Yunjia He
5/28/25
IT FND 110 A
Assignment06
https://github.com/Yunjia13/IntroToProg-Python-Mod06.git

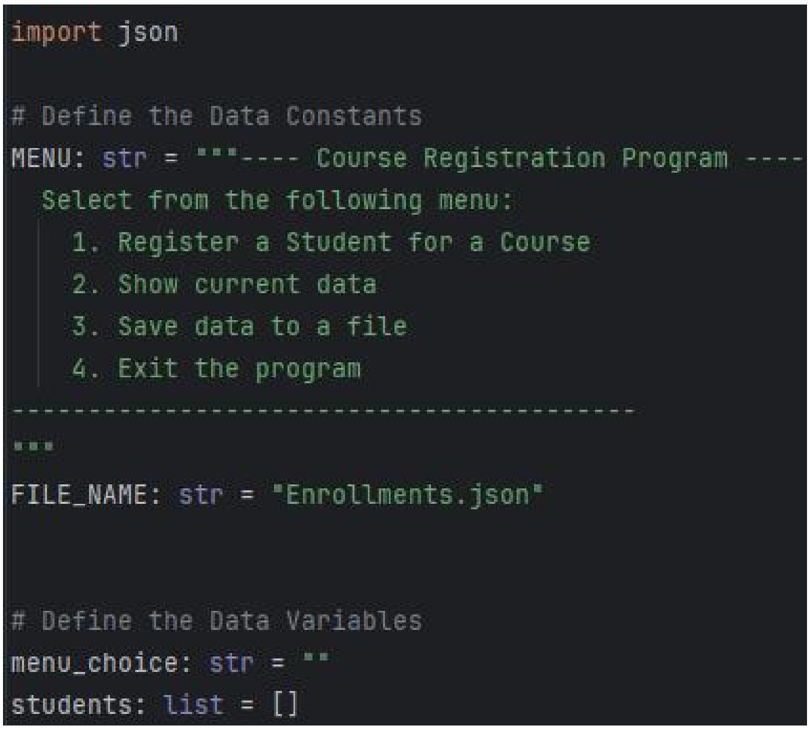
Functions

Introduction

This week, I learned the using functions. I created a script with menu, and user can make selection. User can register students for course or save data to a file. This script can save data in dictionaries and json file. The following information is a breakdown of how I wrote this program.

Creating the Program

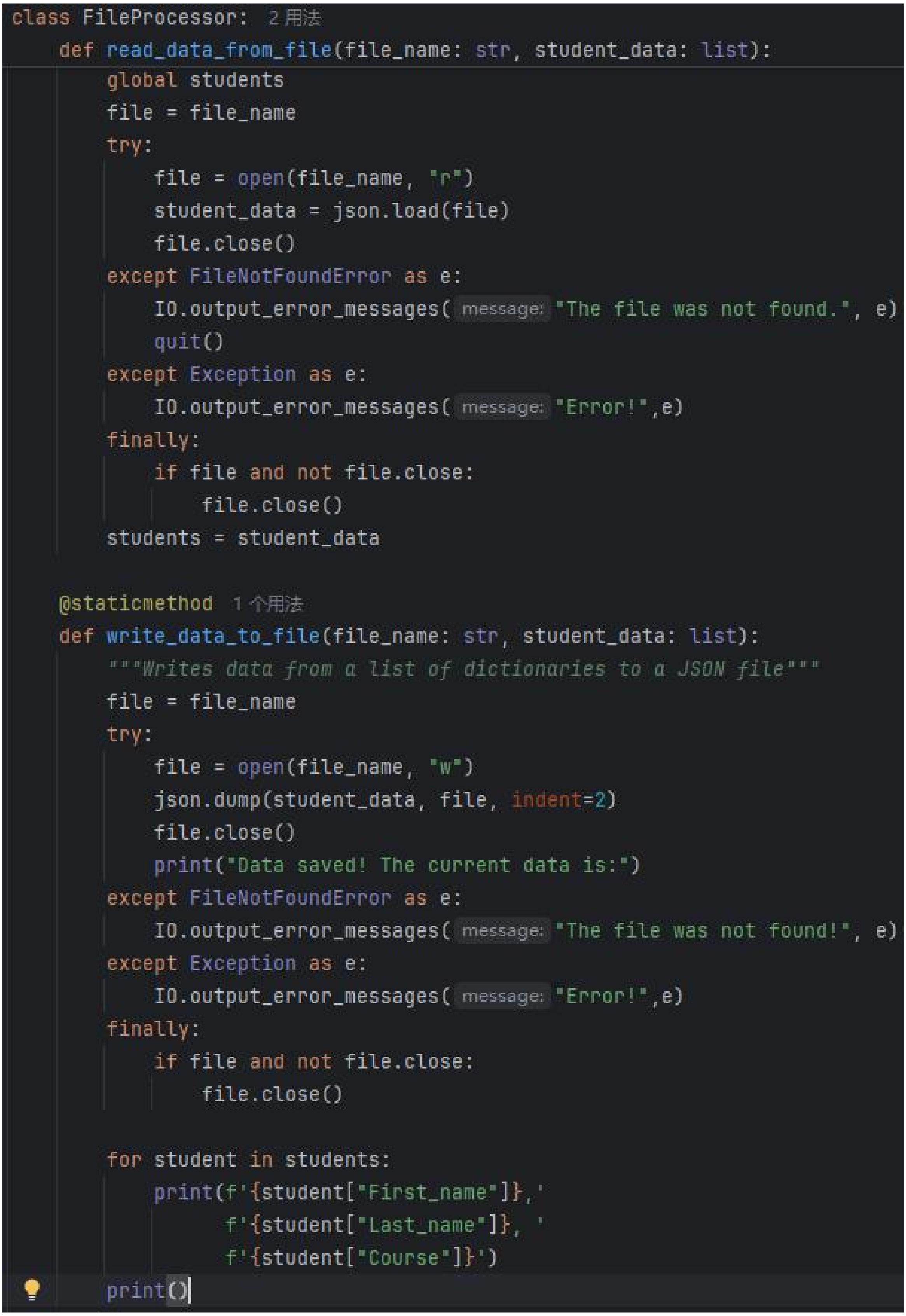
I start my program off by importing "json". Then, I define constants and variables with type hints. I set "MENU" and "FILE_NAME" as string type data constant. I set "menu_choice" and "students" data variables.(Figure 1.1)



```
import json

# 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
-----------------------------------------
"""
FILE_NAME: str = "Enrollments.json"


# Define the Data Variables
menu_choice: str = ""
students: list = []
```

