Ryan Choi

November 27, 2024

Foundations of Programming: Python

Assignment 07

https://github.com/megaryanc/IntroToProg-Python-Mod07

Classes and Objects

Introduction

Assignment 07 focused on learning how to do object-oriented programming. It added a technique of creating classes and improving validation methods. PyCharm IDE was used for the script with basic outline provided as a starter file.

Classes Created

A person and a student class were created for the script to perform same functions while practicing how a class could inherit. Figure 1 shows the created classes and definitions. The following classes were referenced from the Lab.

```python
class Person:  1 usage
    """
    A class representing person data.

    Properties:
        first_name (str): The student's first name.
        last_name (str): The student's last name.

    ChangeLog:
        - Ryan Choi, 11.27.2024: Created the class.
    """
    def __init__(self, first_name: str = '', last_name: str = ''):
        self.first_name = first_name
        self.last_name = last_name

    @property  5 usages (3 dynamic)
    def first_name(self)->str:
        return self._first_name.title()

    @first_name.setter  4 usages (3 dynamic)
    def first_name(self, value: str) -> None:
        if value.isalpha():
            self._first_name = value
        else:
            raise ValueError("First name must be alphabetic")

    @property  5 usages (3 dynamic)
    def last_name(self) -> str:
        return self._last_name.title()

    @last_name.setter  4 usages (3 dynamic)
    def last_name(self, value: str) -> None:
        if value.isalpha():
            self._last_name = value
        else:
            raise ValueError("Last name must be alphabetic")

    def __str__(self) -> str:
        return f'{self.first_name},{self.last_name}'
```

```python
class Student(Person): # Inherit from Person  3 usages
    """
    A class representing student data.

    Properties:
    - first_name (str): The student's first name.
    - last_name (str): The student's last name.
    - course_name(str): The course name that the student registered in.

    ChangeLog:
    Ryan Choi,11.27.2024, Created the class.
    """
    def __init__(self, first_name: str, last_name: str, course_name: str):
        super().__init__(first_name, last_name)
        self.course_name = course_name

    @property  6 usages (3 dynamic)
    def course_name(self) -> str:
        return self._course_name

    @course_name.setter  4 usages (3 dynamic)
    def course_name(self,value) -> None:
        self._course_name  = value

    def __str__(self) -> str:
        return f'{super().__str__()},{self.course_name}'
```

Figure 1. The two created classes for assignment 07.

Processing

Processing of the file was updated to run more seamlessly by converting json inputs to be student objects shown in figure 2. Write_data_to_file converted student objects to json file format as shown in figure 3. Both had a validating measure with error messages.

```python
@staticmethod  1 usage
def read_data_from_file(file_name: str, student_data: list) :
    """ This function reads data from a json file and loads it into a list of dictionary rows

    ChangeLog: (Who, When, What)
    RRoot,1.1.2030,Created function

    :param file_name: string data with name of file to read from
    :param student_data: list of dictionary rows to be filled with file data

    :return: list
    """

    try:
        file = open(file_name, "r")
        list_of_dictionary_data = json.load(file)
        for student in list_of_dictionary_data:
            student_object: Student = Student(first_name=student["FirstName"],
                                              last_name=student["LastName"],
                                              course_name=student["CourseName"])
            student_data.append(student_object)
    except Exception as e:
        IO.output_error_messages(message="Error: There was a problem with reading the file.", error=e)
    finally:
        if file.closed == False:
            file.close()
    return student_data
```

Figure 2. Created student object from dictionary.

```python
@staticmethod  1 usage
def write_data_to_file(file_name: str, student_data: list):
    """ This function writes data to a json file with data from a list of dictionary rows

    ChangeLog: (Who, When, What)
    Ryan Choi,11.27.2024,Created function

    :param file_name: string data with name of file to write to
    :param student_data: list of dictionary rows to be writen to the file

    :return: None
    """
    try:
        list_of_dictionary_data: list = []
        for student in student_data: # Convert list of student objects to dictionary rows for JSON file
            enrollments_json: dict = {"FirstName": student.first_name,
                                      "LastName": student.last_name,
                                      "CourseName": student.course_name}
            list_of_dictionary_data.append(enrollments_json)

        file = open(file_name, "w")
        json.dump(list_of_dictionary_data, file)
        file.close()
        IO.output_student_and_course_names(student_data=student_data)
    except Exception as e:
        IO.output_error_messages(message=message,error=e)
    finally:
        if file.closed == False:
            file.close()
```

Figure 3. Converts student object into json then saves.

