

Anthony Cabral

11/15/2024

IT FDN 110 A Au 24

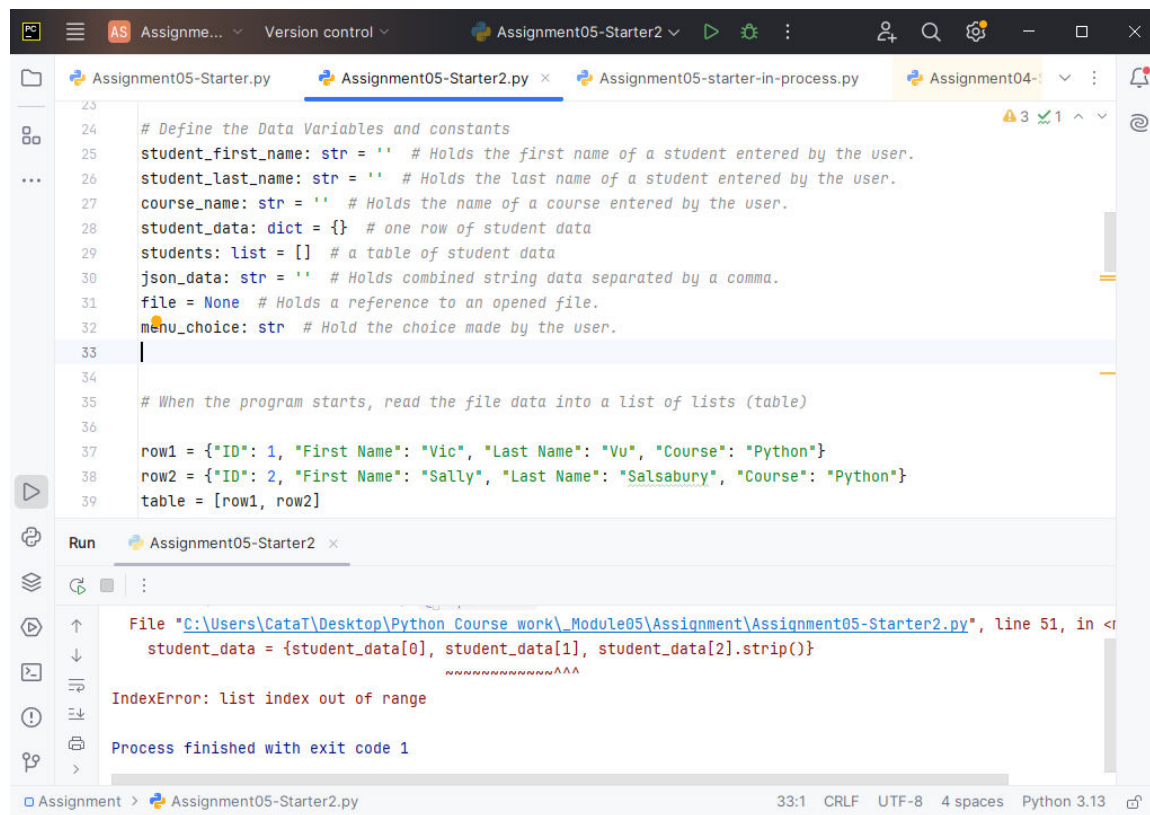
Module 5

## Module 5 Breakdown

### Process Introduction

Choice Adding correct variables and constants. Remove trace of CSV from the script and replace with JSON. Imported json at the start of the script.

I restarted the class. On my own, as in Ive redone the course modules because Im not getting as good a grip as I thought I had by this module. The last 3 I mean. Anyways heres module 5, it starts off really bad.



The screenshot shows a Python IDE with the following code in Assignment05-Starter2.py:

```
23
24 # Define the Data Variables and constants
25 student_first_name: str = '' # Holds the first name of a student entered by the user.
26 student_last_name: str = '' # Holds the last name of a student entered by the user.
27 course_name: str = '' # Holds the name of a course entered by the user.
28 student_data: dict = {} # one row of student data
29 students: list = [] # a table of student data
30 json_data: str = '' # Holds combined string data separated by a comma.
31 file = None # Holds a reference to an opened file.
32 menu_choice: str # Hold the choice made by the user.
33
34
35 # When the program starts, read the file data into a list of lists (table)
36
37 row1 = {"ID": 1, "First Name": "Vic", "Last Name": "Vu", "Course": "Python"}
38 row2 = {"ID": 2, "First Name": "Sally", "Last Name": "Salsabury", "Course": "Python"}
39 table = [row1, row2]
```

