Name: Nelida Romo Tijerina
Date: November 09, 2023
Course: IT FDN 110 B Au 23: Foundations of Programming: Python

Assignment 05 – Advanced Collections and Error Handling

Introduction

The objective of this assignment is to create a Python program that demonstrates using constants, variables, and print statements to display a message about a student's registration for a Python course.

This program is similar to Assignment03, but it also includes: **the use of data processing using dictionaries and error handling.**

This document includes a description of the challenges that I faced while coding this assignment.

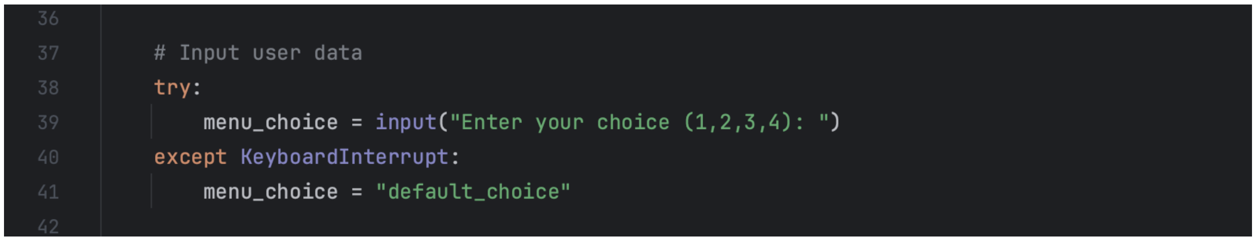
Topics included in this document are about challenges that I had while creating the dictionary from user's input and displaying the values in the dictionary and converting this data to print a coma separated string.

The program also allows to register multiple students and display them and save them from the dictionary to a JSON file and read from the JSON file to construct the dictionary again.

The program also provides structured error handling when the file is read into the list of dictionary rows and when the dictionary rows are written to the file.

1. Handling errors with exceptions from user menu-choices

At the very beginning of this assignment, I added protection to handle errors when the user inputs menu-choices. In the Assignment04, I had problems when the program was expecting the menu-choice and the program was abruptly ending. The type of exception was for keyboard-interruptions. **Figure 1**.



```python

    # Input user data
    try:
        menu_choice = input("Enter your choice (1,2,3,4): ")
    except KeyboardInterrupt:
        menu_choice = "default_choice"

```

Figure 1. KeyboardInterrrupt Exception Handling

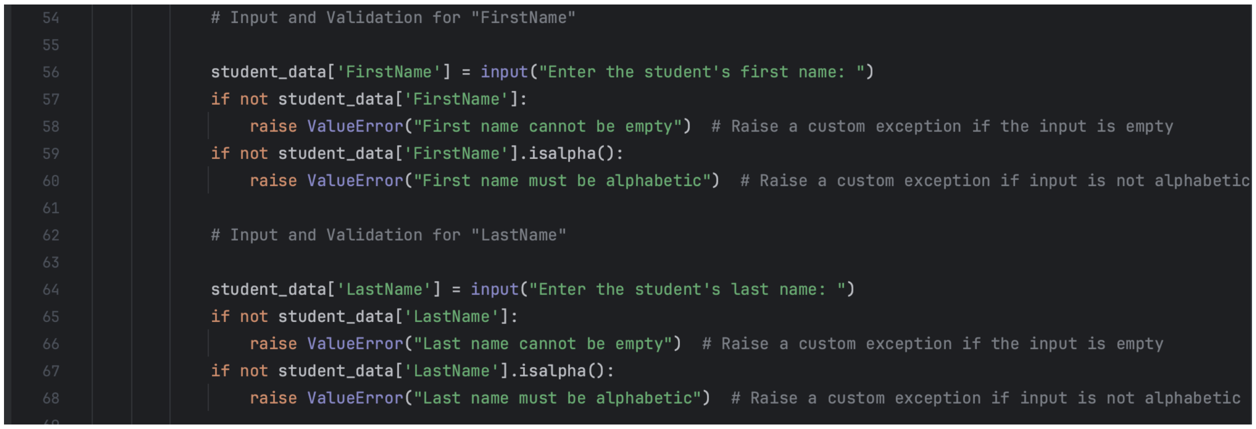
2. Handling errors with exceptions for First Name and Last Name

As required for this assignment, what I did was to protect the program in case of empty inputs, for first name and last name, raising a customized exception for this.

Also, another with **ValueError** that is an exception type that is raised when a function an argument of the correct type of data but it is not an expected or appropriate value.

