Kayden Ward

11/013/2023

Python 100

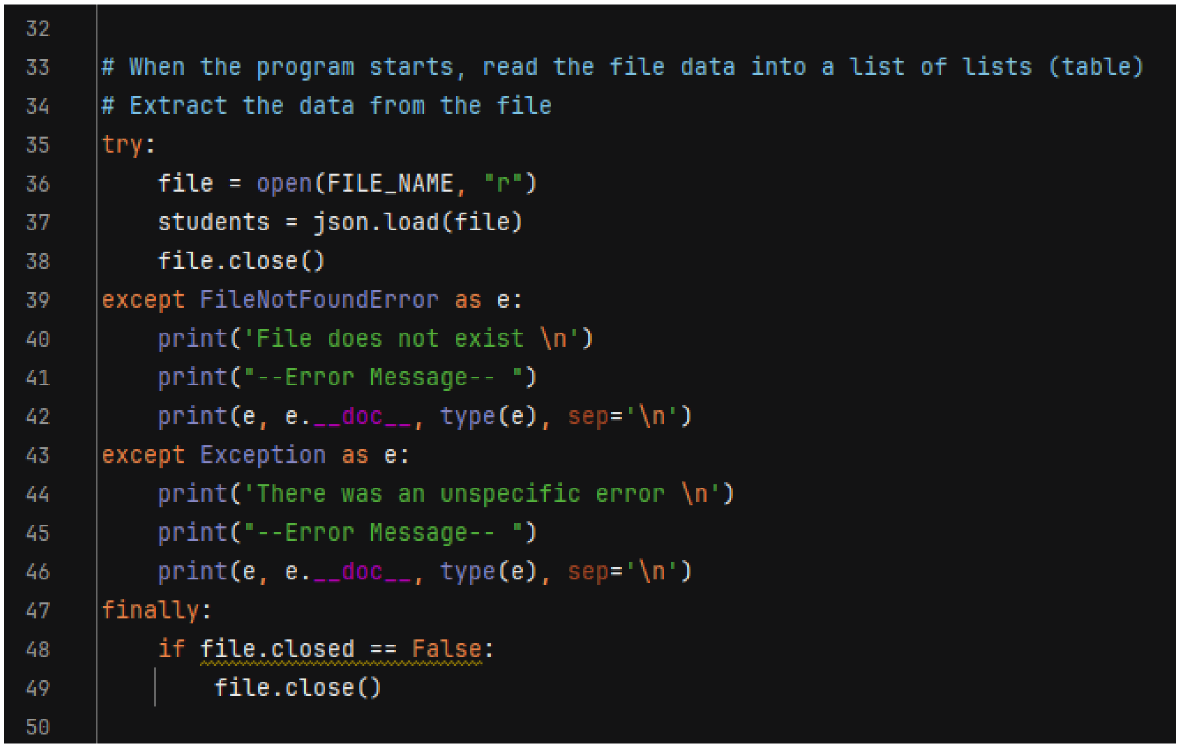
Assignment05

Advanced Collections and Error Handling

In this assignment I implement the use of dictionaries to store sets of data. I also use error handling methods, to make sure the program runs well.

