Name: Blaise Konzel

Date: May 21, 2025

Course: Foundation of Programming (Python): Python 100

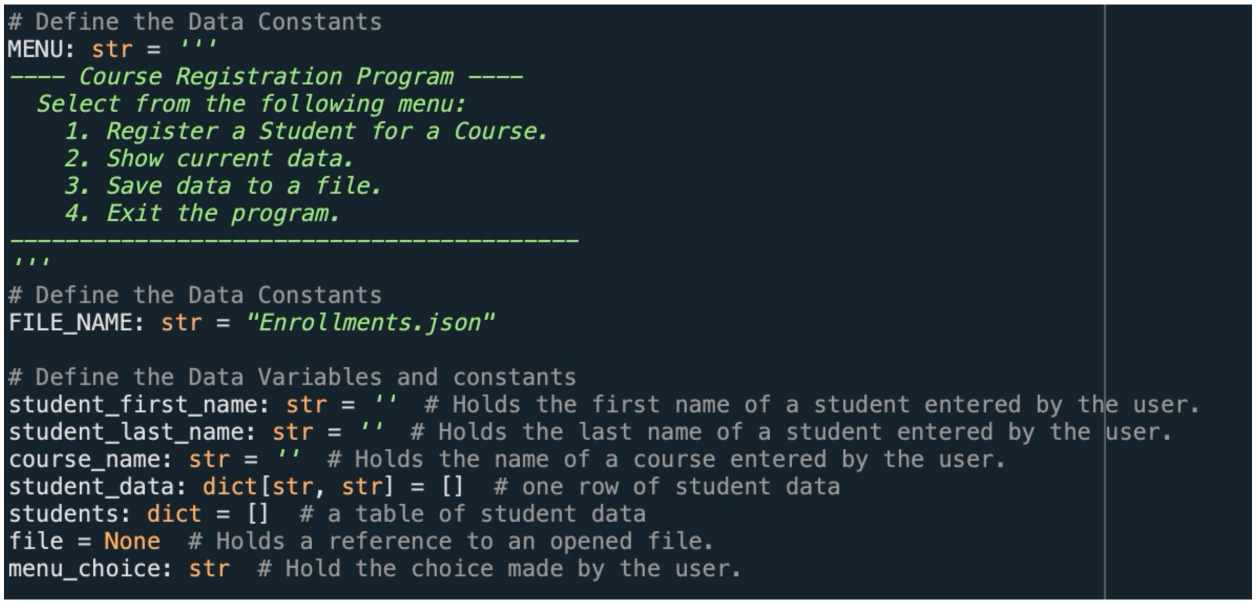
Assignment 05

Introduction

This assignment introduced working with JSON files as well as including proper provisions for error handling during the use of the course registration database code.

Code

1) Defining constants and variables

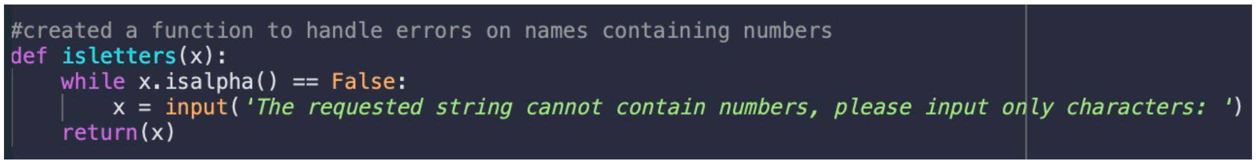


```
# Define the Data Constants
MENU: str = '''
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
-----------------------------------------
'''
# Define the Data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

# Define the Data Variables and constants
student_first_name: str = ''  # Holds the first name of a student entered by the user.
student_last_name: str = ''  # Holds the last name of a student entered by the user.
course_name: str = ''  # Holds the name of a course entered by the user.
student_data: dict[str, str] = []  # one row of student data
students: dict = []  # a table of student data
file = None  # Holds a reference to an opened file.
menu_choice: str  # Hold the choice made by the user.
```

Exactly the same as last assignment, this assignment began with defining the constants and variables we will use later on in our code. The big difference in this assignment compared to the previous assignment is the change in the initialization of 'students' and 'student_data'. We now initialize these as a dictionaries.

