

Richard Khan
11/27/2023
Foundation of Programming: Python
Assignment07

Creating Python Scripts

Introduction

Assignment 7 introduced new terminology and tools for python programming. This assignment and script is similar to previous assignment but now delves into class inheritance, the reuse of code and overridden methods.

Assignment

As mentioned, this week's assignment was similar to assignment06. The constants and variables have stayed the as shown in **Figure 1** and **Figure 2**:

```
# Define the Data Constants
MENU:str=\
    "--Course Registration Program--\n\
    Select from the following menu\n\
    1. Register a Student for a Course\n\
    2. Show current data\n\
    3. Save data to a file\n\
    4. Exit the program"
```

Figure 1: Defining Constants

```
...
FILE_NAME: str = "Enrollments.json"

# Define the Data Variables
students: list = []
menu_choice: str # Hold the choice made by the user.
```

Figure 2: Defining Variables

Separation of Concerns was introduced in our last assignment to help our script be organized and be able to read clearer. This pattern promotes modularity, reduces code complexity and makes it easier to manage and extend software systems. I kept the processing and presentation layer from last assignment with no changes made to that part of the script. Those scripts are not pictured here.

The new classes for assignment07 is called **Person** and **Student**. These are considered data classes because they are attributes, instead of processing classes which focuses on performing certain actions.

Person class represents information about people, in this case, the students first and last name. **Figure 3** shows the code for Person class. It begins with the definition of a class called Person and the two attributes, first and last name. The code is first written using a constructor “__init__”. This is used so we do not have to declare the variables as class-level variables like we have done with the previous assignment. The “self” keyword was used to refer to data found in an object instance and not directly in the class.

After, I created a getter and setter for the first and last name and override method to return the Person data. The getter and setter is for getting data and for setting data. These two code are written separately.

```

class Person:
    """
    A class representing person data.

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

    Change log:
    RichardK, 11/27/2023: Created the Class.
    """
    def __init__(self, first_name: str = '', last_name: str = ''):
        self.first_name = first_name
        self.last_name = last_name

    @property
    def first_name(self):
        return self.__first_name.title()

    @first_name.setter
    def first_name(self, value: str):
        if value.isalpha() or value == "":
            self.__first_name = value
        else:
            raise ValueError("The last name should not contain numbers.")

    @property
    def last_name(self):
        return self.__last_name.title()

    @last_name.setter
    def last_name(self, value: str):
        if value.isalpha() or value == "":
            self.__last_name = value
        else:
            raise ValueError("The last name should not contain numbers.")

    def __str__(self):
        return f'{self.first_name},{self.last_name}'

```

Figure 3: Person Class

Figure 4 created the Student class that inherits from the Person class. The code is written differently after the `__init__` function, using the `super()` method. This connects the first and last name from the Person class, so we no longer need to call out `first_name` and `last_name` in the student constructor. We do need to call out the `course_name`. The getter and setter is no longer needed in the student constructor as it is already call out in the Person class.

```

class Student(Person):
    """
    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 the student is registered for.

    Change log:
    Richardk, 11/27/2023, moved first_name and last_name into a parent class
    """
    def __init__(self, first_name: str = '', last_name: str = '', course_name: str = ''):
        super().__init__(first_name=first_name, last_name=last_name)
        self.course_name = course_name

    def __str__(self):
        return f'{self.first_name},{self.last_name},{self.course_name}'

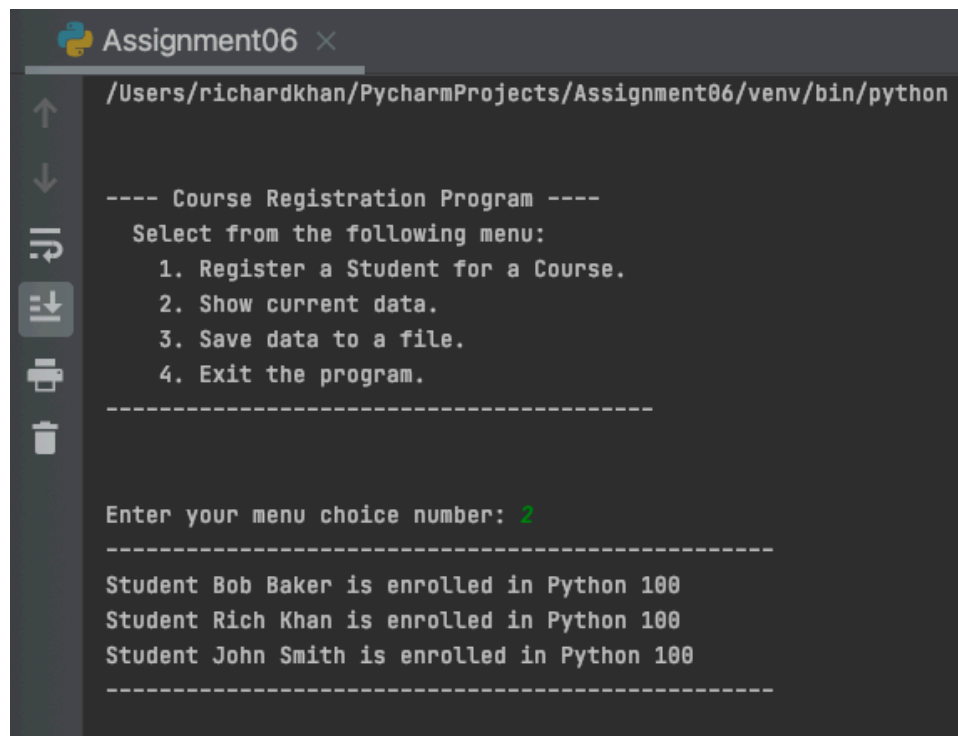
```

