Bogdan Kondratenko
13/11/2025
Foundations Of Python Programming
Assignment 05
Github link: https://github.com/UWStudentAcc/IntroToProg-Python-Mod05

The creation of Assignment05

Introduction

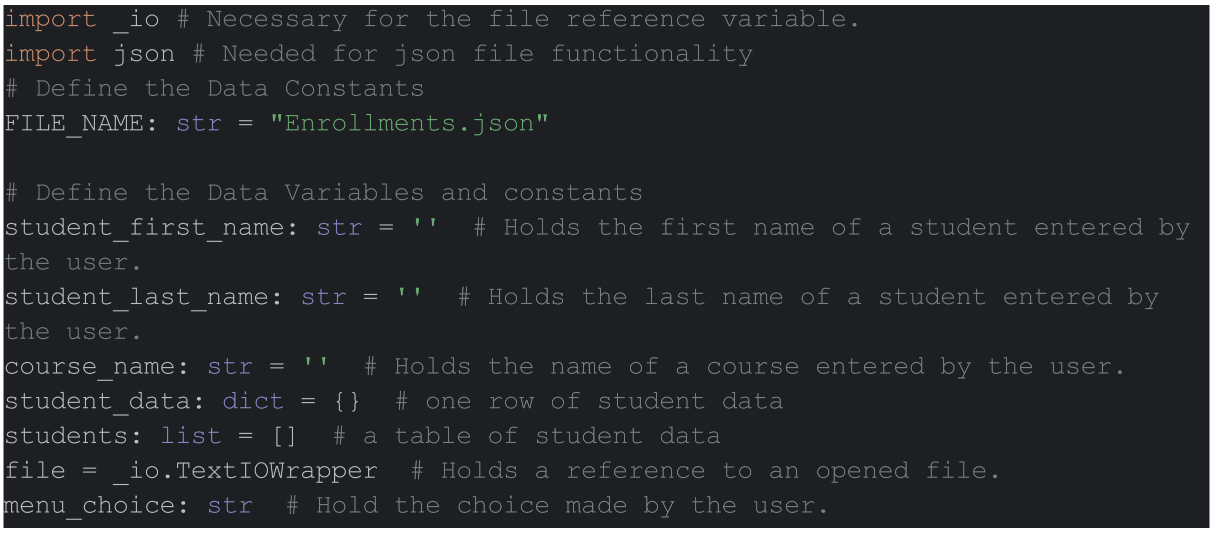
In this document I will do a step-by-step overview of the Assignment05.py file and its creation.

Creating the file

I began the process by opening the Assignment05-Starter.py file, then renaming it to Assignment05.py. This simplifies the process of working on the assignment, as it already has a prewritten header and starting code.

Setting up the script

In order to do what the assignment asks of, several things must be prepared beforehand. (Figure 1)
Of the 2 imports, _io allows the "file" variable to have a new value that allows it to perform better than when it was set to "None". Then, json is imported to allow the script to interact with json files which is a requirement for the assignment.