Input_student_data was also updated to incorporate the new additions for the script as shown in figure 4. It also updated with the improved IO functions.

```python
@staticmethod  1 usage
def input_student_data(student_data: list):
    """ This function gets the student's first name, last name, and course name from the user """

    try:
        first_name = input("Enter the student's first name: ")
        last_name = input("Enter the student's last name: ")
        course_name = input("Please enter the name of the course: ")

        # Create the Student object
        student = Student(first_name=first_name, last_name=last_name, course_name=course_name)

        # Add the Student object to the list
        student_data.append(student)
        print(f"\nYou have registered {student.first_name} {student.last_name} for {student.course_name}.")

    except ValueError as e:
        IO.output_error_messages(message="Input validation failed! Please enter valid alphabetic names.", error=e)
    except Exception as e:
        IO.output_error_messages(message="Error: There was a problem with your entered data.", error=e)
    return student_data
```

Figure 4. updated def input_student_data.

Testing

The program was tested in PyCharm and CMD with the results shown in figure 5 and 6. The proper error handling was examined per the requirement of assignment 7. The file output was to json with the result show in in figure 7.

```
  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: Herman
Enter the student's last name: Miller
Please enter the name of the course: Python 100

You have registered Herman Miller for 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
--------------------------------------------------
Student Bob Smith is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student Ryan Choi is enrolled in Python 100
Student Qwe Qwer is enrolled in Python 100
Student Herman Miller is enrolled in 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.
```

Figure 5. Demonstrates the requirements being met on PyCharm with error handling and multiple registrations.

