

Name: DS

Date: February 18, 2026

Course: IT FDN 110 GP Wi 26: Foundations of Python Programming

GitHub repository: <https://github.com/uw1p/IntroToProg-Python-Mod05>

# Assignment 05 – Advanced Collections and Error Handling

## Introduction:

This is the fifth assignment, involving exception handling, dictionaries, json files, and using GitHub. The assignment is to be posted and peer reviewed on GitHub in addition to submission for grading on Canvas.

## Creating the Script:

I started with the provided starter script, which included a standardized version of code for Assignment 4. Unlike in Assignment 4, we are reading from and writing to a json file, and using dictionaries inside a list instead of a list of lists, and requires error checking of the file reading and writing. Accordingly I changed the initial reading of the file to use code appropriate to a json file:

```
try:
    file = open(FILE_NAME, "r")
    students = json.load(file)
except FileNotFoundError as e:
    print(f"{FILE_NAME} not found.")
except Exception as e:
    print(f"There was an error opening the file {FILE_NAME}.")
    print(e,e.__doc__)
finally:
    if file is not None and file.closed == False:
        file.close()
```

Similarly I placed the write code in option 3 within a try loop and changed the file writing to use the json.dump() command:

```
file = open(FILE_NAME, "w")
json.dump(students, file)
```

For option 1, the assignment requested the addition of error checking on name entry to require only letters in the first and last names. This seems like a bad idea in practice, since

there are names that contain non-alphabetic characters, most commonly hyphens, spaces, and apostrophes, but in accordance with the assignment requirements, I added the following code to the first name entry section, and nearly identical code for last name entry:

```
if menu_choice == "1":  
    try:  
        student_first_name = input("Enter the student's first name: ")  
        if not student_first_name.isalpha(): #This is a bad assumption for names.  
            raise ValueError("First name must only contain letters.")  
    except ValueError as e:  
        print("Invalid first name entry. Continuing.")
```

The data are then saved to a dictionary line within a list:

```
student_data = {"FirstName": student_first_name, "LastName": student_last_name,  
"CourseName": course_name}  
students.append(student_data)
```

Since the assignment used a list of dictionaries instead of a list of lists and the assignment requested printing the data in comma separated format, I modified the print statements in options 2 and 3 to present the data using:

```
print("-"*50)  
print("student first name, student last name, course name") # print headers  
for student in students:  
    print(f"{student['FirstName']}, {student['LastName']}, {student['CourseName']}")  
print("-"*50)
```

The majority of the remaining code remained similar to Assignment 4.

## Testing:

I ran the completed code in a file called Assignment05.py and found that the error reading properly displayed error codes, but because of the “except ValueError as e:” statements it continued running despite the error notifications. This included saving names containing non-alphabetic characters despite issuing an error statement. After attending the weekly office hours I adjusted the name entry code to prevent this:

```
try:  
    student_first_name = input("Enter the student's first name: ")  
    if not student_first_name.isalpha():  
        raise ValueError("First name must only contain letters.")  
    student_last_name = input("Enter the student's last name: ")  
    if not student_last_name.isalpha():  
        raise ValueError("Last name must only contain letters.")  
    else:  
        course_name = input("Please enter the name of the course: ")  
        student_data = {"FirstName": student_first_name,  
                      "LastName": student_last_name, "CourseName": course_name}  
        students.append(student_data)
```

```
        print(f"You have registered {student_first_name} {student_last_name} for  
{course_name}.")  
        continue  
    except ValueError as e:  
        print(f"Invalid name entry. {e} Continuing.")
```

Removing the json file resulted in the expected “file not found” exception, but did not stop the program from running. Since the program also continues after this error the file will be created on saving the data using menu option 3. Similarly when I provided incorrectly formatted data in the json file it did not import any data but continued and overwrote the bad file on choosing option 3 from the menu.

## Takeaways:

I learned the following about Python and the PyCharm IDE:

1. In a dictionary “keys” are like headers, providing labels for each data item.
2. JSON files include identifiers with each element, with each row in curly brackets with headers ahead of the elements.
3. Errors in Python are called exceptions.
4. GitHub is used for sharing code.
5. TextIO Wrappers include the pointers used in working with files
6. The use of “is not” instead of != is useful in some commands.
7. “try/except/finally” statements work well for structured error handling
8. Writing from a dictionary to a json file is accomplished with dump statements.

## Summary:

I learned about exceptions as well as the use of dictionaries and json files in Python, but this assignment seems to raise additional questions about when to have a program continue despite an error, or if we should give an option to either retry or exit. In order to upload to GitHub at <https://github.com/uw1p/IntroToProg-Python-Mod05> I first removed my name from this assignment page since I did not want my name posted on GitHub.