Jeff Malick

11/27/2023

IT FDN 110 A

Assignment07

Github: https://github.com/jeffmalick/Nov292023PY

Classes and Objects - Writeup

Introduction

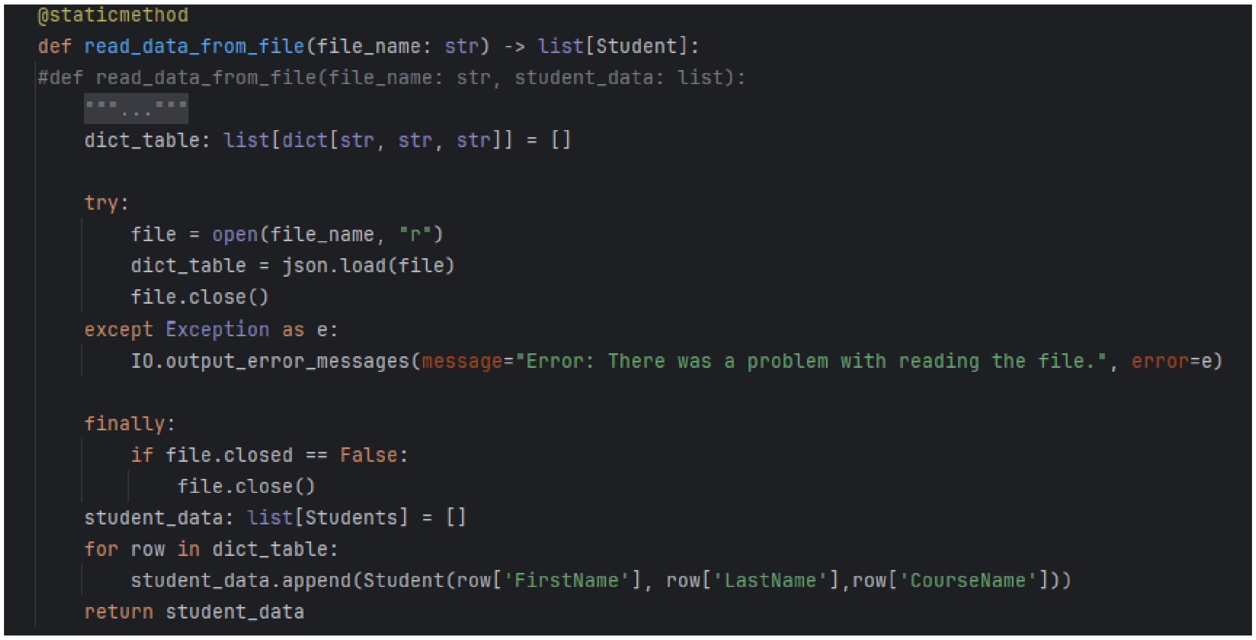
This week's assignment continues with the past assignment use case of managing student enrollments. The project aims to introduce a bit more complexity and concepts of object-oriented programming, data handling, file operations, and user input validation. The project entails creating classes with getters and setters, and learning about inheritance. As usual the program will require defining functions, organizing code, and incorporating error handling mechanisms to ensure a robust and well-structured program with a Separation of Concerns and a goal.

As a bonus, version control was introduced using Pycharm and Github. The goal was to be able to understand some of the version control features in Pycharm and how that integrates with Github (a highlight for me).

Object Oriented challenges

It is clear that Object-oriented programming (OOP) offers a powerful paradigm for software development, but it also presents challenges for novice programmers, particularly like my…self (pun intended). These challenges stem from the dynamic nature of object interactions along with learning a new syntax. Being able to track the quickly increasing complexity with my lack of confidence that I am actually doing it properly was my biggest challenge.