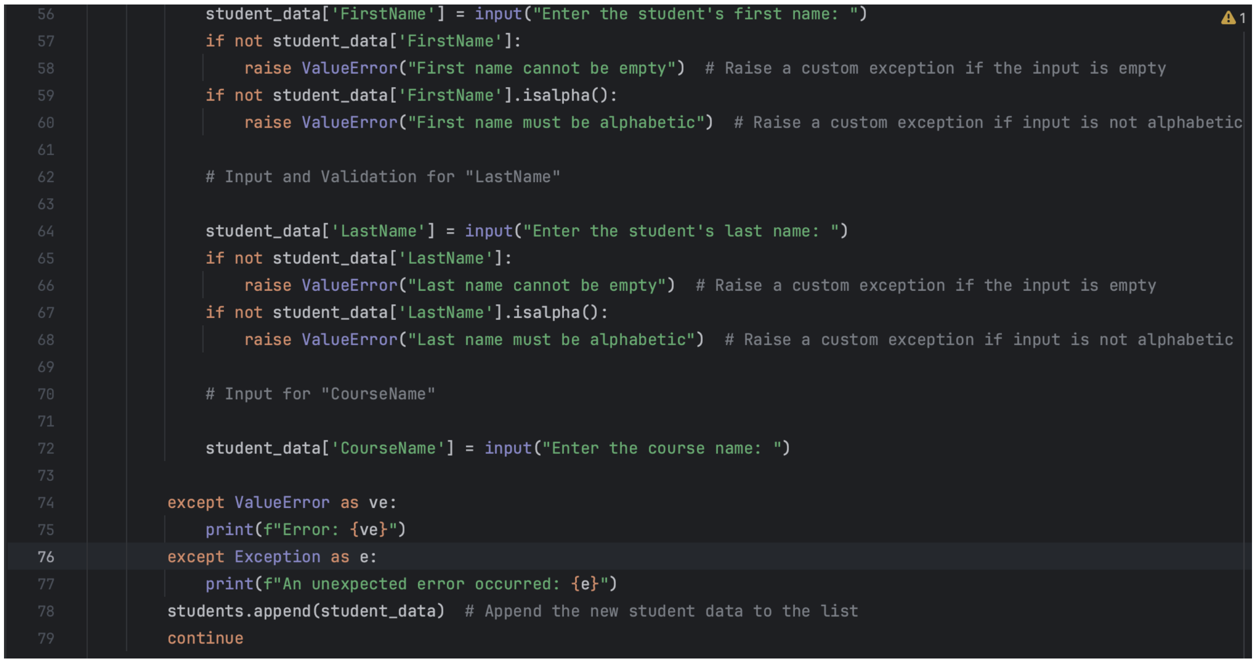
Finally, with **Exception** to catch any other possible type of error while executing this part of the program when the user inputs first name and last name.

Figure 2a and Figure 2b included below, are showing the code that I wrote to do that.



```python
        # Input and Validation for "FirstName"

        student_data['FirstName'] = input("Enter the student's first name: ")
        if not student_data['FirstName']:
            raise ValueError("First name cannot be empty")  # Raise a custom exception if the input is empty
        if not student_data['FirstName'].isalpha():
            raise ValueError("First name must be alphabetic")  # Raise a custom exception if input is not alphabetic

        # Input and Validation for "LastName"

        student_data['LastName'] = input("Enter the student's last name: ")
        if not student_data['LastName']:
            raise ValueError("Last name cannot be empty")  # Raise a custom exception if the input is empty
        if not student_data['LastName'].isalpha():
            raise ValueError("Last name must be alphabetic")  # Raise a custom exception if input is not alphabetic
```

Figure 2a. ValueError and Exception for first name and last name



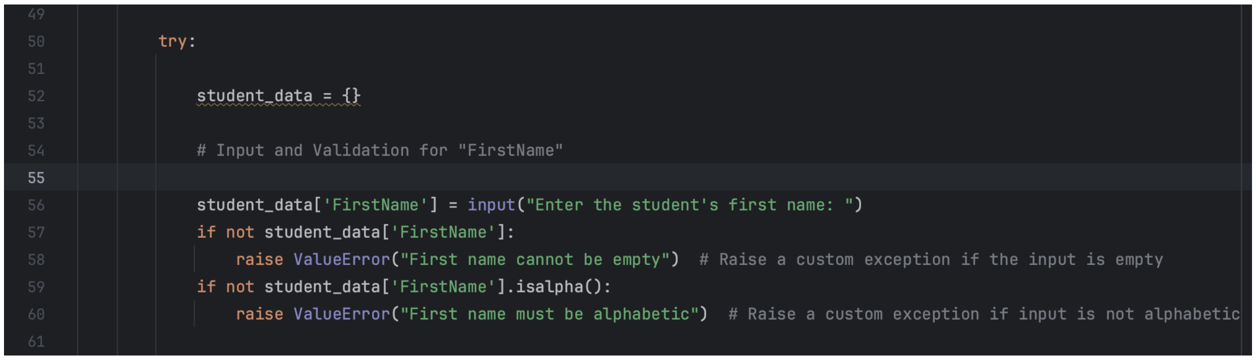
```python
        student_data['FirstName'] = input("Enter the student's first name: ")
        if not student_data['FirstName']:
            raise ValueError("First name cannot be empty")  # Raise a custom exception if the input is empty
        if not student_data['FirstName'].isalpha():
            raise ValueError("First name must be alphabetic")  # Raise a custom exception if input is not alphabetic

        # Input and Validation for "LastName"

        student_data['LastName'] = input("Enter the student's last name: ")
        if not student_data['LastName']:
            raise ValueError("Last name cannot be empty")  # Raise a custom exception if the input is empty
        if not student_data['LastName'].isalpha():
            raise ValueError("Last name must be alphabetic")  # Raise a custom exception if input is not alphabetic

        # Input for "CourseName"

        student_data['CourseName'] = input("Enter the course name: ")

    except ValueError as ve:
        print(f"Error: {ve}")
    except Exception as e:
        print(f"An unexpected error occurred: {e}")
    students.append(student_data)  # Append the new student data to the list
    continue
```

