Name: Celia Louie
Date: November 20, 2023
Course: IT FDN 110 B
GitHubURL: https://github.com/celialouie/IntroToProg-Python-Mod6

Assignment 06 - Functions

Introduction

For this assignment, I created a Python program that builds on what we've learned so far and adds on the use of functions, classes, and using the separation of concerns pattern for students' course registration.

Creating a program

I started out by using the starter file provided and filling out the script header. This assignment builds on Assignment 5 and keeps the data constants and 2 of the data variables (menu_choice and students) since I will use the variables locally in our functions. I also imported json as we will be using the json module in this assignment. Since this assignment built on our past assignments, I will highlight the new additions.

```
# ------------------------------------------------------------------------------------------ #
# Title: Assignment06
# Desc: This assignment demonstrates using functions with structured error handling
# Change Log: (Who, When, What)
#   Celia Louie, 11/19/2023, Created script
#   <Your Name Here>,<Date>,<Activity>
# ------------------------------------------------------------------------------------------ #
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.
-------------------------------------
'''
# Define the Data Constants
FILE_NAME: str = "Enrollments.json"

# Define the Data Variables and constants
menu_choice: str
students: list = []

```

Figure 1: Added script header and defined constants and variables.

This assignment outlined two classes – FileProcessor and IO for the processing and presentation layers. I created a class named FileProcessor for my processing functions and used the @staticmethod decorator for each function since the function code will be static and I

included doc strings for a description. The function read_data_from_file takes in the parameters file_name and student_data to read a json file into a table. File_name is a string which will be our json file and student_data is the dictionary. The function write_data_to_file takes the same parameters and will write data to the json file. I'm not sure why PyCharm kept giving me yellow swigglys but it wanted me to add a global statement. After I created the IO class and functions, I came back to these two functions to add the IO.output_error_messages function to print out a custom error message for my error handling.

```python
# Processing
class FileProcessor:
    """
    A collection of processing layer functions that process a json file

    ChangeLog: (Who, When, What)
    Celia Louie,11.20.2023,Created class and functions
    """

    @staticmethod
    def read_data_from_file(file_name: str, student_data: list):
        """
         This function reads in data from a json file into a table
         """
        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.")
        finally:
            if not file.closed:
                file.close()
        return student_data

    @staticmethod
    def write_data_to_file(file_name: str, student_data: list):
        """
        This function writes data to the json file
        """
        try:
            file = open(file_name, "w")
            json.dump(student_data, file)
            file.close()
            IO.output_student_courses(student_data=student_data)
        except Exception as e:
            IO.output_error_messages(message="Error: There was a problem with writing to the file. \n "
                                             "Please check that the file is not open by another program.", error=e)
            if not file.closed:
                file.close()
        finally:
            if not file.closed:
                file.close()

```

Figure 2: Processing layer with the class FileProcessor and functions

Next, I created the presentation layer. The IO class holds the functions for taking the user input and presenting the data. The first function within this class was output_error_message which takes in the parameters message and error to print out a custom error message.

```python
# Presentation
class IO:
    """
    A collection of presentation layer functions that manager user input and output

    ChangeLog: (Who, When, What)
    Celia Louie,11.20.2023,Created class and functions
    """

    @staticmethod
    def output_error_messages(message: str, error: Exception = None):
        """
        This function prints a custom error message
        """
        print(message, end="\n\n")
        if error is not None:
            print("-- Technical Error Message -- ")
            print(error, error.__doc__, type(error), sep='\n')

```

Figure 3: Presentation layer with error message function

The next few functions take what we've done so far with the menu options with the user input and data output but rewritten as a function with parameters. The function output_menu prints out the menu choices. The function input_menu_choice prompts the user for a choice and we will see this in action later on in the if-elif statement with the main body of the script as we return the user's choice. These next few functions call the same IO.output_error_messages function for the error handling. The function output_student_courses takes the student_data list and prints out each student's registration. This is where I kept running into errors with the key but realized that my json file needed to be modified to reflect the keys in this assignment starter file rather than what I originally had from the last assignment. Lastly, the function input_student_data is intended to prompt the user for the student's first name, last name, and course name for registration and return the student_data list.

```python
    @staticmethod
    def output_menu(menu: str):
        """
        This function displays the menu choices
        """
        print(menu)

    @staticmethod
    def input_menu_choice():
        """
        This function takes in the user's input
        """
        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, or 4")
        except Exception as e:
            IO.output_error_messages(e.__str__())
        return choice

    @staticmethod
    def output_student_courses(student_data: list):
        """
        This function displays the registered students
        """
        print("-" * 50)
        for student in student_data:
            print(f'Student {student["FirstName"]} {student["LastName"]} is enrolled in {student["CourseName"]}')
        print("-" * 50)

    def input_student_data(student_data: list):
        """
        This function takes the user input for student registration
        """
        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("Please enter the name of the course: ")
            student = {"FirstName": student_first_name,
                       "LastName": student_last_name,
                       "CourseName": course_name}
            students.append(student)
            print(f"You have registered {student_first_name} {student_last_name} for {course_name}.")
        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: Presentation layer continued

The menu of options in this assignment are similar to what we did in Assignment 3 and 4 so I will highlight the new additions. Menu choice 1 prompts the user to input the student's first name, student's last name, and the course course name. I included isalpha() to check that the input for the student's first and last name doesn't contain any numbers. I added the user's input as a dictionary to the empty dictionary student_data we defined earlier in the script.

Figure 3: Prompt user for input with error handling

For menu choice 2, to present the data I used the print function and a for loop to print out the data with a custom message.

Figure 4: Present the data

Figure 5: Save the data to json file

Testing

I tested the program to make sure it ran correctly by taking the user's input, displaying the inputs, saving the input as a json file, showing error messaging, and allowing for multiple registrations. I ran the program in both PyCharm and terminal.

Summary

I created a program using lists and files to display a menu option for the user to input a student's registration for a Python course with functions, classes, and using the separation of concerns pattern. I tested the program to make sure it runs correctly in PyCharm.