Writing the Code

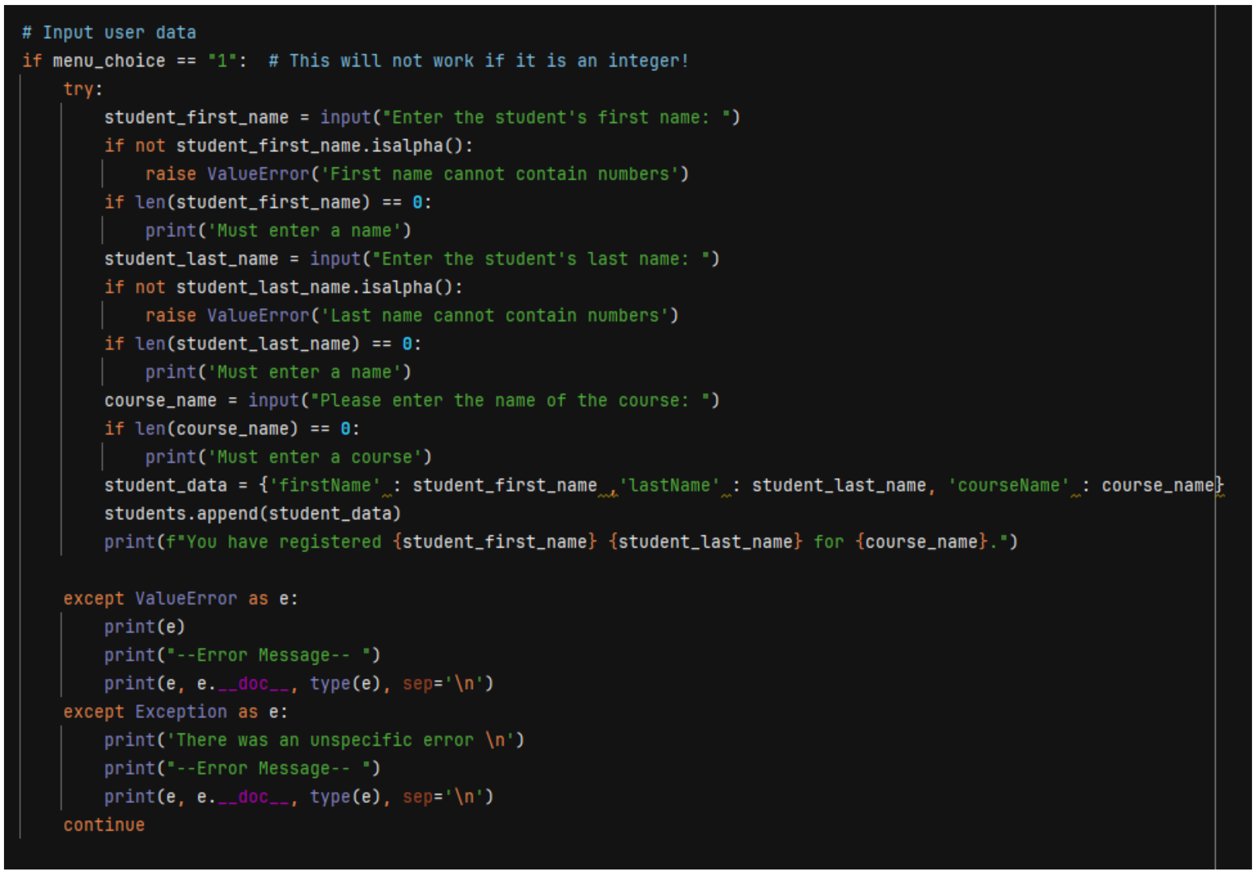
I started with switching the student_data list to a dictionary, and changing the file name to call a json file. Then I wrote the code to put the contents of the file into the list of students. I opened the file in ‘r’ mode and used json.load() for that. I added in the exception handling for if the file doesn't exist, and any other error that may occur. It prints a message saying what happened, and some details of the error. In the finally statement I make sure the file gets closed.



```python

# When the program starts, read the file data into a list of lists (table)
# Extract the data from the file
try:
    file = open(FILE_NAME, "r")
    students = json.load(file)
    file.close()
except FileNotFoundError as e:
    print('File does not exist \n')
    print("--Error Message-- ")
    print(e, e.__doc__, type(e), sep='\n')
except Exception as e:
    print('There was an unspecific error \n')
    print("--Error Message-- ")
    print(e, e.__doc__, type(e), sep='\n')
finally:
    if file.closed == False:
        file.close()

```

Figure 1: Loading in JSON file contents

Option 1 got the addition of error handling there as well. I put the input lines into a try statement, used raise ValueError to make sure no numbers were entered. I also check to make sure there was something entered using ‘len’ to make sure the value is greater than 0. With the entered data, I format it into student_data as a dictionary.

