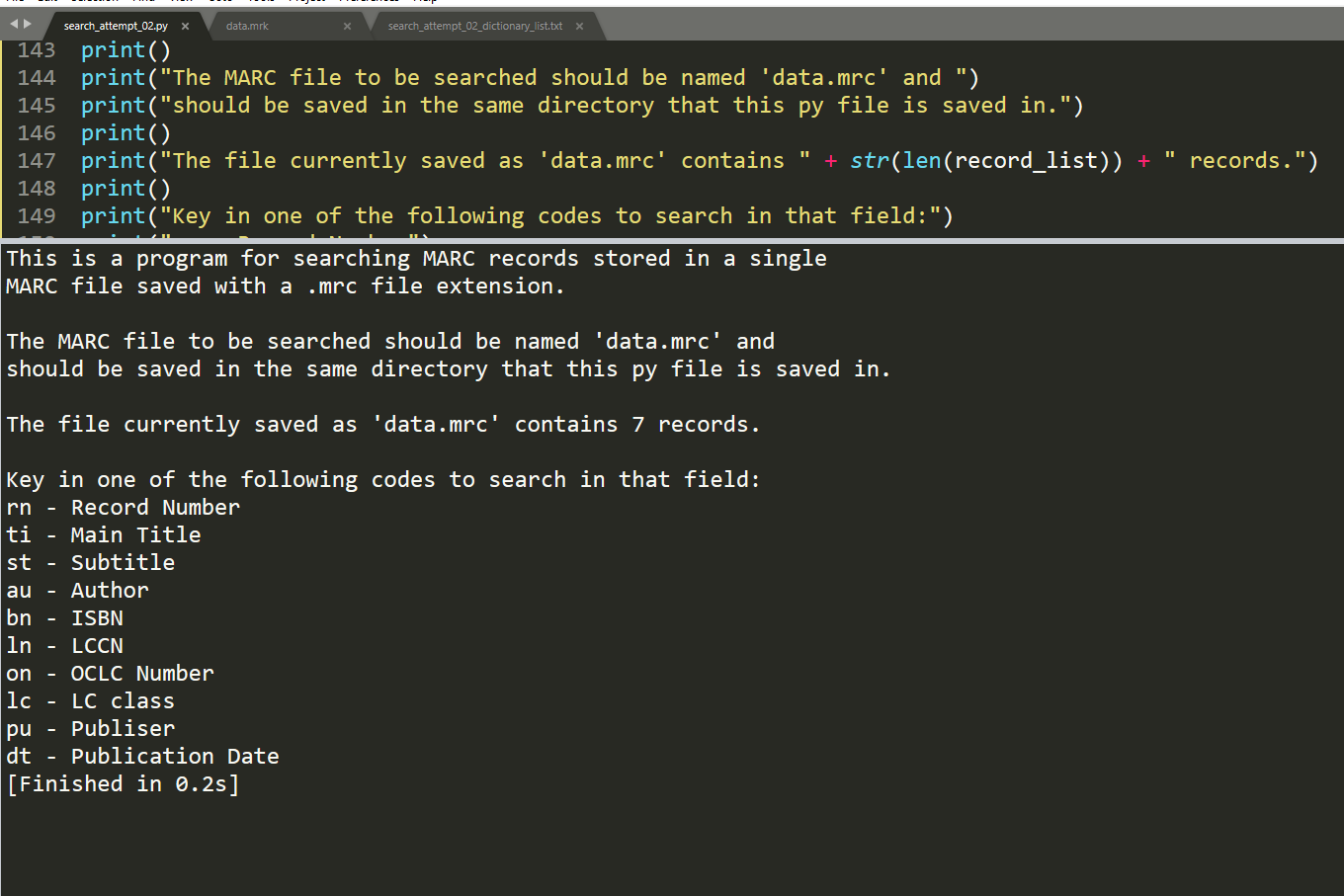
Now that THAT’S done…time for the hard part.

Now, I want to make the program do the following:

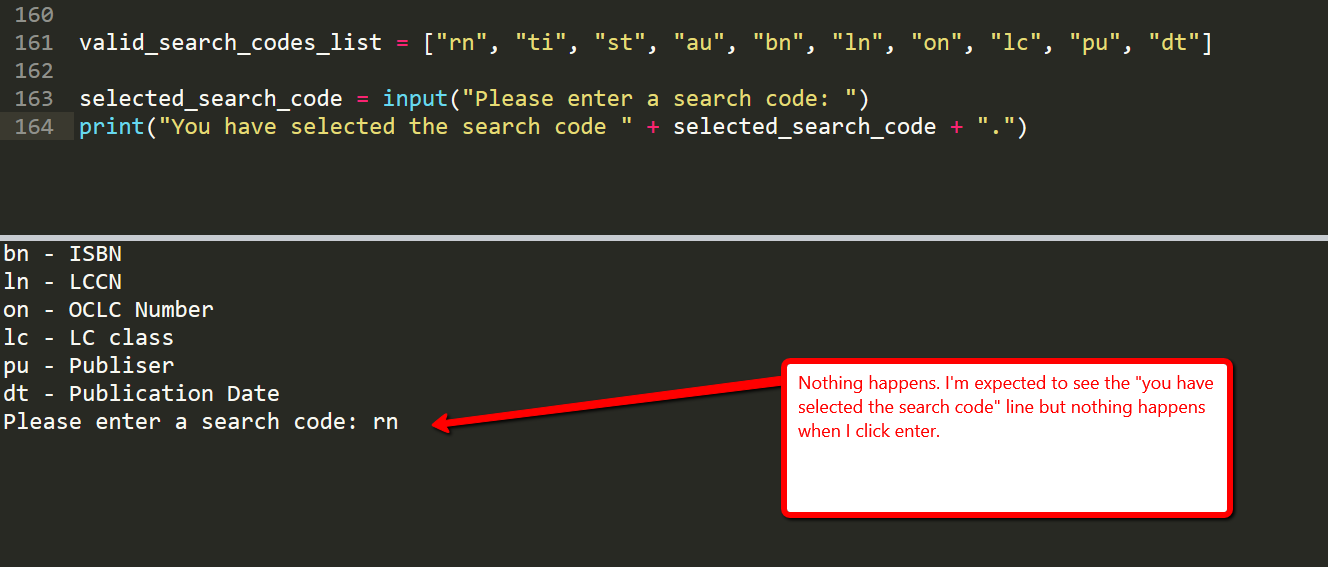
1. Prompt the user for a field to search on.
2. Prompt the user for a search string (the program will try to search case insensitive)
3. Return all of the records that met the search criteria

Now I also have another problem. I still don’t know how to use PyCharm to do this project because it doesn’t load the pymarc module. I also don’t know how to make inputs work correctly in Sublime, so this is going to be a bit of a puzzle. Need to make the inputs work correctly in sublime.