The Run console shows the following error:

```
File "C:\Users\Catal\Desktop\Python Course work\Module05\Assignment\Assignment05-Starter2.py", line 51, in <
student_data = {student_data[0], student_data[1], student_data[2].strip()}
IndexError: list index out of range
Process finished with exit code 1
```

The status bar at the bottom indicates: Assignment > Assignment05-Starter2.py 33:1 CRLF UTF-8 4 spaces Python 3.13

This part makes me a little hesitant because I'm adding a list of dicts and writing them to the JSON file. Just so the program can run.

```
33
34
35 # When the program starts, read the file data into a list of lists (table)
36
37 row1 = {"ID": 1, "First Name": "Vic", "Last Name": "Vu", "Course": "Python"}
38 row2 = {"ID": 2, "First Name": "Sally", "Last Name": "Salsabury", "Course": "Python"}
39 table = [row1, row2]
40
41 file = open("data.json", 'w')
42 json.dump(table, file)
43 file.close()
44
45
46 # Extract the data from the file
47 file = open(FILE_NAME, "r")
48 for row in file.readlines():
49     # Transform the data from the file
```

Run Assignment05-Starter2

File "C:\Users\CataT\Desktop\Python Course work\Module05\Assignment\Assignment05-Starter2.py", line 51, in <br> student\_data = {student\_data[0], student\_data[1], student\_data[2].strip()}<br> ~~~~~~<br> IndexError: list index out of range<br><br> Process finished with exit code 1

Assignment > Assignment05-Starter2.py 43:13 CRLF UTF-8 4 spaces Python 3.13

So now I've added it as suggested by the assignment. But the instructions say "Make sure to put some starting data into the file or you will get an error!" So starting data placed inside the json file? Or the script adding starting data, aka the list of dicts? Seemed unclear, so I figured I'd do both.

```
36
37 row1: dict[str, str] = {"ID": 1, "First Name": "Vic", "Last Name": "Vu", "Course": "Python"}
38 row2: dict[str, str] = {"ID": 2, "First Name": "Sally", "Last Name": "Salsabury", "Course": "Python"}
39 students = [row1, row2]
40
41 file = open("data.json", 'w')
42 json.dump(students, file)
43 file.close()
44
45
46 # Extract the data from the file
47 file = open(FILE_NAME, "r")
48 for row in file.readlines():
49     # Transform the data from the file
50     #student_data = row.split(',')
51     #student_data = {student_first_name, student_last_name, course_name.strip()}
52     # Load it into our collection (list of lists)
```

**Problems** File 9 Project Errors Server-Side Analysis **New** Vulnerable Dependencies

Assignment05-Starter2.py C:\Users\Cata\Desktop\Python Course work\Module05\Assignment 9 problems

- ⚠ Redeclared 'students' defined above without usage :39
- ⚠ Redeclared 'file' defined above without usage :41
- ⚠ Expected type 'SupportsWrite[str]', got 'TextIO' instead :42
- ⚠ Expected type 'dict', got 'set[str]' instead :68
- ⚠ Redeclared 'student' defined above without usage :86
- ⚠ Redeclared 'student' defined above without usage :87
- ⚠ Expected type 'SupportsWrite[str]', got 'TextIO' instead :88

Assignment > Assignment05-Starter2.py 43:13 CRLF UTF-8 4 spaces Python 3.13

Also went back and wrote the dict lines correctly. String/string, as opposed to string float. Makes sense. Opens the file, dumps it to json, closes. Now I can add a try/except line and change the write to read as the assignment suggests.

Ok so the code has changed and I've added my first try. First name and last name have been added as key values, as has course. Student data is correctly formatted. I hope. It works. And then it appends the data. Now as for the "try":

The screenshot shows a code editor with three tabs: 'Assignment05-Starter2.py', 'Test.py', and 'Assignment05-StarterOG.py'. The active tab 'Assignment05-StarterOG.py' contains the following Python code:

```
36 # When the program starts, read the file data into a list of lists (table)
37 # Extract the data from the file
38
39 try:
40     file = open(FILE_NAME, 'r')
41     students = json.load(file)
42     students.append(student_data)
43     file.close()
44 except FileNotFoundError as e:
45     print("Text file must exist before running this script!\n")
46     print("Built-In Python error info: ")
47     print(e, e.__doc__, type(e), sep='\n')
48     file.close()
49
50 # Present and Process the data
51 while (True):
52
```

Below the code editor is a 'Run' window for 'Assignment05-StarterOG.py'. It shows the following output:

```
-----
What would you like to do: 4
Program Ended

Process finished with exit code 0
```

The status bar at the bottom indicates the file is 'Assignment05-StarterOG.py', with a line length of 73:10, CRLF line endings, UTF-8 encoding, 4 spaces for indentation, and Python 3.13.