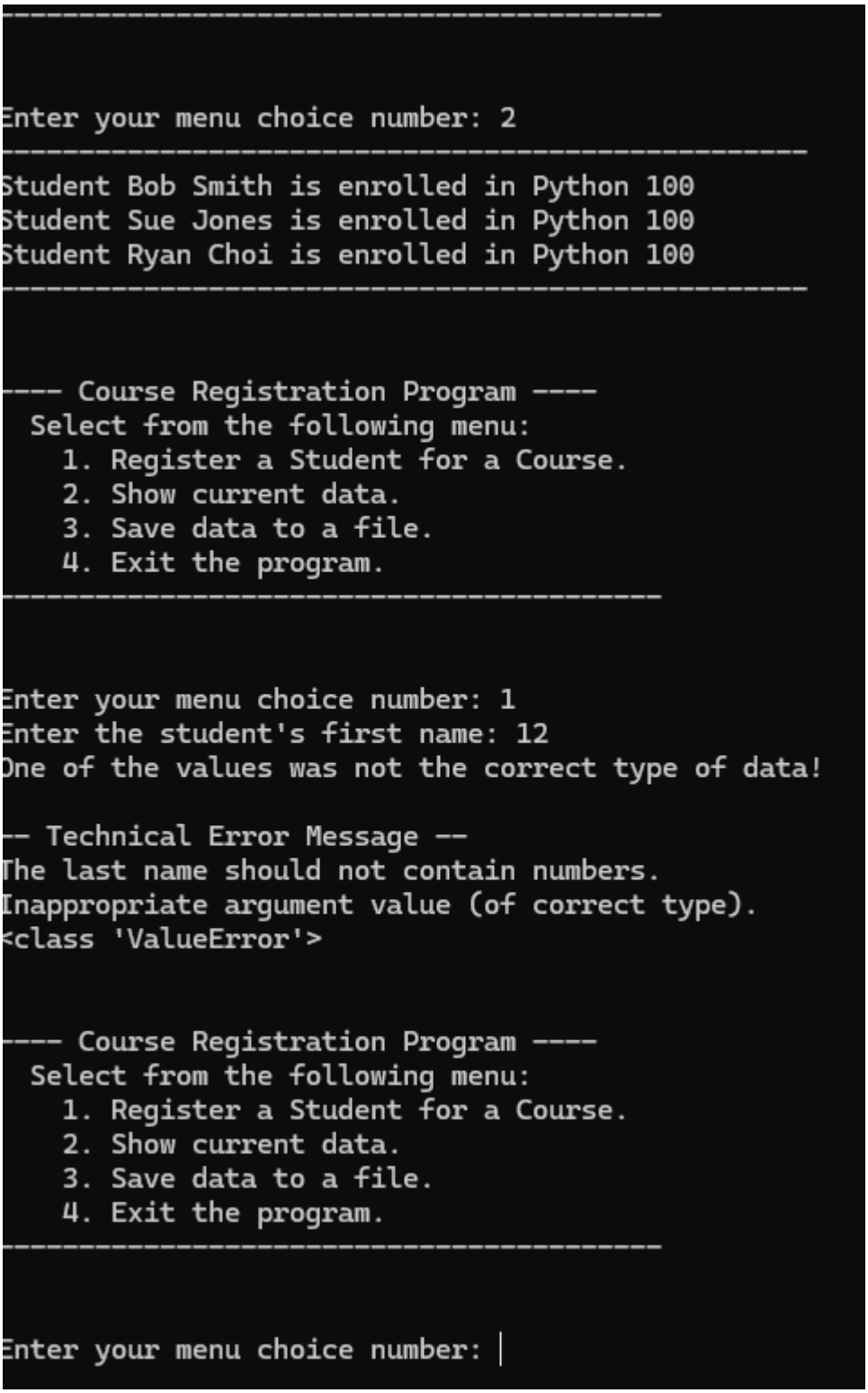

Figure 6. Demonstrates the requirements being met on CMD.

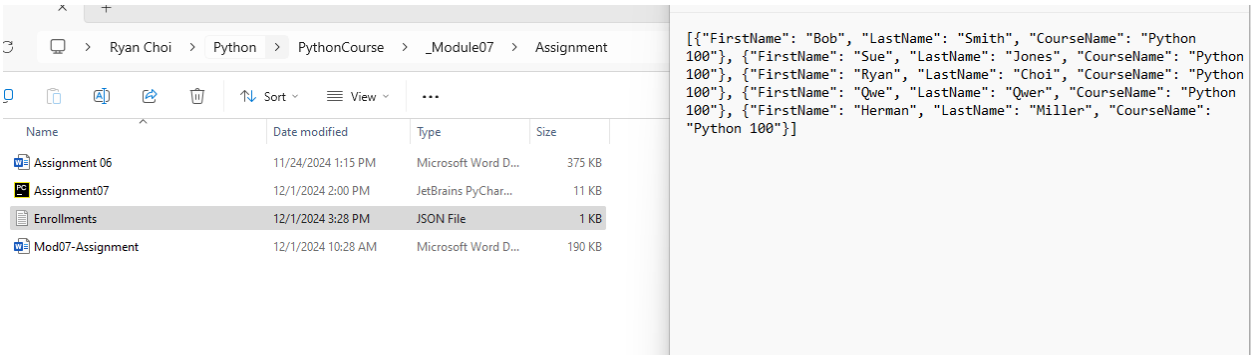

Figure 7. Data recorded in the json.

Summary

The assignment was useful in learning classes and objects while setting up a script to be scalable in the future. Instead of having to write every single class and function, I learned how they could all be incorporated seamlessly with improved error validation. It also set the script up to become more maintainable if anyone else looked at it.