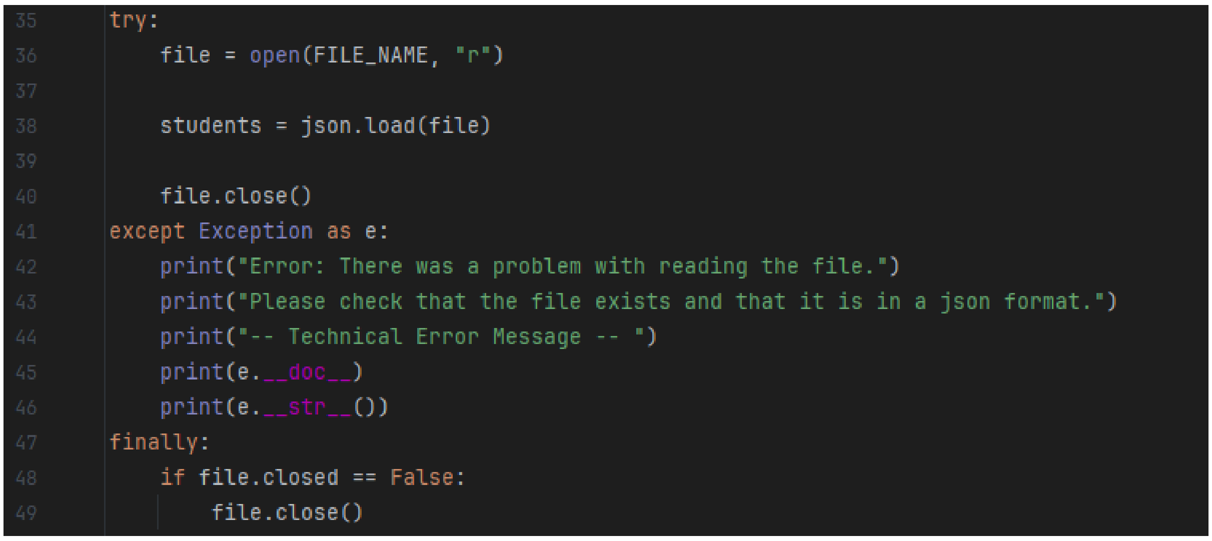


```python
import _io # Necessary for the file reference variable.
import json # Needed for json file functionality
# Define the Data Constants
FILE_NAME: str = "Enrollments.json"

# 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.
student_data: dict = {}  # one row of student data
students: list = []  # a table of student data
file = _io.TextIOWrapper  # Holds a reference to an opened file.
menu_choice: str  # Hold the choice made by the user.
```

(Figure 1) The script setup. The variables and constants don't require much explanation, as the setup is mostly to define the types of variables they are.

Main script body

First, I wrote a way for the script to read the contents of the .json file and input them into the “students” variable. (Figure 2)



```python
try:
    file = open(FILE_NAME, "r")

    students = json.load(file)

    file.close()
except Exception as e:
    print("Error: There was a problem with reading the file.")
    print("Please check that the file exists and that it is in a json format.")
    print("-- Technical Error Message -- ")
    print(e.__doc__)
    print(e.__str__())
finally:
    if file.closed == False:
        file.close()
```

(Figure 2) The try, except and finally are part of the error handling of this specific function.

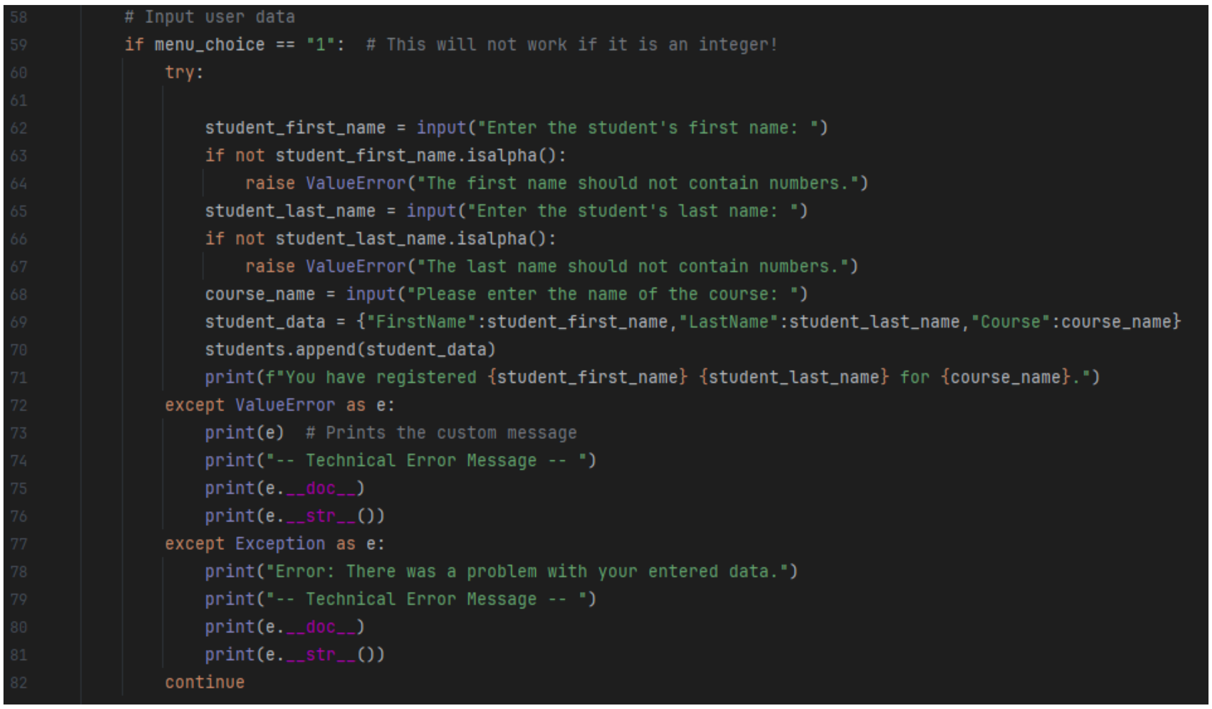
Subtopic A: Option 1, User data collection and processing