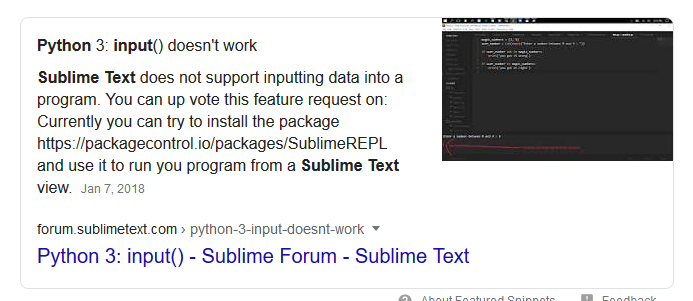
Here’s what I have so far:



Now I need to do my first input.

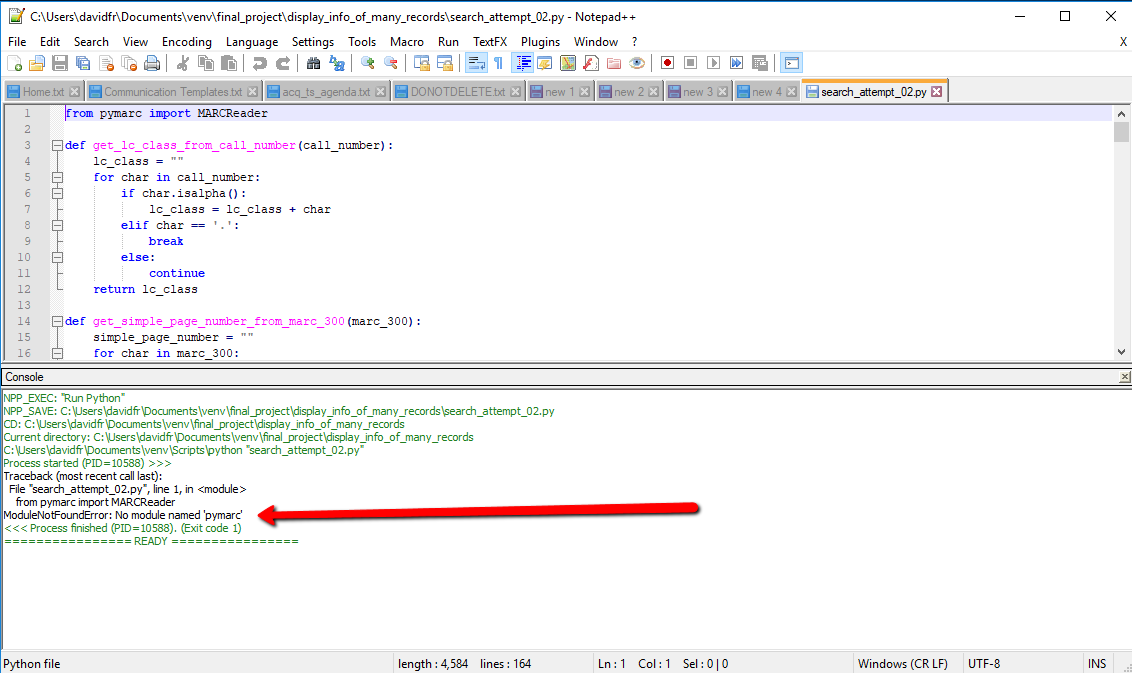


NOOOOOOO!!!!!!!!

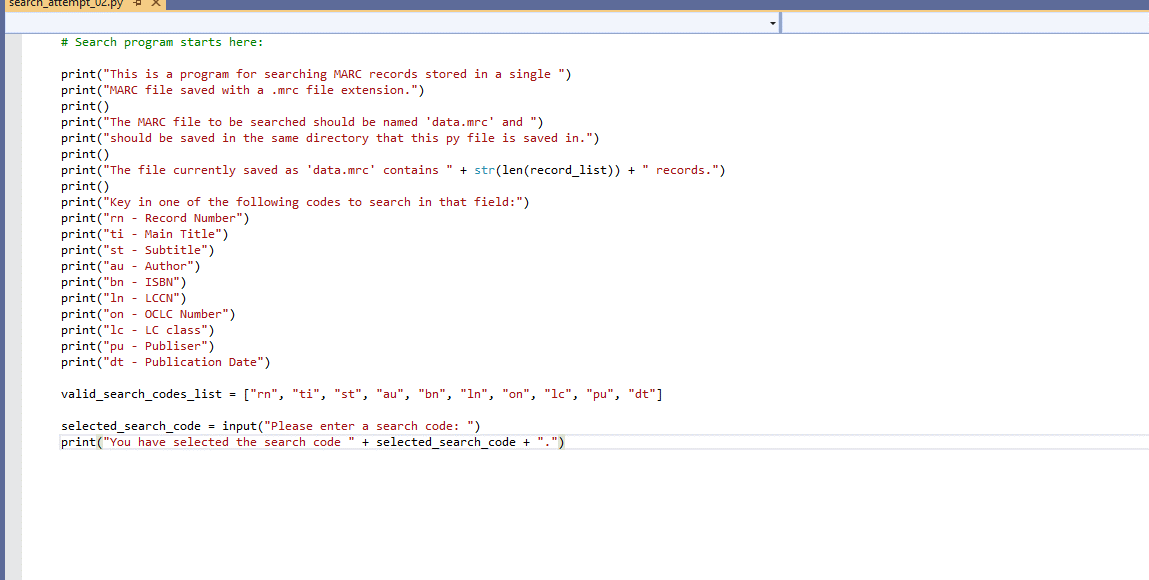


I’m bummed. I can’t use PyCharm. I can’t use Sublime. Need to fine another environment…

Notepad ++?

