Ronald Mursewick
November 13, 2023
Foundations of Programming:
Python Assignment 05

# Lists, try-raise-except, and Writing .json files and github.

## Introduction

The purpose of this writing is to document some enhancements to our prior program. One of these extensions is to use a list to hold our temporary student data before we write it to a .json file. The change here is writing to a json file. JSON stands for Java Script Object Notation. JSON is a more extensible format structure. It is used widely in web applications. In assignment 04 we incorporated pandas methods for managing out in program data structure. There a series of methods that make a two dimensional list very east to manipulate. The selection of methods are called Pandas. I chose to use these because of the ease they bring to table handling. Pandas also has methods for handling json data structures. So we build upon the existing structure and add output methods to create our json class registration file. In this lesson we will also use try-raise-except to verify that we are getting clean data into out class file.

## Pandas

```
# import pandas to help use 2 dimensional arrays easily
import pandas as pd

#
# Create the dataframe with the header row

students_pd = pd.DataFrame(columns=("FirstName", "LastName", "CourseName", "CoursePrice", "CourseCost"))
```

***Figure 1. Importing Pandas methods and creating the student data DataFrame***

Our DataFrame will be 5 columns wide and able to have rows added indefinitely. In the DataFrame creation above we are creating the header row for the data.

```
#
# we will append the newly added student to the dataframe
# this DataFrame can extend indefinitly

students_pd.loc[len(students_pd)] = student_data
```

***Figure 2. Appending the newly added student to the DataFrame***

One of the changes is using pandas over variables. Notice the “0” just to the left of the last name “Harrison”. When the operator selects option “2”, we print the DataFrame. With indexing the “zero” row is the first row of the array. Since Harrison was our first student entered he is saves in the table in row zero

```
Total Students registered so far is:  1

  FirstName LastName  CourseName  CoursePrice  CourseCost
0  Harrison     Ford  Python 100       999.98     1089.98
```

***Figure 3. Shows how data is stored in the DataFrame***

Moving the student data from the DataFrame we build with the information the operator entered during the program run. Another reason for using pandas is ease of data handling. Here we see that with a one line command (method) we are able to write the entire contents of the DataFrame to the .json file. We choose the mode=”a” so we append our new data to any other data that has been entered into the file. We will also use the orient=”records” to keep our data in a tabular format.

```
173         #
174         # now we will append the newly added students to the .json file
175
176         students_pd.to_json(FILE_NAME, mode="a", index=False, lines=True, orient="records")
177
```

***Figure 4. Use of the DataFrame to .json file method.***

Another process introduced in this lesson is validating data with the try-raise-except. The way we use the check in out code is to verify that the first and last names are alphabetical.  We use the .isalpha method to test the operators input to confirm that it is a name.

```
    #
    # confirm the student first name is alphebetic other wise go back and ask again

    try:
        if not student_first_name.isalpha():
            raise ValueError("The first name must be alphabetic. ")
    except ValueError as e:
            print("\n", e, "\n")
            continue
```

***Figure 5. using try-raise-except example***

## Program Operation

First we will choose operation 1 to register a student.

```
C:\Users\Ron\PycharmProjects\pythonProject>python 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 Selection:
1

Enter the students first name:
neil
Enter the students last name:
armstrong
```

***Figure 6. Menu selection 1***

Now the operator chooses select 2, so we show the operator what data has been entered.

```
---- 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 Selection:
2


Neil Armstrong is registered in the class Python 100 and has paid the fee: $1089.98.
```

***Figure 7. Menu selection 2***

In menu select 3, we save the data to the .csv file. In this case another operator has entered two other students. In the beginning of the program we checked to see if a student data file existed. If the file

existed we opened the for “append” to we could add data to the already registered student list. After we write the latest student to the file we show all of the students that have been registered for the class. Another change in the output for selection 3, was to change $stdout to a file and print the enrollment list to a file. Once the output is complete change $stdout back to the console for dialogue to the operator.