(Figure 3) Is the code for option 1 of the menu, it handles user input for new data, handles errors, and lastly appends the new data into the students table.

To break down this large block of code, let's start with the error handling elements:
The 2 “except” at the bottom of Figure 3 print out error messages if the error they’re looking for happens, with the first one informing the user that the first and last names should not contain numbers. The second one is for any unexpected exceptions.

The “if not” below the inputs for first and last name are what raises the ValueError error type if they detect that the first or last names aren’t just alphabetic letters.

With the error handling out of the way, there’s not much left to talk about: the input lines collect user input data for the first, last and course names. They are then formatted as a dictionary, appended into the students table and lastly the dictionary is printed out to the user



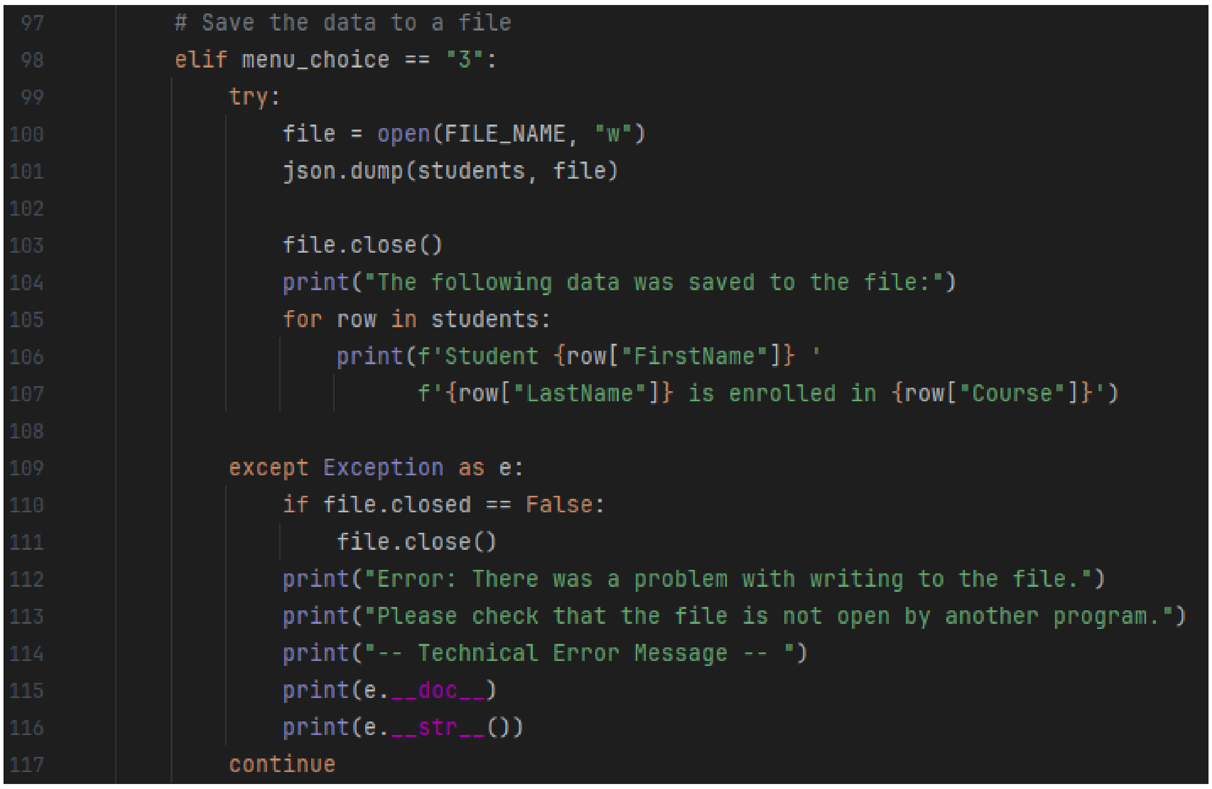
```python
    # Input user data
    if menu_choice == "1":  # This will not work if it is an integer!
        try:

            student_first_name = input("Enter the student's first name: ")
            if not student_first_name.isalpha():
                raise ValueError("The first name should not contain numbers.")
            student_last_name = input("Enter the student's last name: ")
            if not student_last_name.isalpha():
                raise ValueError("The last name should not contain numbers.")
            course_name = input("Please enter the name of the course: ")
            student_data = {"FirstName":student_first_name,"LastName":student_last_name,"Course":course_name}
            students.append(student_data)
            print(f"You have registered {student_first_name} {student_last_name} for {course_name}.")
        except ValueError as e:
            print(e)  # Prints the custom message
            print("-- Technical Error Message -- ")
            print(e.__doc__)
            print(e.__str__())
        except Exception as e:
            print("Error: There was a problem with your entered data.")
            print("-- Technical Error Message -- ")
            print(e.__doc__)
            print(e.__str__())
        continue
```