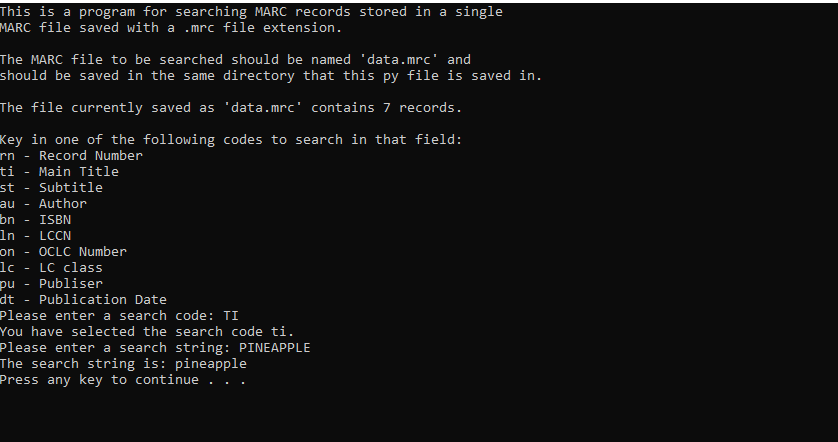


Okay for the time being I am saved by Visual Studio. Here’s my file:

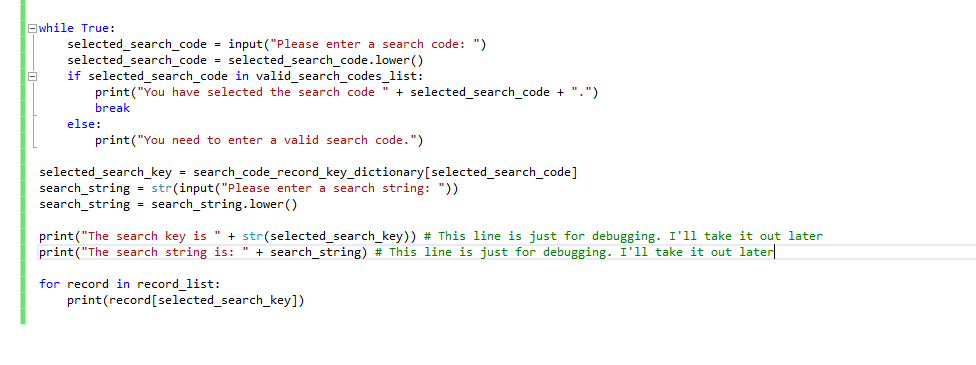


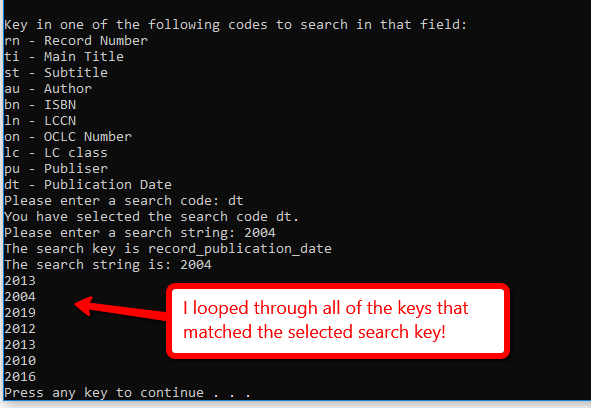
When I right-click on the script and select “Run without Debugging”, it starts running in the command line.

Okay so I have made some progress. I have validated the search code selection and made that non-case sensitive. I have also configured entering in the search string and making the input case insensitive.



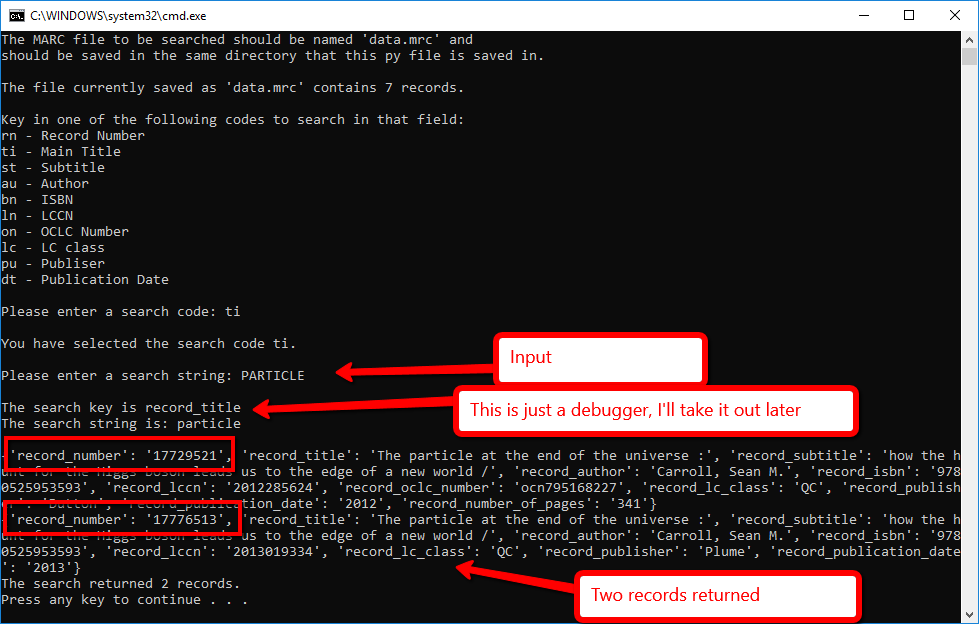
Okay so I have just made lots of progress without really noting it down here because I was in the zone. I’m getting closer. I still don’t have a mechanism for matching but I did figure out how to loop over the keys that match the selected search key!