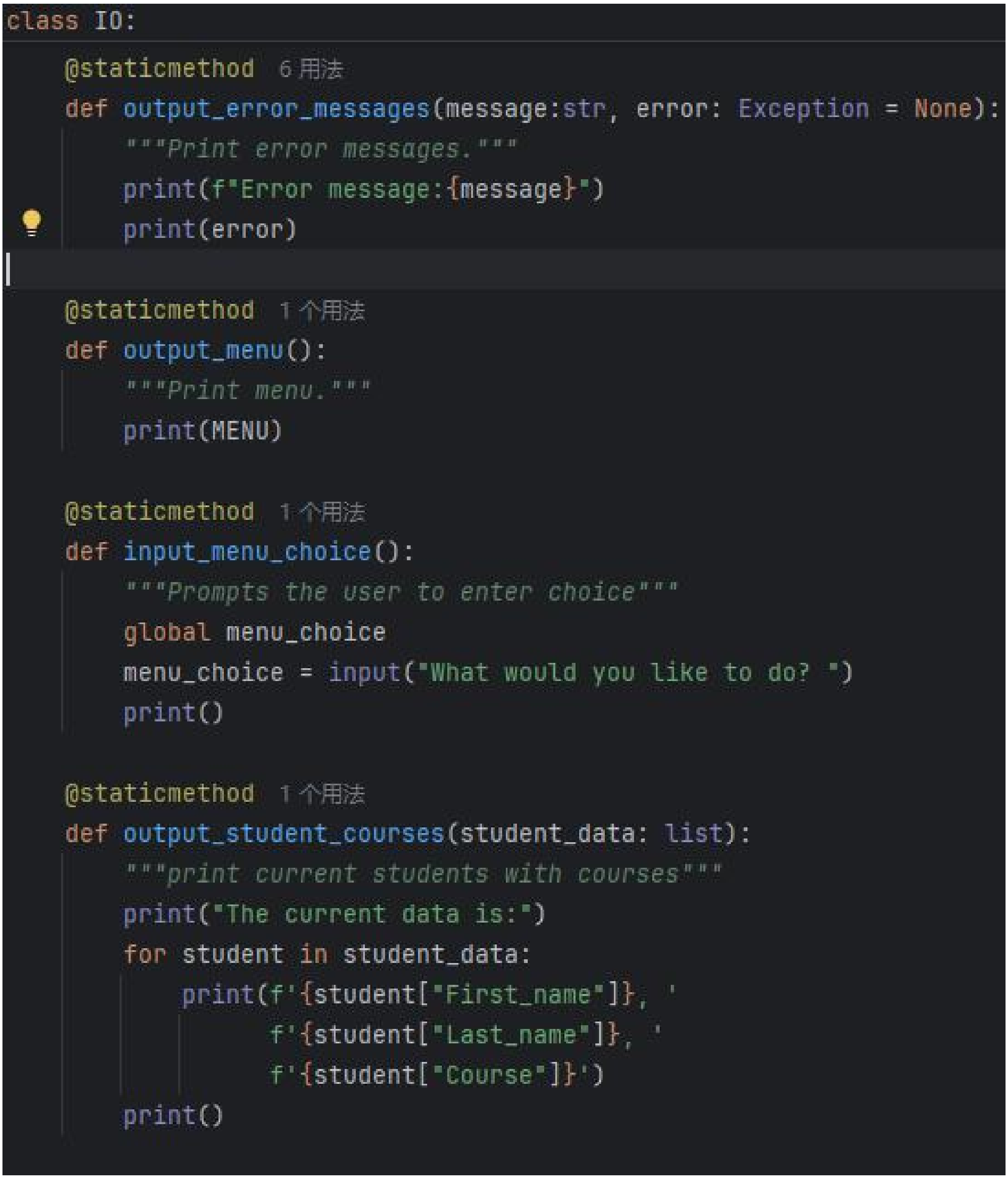
Figure 1.1: Define the data

Next step,I create two classes. First, I create first class named "FileProcessor" to manage functions that run with file. Second, I create second class named "IO" to manage functions of user input and output. After that, I create functions and put them in class.(Figure 1.2~1.3)