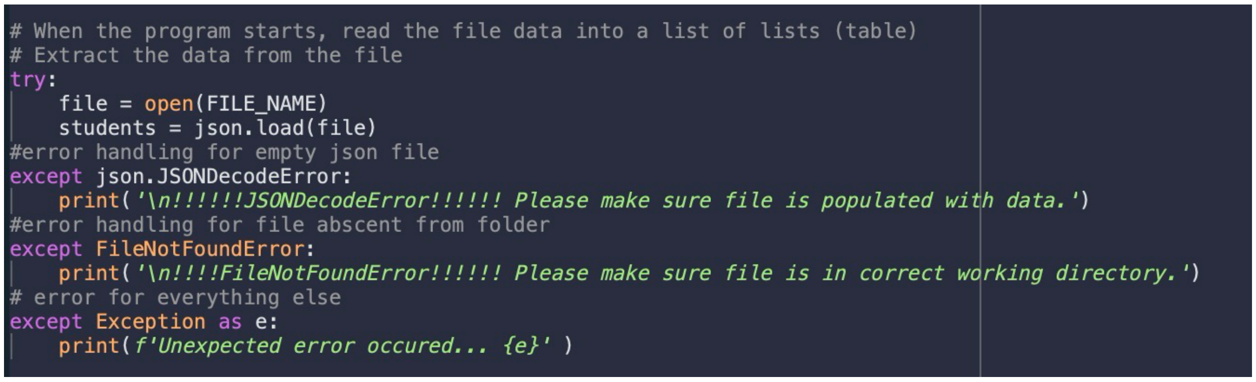
2) Defining an error handling function



```python
#created a function to handle errors on names containing numbers
def isletters(x):
    while x.isalpha() == False:
        x = input('The requested string cannot contain numbers, please input only characters: ')
    return(x)
```

I defined a function to ensure each name that was entered was strictly character-based. If any numbers are included in the name, this function will output a message specifying characters only until the user gets it right. This function was called for the student_first_name and student_last_name variables but not the course_name variable as that can contain numbers.

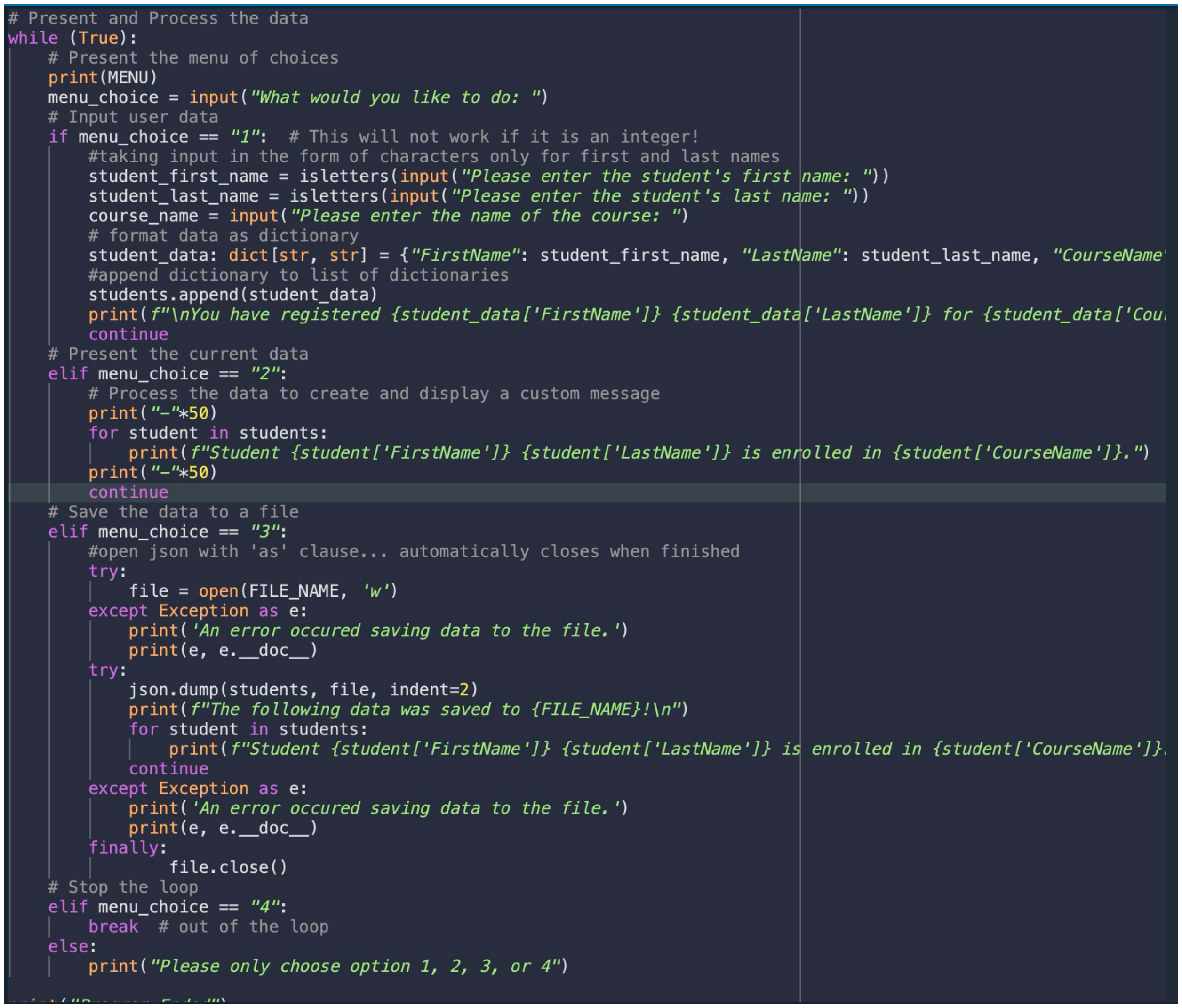
3) Open JSON file initially



```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)
    students = json.load(file)
#error handling for empty json file
except json.JSONDecodeError:
    print('\n!!!!!!JSONDecodeError!!!!!! Please make sure file is populated with data.')
#error handling for file abscent from folder
except FileNotFoundError:
    print('\n!!!!FileNotFoundError!!!!!! Please make sure file is in correct working directory.')
# error for everything else
except Exception as e:
    print(f'Unexpected error occured... {e}' )
```