(Figure 3 above)

Subtopic B: Option 3, Saving the data to a .json file

(Figure 4) Is the code block responsible for saving the students table into the .json file. About half of it is the error handling, similar to option 1. In case an exception of some kind prevents option 3 from properly working, the “except” line will close the potentially opened file to prevent further issues and will then give the user error messages.

The rest of the code simply opens the .json file in write mode, dumps the contents of the students variable into it and then closes the file. Then it prints out the current data of the students table to the user in a presentable way.



```python
# Save the data to a file
elif menu_choice == "3":
    try:
        file = open(FILE_NAME, "w")
        json.dump(students, file)

        file.close()
        print("The following data was saved to the file:")
        for row in students:
            print(f'Student {row["FirstName"]} '
                  f'{row["LastName"]} is enrolled in {row["Course"]}')

    except Exception as e:
        if file.closed == False:
            file.close()
        print("Error: There was a problem with writing to the file.")
        print("Please check that the file is not open by another program.")
        print("-- Technical Error Message -- ")
        print(e.__doc__)
        print(e.__str__())
    continue
```

(Figure 4)

Subtopic C: Options 2 and 4, printing out the current data and closing the script

Option 2 prints out the data currently held in the variables that hold input gathered from option 1 as well as the contents of the students table presented in a similar manner to option 3. (Figure 5)

```python
elif menu_choice == "2":

   # Processes the data to create and display a custom message
   print("-" * 50)
   print(f"The most recent registered student is {student_first_name} "
         f"{student_last_name} going for {course_name}.")
   for row in students:
       print(f'Student {row["FirstName"]} '
             f'{row["LastName"]} is enrolled in {row["Course"]}')
   print("-" * 50)
   continue
```