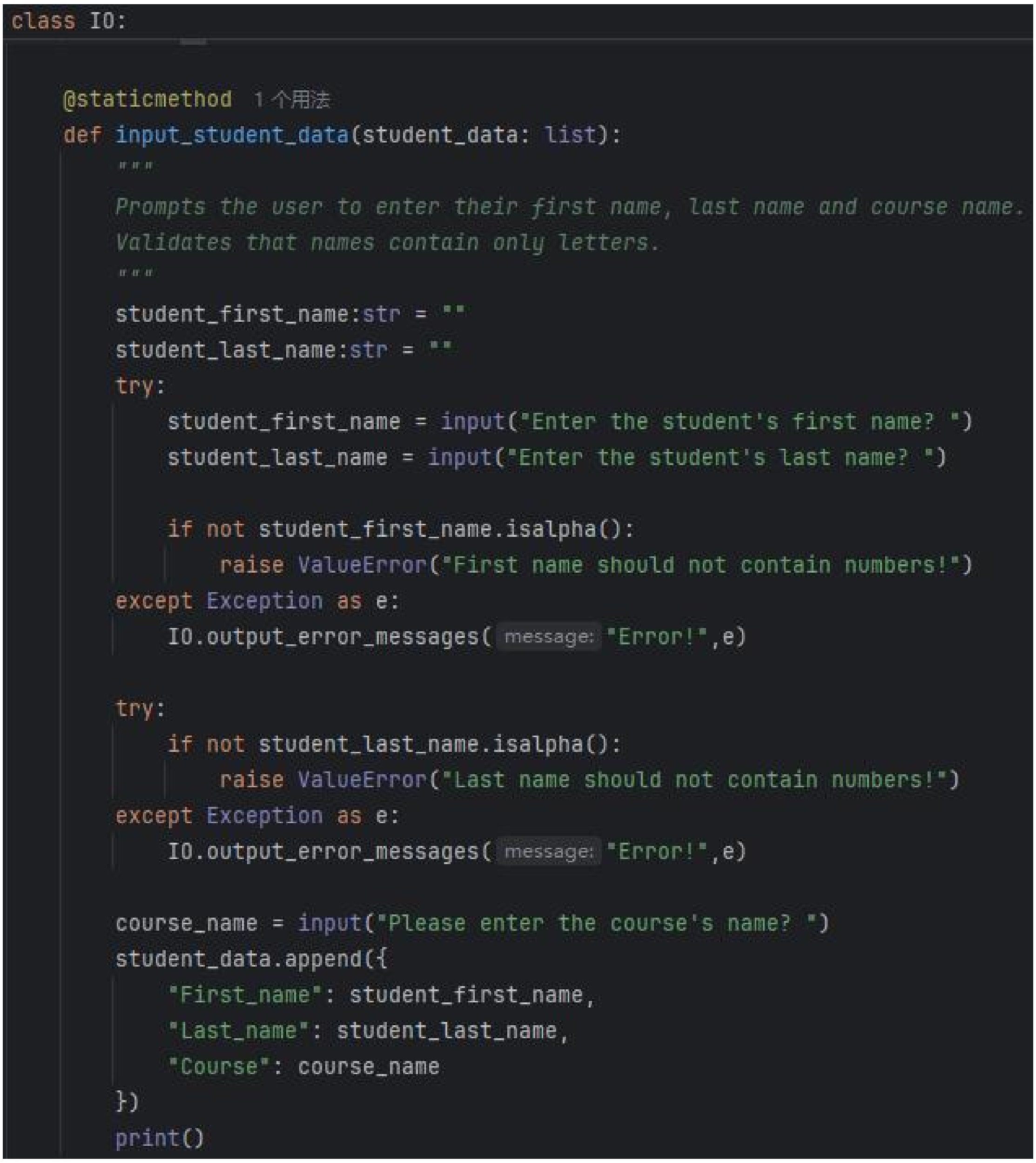


```
class FileProcessor:  2 用法
    def read_data_from_file(file_name: str, student_data: list):
        global students
        file = file_name
        try:
            file = open(file_name, "r")
            student_data = json.load(file)
            file.close()
        except FileNotFoundError as e:
            IO.output_error_messages( message: "The file was not found.", e)
            quit()
        except Exception as e:
            IO.output_error_messages( message: "Error!",e)
        finally:
            if file and not file.close:
                file.close()
        students = student_data

    @staticmethod  1 个用法
    def write_data_to_file(file_name: str, student_data: list):
        """Writes data from a list of dictionaries to a JSON file"""
        file = file_name
        try:
            file = open(file_name, "w")
            json.dump(student_data, file, indent=2)
            file.close()
            print("Data saved! The current data is:")
        except FileNotFoundError as e:
            IO.output_error_messages( message: "The file was not found!", e)
        except Exception as e:
            IO.output_error_messages( message: "Error!",e)
        finally:
            if file and not file.close:
                file.close()

        for student in students:
            print(f'{student["First_name"]},'
                  f'{student["Last_name"]}, '
                  f'{student["Course"]}')
        print()
```