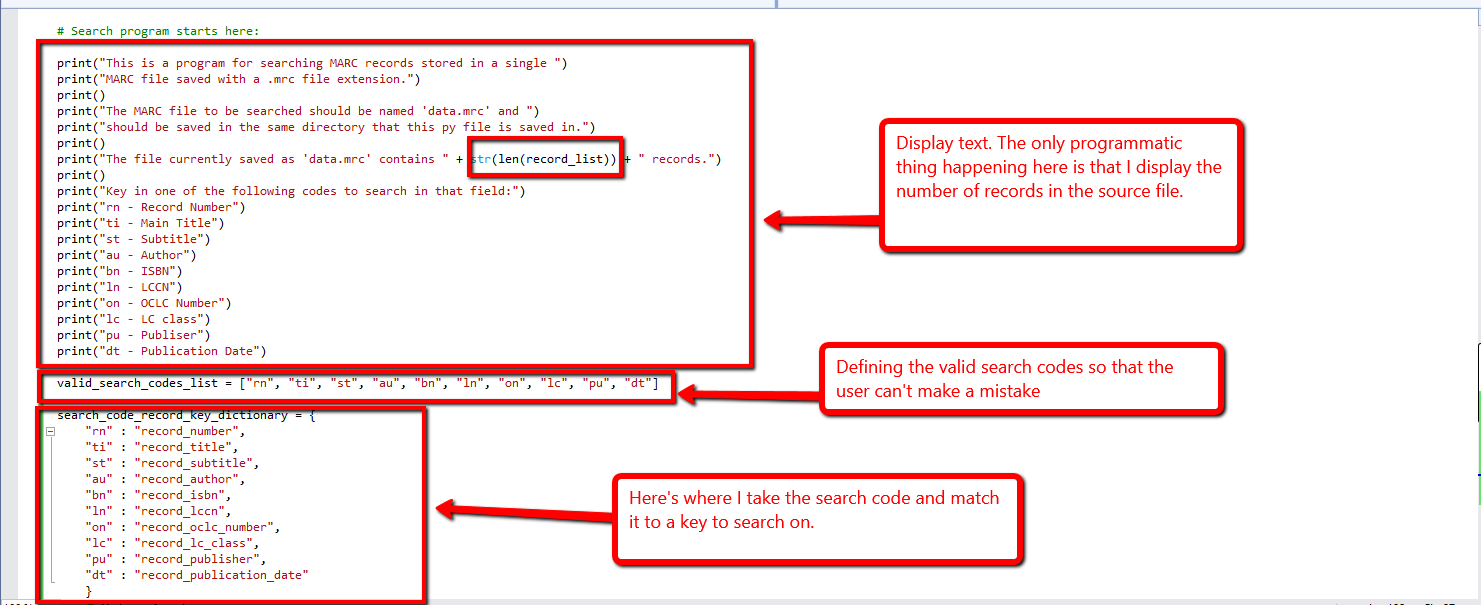


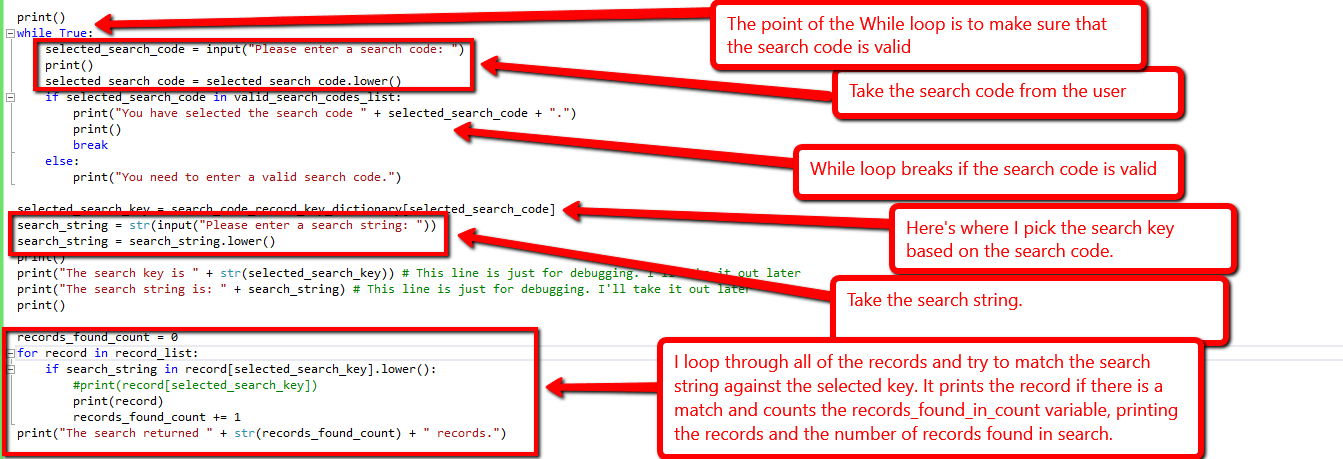
Okay so it’s still buggy but I have the title search working. I still need to display the records better but I have it returning the correct dictionaries.

I don’t know if you’ve noticed but I chose the test records deliberately to include duplicates. So let’s try looking for the search term ‘Particle’, which should return two records:

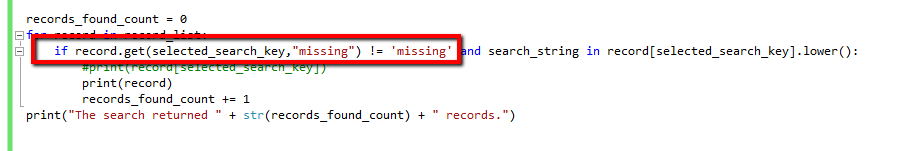


Here’s the code that did that:





I just fixed a bug. Before, if I searched on OCLC number it threw an error because not every record had an oclc number. I fixed that by adding another condition to the for loop:



Now it doesn’t get mad at me if it loops over something that doesn’t have the key that is stored in selected\_search\_key because I’m using the get method to assign that as missing. Then I’m using the != operator along with the AND in the condition to only search on records that have the given key. Woo!

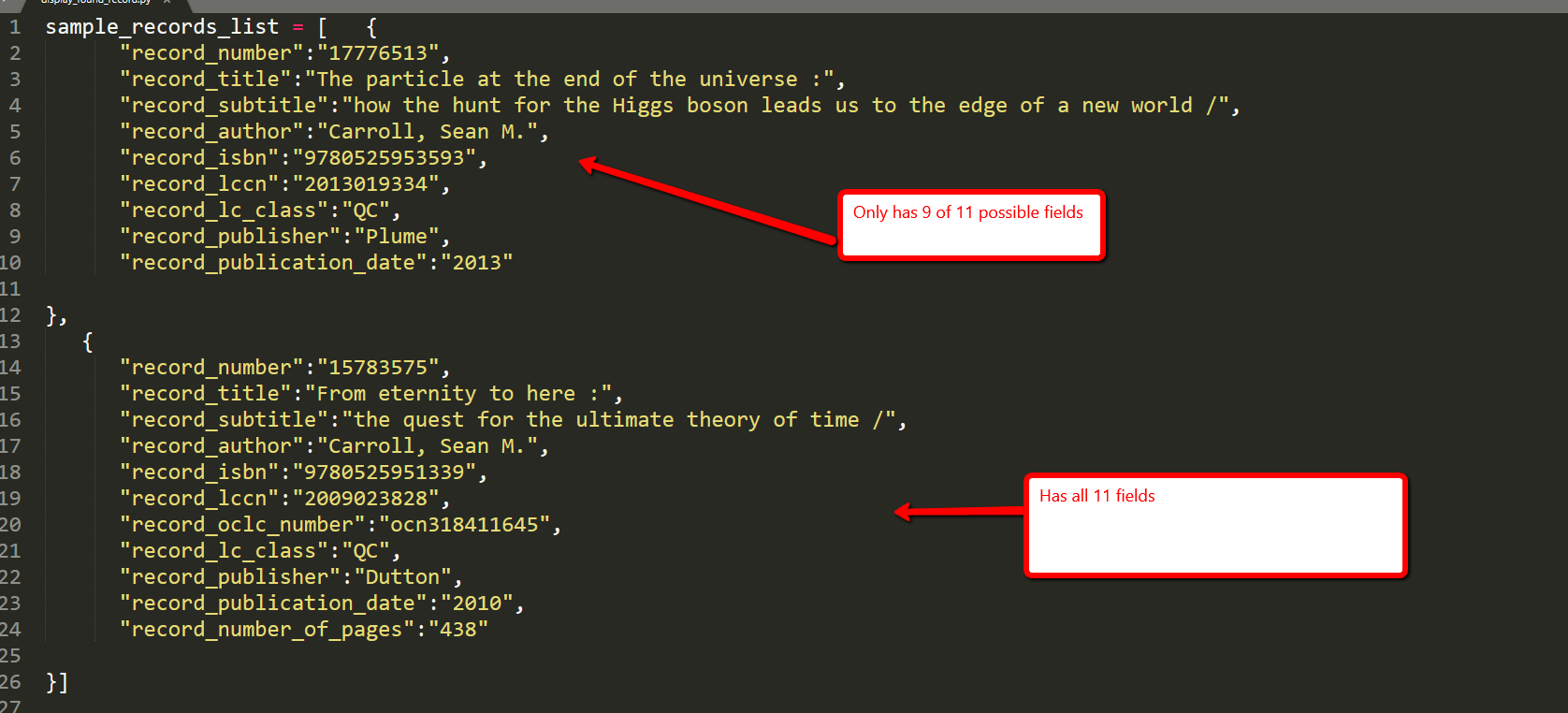
The last things I want to do are to:

1. Create a function that displays the records nicely instead of just the dictionaries.
2. Create a big ole loop so that I can add “Perform another search?” functionality. Basically I need to loop over my whole program…

Okay so I don’t want to do too much editing on my script right now, so I’m going to write the display\_found\_record function in another script….

Going back to sublime and using ‘display\_found\_record.py’ for testing this out:

I’ll start this script out with a variable called sample\_records\_list. One record has all of the the possible fields and one record is deficient:

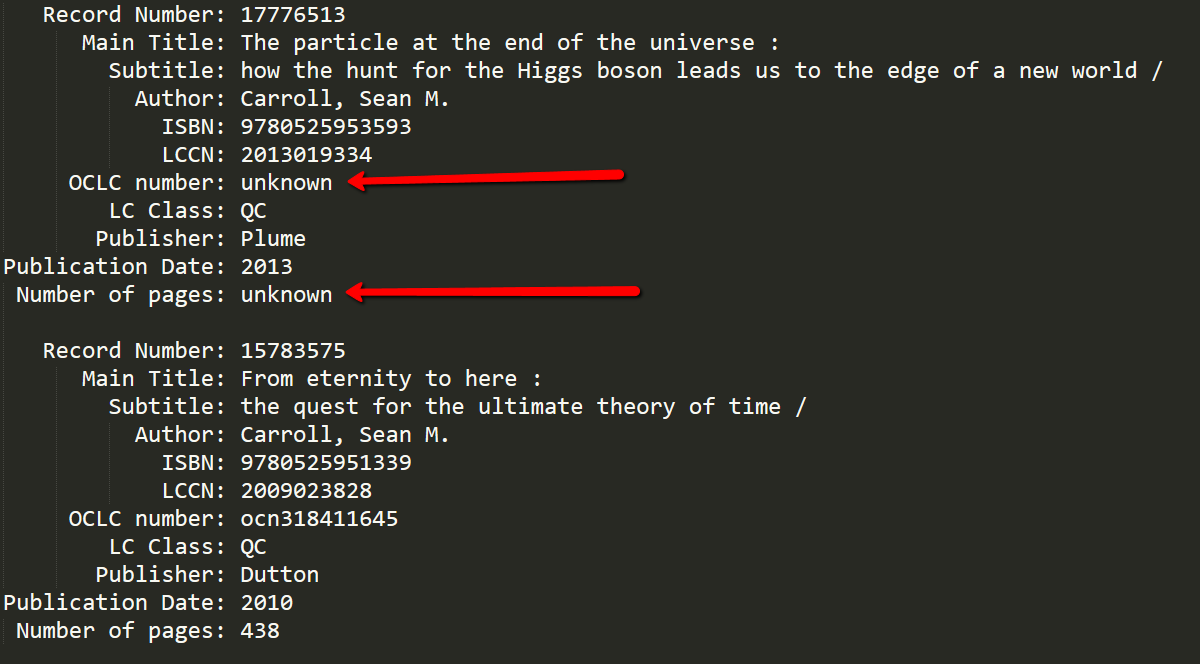


Let’s get cracking! I have no idea how I’m going to do this…

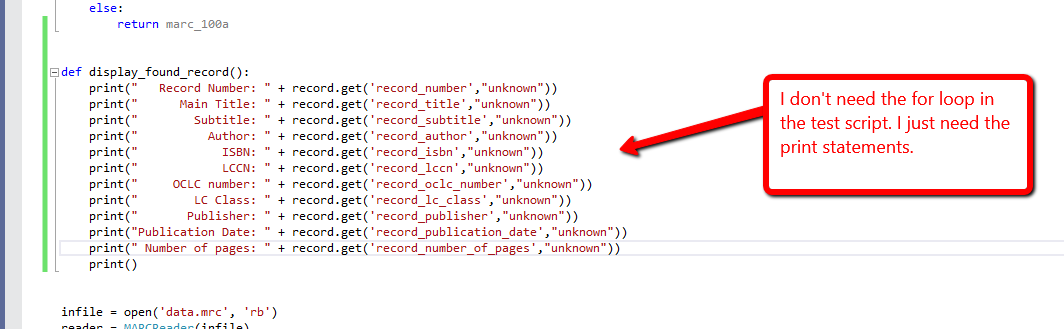
I’m okay with this:



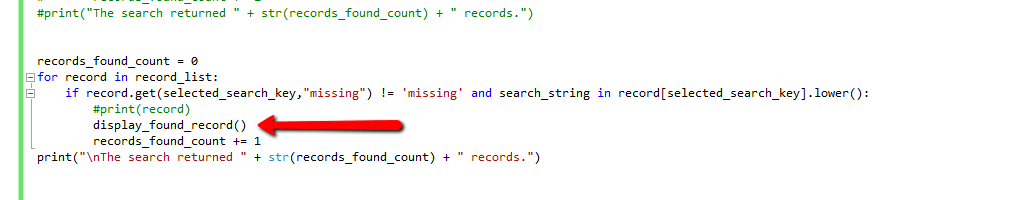
Here’s what that yielded from my sample records:



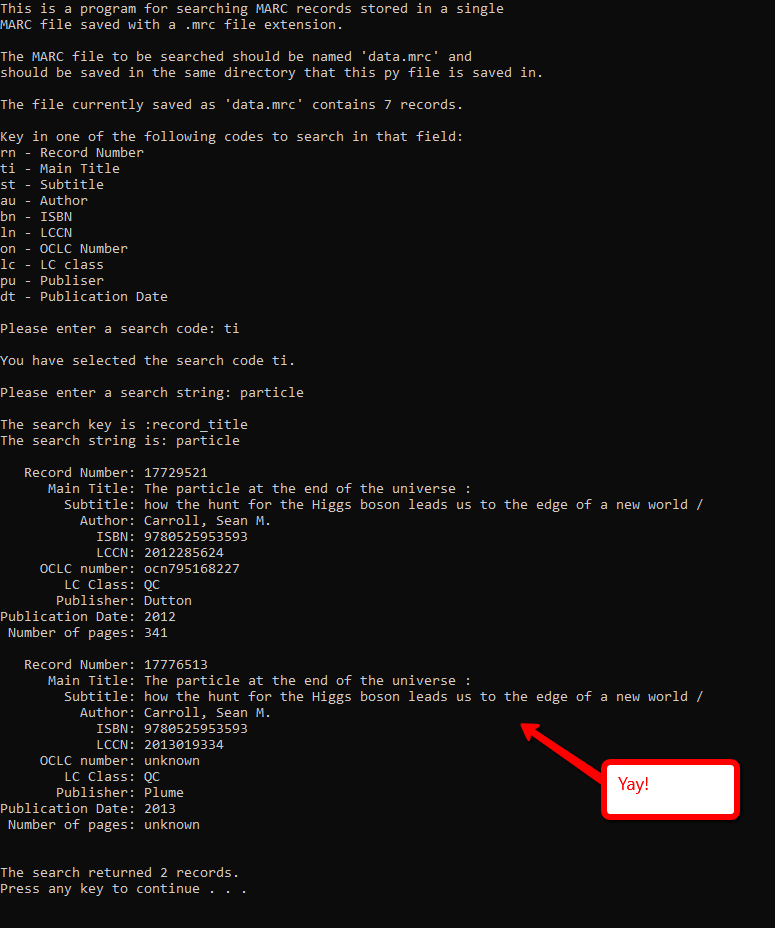
Let’s see how that behaves when I throw it back into my script:



Here is where I call it (instead of printing the dictionary):



And it worked!

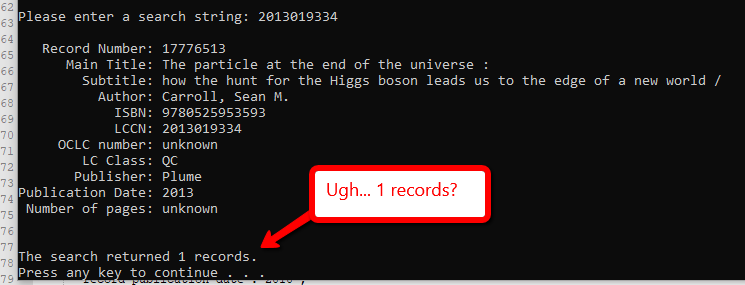


Oops, I misspelled Publisher in the list of available search codes. Fixed!

Now to loop over my whole program so I can add the “Perform another search?” functionality… oy!

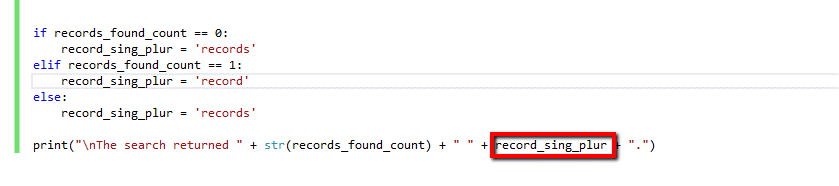
I’m going to start a whole new script just in case I mess this up. This one is ‘perform\_another\_search.py’. I’m just going to copy/paste the whole script to start.

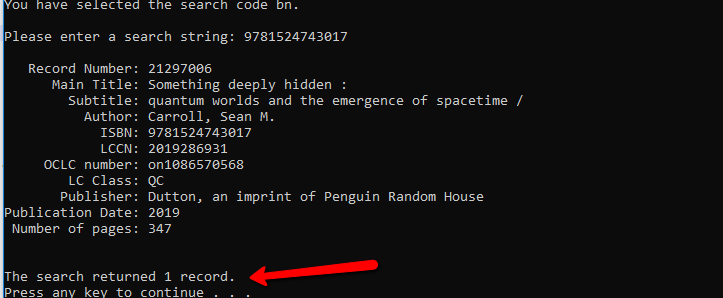
So I just ran ‘perform\_another\_search.py’ as a test to make sure I copied it right and I found something I hate. I hardcoded the word “records” in my statement at the end where I say how many records were returned by the search:



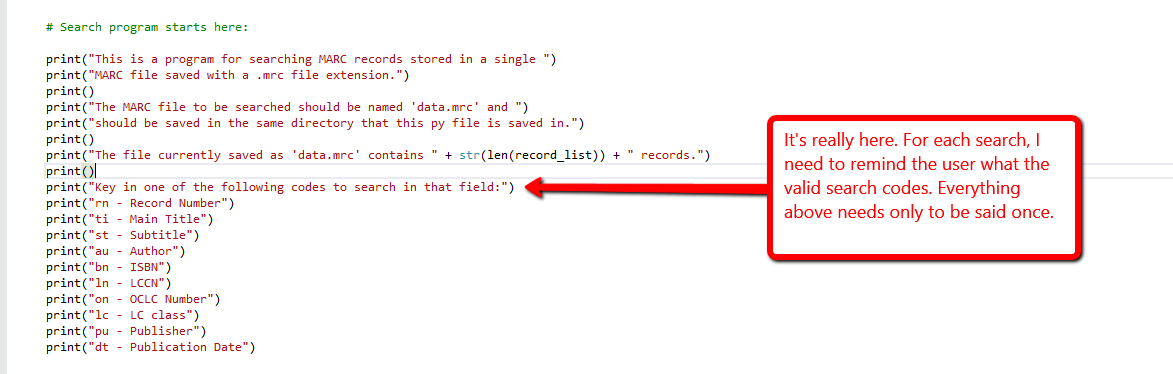
Let’s fix that.

Fixed!

