Evan Peper

November 20th, 2024

IT FDN 110 A

Assignment 06

Using Classes and Functions in Python

Intro

This week, we learned about classes, functions and how they better organize your code in Python, focusing on creating an interactive menu that will read/write to a JSON file in a more structured way than our previous assignments. Using this knowledge, I opened and read a JSON file, generated a script that would create an interactive menu by using a while loop that would allow a user to generate data that would be written to the JSON file for the recording of student enrollment data, and display said data in a much cleaner manner than before.

Creating the Program and Importing JSON

To start, I created the header for my script on lines 1-7. This is important information for both my future self and other potential programmers who are reviewing my script to know who created the script and what it's used for. I then imported json on line 8 so pycharm could interpret and write to a JSON file.

```
1  # ------------------------------------------------------------------------------------------ #
2  # Title: Assignment06
3  # Desc: This assignment demonstrates using functions
4  # with structured error handling
5  # Change Log: (Who, When, What)
6  #   EPeper,11/20/2024,Created Script
7  # ------------------------------------------------------------------------------------------ #
8  import json
```

Figure 1.1 – Header

After that, I went on to the next portion of the assignment, which was to write the main body of my script. The first step of that was to define my data constants. On lines 10-22, I wrote out the constants "MENU" and "FILE_NAME" in all caps to indicate that they are constants, while also putting type hints to indicate what the data types are. I then set the values to those described in Assignment 06.

```
# 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.
----------------------------------------
'''
# Define the Data Constants
# FILE_NAME: str = "Enrollments.csv"
FILE_NAME: str = "Enrollments.json"
```

Figure 1.2 – The Constants

I then went on to create my variables on lines 24-26. I set up my list “student_data”, and variable “menu_choice” that will hold the string data generated by the user. These will be my global variables for my classes and functions below.

```
# Define the Data Variables and constants
students: list = []  # a table of student data
menu_choice: str  # Hold the choice made by the user.
```

Figure 1.3 – The Variables

Now that my variables and constants have been created, I set up my first class, FileProcessor, on lines 29-71. Within file processer I wrote two static functions. The first, “read_data_from_file” is to open and read the file, “enrollments.json”. The second, write_data_to_file” is to write new student information to “enrollments.json”. This is my processing concern.

```
# Processing ---------------------------------- #
2 usages
class FileProcessor:
    """
    A collection of processing layer functions that work with Json files

    ChangeLog: (Who, When, What)
    EPeper,11.20.2024,Created Class
    """
    global students

    1 usage
    @staticmethod
    def read_data_from_file(file_name: str, student_data: list):

        try:
            file = open(FILE_NAME, "r")
            student_data = json.load(file)
        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()
        return student_data

    1 usage
    @staticmethod
    def write_data_to_file(file_name:str, student_data: list):
        try:
            file = open(FILE_NAME, "w")
            json.dump(students, file)
            file.close()
            print("The following data was saved to file!")
            for student in students:
                print(f'Student {student["FirstName"]} {student["LastName"]} is enrolled in {student["CourseName"]}')
        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__())
```