Figure 2b. ValueError and Exception for first name and last name

3. Register a Student for a Course – Menu Option 1

In this part the challenge that I had is that some times when I entered data for a student here, the student appeared duplicated in the list students, after saving and then selecting Menu Option 2.

What I did in this section that solved the problem is that I created and empty dictionary for each student registered, including this line inside the loop for each iteration. Please see Line 52 in **Figure 3**.



```
49
50          try:
51
52              student_data = {}
53
54              # Input and Validation for "FirstName"
55
56              student_data['FirstName'] = input("Enter the student's first name: ")
57              if not student_data['FirstName']:
58                  raise ValueError("First name cannot be empty")  # Raise a custom exception if the input is empty
59              if not student_data['FirstName'].isalpha():
60                  raise ValueError("First name must be alphabetic")  # Raise a custom exception if input is not alphabetic
61
```

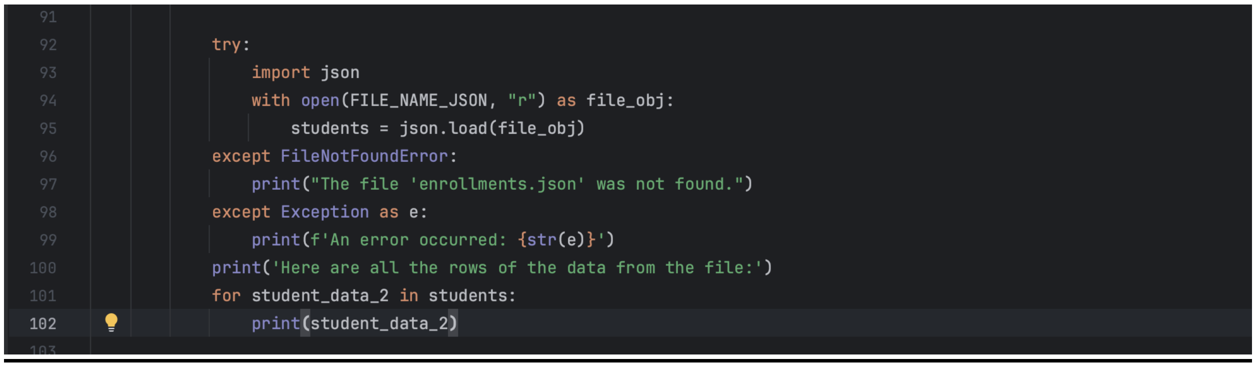
Figure 3. Setting to empty the dictionary and validations for first and last names

Also, I realized that when I skyped first name, while registering a student, and only entered last name and course name, in the list students, appeared the previous first name for the previous student and last name was correct, however that was not working.

To solve this, I decided to raise an exception when first or last name are empty. Please see Lines 57 to 60 in **Figure 3** above**.**

4. <u>Presenting Data – Menu Option 2</u>

For me what worked well to open and read the JSON file, was using the statement in Line 94 illustrated in **Figure 4**.



