Andrew Hopkins
January 25, 2024
IT FDN 110 A
Assignment 05
https://github.com/afhop18/IntroToProg-Python-Mod05

Advanced Collections and Error Handling

Introduction

I would consider this week's material and assignment to be, conceptually, the most difficult for me so far due to the more complex nature of the concepts that we were working with but also the sheer amount of new information, even if some of it was building on previous concepts. I had to learn about, and how to work with, Dictionaries, JSON Files, Error Handling, and GitHub. Dictionaries weren't too bad. The structure is obviously more complicated than a simple list and it takes a little bit more effort to get it exactly right, but the benefits of the dictionary structure clearly outweigh that of a list of a tuple. JSON files were a revelation to me. I've worked with Excel / CSV files my entire life so those are what I was most familiar with. I was aware that JSON files existed but really had no knowledge of their structure or purpose until this lesson. I couldn't be happier to have learned about them because from a coding perspective they made so much more sense than working with CSV files in more complex situations. I was also pleased to learn that the Python Dictionary format and JSON formats were the same. Error handling took me the most time to wrap my head around during this assignment. There were different aspects of the Try/Except loops that were shared with other concepts I have learned so far and it took me a second to recognize the similarities but process how the code worked in this new situation, as well as making sure that each line required for it went into the right place. GitHub was also a fun segment for me. Again, it was one of those things that I was aware of the existence of but had never used so I'm glad that I have a chance to start working with it.

Writing the Script.

Establishing Variables and Constants

Declaring the variables and the constants for this script was fairly straightforward and easy as most of them were provided in the assignment starter file. The major updates / changes that needed to occur were that instead of a list of lists like in the previous assignment I needed to declare a dictionary and a list of dictionaries because I would be interacting with JSON files. This wasn't really any different from declaring other variable types to null values in the past. Other small updates were changing the file name to be .JSON instead of .CSV and updating a string variable name to be more clear about what was being stored in it.

```
# Define the Data Constants
MENU: str = '''
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------
'''
# Define the Data Constants
FILE_NAME: str = "Enrollments.json"

# Define the Data Variables and constants
student_first_name: str = ''  # Holds the first name of a student entered by the user.
student_last_name: str = ''  # Holds the last name of a student entered by the user.
course_name: str = ''  # Holds the name of a course entered by the user.
json_data: str = ''  # Holds the data for a json file.
file = None  # Holds a reference to an opened file.
menu_choice: str = ''  # Hold the choice made by the user.
student_data: dict = {}  # Dictionary of student data.
students: list = []  # a table of student data
```

Figure 1: Establishing constants and Variables for the script.

Reading in data from a JSON File

Reading in data to a file from JSON has got to be one of the best pieces of code I've gotten to write so far. So long as I recognize that the data I'm reading in is going to be in Python's dictionary format and I store it appropriately, it's as simple as importing JSON at the beginning of the program, opening the file, saving data to a list of dictionaries using the json.load() function, then closing the file. This is so much simpler than iterating through a file with a for-in loop.

```
import json

file = open(FILE_NAME, "r")
students = json.load(file)
file.close()
```

Figure 2: Reading in a JSON file.

Saving Data to a Dictionary

Saving input data to a dictionary and a list of dictionaries was also fairly straightforward as the structure of the code was more or less the same as code I have written for other assignments. The difference that I experienced in saving data to a dictionary was that it was slightly more complicated than saving it to a list or a truple because of the dictionary structure and required syntax to get it just right. One thing that is particularly hard is that having to reference the key

and value pairs when you are trying to save new data to it because you have to have the exact string on hand for the keys otherwise it won't work. I can also see this format getting incredibly lengthy if you have more than just a couple of key/value pairs. Storing dictionaries to a list of dictionaries thankfully was as simple as appending with the dictionary.

```
# Input user data
if menu_choice == "1":  # This will not work if it is an integer!
    student_first_name = input("Enter the student's first name: ")
    student_last_name = input("Enter the student's last name: ")
    course_name = input("Please enter the name of the course: ")
    json_data += f'{student_first_name}, {student_last_name}, {course_name}\n'
    student_data = {"First Name": student_first_name, "Last Name": student_last_name, "Course": course_name}
    students.append(student_data)
    continue
```

Figure 3: Saving user input to a dictionary and a list of dictionaries.