Figure 1.4 – FileProcessor Class

I then set up my next class, IO, on lines 75-180. This class holds my presentation concern and is a collection of functions that handles input/output. The function “output_error_messages” displays custum error messages to the user when called. Function “output_menu” displays the menu of choices to the user. Function “input_menu_choice” receives a menu choice from the user. Function “input_student_data” prompts the user for information. Function “output_student_courses” displays student information.

```
# Presentation ------------------------------------- #
5 usages
class IO:
    """
    A collection of presentation layer functions that manage user input and output

    ChangeLog: (Who, When, What)
    EPeper,11.20.2024,Created Class
    """
    global students

    1 usage
    @staticmethod
    def output_error_messages(message: str, error: Exception = None):
        """
        This function displays a custom error messages to the user

        ChangeLog: (Who, When, What)
        EPeper,11.20.2024,Created function

        :return: None
        """
        print(message, end="\n\n")
        if error is not None:
            print("-- Technical Error Message -- ")
```

```
            print(error, error.__doc__, type(error), sep='\n')

    1 usage
    @staticmethod
    def output_menu(menu: str):
        """
        This function displays the menu of choices to the user

        ChangeLog: (Who, When, What)
        EPeper,11.20.2024,Created function

        :return: None
        """
        print()
        print(menu)
        print()

    1 usage
    @staticmethod
    def input_menu_choice():
        """
        This function gets a menu choice from the user

        ChangeLog: (Who, When, What)
        EPeper,11.20.2024,Created function

        :return: string with the users choice
        """
        choice = "0"
        try:
            choice = input("Enter your menu choice number: ")
            if choice not in ("1", "2", "3", "4"):
                raise Exception("Please, choose only 1, 2, 3, 4")
        except Exception as e:
            IO.output_error_messages(e.__str__())

        return choice

    1 usage
    @staticmethod
    def input_student_data(student_data: list):
        """
        This function displays prompts the user for student information to be stored

        ChangeLog: (Who, When, What)
        EPeper,11.20.2024,Created function

        :return: None
        """
        try:
            student_first_name = input("Enter the student's first name: ")
            if not student_first_name.isalpha():
```

```
146                 raise ValueError("The last name should not contain numbers.")
147             student_last_name = input("Enter the student's last name: ")
148             if not student_last_name.isalpha():
149                 raise ValueError("The last name should not contain numbers.")
150             course_name = input("Please enter the name of the course: ")
151             student_data = {"FirstName": student_first_name,
152                     "LastName": student_last_name,
153                     "CourseName": course_name}
154             students.append(student_data)
155             print(f"You have registered {student_first_name} {student_last_name} for {course_name}.")
156         except ValueError as e:
157             print(e)  # Prints the custom message
158             print("-- Technical Error Message -- ")
159             print(e.__doc__)
160             print(e.__str__())
161         except Exception as e:
162             print("Error: There was a problem with your entered data.")
163             print("-- Technical Error Message -- ")
164             print(e.__doc__)
165             print(e.__str__())
166
        1 usage
167     @staticmethod
168     def output_student_courses(student_data: list):
169         """
170          This function displays student information
```

```
171
172          ChangeLog: (Who, When, What)
173          EPeper,11.20.2024,Created function
174
175          :return: None
176          """
177         print("-" * 50)
178         for student in students:
179             print(f'Student {student["FirstName"]} {student["LastName"]} is enrolled in {student["CourseName"]}')
180         print("-" * 50)
```

Figure 1.5 – IO Class

I then set up my while statement on lines 190-197, which I set to True. I then present the menu of choices by calling the IO.output_menu function, and set my parameter as the variable menu= the constant MENU. I then set my variable menu_choice equal to the function IO.input_menu_choice() so the user can select a choice.

```
190 # Present and Process the data
191 while True:
192
193     # Present the menu of choices
194
195     IO.output_menu(menu=MENU)
196
197     menu_choice = IO.input_menu_choice()
```

Figure 1.6 – Setting up the menu

I then set up option 1 for my menu on lines198-202. I set up an if statement and have menu choice equal to the conditional statement of one. I them call my function “IO.input_student_data” to bring up the prompts the user needs to answer and set my parameter as student_data=students.

```
198     # Input user data
199
200     if menu_choice == "1":
201         IO.input_student_data(student_data=students)
202         continue
```

Figure 1.7 – Prompting User Information

For option 2, I created an elif statement and I set the variable menu_choice to a conditional statement of 2. This option displays data to the user. To do this, I call the function "IO.output_student_courses" to display the data and set the parameter as student_data=students. This was done on lines 204-207.

```
204     # Present the current data
205     elif menu_choice == "2":
206         IO.output_student_courses(student_data=students)
207         continue
```

Figure 1.8 – Presenting User Data

For option 3, I created an elif statement and I set the variable menu_choice to a conditional statement of 3. This option saves the current data to the file. To do this, I call the function "FileProcesor.write_data_to_file" save the data and set the parameters as file_name=FILE_NAME and student_data=students. This was done on lines 209-212.

```
209     # Save the data to a file
210     elif menu_choice == "3":
211         FileProcessor.write_data_to_file(file_name=FILE_NAME, student_data=students)
212         continue
```

Figure 1.9 – Presenting User Data

Lastly, we need to stop the loop on lines 214-220. I create an elif statement and I set the variable "menu_choice" to a conditional statement of "4". I then set my break here to close the loop. I also put an else statement, where if the user inputs any other number at the menu besides 1-4, they will receive an error message.

```
214     # Stop the loop
215     elif menu_choice == "4":
216         break  # out of the loop
217     else:
218         print("Please only choose option 1, 2, or 3")
219
220 print("Program Ended")
```

Figure 1.10 – Closing the Loop

Testing the Script

To test the script, I run the command console. I then type in "CD" to change the directory to the location of my script. I type in the "Python" command followed by the name of our script, Assignment06.py. I'm then given the menu options.