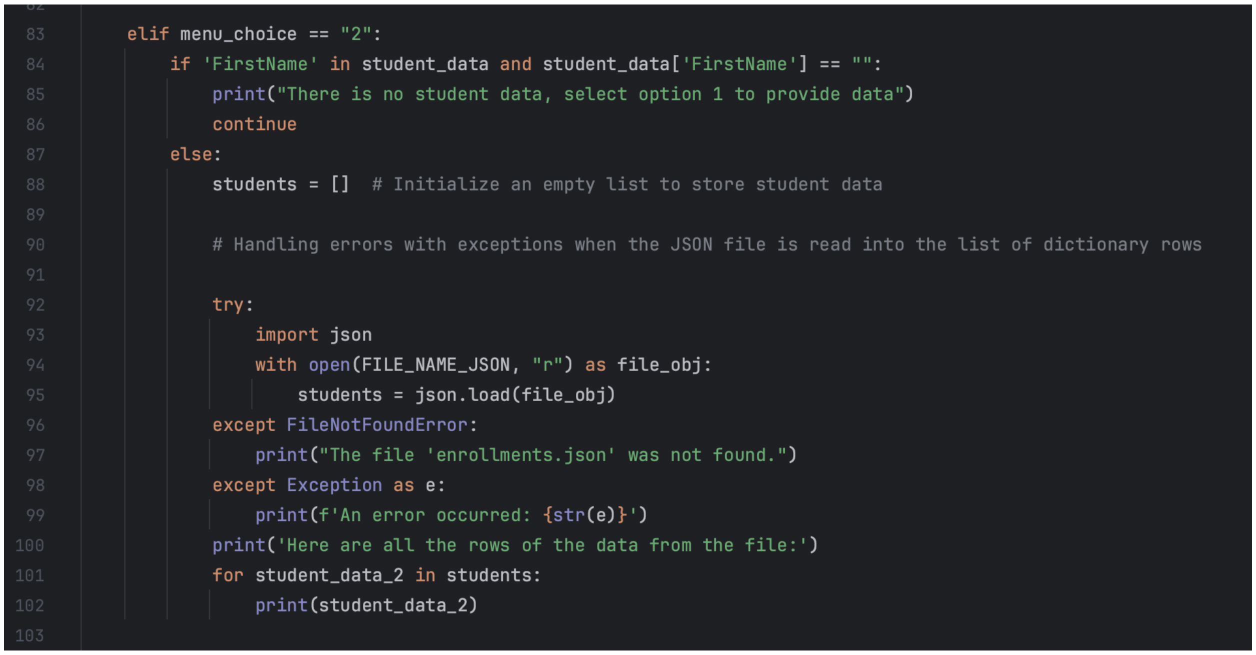
```
91
92              try:
93                  import json
94                  with open(FILE_NAME_JSON, "r") as file_obj:
95                      students = json.load(file_obj)
96              except FileNotFoundError:
97                  print("The file 'enrollments.json' was not found.")
98              except Exception as e:
99                  print(f'An error occurred: {str(e)}')
100             print('Here are all the rows of the data from the file:')
101             for student_data_2 in students:
102                 print(student_data_2)
103
```

Figure 4. Open the JSON file to read and validation

In this section another challenge that I had was that when the user selected Menu Option 1 then Menu Option 3, that worked perfect. However, when selecting Menu Option 1 then Option 2 and later Option 3, the data from the user for the last student registered was lost.

You can see my code for this section in **Figure 5.**

Also, as a requirement for this assignment, I validated that there was a JSON file, handling the error by exceptions, that code is illustrated in **Figure 5** as well.



```
    elif menu_choice == "2":
        if 'FirstName' in student_data and student_data['FirstName'] == "":
            print("There is no student data, select option 1 to provide data")
            continue
        else:
            students = []  # Initialize an empty list to store student data

            # Handling errors with exceptions when the JSON file is read into the list of dictionary rows

            try:
                import json
                with open(FILE_NAME_JSON, "r") as file_obj:
                    students = json.load(file_obj)
            except FileNotFoundError:
                print("The file 'enrollments.json' was not found.")
            except Exception as e:
                print(f'An error occurred: {str(e)}')
            print('Here are all the rows of the data from the file:')
            for student_data_2 in students:
                print(student_data_2)
```

Figure 5. Menu Option 2: Presenting Data

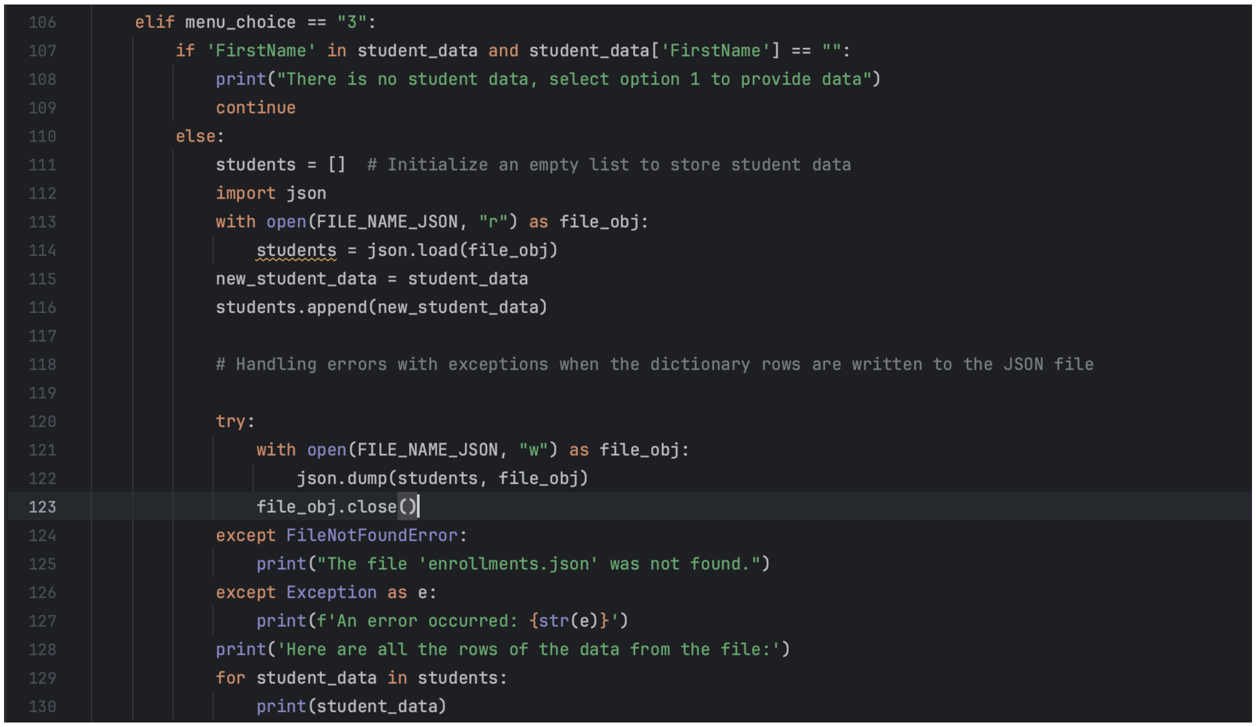
5. <u>Save data to a file and print data from the JSON file – Menu Option 3</u>