Writing to a JSON File

Again, working with JSON files in Python is such a treat because of how simple it is. And again, as long as I am cognizant of the fact that what I'm going to be writing to the JSON file is in the correct format (Python Dictionary), and in this case I am certain that the list of data I'm passing is a list of dictionaries, then saving this data to the file is just a matter of opening the file in write mode, utilizing the json.dump() function, then closing the file.

```
# Save the data to a file
elif menu_choice == "3":
    file = open(FILE_NAME, "w")
    json.dump(students, file)
    file.close()
    print("The following data was saved to file!")
    print(json_data)
    continue
```

Figure 4: Writing user input to a JSON file.

Handling Exceptions

Dealing with custom exceptions in the assignment has to be the most complicated concept that I've work with so far. Using Try-Except loops to handle exceptions was difficult because they work like a combination of other loops but don't actually add any real substance to the code, and they can't just nicely wrap around a block of code, they have to be integrated neatly into it. The most complicated exception handling I had to do for the assignment was around the user input for entering the first and last names. I just took this step by step to make sure that each line was in the right spot. I started by making sure to enter the if loop with the menu selection, immediately followed by a try statement. Within the try statement I created a specific error handling exception for both the first and the last name variables. In both of these situations if the input from the user was not strictly alphabetical then it had to raise a specific Value Error with a stored string. If the code made it past this point then the remainder of the menu 2 selection would run as intended,

but if a Value Error was raised due to incorrect input from the first or last name then it would have to jump out of the try loop, pass the ValueError string to the Except statements and print that passed information, as well as a few other piece of data, via something similar to a for-in loop (but with exceptions).

```
# Input user data
if menu_choice == "1":  # This will not work if it is an integer!
    try:
        student_first_name = input("Enter the student's first name: ")
        if not student_first_name.isalpha():
            raise ValueError("The student's first name should only contain letters")

        student_last_name = input("Enter the student's last name: ")
        if not student_last_name.isalpha():
            raise ValueError("The student's last name should only contain letters")

        course_name = input("Please enter the name of the course: ")
        json_data += f'{student_first_name}, {student_last_name}, {course_name}\n'
        student_data = {"First Name": student_first_name, "Last Name": student_last_name, "Course": course_name}
        students.append(student_data)
    except ValueError as e:
        print("-- Technical Error Message -- ")
        print(e.__doc__)
        print(e.__str__())
    except Exception as e:
        print("There was a non-specific error!\n")
        print("-- Technical Error Message -- ")
        print(e, e.__doc__, type(e), sep='\n')

    continue
```

Figure 5: Example of handling exceptions during user input.

Saving to GitHub

This was the first assignment where I have had to save my files to GitHub. To do that I followed the instructions provided as best as I could. I created a new repository under my owner name, gave the repository a description and made it public, made sure to include a README file, uploaded my Assignment.py file and committed the change, then verified that the file had been saved. I will upload my knowledge document as well but I had to record this process with the .py file to add it into the knowledge document.