Figure 1.2: class FileProcessor



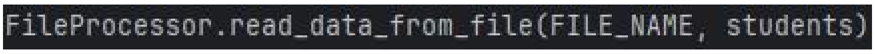
```python
class IO:
    @staticmethod
    def output_error_messages(message:str, error: Exception = None):
        """Print error messages."""
        print(f"Error message:{message}")
        print(error)

    @staticmethod
    def output_menu():
        """Print menu."""
        print(MENU)

    @staticmethod
    def input_menu_choice():
        """Prompts the user to enter choice"""
        global menu_choice
        menu_choice = input("What would you like to do? ")
        print()

    @staticmethod
    def output_student_courses(student_data: list):
        """print current students with courses"""
        print("The current data is:")
        for student in student_data:
            print(f'{student["First_name"]}, '
                  f'{student["Last_name"]}, '
                  f'{student["Course"]}')
        print()
```



```
class IO:

    @staticmethod  1 个用法
    def input_student_data(student_data: list):
        """
        Prompts the user to enter their first name, last name and course name.
        Validates that names contain only letters.
        """
        student_first_name:str = ""
        student_last_name:str = ""
        try:
            student_first_name = input("Enter the student's first name? ")
            student_last_name = input("Enter the student's last name? ")

            if not student_first_name.isalpha():
                raise ValueError("First name should not contain numbers!")
        except Exception as e:
            IO.output_error_messages( message: "Error!",e)

        try:
            if not student_last_name.isalpha():
                raise ValueError("Last name should not contain numbers!")
        except Exception as e:
            IO.output_error_messages( message: "Error!",e)

        course_name = input("Please enter the course's name? ")
        student_data.append({
            "First_name": student_first_name,
            "Last_name": student_last_name,
            "Course": course_name
        })
        print()
```

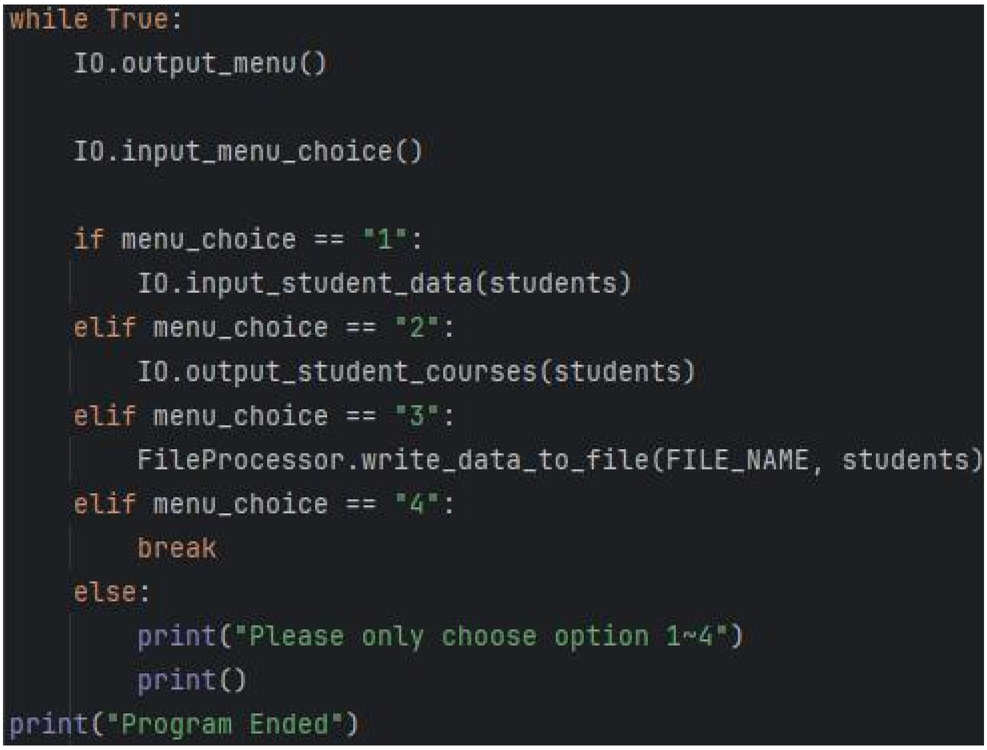
Figure 1.3: class IO

Now the functions are all ready. I start the main by using “FileProcessor.read_ data_from _file.” Next, I use “while” loop to keep user on menu and use “if” to direct to different functions. When “menu_choice == 1,” the program call “IO.input_student_data.” When “menu_choice == 2,” the program call “IO.output_student_courses” to display current data. When “menu_choice == 3,” the program call “FileProcessor.write_data_to_file” to save data. When “menu_choice == 4,” the program will end. (Figure 1.4~1.5)



```
FileProcessor.read_data_from_file(FILE_NAME, students)
```

