Jon Bennefeld
May 29, 2024
Foundations of Programming: Python
IT FDN 110 A
Assignment07
https://github.com/bentleybear17/IntroToProg-Python-Mod07

Working with classes and constructors using a Python script

Introduction

For this assignment, I wrote a Python script that reads from a JSON file, collects data from the user, and writes the collected data back to the JSON file. It performs the same operations as last week's program, except the script is written with classes, constructors, and inherited code, all of which were introduced as new or expanded features for this week.

Working with classes

I read the notes for this week's module a number of times before some of the concepts clicked with me. I didn't understand the concept of objects, or of using the self keyword. It made sense when I watched on of the videos, and the instructor described "self" as belonging to an instance of an object. That, along with the idea of cookie cutters and cookies, helped it make sense to me. Following that, the idea of parent and child classes and inheritance make senses, as using a template of something that already exists and adding more on to it.

Working with constuctors

I understand the idea of initializing variables explicitly, to what values we want them to have at first. The idea of private attributes as just a marker to indicate that the attributes should not be accessed outside the class make sense to me. This week, the new concepts are beyond what I had learned in programming classes as an undergraduate student a long time ago, so instead of thinking, "let's see how python handles concepts that I already know," I've shifted to thinking, "what is this new concept used for and how is its use an improvement on older languages that I have used?"

Code Excerpts

As the scripts now getting longer than 200 lines, I will be including excerpts of code here rather than the full scripts. Figure 1 shows excerpts of myPython script for this week's assignment.

```python
# ------------------------------------------------------------------------------------------ #
# Title: Assignment07
# Description: This assignment demonstrates using data classes with structured error handling
# Change Log: (Who, When, What)
#   Jon Bennefeld, 5/28/2024, Created Script
#   Jon Bennefeld, 5/29/2024, removed duplicate error message for menu choice
# ------------------------------------------------------------------------------------------ #

# ------------------------------------------------------------------------------------------ #
# startup code: a file named Enrollments.json must pre-exist, with some starting data in it
# ------------------------------------------------------------------------------------------ #


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
students: list = []  # a table of student data
menu_choice: str = ""  # Hold the choice made by the user.


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

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

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

    ChangeLog: (Who, When, What)
    Jon Bennefeld, 5/28/2024, created the class.
    """
    @staticmethod
    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)
        Jon Bennefeld, 5/28/2024, created function

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

        :return: list
        """

        try:
            file = open(file_name, "r")
            student_data = json.load(file)
            file.close()
        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
```

```python
    @staticmethod
    def output_student_and_course_names(student_data: list):
        """ This function displays the student and course names to the user

        ChangeLog: (Who, When, What)
        Jon Bennefeld, 5/28/2024, created function

        :parameter student_data: list of dictionary rows to be displayed

        :return: None
        """

        print("-" * 50)
        for student in student_data:
            print(f'Student {student["FirstName"]} '
                  f'{student["LastName"]} is enrolled in {student["CourseName"]}')
        print("-" * 50)

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

        ChangeLog: (Who, When, What)
        Jon Bennefeld, 5/28/2024, created function

        :param student_data: list of dictionary rows to be filled with input data

        :return: list
        """

        try:
            student_first_name = input("Enter the student's first name: ")
            if not student_first_name.isalpha():
                raise ValueError("The last 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("Enter the name of the student's course: ")
```

```python
# Start of main body

# when the program starts, read the file data into a list of lists (table)
# extract the data from the file
students = FileProcessor.read_data_from_file(file_name=FILE_NAME, student_data=students)

# present and process the data
while (True):

    # present the menu of choices
    IO.output_menu(menu=MENU)

    menu_choice = IO.input_menu_choice()

    # input user data
    if menu_choice == "1":  # This will not work if it is an integer!
        students = IO.input_student_data(student_data=students)
        continue

    # present the current data
    elif menu_choice == "2":
        IO.output_student_and_course_names(students)
        continue

    # save the data to a file
    elif menu_choice == "3":
        FileProcessor.write_data_to_file(file_name=FILE_NAME, student_data=students)
        continue

    # stop the loop
    elif menu_choice == "4":
        break  # out of the loop
    else:
        pass

print("Program Ended")
```