Opening the JSON file using json.load. This requires the ‘import json’ call. Error handling is included here to ensure the program continues to run even if 1) there is no file named ‘Enrollments.json’ in the directory, or ‘Enrollments.json’ is empty. Neither of these errors prevents the program from taking data and writing it to a JSON file.

4) Present menu options and process input



```
# Present and Process the data
while (True):
    # Present the menu of choices
    print(MENU)
    menu_choice = input("What would you like to do: ")
    # Input user data
    if menu_choice == "1":  # This will not work if it is an integer!
        #taking input in the form of characters only for first and last names
        student_first_name = isletters(input("Please enter the student's first name: "))
        student_last_name = isletters(input("Please enter the student's last name: "))
        course_name = input("Please enter the name of the course: ")
        # format data as dictionary
        student_data: dict[str, str] = {"FirstName": student_first_name, "LastName": student_last_name, "CourseName'
        #append dictionary to list of dictionaries
        students.append(student_data)
        print(f"\nYou have registered {student_data['FirstName']} {student_data['LastName']} for {student_data['Cou
        continue
    # 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 {student['FirstName']} {student['LastName']} is enrolled in {student['CourseName']}.")
        print("-"*50)
        continue
    # Save the data to a file
    elif menu_choice == "3":
        #open json with 'as' clause... automatically closes when finished
        try:
            file = open(FILE_NAME, 'w')
        except Exception as e:
            print('An error occured saving data to the file.')
            print(e, e.__doc__)
        try:
            json.dump(students, file, indent=2)
            print(f"The following data was saved to {FILE_NAME}!\n")
            for student in students:
                print(f"Student {student['FirstName']} {student['LastName']} is enrolled in {student['CourseName']}.
            continue
        except Exception as e:
            print('An error occured saving data to the file.')
            print(e, e.__doc__)
        finally:
                file.close()
    # Stop the loop
    elif menu_choice == "4":
        break  # out of the loop
    else:
        print("Please only choose option 1, 2, 3, or 4")
```