I used the format from the labs:

```
except Exception as e:
    print("Error! Please check you are not dividing by zero.\n")
    print("-- Technical Error Message -- ")
    print(e)           # Print the exception object (typically includes the error message)
    print(type(e))      # Print the type of the exception object
    print(e.__doc__)    # Print the documentation string of the exception type
    print(e.__str__())
```

Using the file not found handler. This example shows the zero error, because I found it redundant to share the exact same code I just used. File not found seemed most appropriate here.

-----

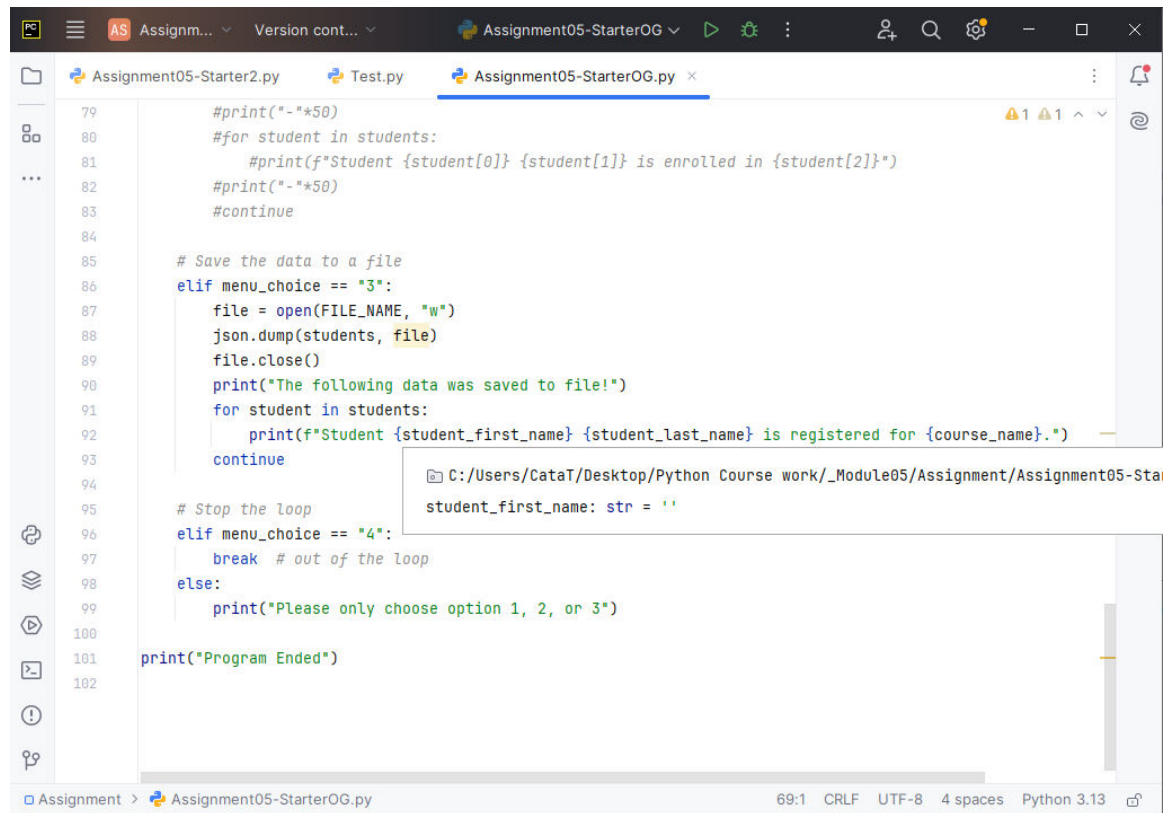
Ok so I looked back and realized the dicts at the start were pointless. I've removed them. The start has the correct formatting and it works now. The earlier screenshots look messy. This is much better. I had to fix everything. The Json file kept throwing errors, probably because it didnt NOT have starting data, the starting data just wasnt correctly formatted. I had to restart from scratch twice. Now its working again.

```
19
20 import json
21
22 # Define the Data Constants
23 FILE_NAME: str = "Enrollments.json"
24
25 # Define the Data Variables and constants
26 student_first_name: str = '' # Holds the first name of a student entered by the user.
27 student_last_name: str = '' # Holds the last name of a student entered by the user.
28 course_name: str = '' # Holds the name of a course entered by the user.
29 student_data: dict = {} # one row of student data
30 students: list = [] # a table of student data
31 json_data: str = '' # Holds combined string data separated by a comma.
32 file = None # Holds a reference to an opened file.
33 menu_choice: str # Hold the choice made by the user.
34
35
36 # When the program starts, read the file data into a list of lists (table)
37 # Extract the data from the file
38
39 try:
40     file = open(FILE_NAME, 'r')
41     students = json.load(file)
42     students.append(student_data)
43     file.close()
44 except FileNotFoundError as e:
45     print("Text file must exist before running this script!\n")
46     print(f"Built-In Python error info: {e}")
```