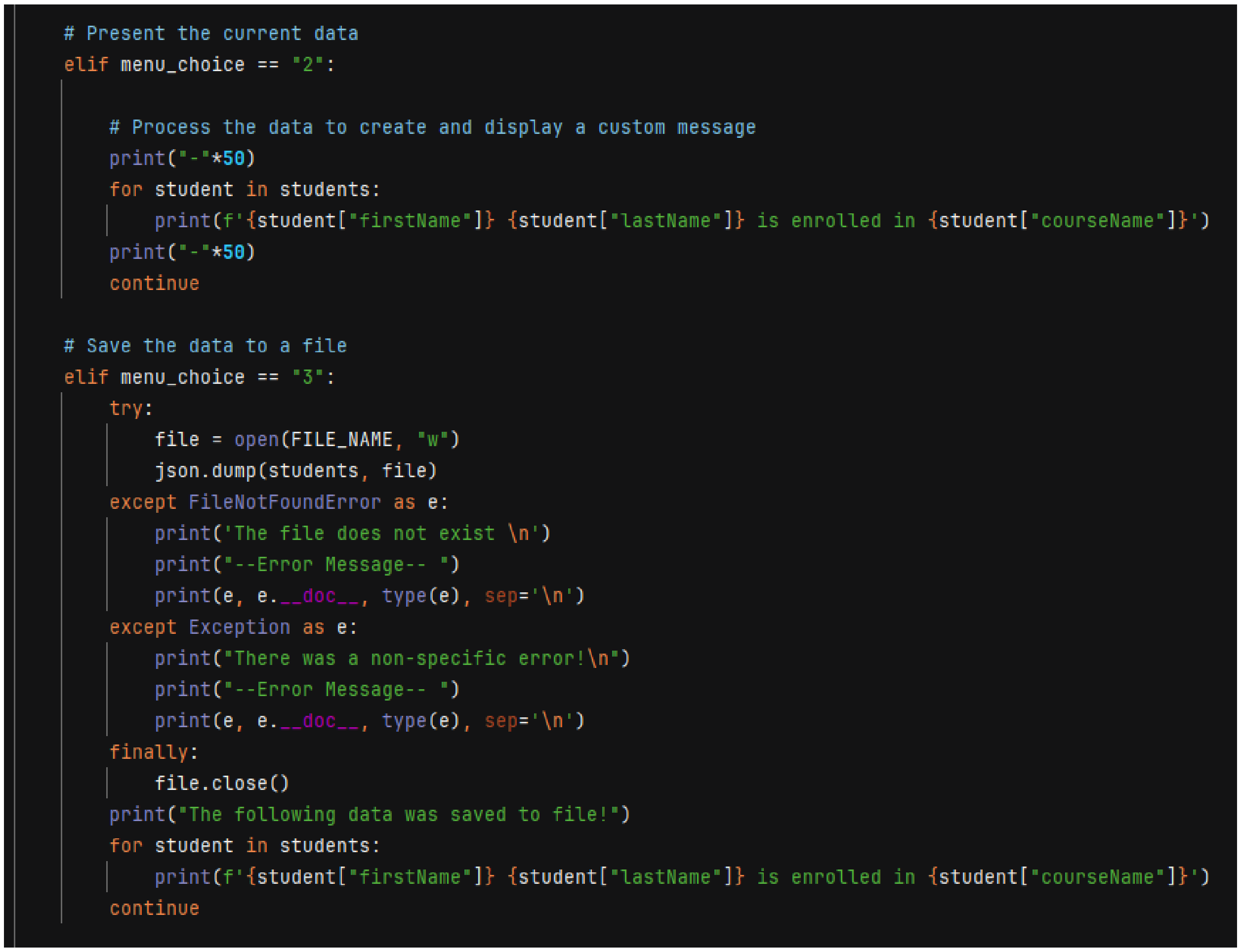


```python
# Input user data
if menu_choice == "1":  # This will not work if it is an integer!
    try:
        student_first_name = input("Enter the student's first name: ")
        if not student_first_name.isalpha():
            raise ValueError('First name cannot contain numbers')
        if len(student_first_name) == 0:
            print('Must enter a name')
        student_last_name = input("Enter the student's last name: ")
        if not student_last_name.isalpha():
            raise ValueError('Last name cannot contain numbers')
        if len(student_last_name) == 0:
            print('Must enter a name')
        course_name = input("Please enter the name of the course: ")
        if len(course_name) == 0:
            print('Must enter a course')
        student_data = {'firstName' : student_first_name ,'lastName' : student_last_name, 'courseName' : course_name}
        students.append(student_data)
        print(f"You have registered {student_first_name} {student_last_name} for {course_name}.")

    except ValueError as e:
        print(e)
        print("--Error Message-- ")
        print(e, e.__doc__, type(e), sep='\n')
    except Exception as e:
        print('There was an unspecific error \n')
        print("--Error Message-- ")
        print(e, e.__doc__, type(e), sep='\n')
    continue
```

Figure 2: Menu option 1

I initially had trouble with option 2, and getting it to give me a formatted output, like I had trouble with last week. At first, at the startup, I was taking the contents from the file one by one using a for loop. For some reason with the for loop, I could not format what it had put into the list of students. When I changed it to json.load(), the formatting worked fine.

For option 3, I opened the file in ‘w’ and used json.dump() to put the contents of students into the file. I used the same error handling as at the start, to handle if there is no file, and any other errors.