Figure 4: Student Class

Once these two classes were incorporated, the rest of the script remained the same and I executed the program as shown below:

Executing the program

Read the data from the Json file:



```

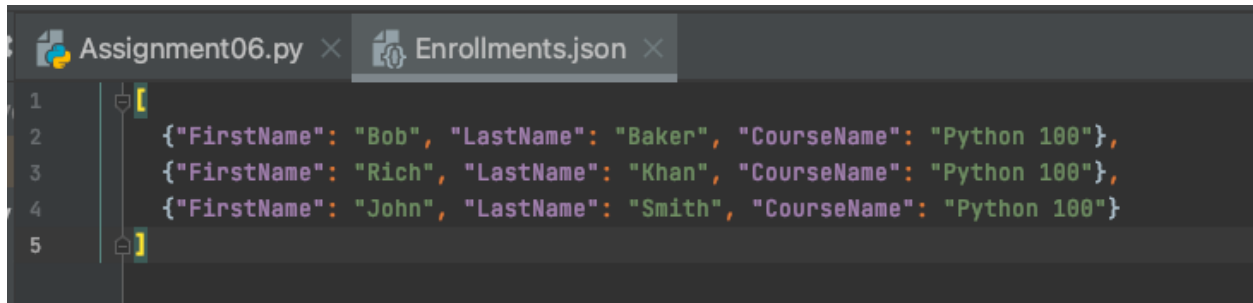
Assignment06 x
/Users/richardkhan/PycharmProjects/Assignment06/venv/bin/python

---- 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.
-----

Enter your menu choice number: 2
-----

Student Bob Baker is enrolled in Python 100
Student Rich Khan is enrolled in Python 100
Student John Smith is enrolled in Python 100
-----

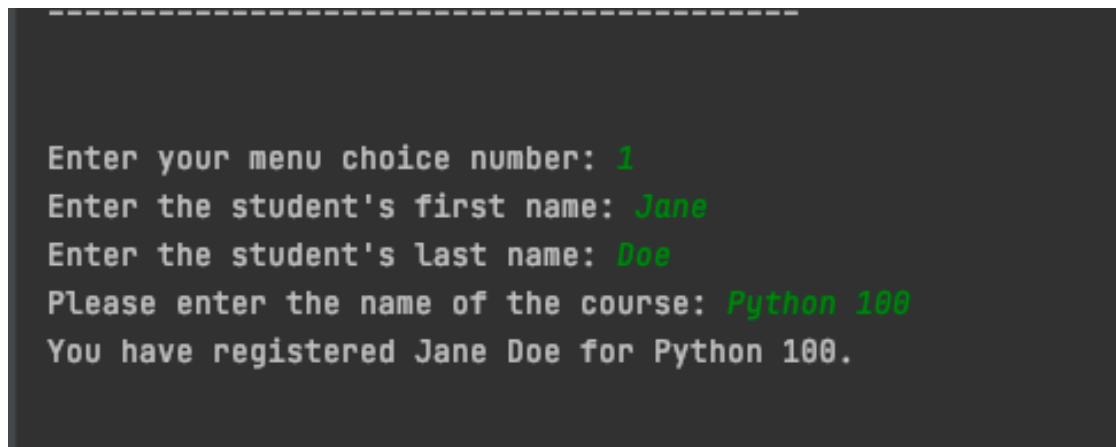
```