Im going straight to option 3 to add the dump function. On the way there I added the dictionary keys and rewrote them about 200 times, getting keyErrors while doing so. It was very fun. Heres the result:

```
54 print(MENU)
55 menu_choice = input("What would you like to do: ")
56
57 # Input user data
58 if menu_choice == "1": # This will not work if it is an integer!
59     student_first_name = input("Enter the student's first name: ")
60     student_last_name = input("Enter the student's last name: ")
61     course_name = input("Please enter the name of the course: ")
62
63     #Dictionary keys
64
65     student_data = {"FirstName": student_first_name, "LastName": student_last_name, "CourseName": course_name}
66     students.append(student_data)
67     print(f"You have registered {student_first_name} {student_last_name} for {course_name}.")
68     continue
69
70 # Present the current data
71 elif menu_choice == "2":
72     print("=" * 50)
73     #student_data = {"FirstName": student_first_name, "LastName": student_last_name, "CourseName": course_name}
74     print(f"Student {student_first_name} {student_last_name} is registered for {course_name}.")
75     print("=" * 50)
76     continue
77
78 # Process the data to create and display a custom message
79 #print("=" * 50)
80 #for student in students:
81     #print(f"Student {student[0]} {student[1]} is enrolled in {student[2]}")
```

I watched and rewatched the lab and module videos to find out why I kept getting keyErrors, and I'm still not sure why. I hope this can be elaborated on for my review. Anyways here is the option 3 requirement:



```
79     #print("-"*50)
80     #for student in students:
81         #print(f"Student {student[0]} {student[1]} is enrolled in {student[2]}")
82     #print("-"*50)
83     #continue
84
85     # Save the data to a file
86     elif menu_choice == "3":
87         file = open(FILE_NAME, "w")
88         json.dump(students, file)
89         file.close()
90         print("The following data was saved to file!")
91         for student in students:
92             print(f"Student {student_first_name} {student_last_name} is registered for {course_name}.")
93         continue
94
95     # Stop the loop
96     elif menu_choice == "4":
97         break # out of the loop
98     else:
99         print("Please only choose option 1, 2, or 3")
100
101 print("Program Ended")
102
```

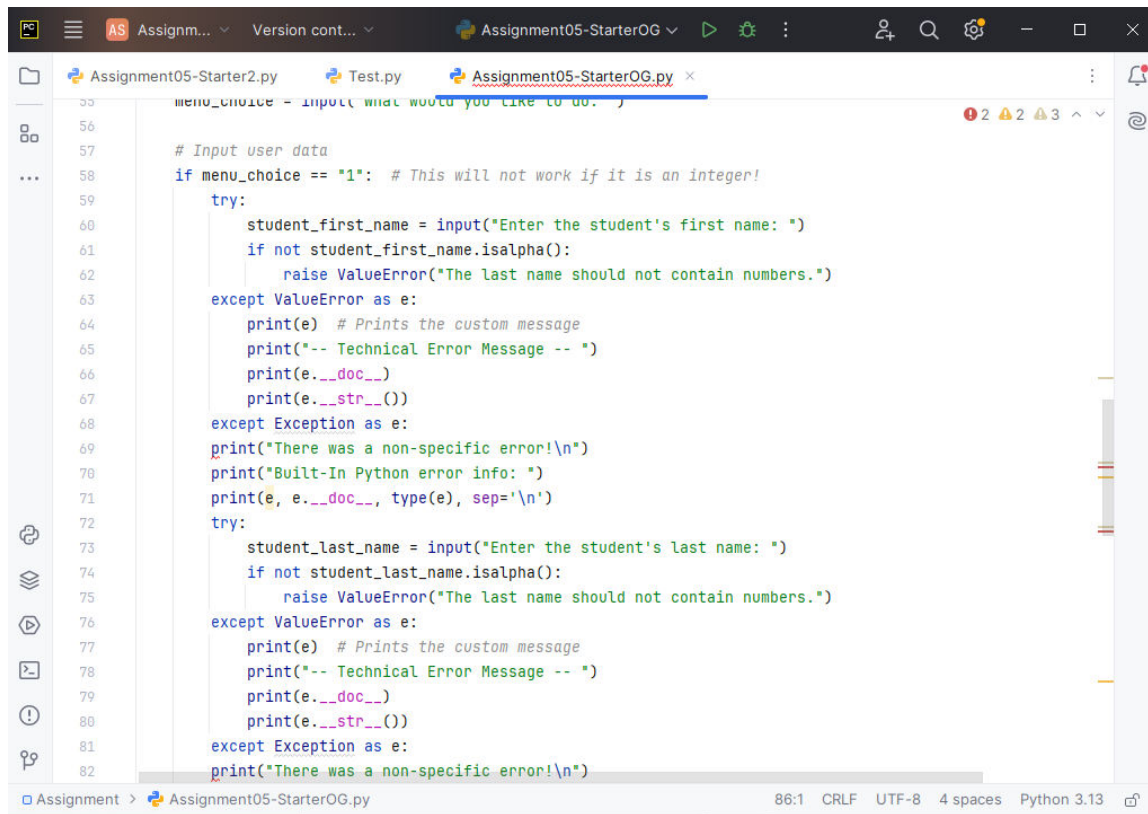
C:/Users/CataT/Desktop/Python Course work/\_Module05/Assignment/Assignment05-StarterOG.py  
student\_first\_name: str = ''