Figure 1.4: Read data from file

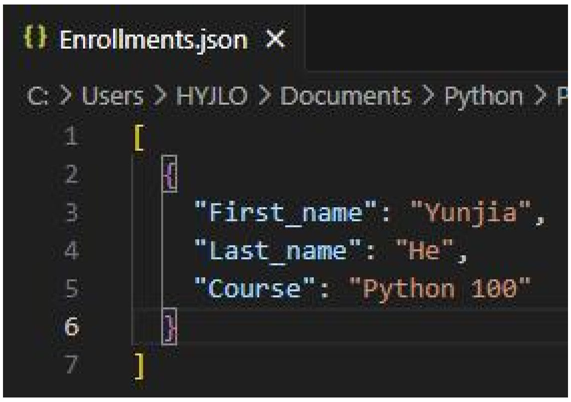


```
while True:
    IO.output_menu()

    IO.input_menu_choice()

    if menu_choice == "1":
        IO.input_student_data(students)
    elif menu_choice == "2":
        IO.output_student_courses(students)
    elif menu_choice == "3":
        FileProcessor.write_data_to_file(FILE_NAME, students)
    elif menu_choice == "4":
        break
    else:
        print("Please only choose option 1~4")
        print()
print("Program Ended")
```

Figure 1.5: while loop

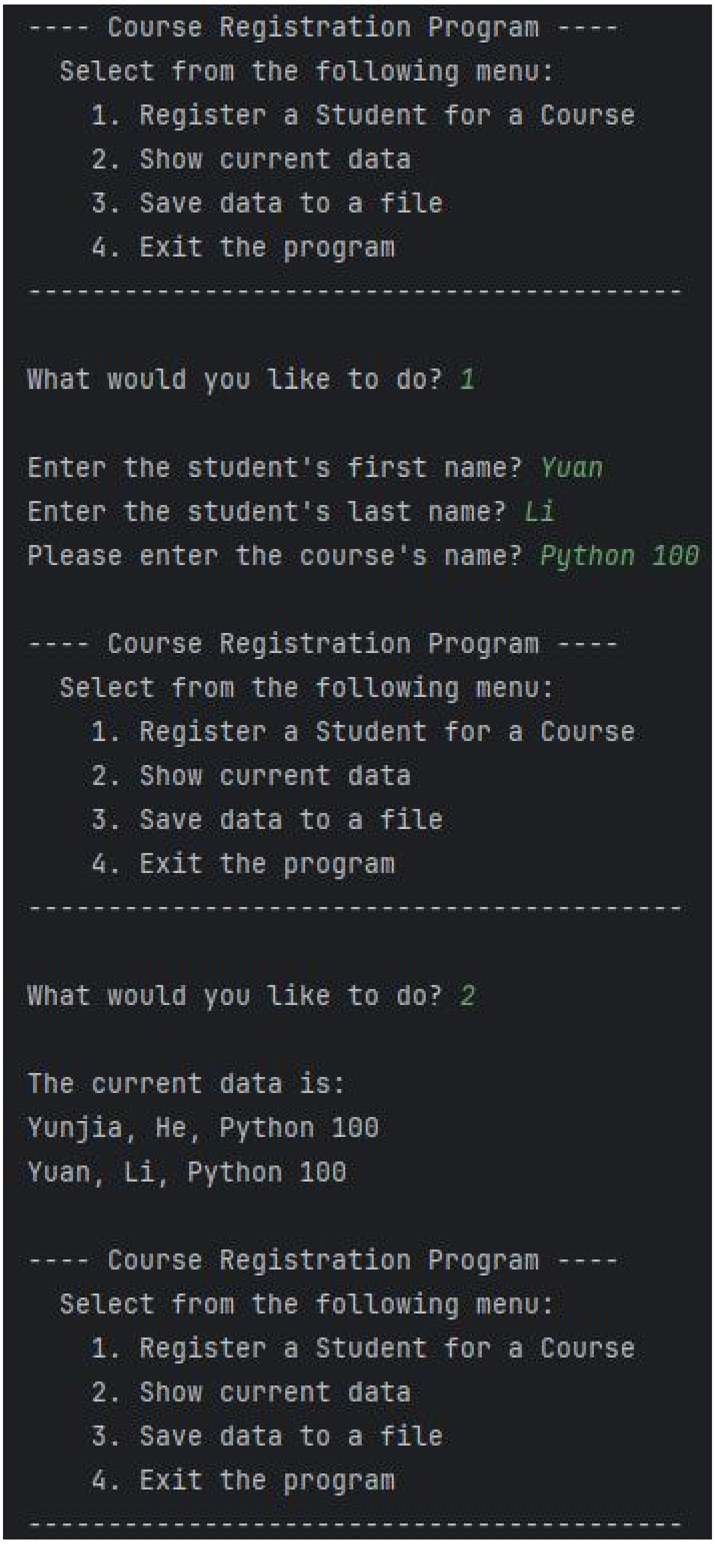
Testing the Program

Now that the code is complete. It's time to run and test it. First, I save the script and run it in Pycharm and CMD. The following information is test result.(Figure 2.1~2.5)