A screenshot of a code editor with two tabs: 'Assignment06.py' and 'Enrollmentments.json'. The 'Enrollmentments.json' tab is active, showing a JSON array of three student enrollment records. The records are for Bob Baker, Rich Khan, and John Smith, all enrolled in 'Python 100'.

```
1 [{"FirstName": "Bob", "LastName": "Baker", "CourseName": "Python 100"},  
2 {"FirstName": "Rich", "LastName": "Khan", "CourseName": "Python 100"},  
3 {"FirstName": "John", "LastName": "Smith", "CourseName": "Python 100"}  
4 ]  
5
```

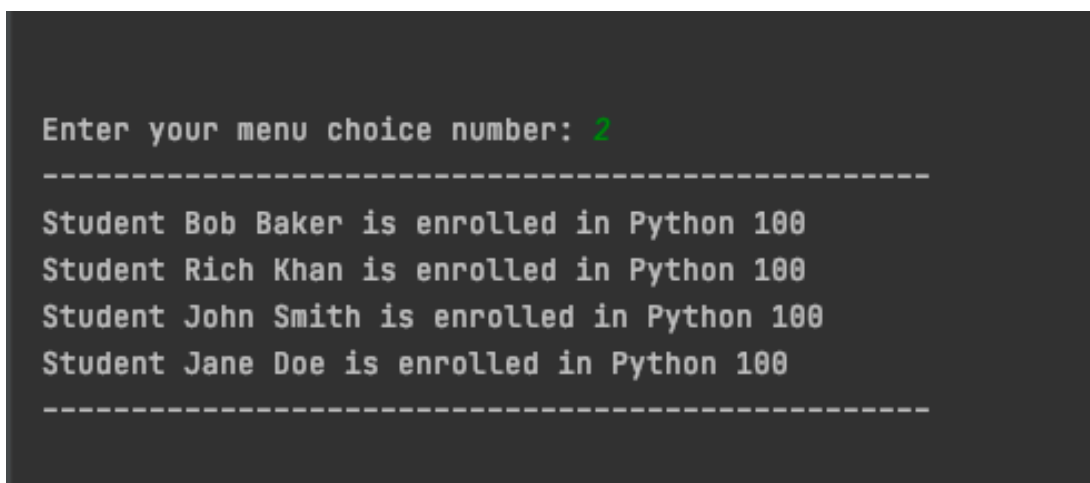
Enrollmentments.json

Select option 1 from the menu choice to incorporate the student first and last name and course name.

A screenshot of a terminal window showing the execution of a program. The user enters '1' for the menu choice, then provides the student's first name 'Jane', last name 'Doe', and course name 'Python 100'. The program outputs a confirmation message.

```
-----  
  
Enter your menu choice number: 1  
Enter the student's first name: Jane  
Enter the student's last name: Doe  
Please enter the name of the course: Python 100  
You have registered Jane Doe for Python 100.
```

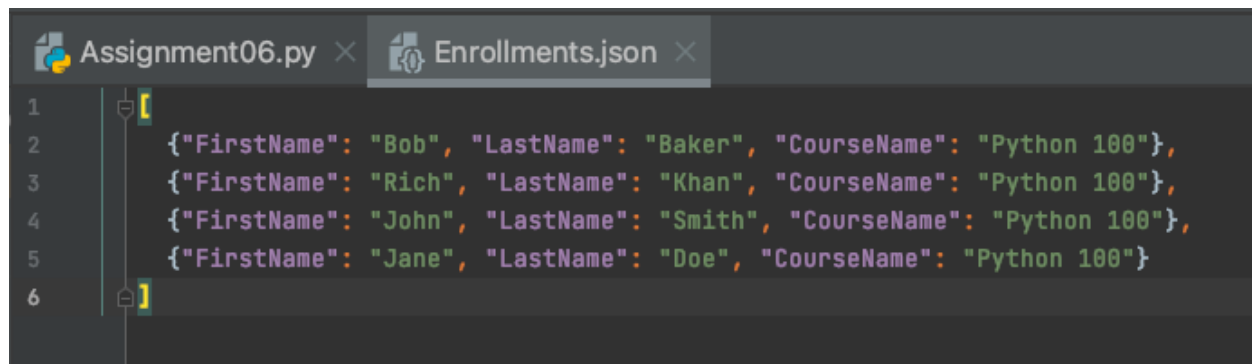
When the “enter your menu choice number” shows again, I inputted 2, in order to present the data.