For this section my challenge was that I was not saving the last information in the JSON file also I was not retrieving data from the JSON file. My code was not working.

My first solution was to initialize my list students to empty at the beginning of the code for this section, Line 111 in **Figure 6**. That solved only some of the issues.

Also, the solution for me was to read from and write to a JSON file using the commands showed in Lines 113 and 121 in **Figure 6** below.
It is worth to mention that I realized that before saving, I needed to load the existing data from the JSON file, reading from it, then append the new student data to the list, writing that information the user recently provided.

In **Figure 6** below you can see the code for this section.

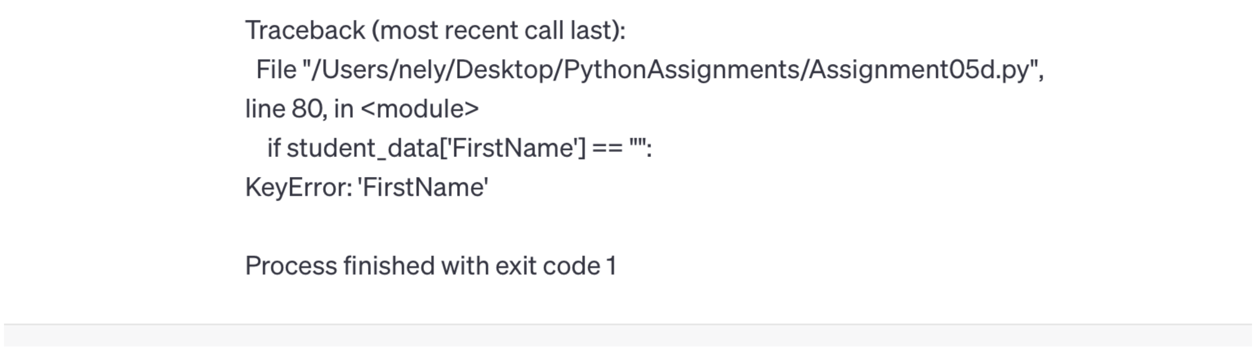


```
106         elif menu_choice == "3":
107             if 'FirstName' in student_data and student_data['FirstName'] == "":
108                 print("There is no student data, select option 1 to provide data")
109                 continue
110             else:
111                 students = []  # Initialize an empty list to store student data
112                 import json
113                 with open(FILE_NAME_JSON, "r") as file_obj:
114                     students = json.load(file_obj)
115                 new_student_data = student_data
116                 students.append(new_student_data)
117 
118                 # Handling errors with exceptions when the dictionary rows are written to the JSON file
119 
120                 try:
121                     with open(FILE_NAME_JSON, "w") as file_obj:
122                         json.dump(students, file_obj)
123                     file_obj.close()
124                 except FileNotFoundError:
125                     print("The file 'enrollments.json' was not found.")
126                 except Exception as e:
127                     print(f'An error occurred: {str(e)}')
128                 print('Here are all the rows of the data from the file:')
129                 for student_data in students:
130                     print(student_data)
```

Figure 6. Menu Option 3 – Save information to a JSON file

Finally, I had also a Traceback error showed **Figure 7** below. This error was related with the corrected Line 107 in **Figure 6** above**.**

My code was checking if student data first name is empty but it could not find the key in the dictionary. This is why I added, if "FirstName" to the if condition.



