Diego Mejia

November 12, 2024

Course: Introduction to Programming with Python

Assignment05 – Advanced Collections and Error Handling

dmejia27/IntroToProg-Python: This Rep. will be used for reviewing homework files

Using dictionaries and Error Handling in Programs

Introduction

The task for this assignment was to Create a Python program that demonstrates using constants, variables, and print statements to display a message about a student's registration for a Python course. This program is very similar to Assignment04, but It adds the use of data processing using dictionaries and exception handling.

We were introduced to structured error handling, which is, Exception Handling in Python deals with the techniques to deal with unwanted and unexpected events occurring during the program execution.

We also learned about GitHub, which is, a cloud-based platform for software development and collaboration. It allows developers to store, manage, and share code using Git for version control. GitHub provides tools for tracking changes, collaborating with other developers, and hosting open-source projects.

Methodology

We begin with a Script Header as we have done in the past with a constant MENU: str that informs the user about the different options they have when running the program. The following is the MENU constant str() used.

---- 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

The file name used in this assignment is the same as Assignment 03, Enrollments.csv. The variables used in this program are as follows.

Variables:

- **student_first_name: str is set to empty string.**
- **student_last_name: str is set to empty string.**
- **course_name: str is set to empty string.**
- **csv_data: str is set to empty string.**
- **file_obj is set to None.**
- **menu_choice: str is set to empty string.**
- **student_data: dict is set to an empty dict**
- **students: list : list is set to and empty list**

The input/output(s) are as follows:

- On menu choice 1, the program prompts the user to enter the student's first name and last name, followed by the course name, using the input() function and stores the inputs in the respective variables.
- On menu choice 2, the presents a comma-separated string by formatting the collected data using the print() function.
- Data collected for menu choice 1 is added to a two-dimensional list table (list of lists).
- All data is the list is displayed when menu choice 2 is used.
- When the program starts, the contents of the "Enrollments.csv" are automatically read into a two-dimensional list table (a list of list rows). (**Tip:** Make sure to put some starting data into the file or you will get an error!)
- On menu choice 3, the program opens a file named "Enrollments.csv" in write mode using the open() function. It writes the content of the csv_data variable to the file using the write() function, then file is closed using the close() method. Then displays what was stored in the file.

- On menu choice 4, the program ends.

Error Handling

- The program provides structured error handling when the file is read into the list of dictionary rows.

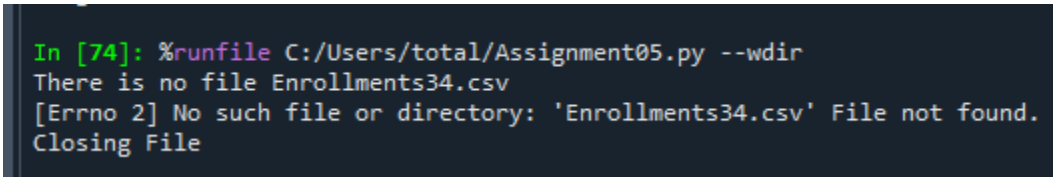

Figure 1: Inserted a file name that did not exist in the current folder "Enrollments34.csv"

- The program provides structured error handling when the user enters a first name.

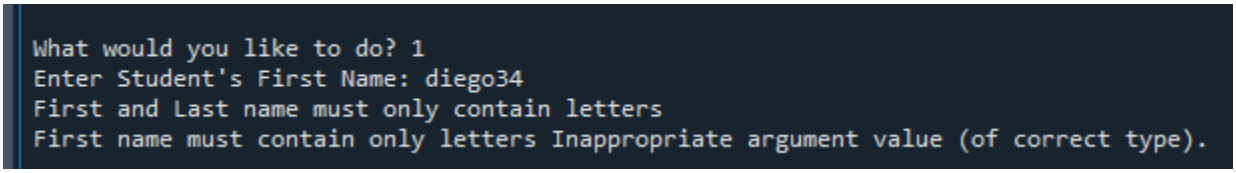

Figure 2:Error when first name contains characters other than letters

- The program provides structured error handling when the user enters a last name.

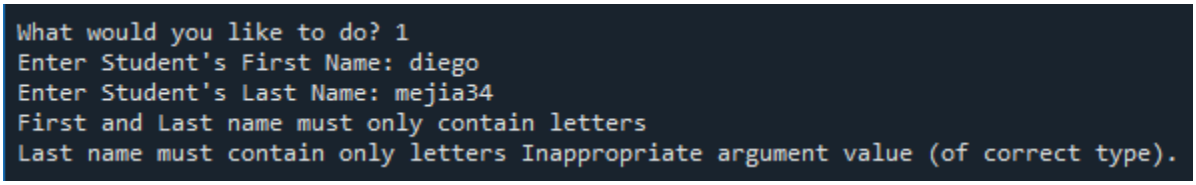

Figure 3: Error when last name contains characters other than letters

- The program provides structured error handling when the dictionary rows are written to the file.