I was stumped for quite some time with the file read method. The added complexity of converting the json data into a list of objects became very abstract. I conceptually understood how to go about it but the syntax threw me off. I was able to get to where I thought I had working functionality but it turns out my list was empty. It was more difficult to see this issue due to the abstraction. As it turns out the problem was simply using an incorrect property name. *fig 1*



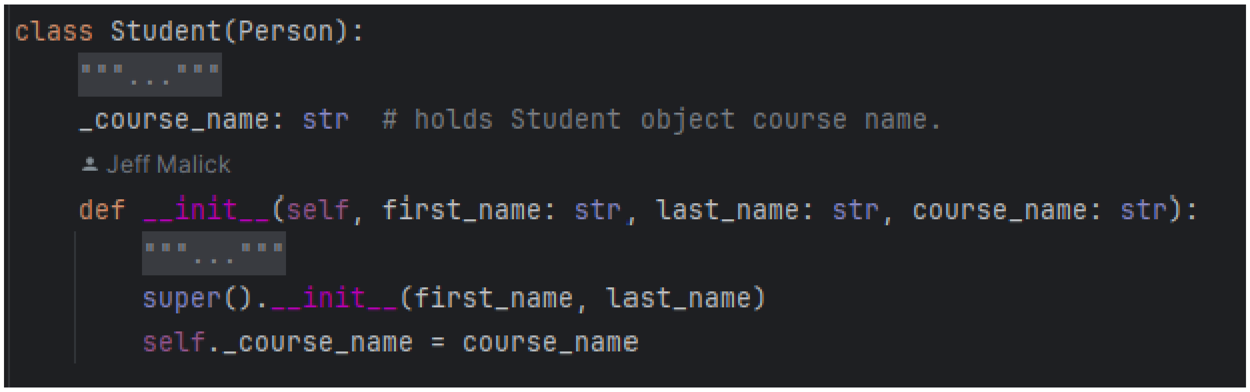
```python
@staticmethod
def read_data_from_file(file_name: str) -> list[Student]:
#def read_data_from_file(file_name: str, student_data: list):
    """..."""
    dict_table: list[dict[str, str, str]] = []

    try:
        file = open(file_name, "r")
        dict_table = json.load(file)
        file.close()
    except Exception as e:
        IO.output_error_messages(message="Error: There was a problem with reading the file.", error=e)

    finally:
        if file.closed == False:
            file.close()
    student_data: list[Students] = []
    for row in dict_table:
        student_data.append(Student(row['FirstName'], row['LastName'],row['CourseName']))
    return student_data
```

Fig 1. shows the working read file method.

Another issue this code snippet *(fig 1.*) reveals the brittleness of the file handling. the keys must match the keys from the file, shown above as "FirstName" … I know this problem could be handled with a bit more dynamic code. And I also know this is not an OOP issue specifically but it does highlight the escalating complexity due to the abstractions. Looking forward I can see the need to carefully the balance flexibility with maintainability.



```python
class Student(Person):
    """..."""
    _course_name: str  # holds Student object course name.
    Jeff Malick
    def __init__(self, first_name: str, last_name: str, course_name: str):
        """..."""
        super().__init__(first_name, last_name)
        self._course_name = course_name
```

Figure 1: shows several features of OO programing

The above code snippet shows several OOP programming features in a very compact way. This is a method, also known as the constructor, is a special method that is called when an object of a class is created. It is responsible for initializing the object's attributes and performing any necessary setup tasks. The method typically takes the object itself referred to as "self" as its first argument, and it can also take additional arguments to initialize the object's attributes.

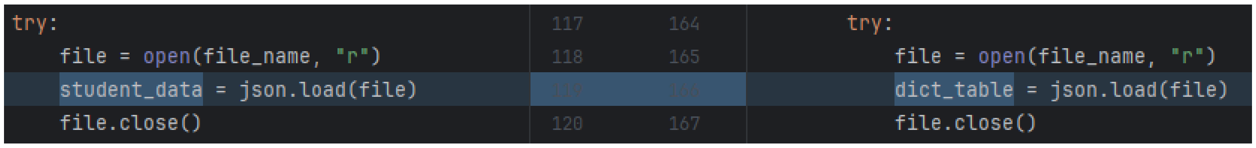
In this example, the __init__ method takes *self and* three arguments initializing the corresponding attributes of the object. This will "construct" a new object which will contain the attributes of first name, last name and course name.