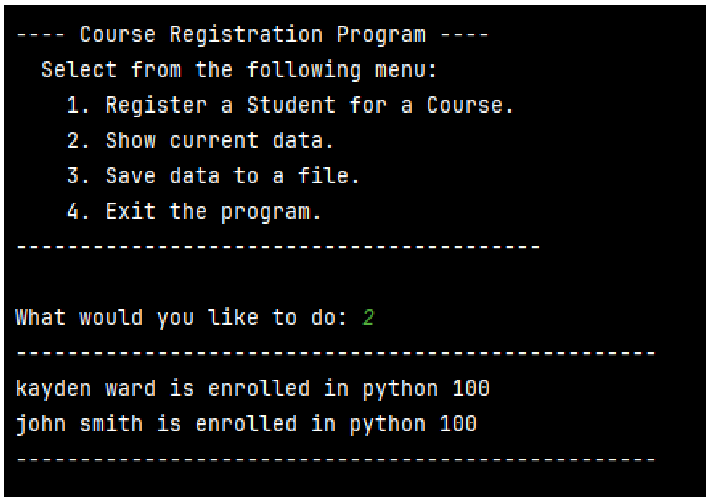


```python
    # Present the current data
    elif menu_choice == "2":

        # Process the data to create and display a custom message
        print("-"*50)
        for student in students:
            print(f'{student["firstName"]} {student["lastName"]} is enrolled in {student["courseName"]}')
        print("-"*50)
        continue

    # Save the data to a file
    elif menu_choice == "3":
        try:
            file = open(FILE_NAME, "w")
            json.dump(students, file)
        except FileNotFoundError as e:
            print('The file does not exist \n')
            print("--Error Message-- ")
            print(e, e.__doc__, type(e), sep='\n')
        except Exception as e:
            print("There was a non-specific error!\n")
            print("--Error Message-- ")
            print(e, e.__doc__, type(e), sep='\n')
        finally:
            file.close()
        print("The following data was saved to file!")
        for student in students:
            print(f'{student["firstName"]} {student["lastName"]} is enrolled in {student["courseName"]}')
        continue
```

Figure 3: Menu options 2 and 3

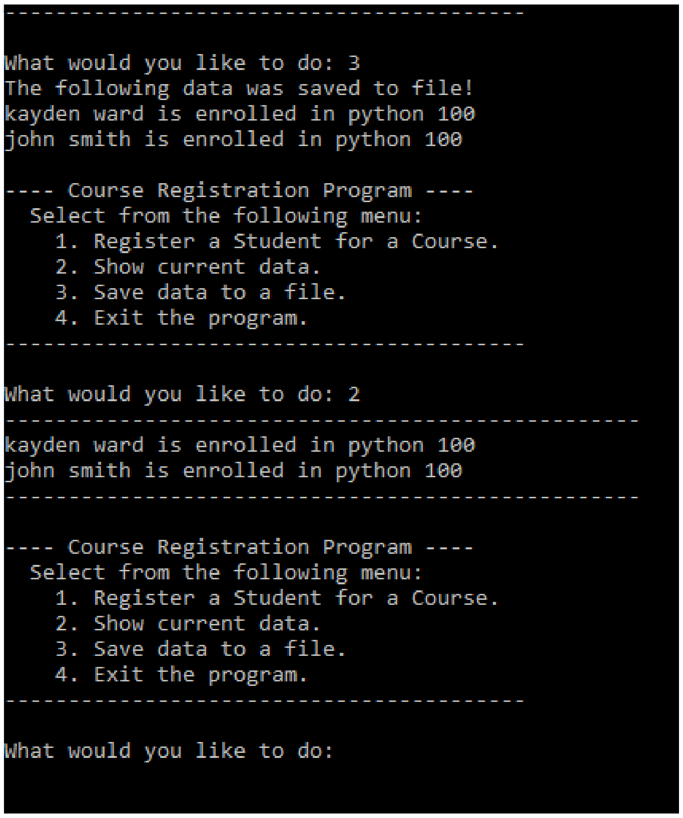
Testing the Code

The code works in both PyCharm and the command prompt. I am able to load the data from the JSON file, add more data, save the new data, and print it all out.



```
---- 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
--------------------------------------------------
kayden ward is enrolled in python 100
john smith is enrolled in python 100
--------------------------------------------------
```

Figure 4: Code running in pyCharm



```
-----------------------------------------

What would you like to do: 3
The following data was saved to file!
kayden ward is enrolled in python 100
john smith 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.
-----------------------------------------

What would you like to do: 2
--------------------------------------------------
kayden ward is enrolled in python 100
john smith 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.
-----------------------------------------

What would you like to do:
```

Figure 5: Code running in command prompt

Summary

I did not have a lot of issues with this assignment, It was pretty easy to convert the starter to using a dict instead of list, and csv to json. Writing error handling was also a lot less harder than I thought it was going to be. I feel confident in my ability to do these things now.