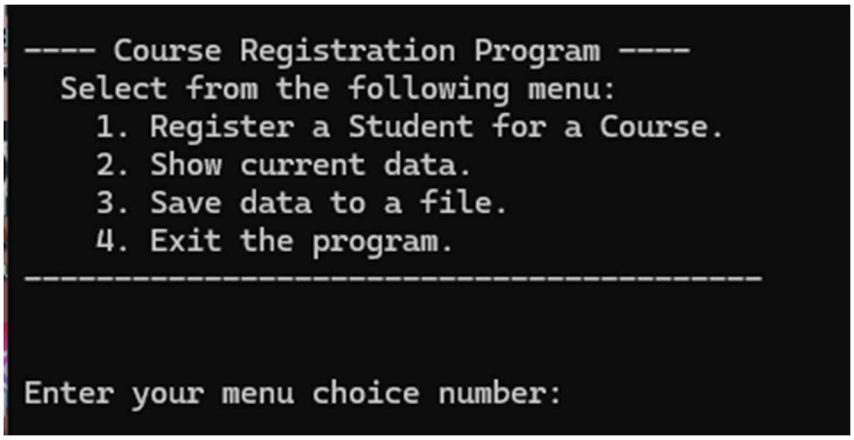

Figure 2.1 – Command Console

I enter "1". I enter my first name, last name and Python 100. I am then presented with the menu again.

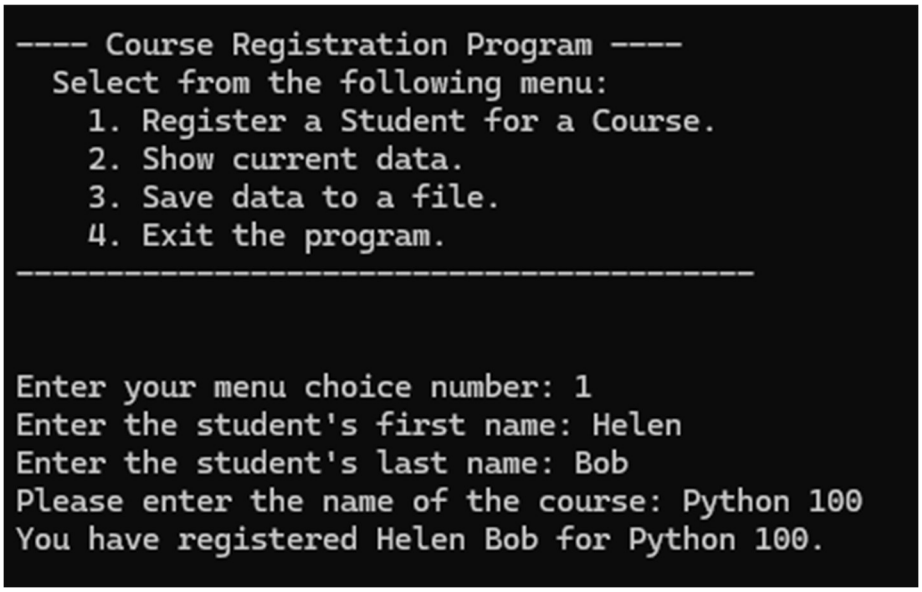

Figure 2.2 – Option 1

I then enter "2" at the menu to display my data.

```
---- 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 menu choice number: 2
------------------------------------------------
Student Bob Smith is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student Evan Peper is enrolled in Python 100
Student Helen Bob is enrolled in Python 100
------------------------------------------------
```

Figure 2.3 – Option 2

I then enter “3” at the menu to save the data to the file.

```
---- 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 menu choice number: 3
The following data was saved to file!
Student Bob Smith is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student Evan Peper is enrolled in Python 100
Student Helen Bob is enrolled in Python 100
```

Figure 2.3 – Option 3

I then check inside my “Enrollments.json” file to show that the data has been recorded.

```
{"FirstName": "Evan", "LastName": "Peper", "CourseName": "Python 100"}]
```

Figure 2.4 – Enrollments.csv

I then enter in a “5” to show my first error message.

```
---- 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 menu choice number: 5
Please, choose only 1, 2, 3, 4

Please only choose option 1, 2, or 3
```