However, note the line starting with a method called "supper()". This is an example of further abstraction where this object will inherit functionality from the passed in "Person" class. further encapsulating functionality. In this case we are constructing an object using this constructor to add the course name attribute and get the student name for the "Person" object

The last thing to point out in this snippet is the use of the underscore before the variable name. As in "_course_name" this is intended to limit the scope of this variable. Note this is not a strict rule but a convention. Pycharm will warn the programer if the variable is used but will not stop its use. These are just a few examples of OO features use in this assignment

Using Pycharm version control and Git integration

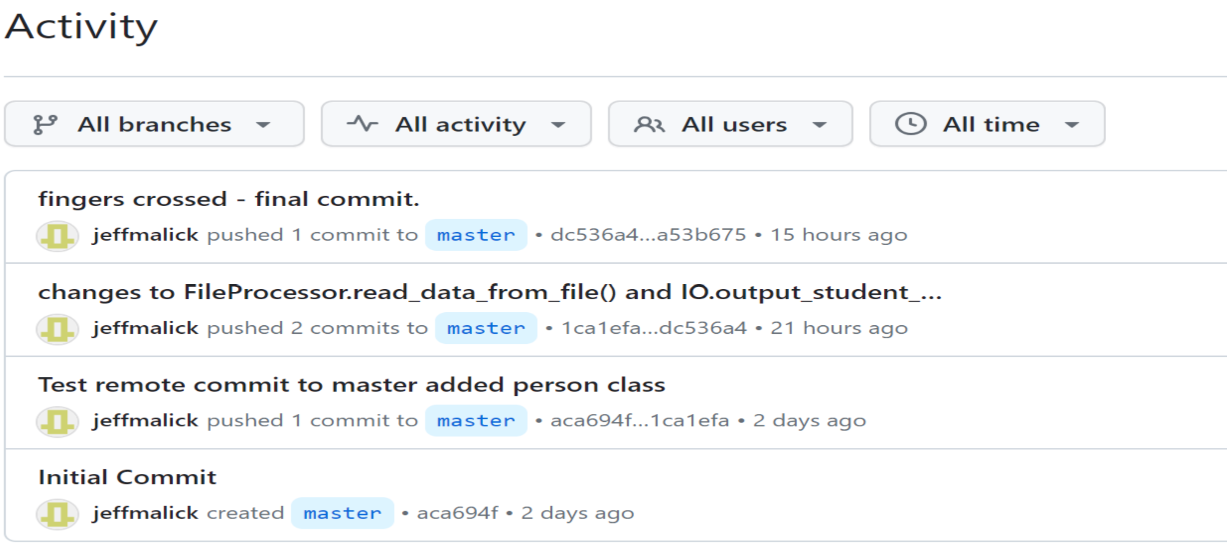
AKA how to make your life easier.



```
try:                                          try:
    file = open(file_name, "r")                   file = open(file_name, "r")
    student_data = json.load(file)                dict_table = json.load(file)
    file.close()                                  file.close()
```

fig 3. shows Pycharm built in version comparison tool

When implementing a version control system, in this case, Git. In Pycharm we can compare versions of our work Identified by commits. When the programmer gets to a certain state they may want to make sure that instance is saved. This Will allow for the comparison. The next level of this feature would be to push this version to a web-based source control system. In this case GitHub. Using GitHub allows us to share With others. Work on the same code with others and resolve conflicts and store code in an off-site location for security and safety. My new favorite pie charm feature is the ability to push code from Pycharm natively to GitHub. *fig 4*



fingers crossed - final commit.
jeffmalick pushed 1 commit to master • dc536a4...a53b675 • 15 hours ago

changes to FileProcessor.read_data_from_file() and IO.output_student_...
jeffmalick pushed 2 commits to master • 1ca1efa...dc536a4 • 21 hours ago

Test remote commit to master added person class
jeffmalick pushed 1 commit to master • aca694f...1ca1efa • 2 days ago

Initial Commit
jeffmalick created master • aca694f • 2 days ago

Fig 4. Shows commits made from Pycharm to GitHub using Pycharm

Summary

My goal for this assignment was to improve my understanding of object-oriented programming (OOP), data handling, file operations, and user input validation. I also wanted to learn more about using Pycharm's version control features and how to integrate them with Github.

I found OOP to be challenging, particularly due to the dynamic nature of object interactions and the new syntax. I also had difficulty with the file read method, as converting the JSON data into a list of objects was abstract. However, I was able to overcome these challenges by carefully tracing the data flow.

I also found Pycharm's version control features to be very helpful. I was able to use it to compare different versions of my code and push my work to Github. I found this to be a valuable tool and can see the value for collaboration and code management.

Overall, I found this assignment to be challenging but rewarding. I learned a lot about OOP, data handling, file operations, user input validation, and version control. I also gained experience using Pycharm and Github.