Jett Magnuson

November 13, 2025

Dictionaries and Exceptions Handling

Assignment05

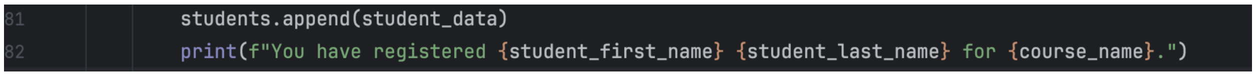
Dictionaries and Exception Handling

Introduction

This assignment expands on assignment04 by introducing dictionaries and exception handling to the course registration program. Instead of using lists to store student information, the program now uses dictionaries with named keys. Additionally, the program includes structured error handling to manage potential issues like missing files or invalid user input.

Using Dictionaries

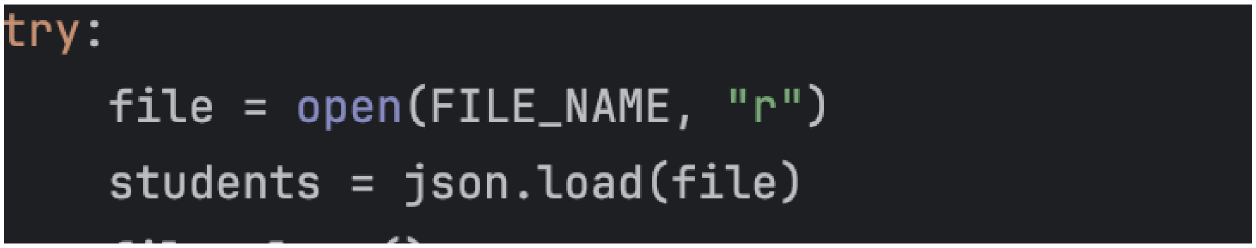
The main change in this assignment was converting lists to dictionaries for storing student data. Previously, student information was stored as a list like {first_name, last_name, course_name], but now it uses dictionary format: {“FirstName”: “Jett”, “LastName”: “Magnuson”, “CourseName”: “Python101”}. Dictionaries make the code more readable because you access data using meaningful keys like student[“FirstName’] instead of remembering that student[0] is the first name. The program stores multiple student dictionaries in a list called students, creating a two-dimensional data structure.



```
        students.append(student_data)
        print(f"You have registered {student_first_name} {student_last_name} for {course_name}.")
```

Working with JSON Files

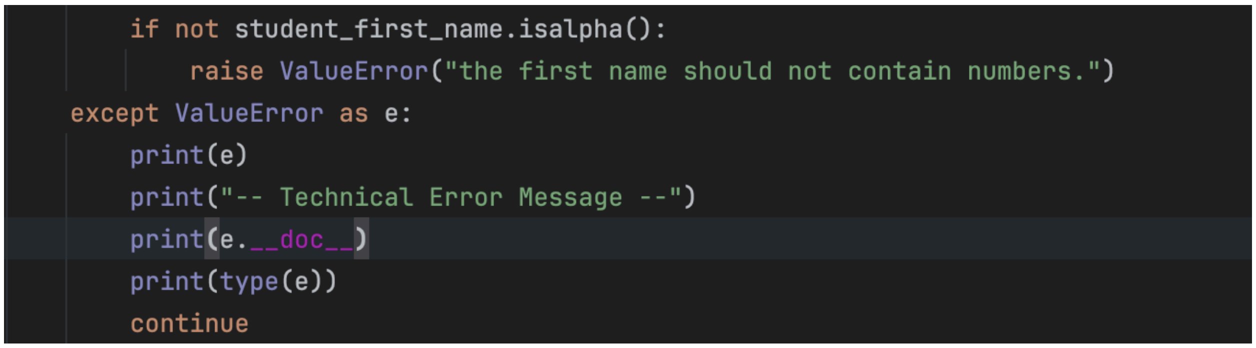
This assignment required using JSON format instead of CSV for file storage. To my knowledge, JSON works naturally with python dictionaries and lists, making it easier to save and load structured data. The program uses json.load() to read data from the “Enrollments.json” when it starts and json.dump() to write data back to the files when the user chooses to save. The JSON file format preserves the dictionary structure, so when I resopen the program, all the data loads back exactly as it was saved.



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

Exception Handling

Exception handling prevents the program from crashing when an error occurs. The program includes try-except-finally blocks in the three key areas, reading the file on startup, validating the user input for names, and writing data to the file. When reading the file, a FileNotFoundError exception catches this situation where “Enrollments.json” doesn’t exist yet, For user input, the program uses ValueError exceptions to catch names that contain numbers, using the isalpha() method to validate that only letters are entered. The finally block ensures that files are properly closed even if an error occurs during reading or writing.



```
        if not student_first_name.isalpha():
            raise ValueError("the first name should not contain numbers.")
    except ValueError as e:
        print(e)
        print("-- Technical Error Message --")
        print(e.__doc__)
        print(type(e))
        continue
```

Conclusion

This Assignment taught me how to use dictionaries to organize data more clearly and how to implement exception handling to make more programs more reliable. Using dictionaries with meaningful key names makes the code easier to understand and maintain compared to using lists with index numbers. Exception handling is essential for creating professional programs that handle errors gracefully instead of crashing unexpectedly. These skills will be valuable for building more complex programs that need to manage data and handle real world situations where things don't go as planned.