A screenshot of a terminal window showing the execution of a program. The user enters '2' for the menu choice. The program outputs a list of all enrolled students and their course names, separated by a dashed line.

```
-----  
  
Enter your menu choice number: 2  
-----  
  
Student Bob Baker is enrolled in Python 100  
Student Rich Khan is enrolled in Python 100  
Student John Smith is enrolled in Python 100  
Student Jane Doe is enrolled in Python 100  
-----
```

When the “enter your menu choice number” shows again, I inputted 3, in order to save the data to enrollments.json.

```
-----  
Enter your menu choice number: 3  
-----  
Student Bob Baker is enrolled in Python 100  
Student Rich Khan is enrolled in Python 100  
Student John Smith is enrolled in Python 100  
Student Jane Doe is enrolled in Python 100  
-----
```



The screenshot shows a code editor with two tabs: "Assignment06.py" and "Enrollments.json". The "Enrollments.json" tab is active, displaying a JSON array of four student enrollment records. The records are for Bob Baker, Rich Khan, John Smith, and Jane Doe, all enrolled in Python 100. The code is as follows:

```
1 [{"FirstName": "Bob", "LastName": "Baker", "CourseName": "Python 100"},  
2 {"FirstName": "Rich", "LastName": "Khan", "CourseName": "Python 100"},  
3 {"FirstName": "John", "LastName": "Smith", "CourseName": "Python 100"},  
4 {"FirstName": "Jane", "LastName": "Doe", "CourseName": "Python 100"}]  
5  
6 ]
```

New Enrollments.json

Execution of script in Terminal

```
Terminal: Local x +
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(venv) Richards-MBP:Assignment07 richardkhan$ Python3 Main.py

---- 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 Baker is enrolled in Python 100
Student Rich Khan is enrolled in Python 100
Student John Smith is enrolled in Python 100
Student Jane Doe is enrolled in Python 100
-----
```

Menu choice 2 to show current 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: 1
Enter the student's first name: Vic
Enter the student's last name: Vu
Please enter the name of the course: Python 100
You have registered Vic Vu 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.
-----
```

Menu choice 1 to add data (first name, last name and course name)

```

Enter your menu choice number: 2
-----
Student Bob Baker is enrolled in Python 100
Student Rich Khan is enrolled in Python 100
Student John Smith is enrolled in Python 100
Student Jane Doe is enrolled in Python 100
Student Vic Vu 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.

```

Menu choice 2 to show current data again with the new addition

```

Enter your menu choice number: 3
-----
Student Bob Baker is enrolled in Python 100
Student Rich Khan is enrolled in Python 100
Student John Smith is enrolled in Python 100
Student Jane Doe is enrolled in Python 100
Student Vic Vu 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.

```

Menu choice 3 to save the data to the json file



```

Main.py x Enrollments.json x
1 [{"FirstName": "Bob", "LastName": "Baker", "CourseName": "Python 100"},
2  {"FirstName": "Rich", "LastName": "Khan", "CourseName": "Python 100"},
3  {"FirstName": "John", "LastName": "Smith", "CourseName": "Python 100"},
4  {"FirstName": "Jane", "LastName": "Doe", "CourseName": "Python 100"},
5  {"FirstName": "Vic", "LastName": "Vu", "CourseName": "Python 100"}
6 ]

```

New Json file

Conclusion

This assignment showed us the importance of inheritance and the reuse of code. As our code gets larger and more complicated, the tools we have learn in the past and with this assignment will help us keep our scripts organized and easy to read. The inheritance portion and the reuse of code is vital going forward because it saves time. If one can copy an existing code and make minor changes is a lot better than writing from beginning where mistakes are sure to occur.