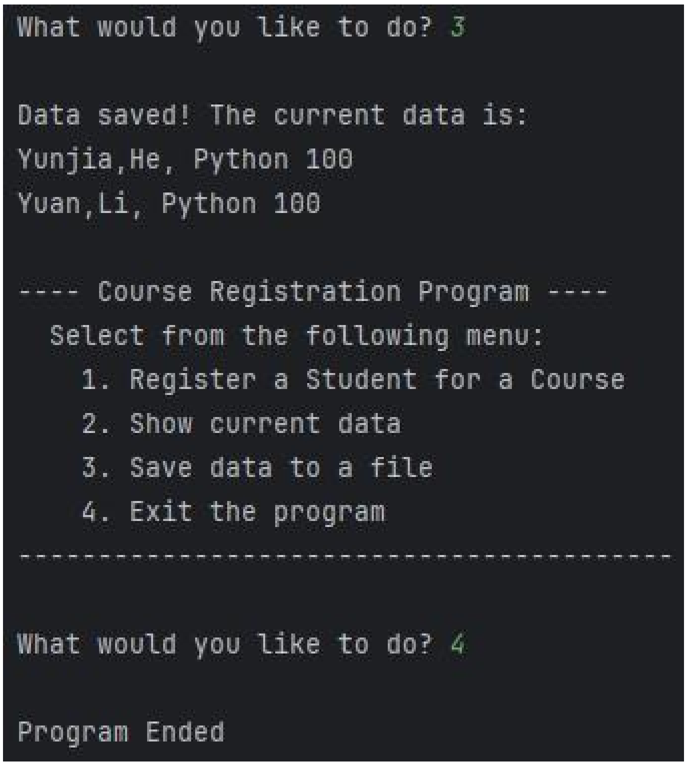


```
[
  {
    "First_name": "Yunjia",
    "Last_name": "He",
    "Course": "Python 100"
  }
]
```

Figure 2.1: Enrollments.json

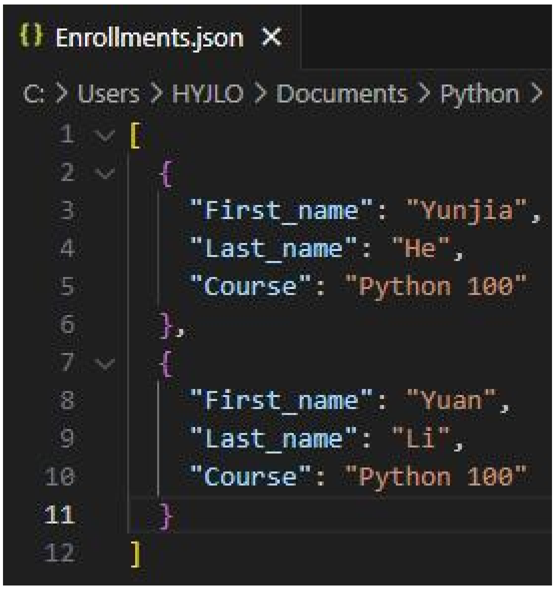


```
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course
    2. Show current data
    3. Save data to a file
    4. Exit the program
-----------------------------------------

What would you like to do? 1

Enter the student's first name? Yuan
Enter the student's last name? Li
Please enter the course's name? Python 100

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

What would you like to do? 2

The current data is:
Yunjia, He, Python 100
Yuan, Li, Python 100

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



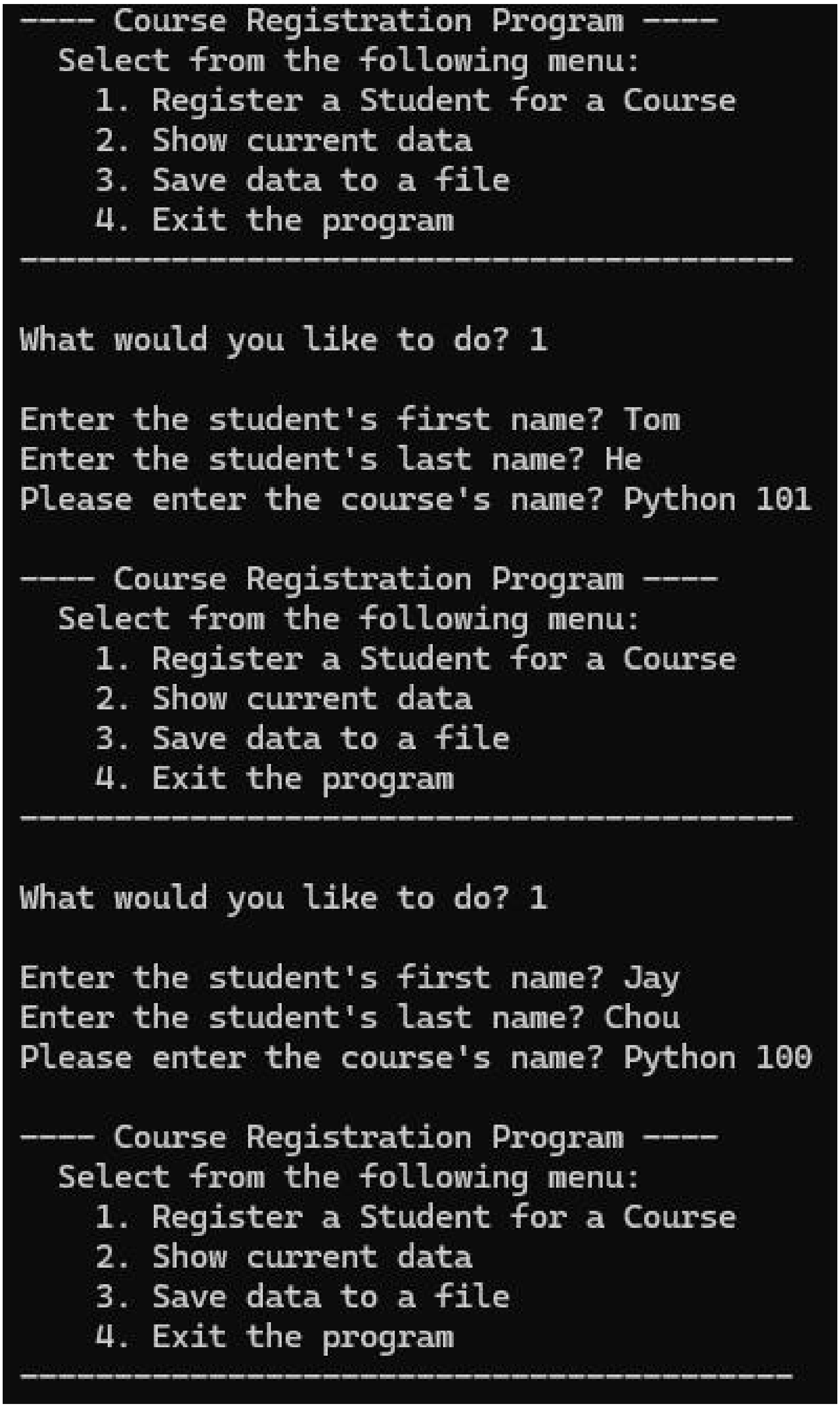
```
What would you like to do? 3