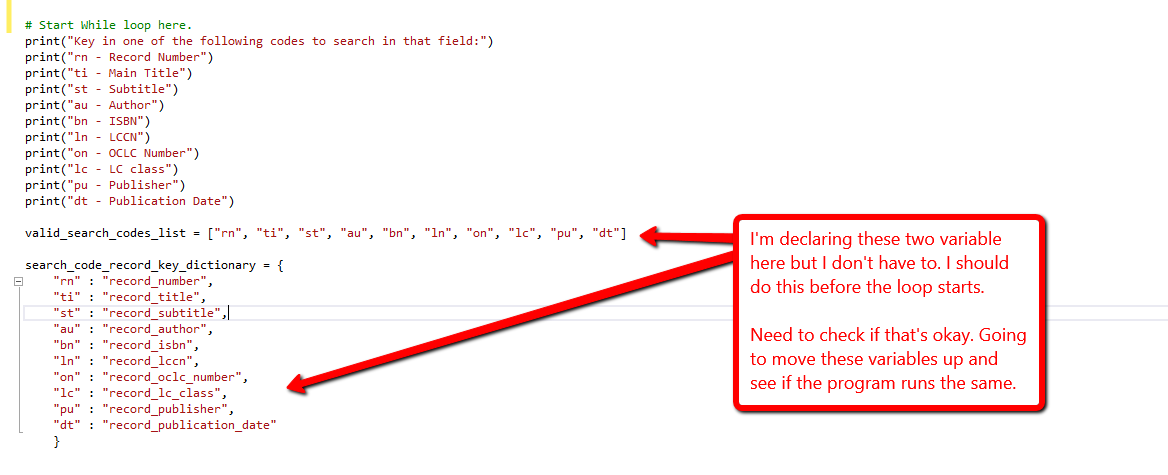




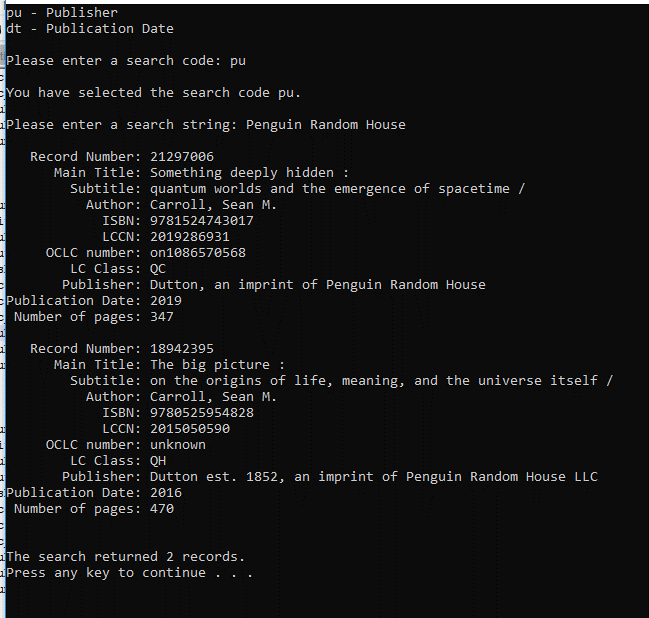
Okay now back to the task at hand. Need to add “perform another search?” functionality. I need to find the part of my code that I need to loop over.



Problem:

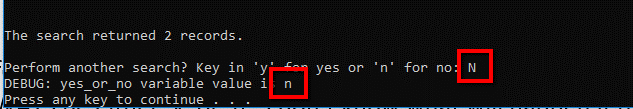


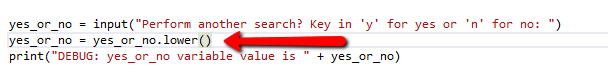
Pass! I moved them up before where I want the loop to start and it worked fine.



Before I actually start the while loop, I want to make sure that I can safely get a variable that will ALWAYS be either ‘y’ or ‘n’. I don’t know what will happen if they key in ‘adfaiuhoiuh’ when I ask “Perform anther search? Key in ‘y’ for yes or ‘n’ for no”

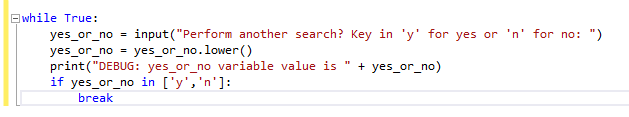
First goal accomplished. I forced the yes\_or\_no variable to be lowercase:



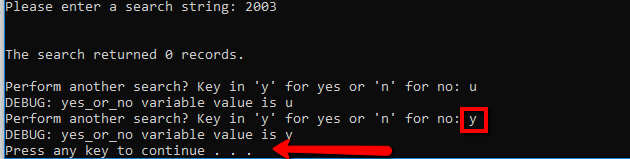


Now I need to force it to be ONLY y OR n.

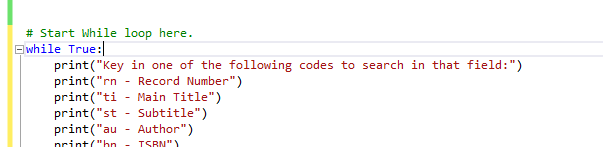
I’m going to try this. I hope I don’t start an infinite loop!!!



Hurray! I got the loop to break when the value was either ‘y’ or ‘n’.



Okay I think I’m done now. Now I just need to wrap up my whole program in a while loop. Let’s see how that goes!



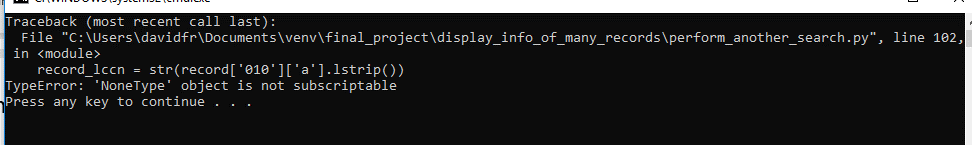
Hurray!!!! WOOOOO!!! It worked!

See video for full walkthrough.

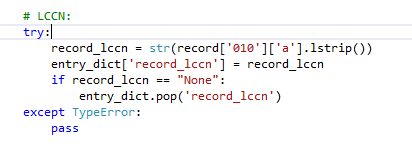
Okay so I have it working with the original 7 record test. Now to use a giant file. I downloaded ‘bpl101.mrc’ this one here: <https://archive.org/download/bpl_marc>

I’m going to debug using this giant file, so I’ll need a new script. This one is called ‘final\_version.py’

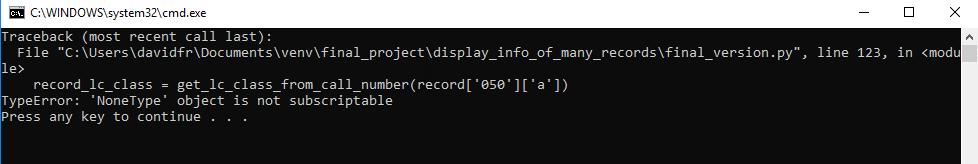
Okay so right off the bat I’m getting errors:



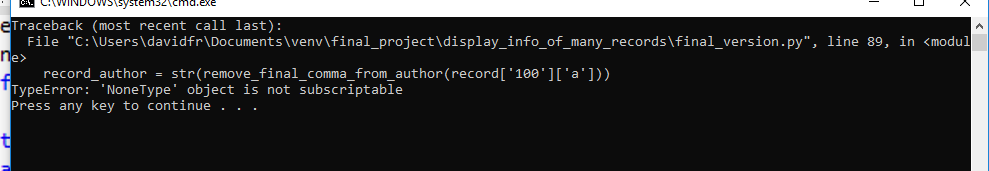
Exception handler?