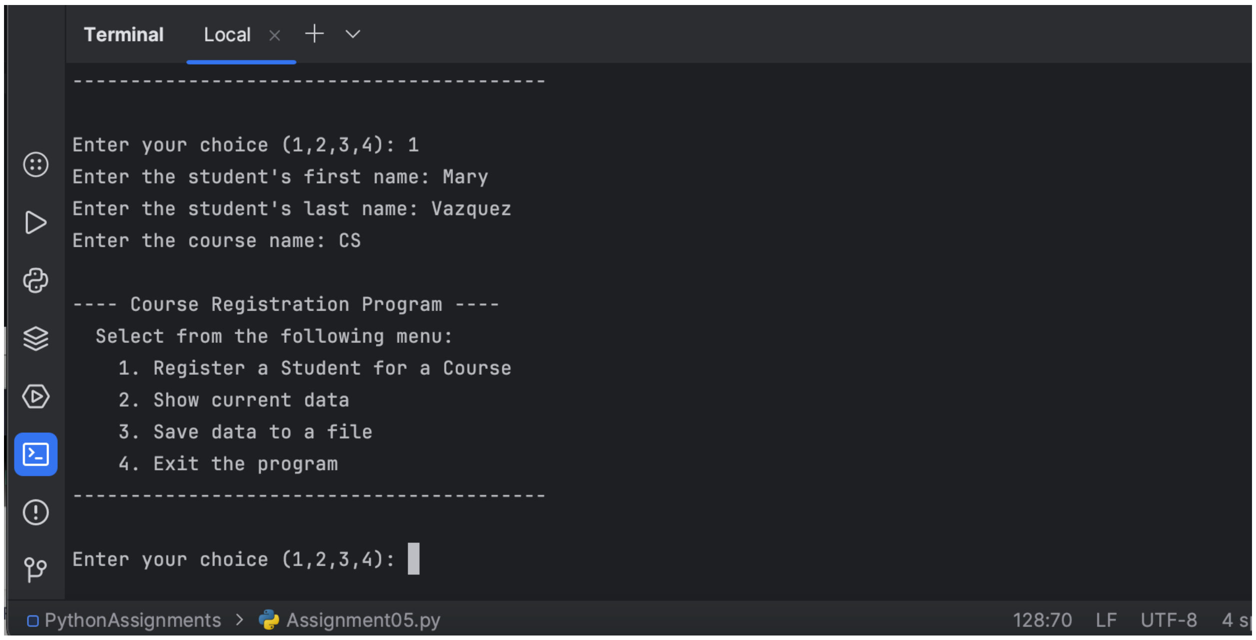
```
Traceback (most recent call last):
  File "/Users/nely/Desktop/PythonAssignments/Assignment05d.py",
line 80, in <module>
    if student_data['FirstName'] == "":
KeyError: 'FirstName'

Process finished with exit code 1
```

Figure 7. Traceback error

6. Running the Program from the Terminal.

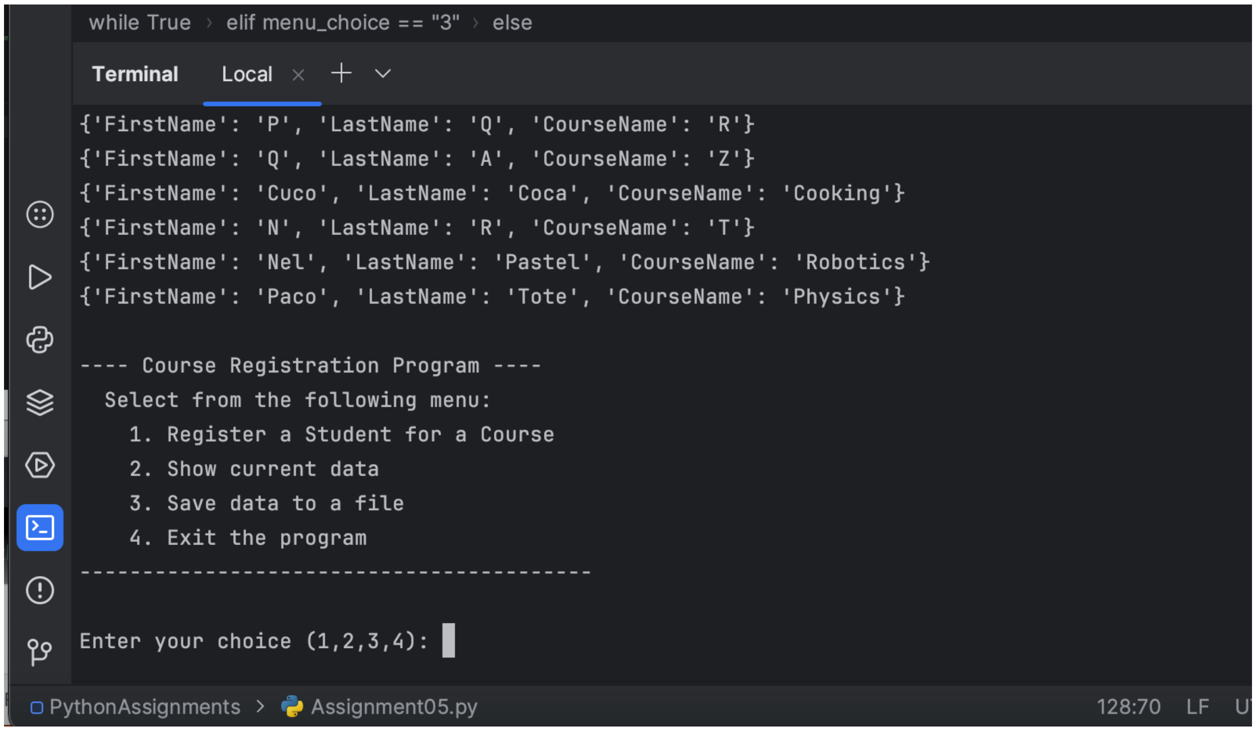
I did not have any challenge while executing my program in the terminal once everything was working well in the PyCharm console.