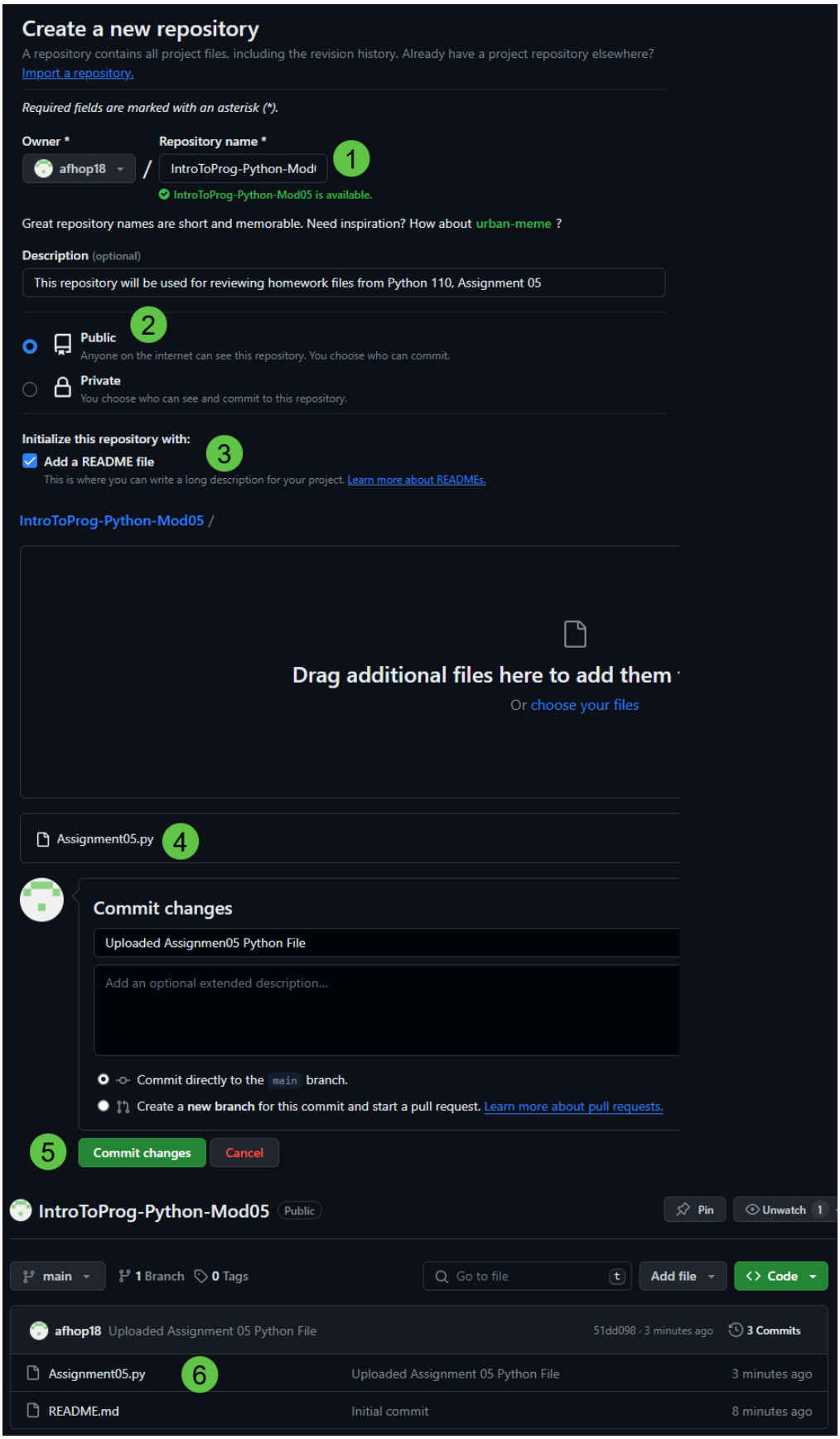
Create a new repository
A repository contains all project files, including the revision history. Already have a project repository elsewhere?
Import a repository.
Required fields are marked with an asterisk (*).
Owner *
Repository name *
afhop18
IntroToProg-Python-Mod
1
IntroToProg-Python-Mod05 is available.
Great repository names are short and memorable. Need inspiration? How about urban-meme ?
Description (optional)
This repository will be used for reviewing homework files from Python 110, Assignment 05
2
Public
Anyone on the internet can see this repository. You choose who can commit.
Private
You choose who can see and commit to this repository.
Initialize this repository with:
3
Add a README file
This is where you can write a long description for your project. Learn more about READMEs.
IntroToProg-Python-Mod05 /
Drag additional files here to add them
Or choose your files
Assignment05.py
4
Commit changes
Uploaded Assignmen05 Python File
Add an optional extended description...
Commit directly to the main branch.
Create a new branch for this commit and start a pull request. Learn more about pull requests.
5
Commit changes
Cancel
IntroToProg-Python-Mod05
Public
Pin
Unwatch 1
main
1 Branch
0 Tags
Go to file
t
Add file
Code
afhop18 Uploaded Assignment 05 Python File
51dd098 · 3 minutes ago
3 Commits
Assignment05.py
6
Uploaded Assignment 05 Python File
3 minutes ago
README.md
Initial commit
8 minutes ago

Figure 6: Saving my assignment file to GitHub.

Notes

- When producing the output for menu selection 2 I opted to print a string of data already saved because it's simpler code and easier to write than iterating through and printing from a dictionary or list of dictionaries. That and the format of the string is already clean
- An observation: reading a JSON file after you have written data to it is not fun. It's just a single line of data so if you have more than a few rows of dictionaries you're going to be scrolling sideways forever. It would be nice to know how to read those files in such a way that new sets of data are printed with carriage returns

Summary

Overall this module and assignment was the most difficult for me so far as far as fully understanding the new concepts that were being introduced as well as the volume of brand new material. Despite both of those things the value of the information that I learned is probably the most useful so far. Dictionaries, while a little weird in structure have huge potential, especially with JSON files which were a big revelation for me as far as file types go. Error handling wasn't particularly fun. I understand the utility of it of course but it seems strangely complicated for it's purpose. And it was nice to finally learn and start to use GitHub which I know will become an extremely useful tool for me in the future.