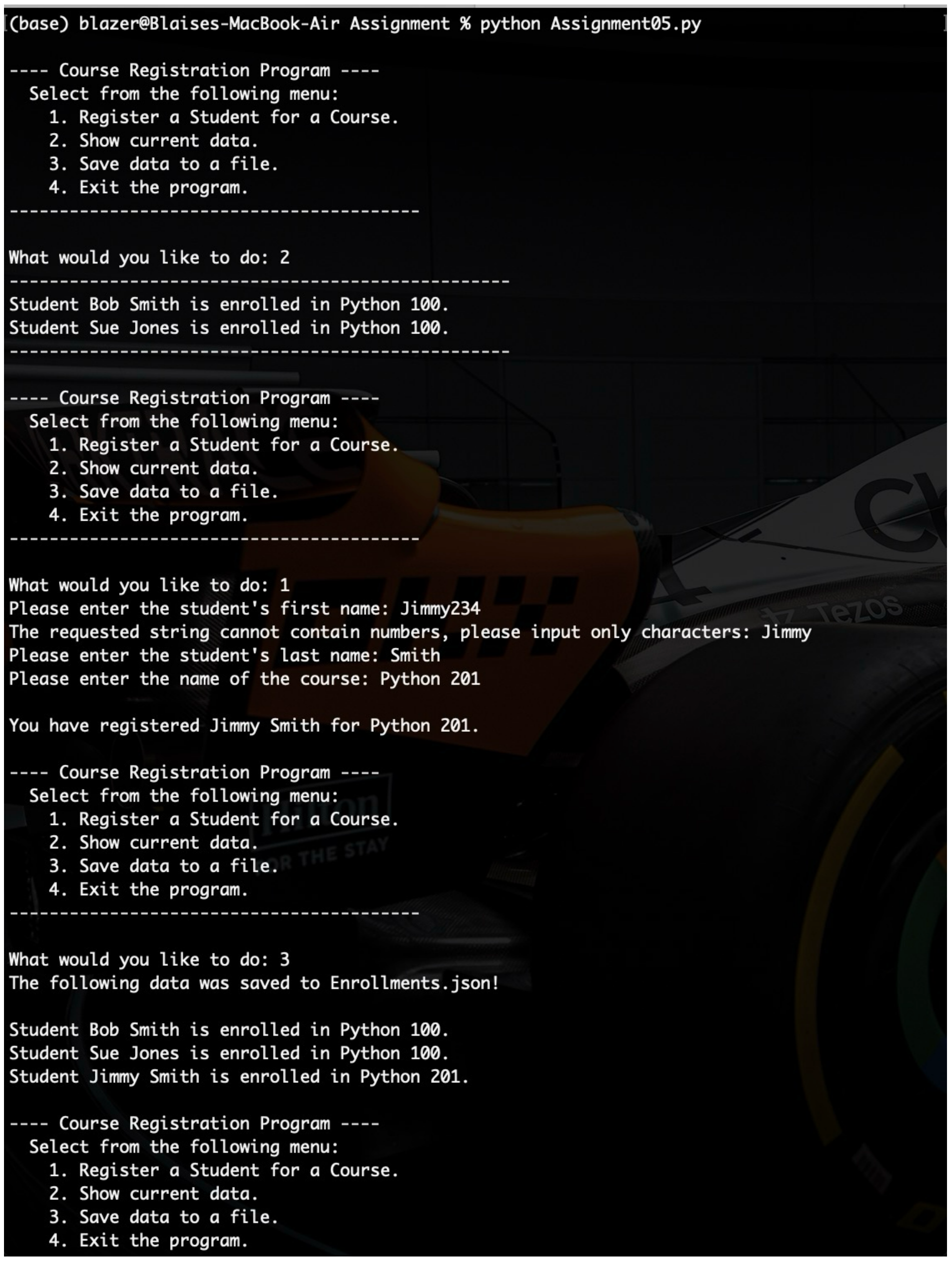
Similar to last time, this while loop continuously runs until the user inputs a ‘4’ in which the *elif* statements breaks the loop. The significant changes in this week’s assignment compared to are:

a. ‘student_data’ now takes the type of a data dictionary in the form of:

{"FirstName": student_first_name, "LastName": student_last_name, "CourseName": course_name}

b. Error handling exceptions are included to make sure the program continues to run after the user has done something to draw an error.
c. The data is stored in a JSON file instead of a csv.

Running the Code



```
(base) blazer@Blaises-MacBook-Air Assignment % python Assignment05.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.
-----------------------------------------

What would you like to do: 2
--------------------------------------------------
Student Bob Smith is enrolled in Python 100.
Student Sue Jones 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: 1
Please enter the student's first name: Jimmy234
The requested string cannot contain numbers, please input only characters: Jimmy
Please enter the student's last name: Smith
Please enter the name of the course: Python 201

You have registered Jimmy Smith for Python 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.
-----------------------------------------

What would you like to do: 3
The following data was saved to Enrollments.json!

Student Bob Smith is enrolled in Python 100.
Student Sue Jones is enrolled in Python 100.
Student Jimmy Smith is enrolled in Python 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.
```

This terminal snippet shows the code taking new data from the user, displaying current data, and successfully writing the data to my local copy of Enrollments.json.

Summary

This assignment really expanded on the previous assignment in its ability to format data. The use of the json file is really no different than that of a csv but requires less maintenance to achieve a high level of readability than that of a csv. The error handling is easily implemented using a function to prevent repeated code. Its also very effective in the event the user wants to create a new json file or does not have any previous data.