In **Figures 8**, **9**, **10** and **11** you can see screenshots for each option on the Menu.



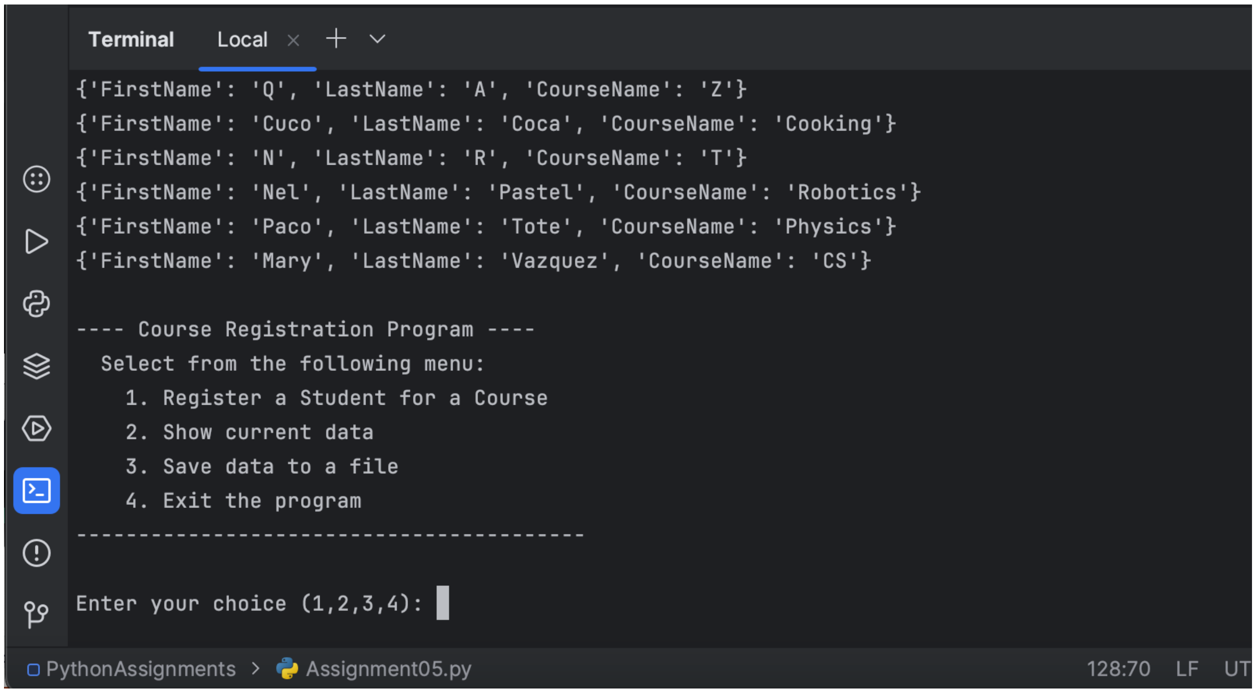
```
-----------------------------------------

Enter your choice (1,2,3,4): 1
Enter the student's first name: Mary
Enter the student's last name: Vazquez
Enter the course name: CS

---- 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
-----------------------------------------

Enter your choice (1,2,3,4):
```