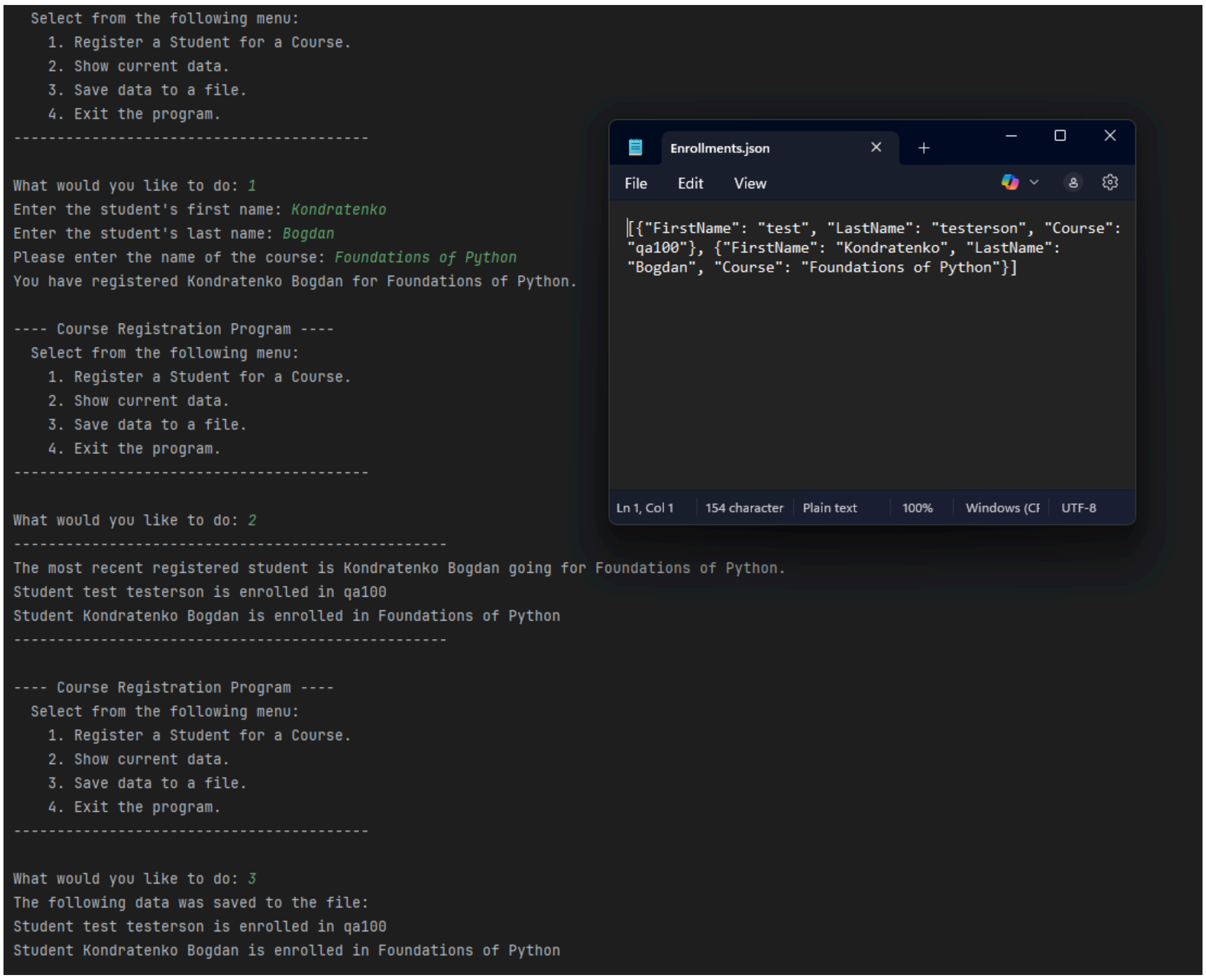
(Figure 5 above)
Option 4 simply ends the loop keeping the script going. (Figure 6)

```
# Stop the loop
elif menu_choice == "4":
    break  # out of the loop
```

(Figure 6)