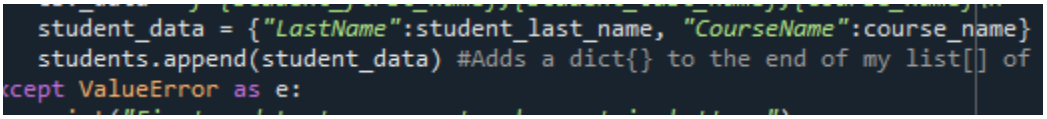

Figure 4: Manipulation of the code allows for the "KeyError," no "FirstName" inside the student_data variable

```
What would you like to do? 3
Closing file
Traceback (most recent call last):

  File c:\users\total\assignment05.py:98
    file_obj.write(f'{student["FirstName"]},{student["LastName"]},
{student["CourseName"]}\n')

KeyError: 'FirstName'


During handling of the above exception, another exception occurred:

Traceback (most recent call last):

  File ~\AppData\Local\spyder-6\envs\spyder-runtime\Lib\site-
packages\spyder_kernels\customize\utils.py:209 in exec_encapsulate_locals
    exec_fun(compile(code_ast, filename, "exec"), globals)

  File c:\users\total\assignment05.py:113
    print(f'{student["FirstName"]} {student["LastName"]} has registered for
{student["CourseName"]} in this last user session!')

KeyError: 'FirstName'
```

Figure 5: Error showing "KeyError"

Testing the Program

The grading criteria for the assignment are as follows:

- The program takes the user's input for a student's first, last name, and course name.

```
In [55]: %runfile C:/Users/total/
Assignment04.py --wdir

--- 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
Enter Student's First Name: diego
Enter Student's Last Name: mejia
Enter Course Name: python 100
```

Figure 6: The program displays the user's input for a student's first, last name, and course name

- The program saves the user's input for a student's first, last name, and course name to a coma-separated string file. (Check this in a simple text editor like notepad.)

```
What would you like to do? 2
Current CSV Data: Last student entered in session
diego,mejia,python 100

Current List Table: Full Table of all students registered
[{'FirstName': 'diego', 'LastName': 'mejia', 'CourseName': 'python 100'}]
```

Figure 7: the IDE shows the input data and presents that the student was successfully registered

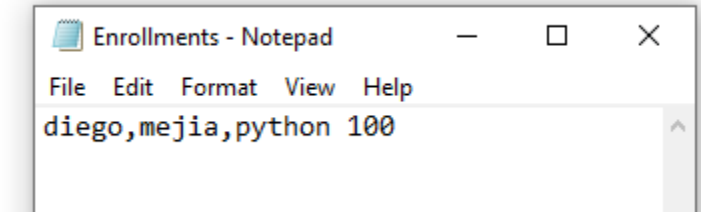

Figure 8: a simple check in a Notepad shows the data was successfully saved in the wanted format

- The program allows users to enter multiple registrations (first name, last name, course name).

```
What would you like to do? 2
Current CSV Data: Last student entered in session
amy,jaime,math 200

Current List Table: Full Table of all students registered
[{'FirstName': 'tony', 'LastName': 'shives', 'CourseName': 'python 100'},
{'FirstName': 'amy', 'LastName': 'jaime', 'CourseName': 'math 200'}]
```

Figure 9: Multiple registrations are shown, and students were successfully saved

- The program allows users to save multiple registrations to a file (first name, last name, course name).

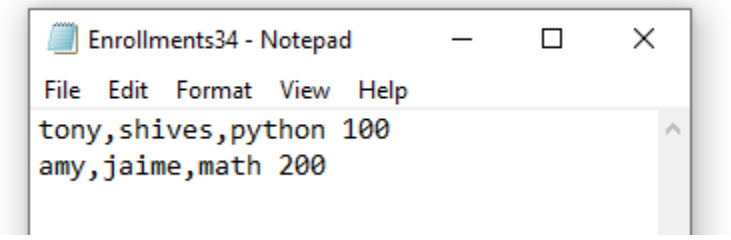

Figure 10: Multiple entries saved to a file in CSV format

- The program runs correctly in both **the IDE and** from the **console or terminal**.

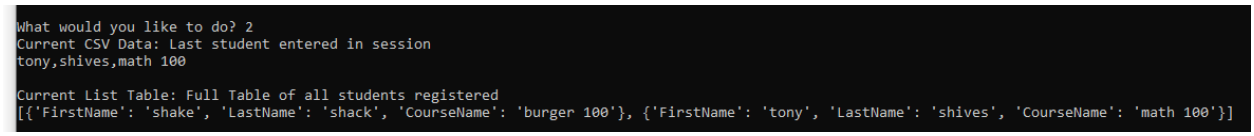

Figure 11: the program successfully runs in the console or terminal

To exit the program we input menu item 4 as follows

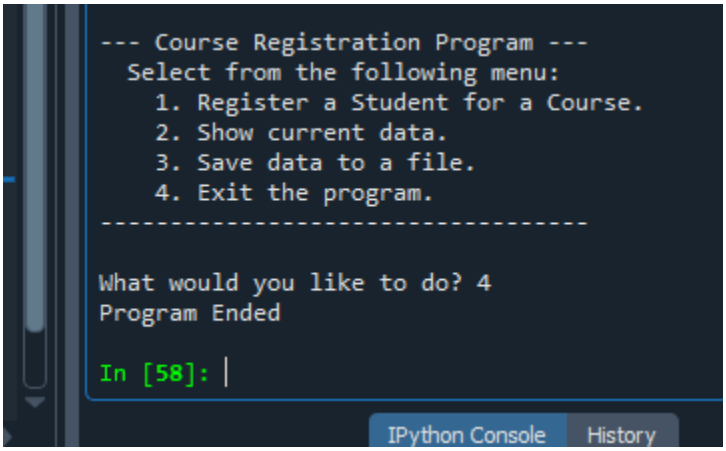

Figure 12: Program ends

Summary

We were able to successfully create a program that registers students to a course and use a while loop with proper conditional statements. We also used a for loop outside the while loop to read the file beforehand and append the new data into it. We also introduced Error Handling using the try: and except command. By using these exceptions, we were able to add functionality that gave an error when the user input anything other than letters to the names. We also coded a FileNotFound in the for loop to make sure the file existed in the folder.