Assignment > Assignment05-StarterOG.py 69:1 CRLF UTF-8 4 spaces Python 3.13

## Action Time

Now I need error handling.





```
55 menu_choice = input("What would you like to do: ")
56
57 # Input user data
58 if menu_choice == "1": # This will not work if it is an integer!
59     try:
60         student_first_name = input("Enter the student's first name: ")
61         if not student_first_name.isalpha():
62             raise ValueError("The last name should not contain numbers.")
63     except ValueError as e:
64         print(e) # Prints the custom message
65         print("-- Technical Error Message -- ")
66         print(e.__doc__)
67         print(e.__str__())
68     except Exception as e:
69         print("There was a non-specific error!\n")
70         print("Built-In Python error info: ")
71         print(e, e.__doc__, type(e), sep='\n')
72     try:
73         student_last_name = input("Enter the student's last name: ")
74         if not student_last_name.isalpha():
75             raise ValueError("The last name should not contain numbers.")
76     except ValueError as e:
77         print(e) # Prints the custom message
78         print("-- Technical Error Message -- ")
79         print(e.__doc__)
80         print(e.__str__())
81     except Exception as e:
82         print("There was a non-specific error!\n")
```

Assignment > Assignment05-StarterOG.py 86:1 CRLF UTF-8 4 spaces Python 3.13

I added the error handling. Just for the first name and last name. Alphabetical characters only. That was much easier to parse than the KeyErrors.

Adding the last error handling:

```
106 #print("\n")
107 #continue
108
109 # Save the data to a file
110 try:
111     elif menu_choice == "3":
112         file = open(FILE_NAME, "w")
113         json.dump(students, file)
114         file.close()
115         continue
116 except TypeError as e:
117     print("Please check that the data is a valid JSON format\n")
118     print("-- Technical Error Message -- ")
119     print(e, e.__doc__, type(e), sep='\n')
120 except Exception as e:
121     print("-- Technical Error Message -- ")
122     print("Built-In Python error info: ")
123     print(e, e.__doc__, type(e), sep='\n')
124 finally:
125     if file.closed == False:
126         file.close()
127
128     print("The following data was saved to file!")
129     for student in students:
130         print(f"Student {student_first_name} {student_last_name} is registered for {course_name}.")
131     continue
132
133 # Stop the loop
```

Checking if false, checking for valid json format.

## Conclusion

Oh no.

```
Run Assignment05-StarterOG x
Student Antoneh Caboolski is registered for Math.
Student Antoneh Caboolski is registered for Math.
Student Antoneh Caboolski is registered for Math.
Student Antoneh Caboolski is registered for Math.
Student Antoneh Caboolski is registered for Math.
Student Antoneh Caboolski is registered for Math.
Student Antoneh Caboolski is registered for Math.
```

This is my achilles. Why does it duplicate the print statement? I had this issue last time. This is a new script taken from the module zip. Its really funny. Its really upsetting. Anyways error handling is not too



bad.