Data saved! The current data is:
Yunjia,He, Python 100
Yuan,Li, Python 100

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


What would you like to do? 4

Program Ended
```

Figure 2.2: Pycharm test result

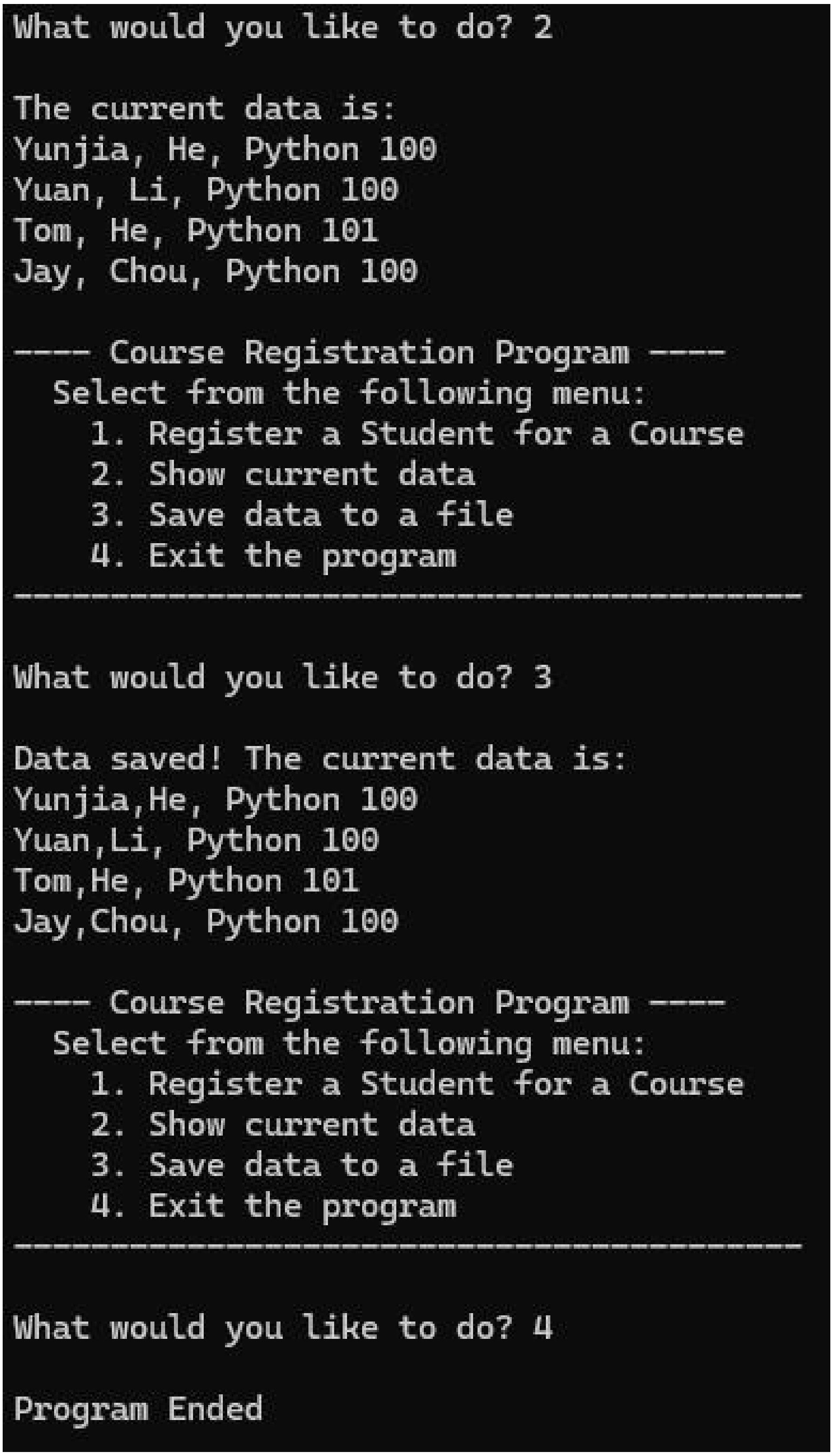