Figure 2.5 – Error Message 1

I then try to run my program without an enrollment.json file available. I get my second error.

```
C:\Python\PythonCourse>Python Assignment05.py
File not found. Creating...
Traceback (most recent call last):
  File "C:\Python\PythonCourse\Assignment05.py", line 47, in <module>
    if not file.closed:
           ^^^^^^^^^^
AttributeError: 'NoneType' object has no attribute 'closed'
```

Figure 2.6 – Error Message 2

I try to add numbers into the name fields to display another error message.

```
---- 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 menu choice number: 1
Enter the student's first name: 123
The last name should not contain numbers.
-- Technical Error Message --
Inappropriate argument value (of correct type).
The last name should not contain numbers.
```

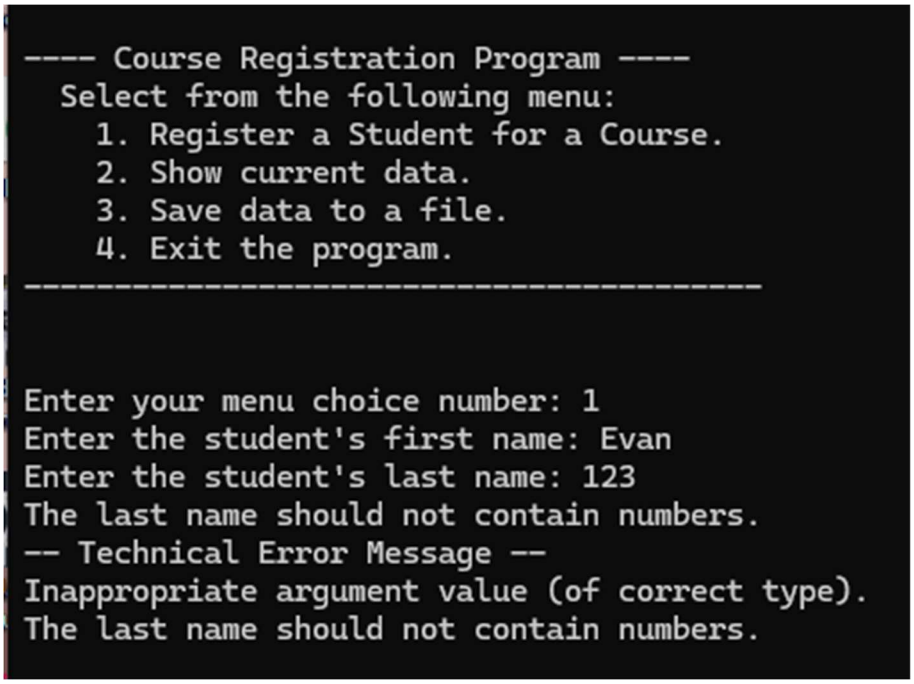

Figure 2.6 – Error Message 3

And lastly I enter "4" to close my program.

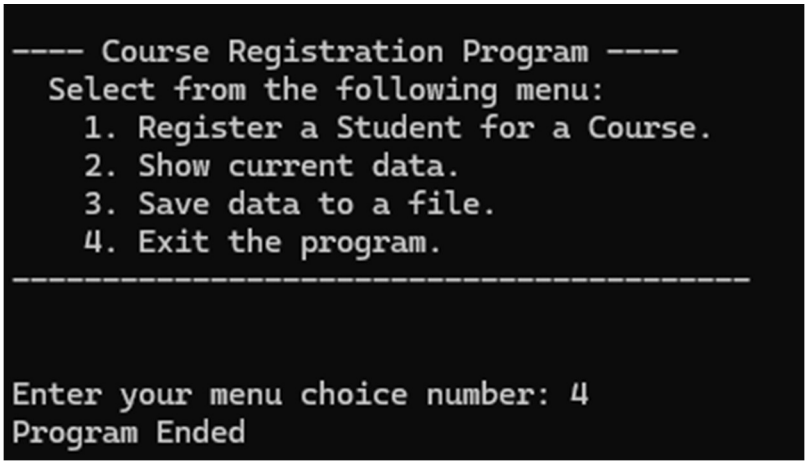

Figure 2.6 – Exiting the program

Summary

In summation, I was able to use my readings, module notes and lecture to give me an understanding of classes and functions. Using this knowledge, I was able to write a script in my IDE to display a menu, open/interpret data from my file "enrollments.json", use a while loop to create an interactive program where a user can store enrollment data and use classes/functions/concerns to better organize my code for future users.

GitHubLink: https://github.com/evanpeps/IntrotoProg-Python-Mod-06