Testing the script

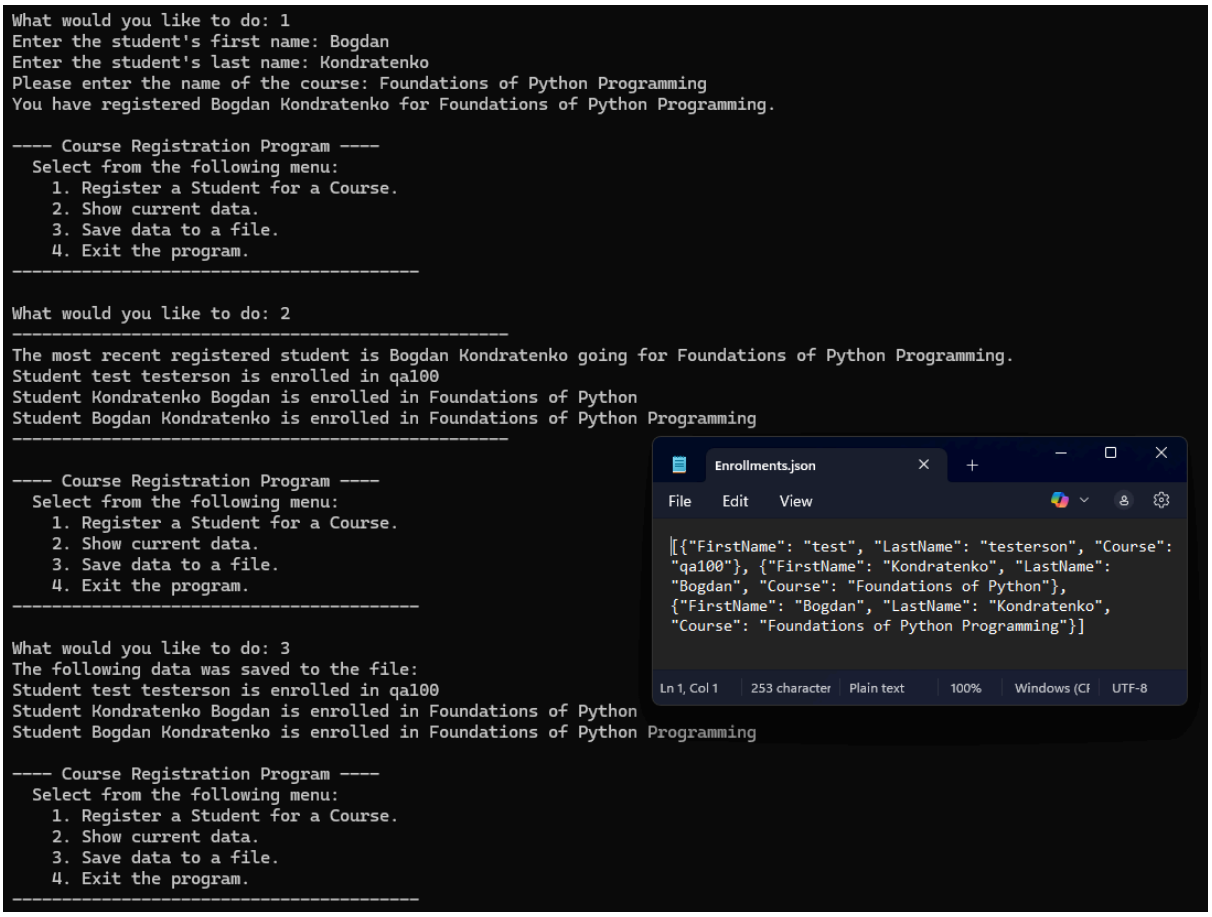
The script was then tested both in PyCharm and the cmd console. (Figure 7 and Figure 8)



```
  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: Kondratenko
Enter the student's last name: Bogdan
Please enter the name of the course: Foundations of Python
You have registered Kondratenko Bogdan for Foundations of Python.

---- 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 most recent registered student is Kondratenko Bogdan going for Foundations of Python.
Student test testerson is enrolled in qa100
Student Kondratenko Bogdan is enrolled in Foundations of Python
--------------------------------------------------

---- 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
The following data was saved to the file:
Student test testerson is enrolled in qa100
Student Kondratenko Bogdan is enrolled in Foundations of Python
```

```
[{"FirstName": "test", "LastName": "testerson", "Course": "qa100"}, {"FirstName": "Kondratenko", "LastName": "Bogdan", "Course": "Foundations of Python"}]
```

(Figure 7) PyCharm test.



```
What would you like to do: 1
Enter the student's first name: Bogdan
Enter the student's last name: Kondratenko
Please enter the name of the course: Foundations of Python Programming
You have registered Bogdan Kondratenko for Foundations of Python Programming.

---- 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 most recent registered student is Bogdan Kondratenko going for Foundations of Python Programming.
Student test testerson is enrolled in qa100
Student Kondratenko Bogdan is enrolled in Foundations of Python
Student Bogdan Kondratenko is enrolled in Foundations of Python Programming
--------------------------------------------------

---- 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
The following data was saved to the file:
Student test testerson is enrolled in qa100
Student Kondratenko Bogdan is enrolled in Foundations of Python
Student Bogdan Kondratenko is enrolled in Foundations of Python Programming

---- 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.
-----------------------------------------
```

```
[{"FirstName": "test", "LastName": "testerson", "Course":
"qa100"}, {"FirstName": "Kondratenko", "LastName":
"Bogdan", "Course": "Foundations of Python"},
{"FirstName": "Bogdan", "LastName": "Kondratenko",
"Course": "Foundations of Python Programming"}]
```

(Figure 8) Cmd console test.

Summary

In this document I did a walk-through of how I created Assignment05.py and elaborated on the error handling.