```
[
  {
    "First_name": "Yunjia",
    "Last_name": "He",
    "Course": "Python 100"
  },
  {
    "First_name": "Yuan",
    "Last_name": "Li",
    "Course": "Python 100"
  }
]
```

Figure 2.3: Enrollments.json(after Pycharm test)

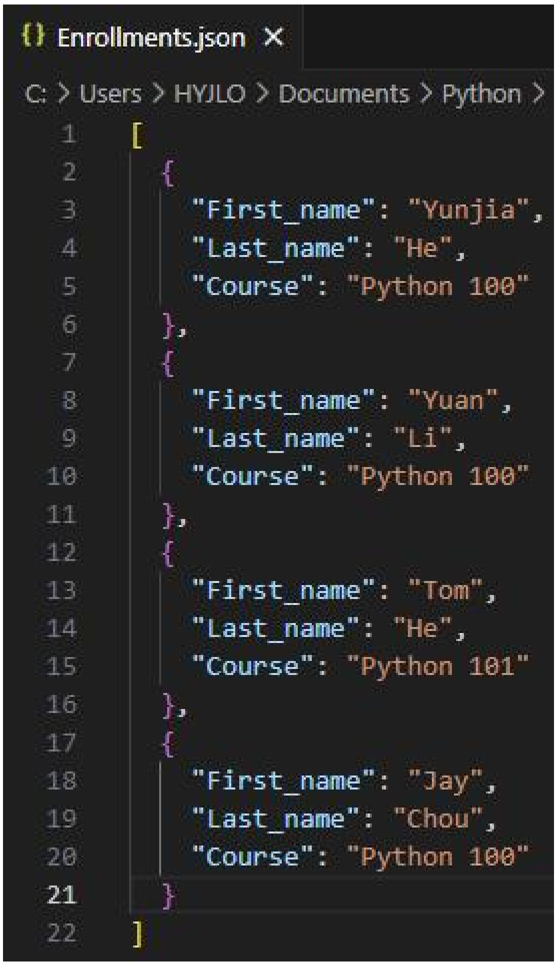


```
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course
    2. Show current data
    3. Save data to a file
    4. Exit the program
-----------------------------------------

What would you like to do? 1

Enter the student's first name? Tom
Enter the student's last name? He
Please enter the course's name? Python 101

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

What would you like to do? 1

Enter the student's first name? Jay
Enter the student's last name? Chou
Please enter the course's name? Python 100

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



```
What would you like to do? 2

The current data is:
Yunjia, He, Python 100
Yuan, Li, Python 100
Tom, He, Python 101
Jay, Chou, Python 100

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


What would you like to do? 3

Data saved! The current data is:
Yunjia,He, Python 100
Yuan,Li, Python 100
Tom,He, Python 101
Jay,Chou, Python 100

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


What would you like to do? 4

Program Ended
```

Figure 2.4: CMD test result



```
[
  {
    "First_name": "Yunjia",
    "Last_name": "He",
    "Course": "Python 100"
  },
  {
    "First_name": "Yuan",
    "Last_name": "Li",
    "Course": "Python 100"
  },
  {
    "First_name": "Tom",
    "Last_name": "He",
    "Course": "Python 101"
  },
  {
    "First_name": "Jay",
    "Last_name": "Chou",
    "Course": "Python 100"
  }
]
```

Figure 2.5: Enrollments.json(after CMD test)

Summary

I review using dictionaries, json files, and exception handling by writing this program. I learn that functions can help program management. It is convenient to organize different program with functions. Using the descriptive document strings in programming is very important. It can help others read program more directly. I also review lists using, file handling, and looping in Python. Lastly, this practice helped me start programming with Python and review knowledge.