Hey that got me to LC class, which means OCLC Number is okay:

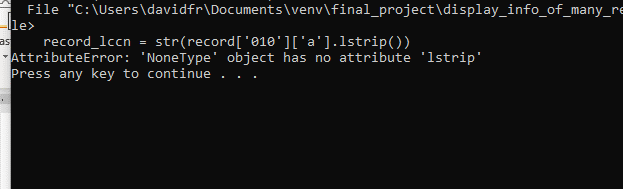


The loop iterated at least once. The error is bumped up the script to author now!

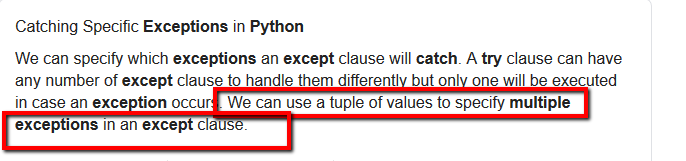


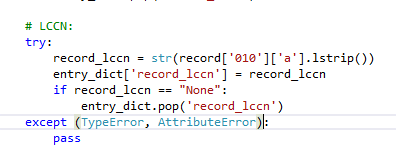
I’m going to keep adding that exception handler until it stops erroring here.

Attribute error!

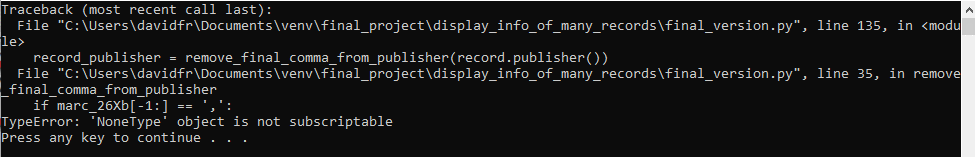


How do I add two types of errors to the same handler? To google!

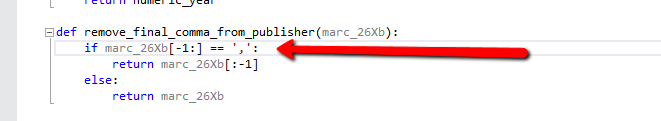


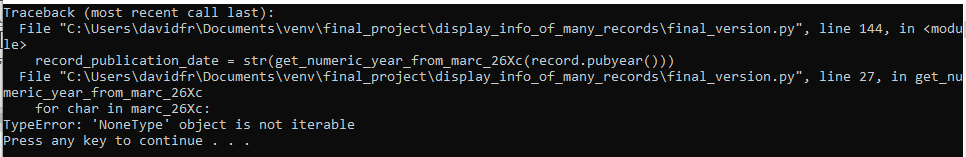


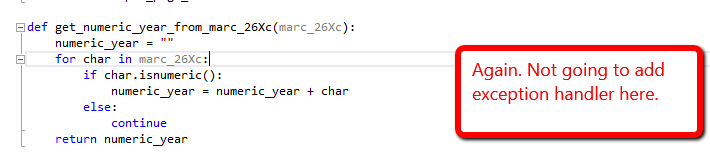
Now it doesn’t like something in my function:



I’m not going to add an exception handler to the function. I’m going to add it to the part of the code where the function is called.

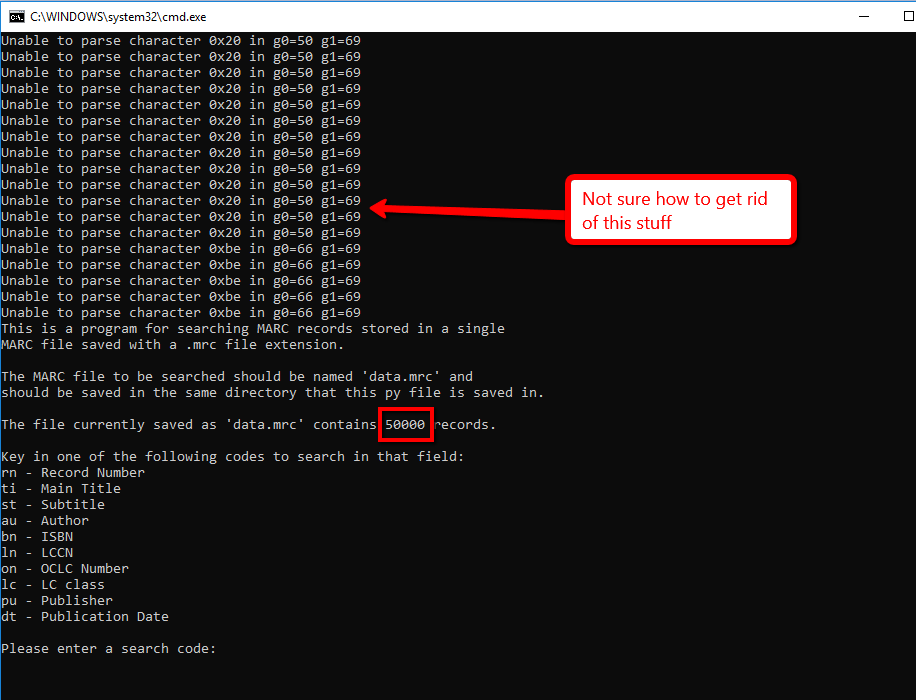


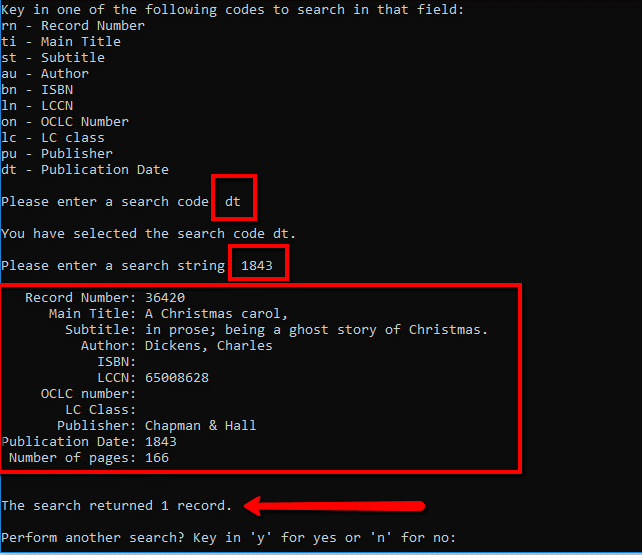




It works if I add the exception handler in the part of the code where the function is called. No need to mess with the function itself.

Woo! I got it working for the 50,000 record file! The program displays some weird stuff at the top that I don’t know how to get rid of, but it works nonetheless!





It works! Video demonstration: <https://youtu.be/EZtlsTgv37c>