Assignment06_David_Levinson

David Levinson
9/3/2024
Foundations of Programming: Python
Assignment 06

https://github.com/dl-11/IntroToProg-Python-Module6.git

Introduction

This week the topics were quite big and included many conceptual changes to our code to-date. We learned about building code segments that are reusable and putting them into functions. I appreciated having the code segments written out early on, but understand the benefits of moving to reusable functions and not duplicating code. Next, return and parameters were both taught to us. Returning a value further simplifies code needed and makes it easier to understand what a function is doing. Parameters allow you to provide different values to a function when you call it. This makes that section of code even more reusable.

We also learned about classes as a way of grouping functions, variables, and constants together into portions of code. In the instructor office hours, it became clear that classes will be used in a much more impactful way than I had understood from the videos. It will be interesting to see week 7 as we learn more about them and how they can be incorporated into our student registration program.

With that, I jumped into re-writing the code for this week's assignment!

Week 6 Assignment

I reviewed the starter code for the week and updated the header, as usual. Next, I wanted to start with moving the code sections into the various functions mentioned by the assignment. Below is the first function as shown in Figure 1.

```
def read_data_from_file():
    try:
        file = open(FILE_NAME, "r")
        # JSON Answer
        students = json.load(file)
        file.close()
    except Exception as e:
        print("Error: There was a problem with reading the file.")
        print("Please check that the file exists and that it is in a json format.")
        print("-- Technical Error Message -- ")
        print(e.__doc__)
        print(e.__str__())
    finally:
        if file.closed == False:
            file.close()

# Present and Process the data
def output_menu():
    while (True):

        # Present the menu of choices
        print(MENU)
        menu_choice = input("What would you like to do: ")
```

Figure 1: First section of code to read a file converted into a function

Next, I added a call to functions that were built out at bottom of code with the general workflow. Given what we learned about not using global variables, and that ultimately they were not preferred, I didn't add global statements to each function with the intent of using them as local variables or passing them in to the function as parameters. You'll see the function calls in the new main section of code in Figure 2:

```
read_data_from_file()
output_menu()
add_data_to_table()
write_data_to_file()
exit()
```

Figure 2: Calls to the functions without parameters

Next, I knew it was important to add the parameters and return statements as we were taught in the course videos. This eliminates the need to use global in each function or to define them locally. As you see in Figure 3 below, this function has two parameters, the file name and the list of students. Because these are used elseward, such as the write file function, it makes the most sense to use them as parameters here.

```
# Extract the data from the file
def read_data_from_file(file_name: str, student_data: list):
    try:
        file = open(file_name, "r")
        # JSON Answer
        student_data = json.load(file)
        file.close()
    except Exception as e:
        print("Error: There was a problem with reading the file.")
        print("Please check that the file exists and that it is in a json format.")
        print("-- Technical Error Message -- ")
        print(e.__doc__)
        print(e.__str__())
    finally:
        if file.closed == False:
            file.close()
    return student_data
```

Figure 3: Convert functions to using parameters w/return statements

I also updated the call to the functions with the added parameters as shown in Figure 4.

```
# Beginning of the main body of this script
students = read_data_from_file(file_name=FILE_NAME, student_data=students)
```

Figure 4: Main section of code with added parameters.

I must note in my learning journey, it was quite nerve racking and while converting this code over to functions. It felt like everything was broken as shown in Figure 5. I tried to stay focused and just work on each section of code at a time. This resolved most of the problems in the code.

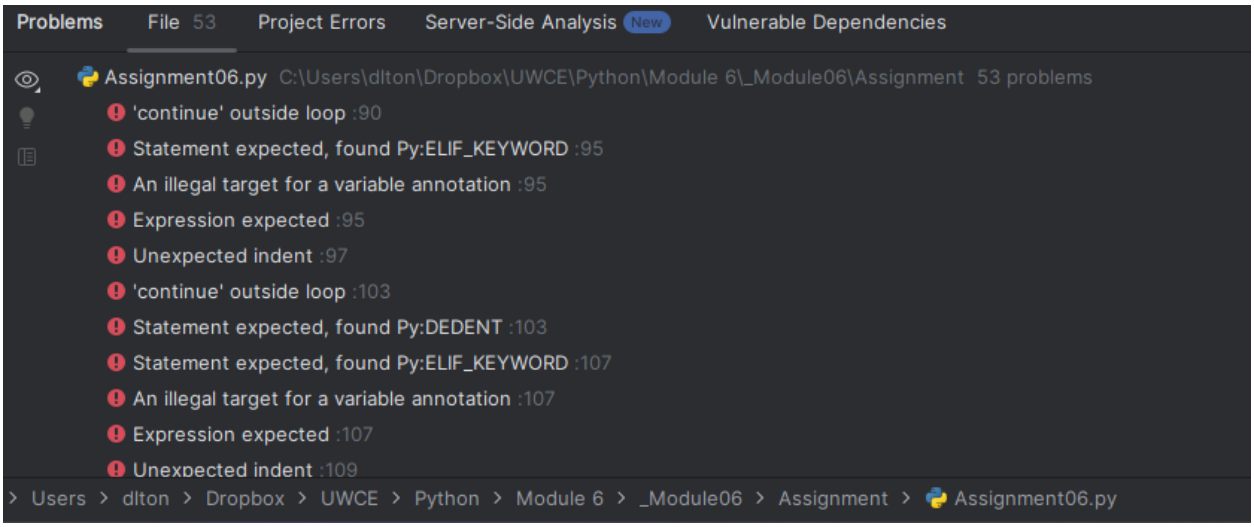

Figure 5: Everything broken! Stressed!

Once I eliminated most of the errors, the code would run but would error out still when printing the student list in option 2. I tried debugger but couldn't get a sense of what was going wrong. I spent at least an hour on this particular error. Ultimately, it turned out to be the json file. It existed, so I wasn't getting an exception, but must have gotten some wrong formatting in it. It was causing the failure in the program. I completed a new json file and it resolved it!

Classes:

With that fixed, the next day I decided to tackle adding classes to the code, along with descriptive document strings. This is shown in Figure 6 below.

```
class FileProcessor:
    """
    File processing class of functions that read and write from the JSON file
    David Levinson, 9/3/2024, added FileProcessor class
    """

    # Function to extract the data from the file
    @staticmethod
```

Figure 6: New FileProcessor class with descriptive document string

As mentioned above, I understood that the class was used to group the functions for file processing and Input/Output. I didn't yet get to clear understanding of how it will further benefit. Looking forward to that this week.

Summary

In summary, this week's new learnings were significant and functions and classes will clearly be a big part of our coding journey. I intend to spend more time reviewing the notes and videos to make sure I have a full understanding of how they can be used.

This week's code is well structured and organized now into functions and classes which I appreciate and like the cleanliness and readability of. I used Github throughout the week and completed 11 commits of the code throughout my build. This felt like a great safety net, knowing I could revert to the code at various points if I got too lost or off track!