Figure 1: Python script excerpts

Testing script in Pycharm

The assignment requires that I successfully run my script in both Pycharm and from the command shell. Figure 2 shows the successful run of my program in Pycharm, with each menu selection chosen at least once, and some of the error handling code tested.

```
C:\Users\bentl\Documents\Python\Pycharm\Module03\pythonProject\.venv\S


---- 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 Vic Vu is enrolled in Math 160
Student Sue Jones is enrolled in Programming 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.
----------------------------------------


Enter your menu choice number:
```

```
Enter your menu choice number: 1
Enter the student's first name: 3
One of the values was not the correct type of data!

-- Technical Error Message --
The first name should not contain numbers.
Inappropriate argument value (of correct type).
<class 'ValueError'>


---- 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: Chad
Enter the student's last name: Bad4
One of the values was not the correct type of data!

-- Technical Error Message --
The last name should not contain numbers.
Inappropriate argument value (of correct type).
<class 'ValueError'>
```

```
Enter your menu choice number: 1
Enter the student's first name: Chad
Enter the student's last name: Bad
Enter the name of the student's course: Chemistry 201

You have registered Chad Bad for Chemistry 201.


---- 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
-------------------------------------------------
Student Vic Vu is enrolled in Math 160
Student Sue Jones is enrolled in Programming 101
Student Chad Bad is enrolled in Chemistry 201
-------------------------------------------------


---- 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: 4
Program Ended

Process finished with exit code 0
```

Figure 2: Script run in Pycharm

Testing script the command shell

Figure 3 shows the successful run of my program in the command shell.

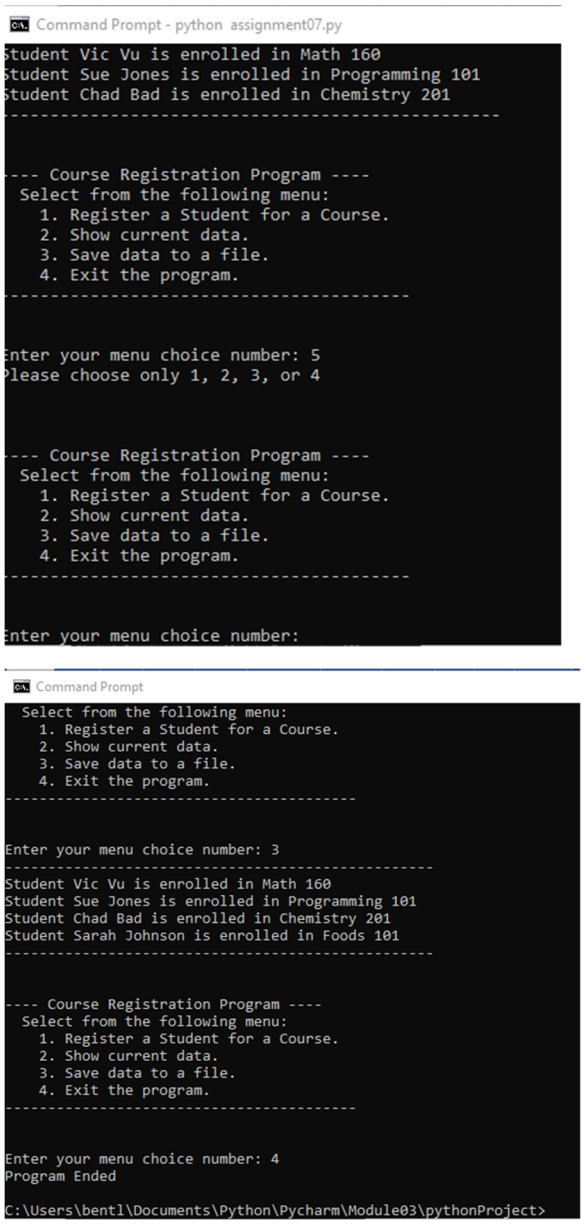

Figure 3: Script run in Command Shell

Output file

Figure 4 shows that my script wrote student information into a JSON file.

Figure 4: Output file opened in Notepad

Summary

This assignment introduced classes, constructors, and properties and naming conventions in Python that were wholly new to me. The concepts were presented as ways to build code that is more organized than the more manual coding techniques from past weeks. I learned about objects, and why Python is an object-oriented language, and how templates for objects are built and initialized.