Figure 8. Executing code from Terminal - Menu Option 1



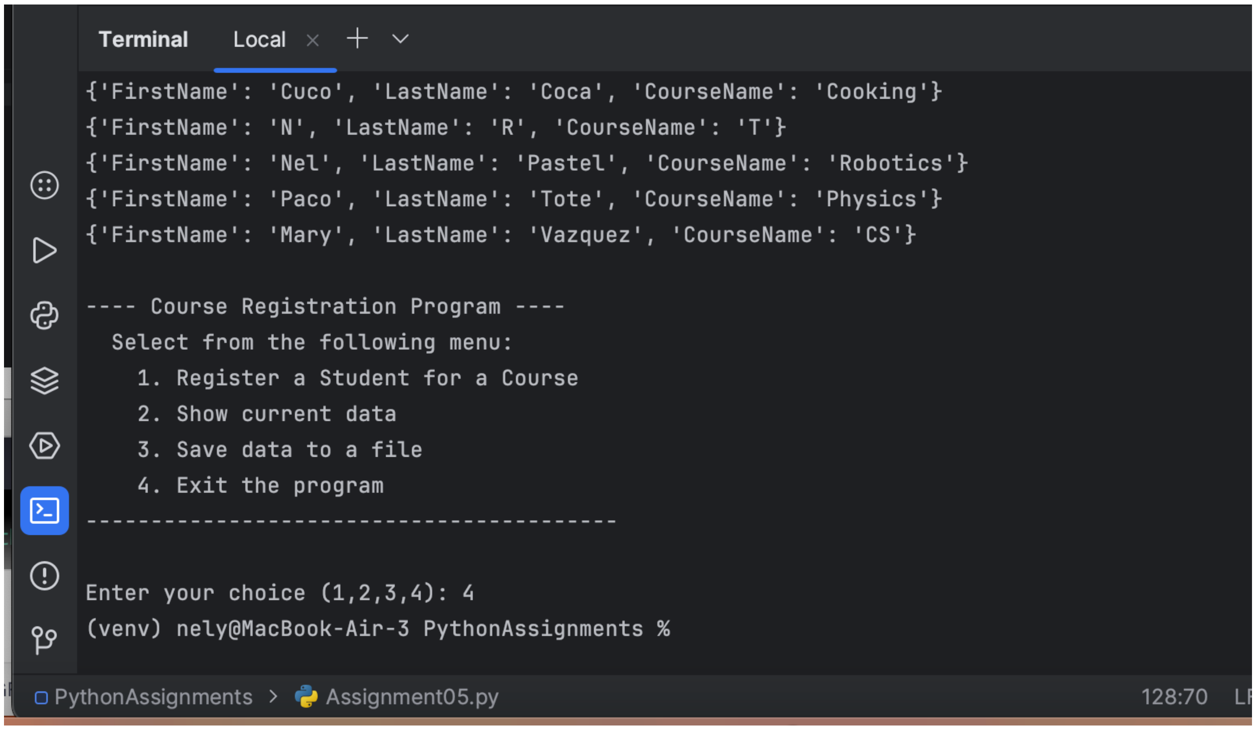
```
{'FirstName': 'P', 'LastName': 'Q', 'CourseName': 'R'}
{'FirstName': 'Q', 'LastName': 'A', 'CourseName': 'Z'}
{'FirstName': 'Cuco', 'LastName': 'Coca', 'CourseName': 'Cooking'}
{'FirstName': 'N', 'LastName': 'R', 'CourseName': 'T'}
{'FirstName': 'Nel', 'LastName': 'Pastel', 'CourseName': 'Robotics'}
{'FirstName': 'Paco', 'LastName': 'Tote', 'CourseName': 'Physics'}

---- 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
-----------------------------------------

Enter your choice (1,2,3,4):
```

Figure 9. Executing code from Terminal - Menu Option 2



```
{'FirstName': 'Q', 'LastName': 'A', 'CourseName': 'Z'}
{'FirstName': 'Cuco', 'LastName': 'Coca', 'CourseName': 'Cooking'}
{'FirstName': 'N', 'LastName': 'R', 'CourseName': 'T'}
{'FirstName': 'Nel', 'LastName': 'Pastel', 'CourseName': 'Robotics'}
{'FirstName': 'Paco', 'LastName': 'Tote', 'CourseName': 'Physics'}
{'FirstName': 'Mary', 'LastName': 'Vazquez', 'CourseName': 'CS'}

---- 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
-----------------------------------------

Enter your choice (1,2,3,4):
```

Figure 10. Executing code from Terminal - Menu Option 3



```
{'FirstName': 'Cuco', 'LastName': 'Coca', 'CourseName': 'Cooking'}
{'FirstName': 'N', 'LastName': 'R', 'CourseName': 'T'}
{'FirstName': 'Nel', 'LastName': 'Pastel', 'CourseName': 'Robotics'}
{'FirstName': 'Paco', 'LastName': 'Tote', 'CourseName': 'Physics'}
{'FirstName': 'Mary', 'LastName': 'Vazquez', 'CourseName': 'CS'}

---- 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
-----------------------------------------

Enter your choice (1,2,3,4): 4
(venv) nely@MacBook-Air-3 PythonAssignments %
```

Figure 11. Executing code from Terminal - Menu Option 4

Summary

The objective of this assignment was as in the previous assignments, to create a Python program that demonstrates using constants, variables, and print statements to display a message about a student's registration for a Python course.

This program was very similar to Assignment04, but it also included: **the use of data processing using dictionaries and error handling.**

This document included a description of the challenges that I faced while performing this assignment.

It is worth to mention that this assignment was very relevant in my learning about data processing of lists and dictionaries and files as well. Also, about how to manipulate the data in JSON files to be read and save data into this file.

This assignment was especially challenging for me. I required triple the time that I needed for the previous assignments. However, I enjoyed greatly while doing this Assignment.

The program also allows to register multiple students and display them and save them from the dictionary to a JSON file and read from the JSON file to construct the dictionary again.

The program also provides structured error handling when the file is read into the list of dictionary rows and when the dictionary rows are written to the file.

Citations

1. Writing professional papers: https://www.youtube.com/watch?v=9ojhSW9ljjo&feature=youtu.be

2. Open AI ChatGPT, Oct. 2023, chat.openai.com/chat: A few aspects of this assignment were informed by queries submitted to the ChatGPT.

3. Slides and videos from class, laboratories and Demo/Videos of the course in this Module 5.