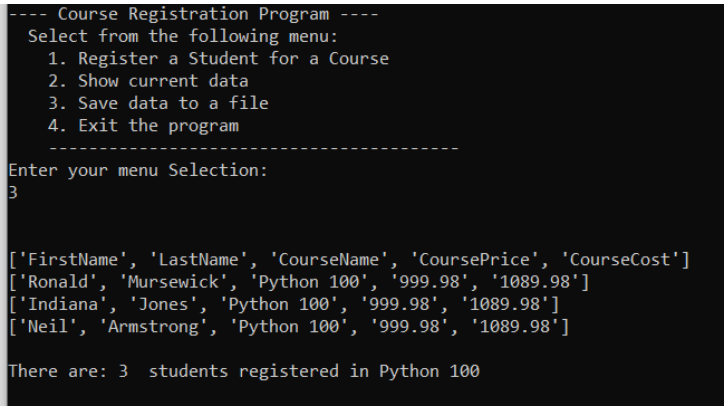


***Figure 8. Menu selection 3***

Menu selection ends the program and returns us to the command prompt.

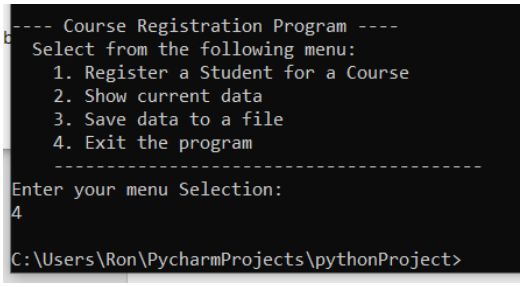


***Figure 9. Menu selection 4***

We put some error processing code in so that if the operator tried to show the current data before a student was registered, we would let them know that they needed to enter student data.

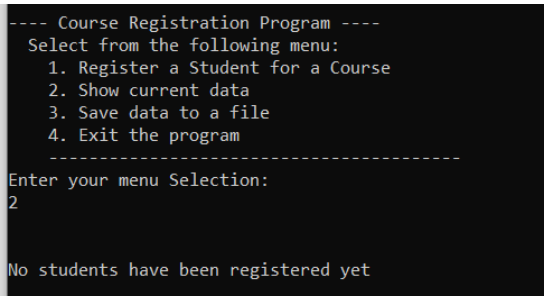


***Figure 10. Menu selection 2, when no student has been registered.***

The Enrollments.json data file as displayed in Notepad.

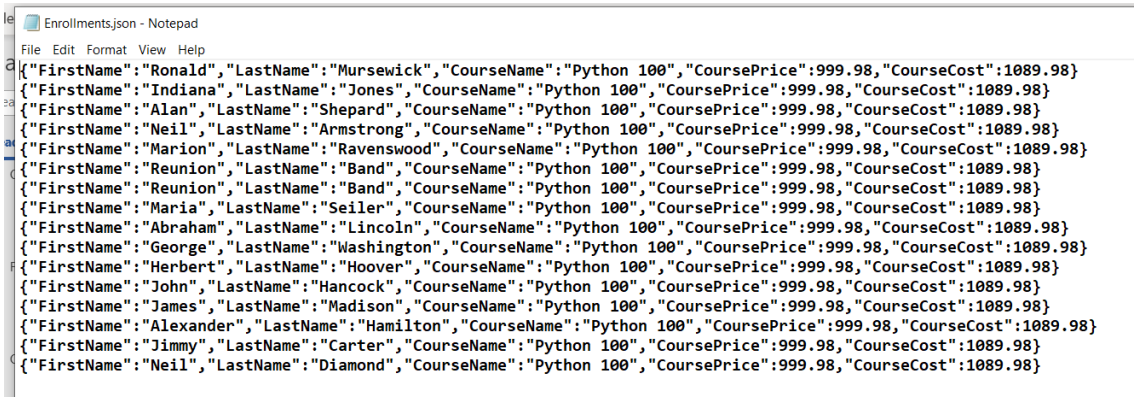


***Figure 11. Enrollments.json data file***

## Summary

We continue to build upon our first assignment. Now we add operations to perform repetitive tasks until a specified condition is met. The tasks are accomplished with the use of “while”, “if” – “else” – “elif”. We also added some enhancement to append to our student data if the file already exists. With adding complexity we want to look for methods that enhance our ability to manage larger amounts of data as efficiently as possible. Pandas is a powerful tool to aid us with data management and manipulation. We change out output file type to json since it is widely is for data manipulation today. We also add data checking to ensure that the data we write to out file is clean or normalized to minimize the chance for errors or misinterpretation of data later in the processing.