Angel Reddy
Luis Conejo Alpizar and Julian Peterson
Foundations in Programming (Python)
18 March 2024

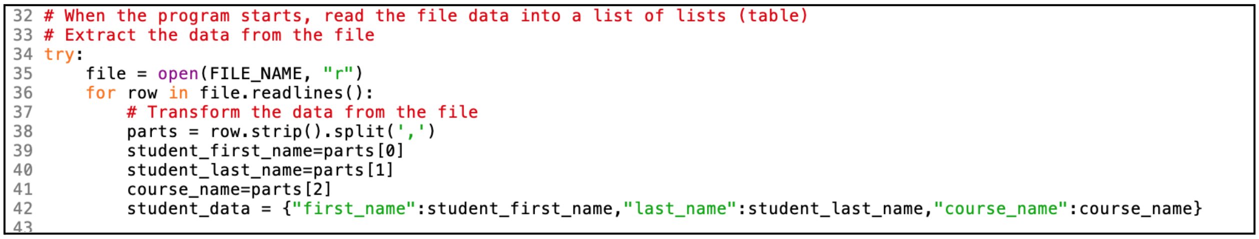
# Assignment 05

## Introduction

In Assignment 05, the goal was to modify the script from assignment 04 to include data processing with dictionaries and exception handling. The purpose of the script remained the same; the program demonstrated the use of constants, variables, and print statements to display a message about registering a student for a course.

## Drafting the Script

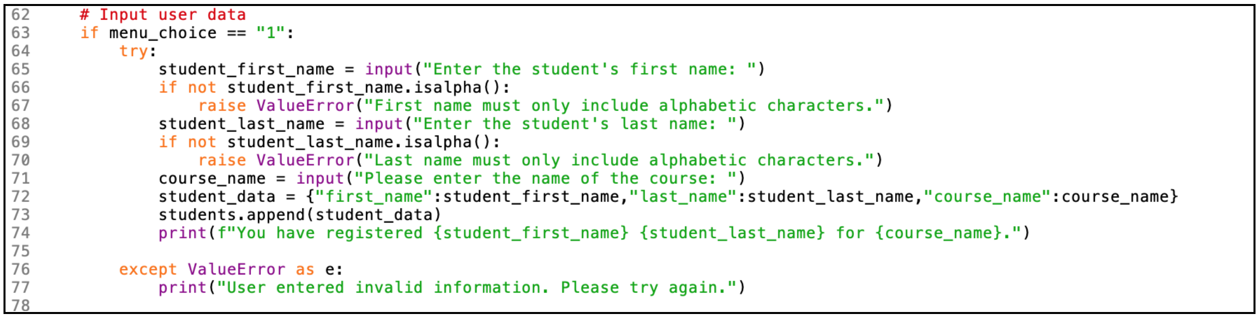
To begin, I used my script from assignment 04 as my starting point. To this script, I updated my use of lists to dictionaries.To do this, I adjusted the variable student_data. I changed the variable to dict[str, str] where a dictionary held the student data inputted by the user. In the file itself, I updated student_data to include new keys, such as “first_name”:student_first_name. Previously, I used a number as an index that indicated to the program the part of the variable I was referencing. I went through the entire script and replaced the previous indexes with the new keys created, such as “last_name”:student_last_name. This allowed me to alter my previous use of lists to dictionaries.



```
# 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")
    for row in file.readlines():
        # Transform the data from the file
        parts = row.strip().split(',')
        student_first_name=parts[0]
        student_last_name=parts[1]
        course_name=parts[2]
        student_data = {"first_name":student_first_name,"last_name":student_last_name,"course_name":course_name}
```

**Figure 1. Updating to dictionaries.** In the variable student_data, I added code that replaced the indexes. This new key allowed the data to be read into a dictionary.

Additionally, I added error handling to the script. The error handling allowed me to alert the user when an error occurred. I added error handling to the opening of the file. For example, if the file does not exist, the program will raise an error message to the user and create the necessary file. I also added error handling to any user input. Another example is that I added an error message for when the user inputs something other than an alphabetic character for the student's first and last name. The error handling helps the program follow a more stringent set of rules for the input of data. Previously, the user could enter a random string of characters and it would be accepted. Error handling prevents this kind of messy input.



```
    # Input user data
    if menu_choice == "1":
        try:
            student_first_name = input("Enter the student's first name: ")
            if not student_first_name.isalpha():
                raise ValueError("First name must only include alphabetic characters.")
            student_last_name = input("Enter the student's last name: ")
            if not student_last_name.isalpha():
                raise ValueError("Last name must only include alphabetic characters.")
            course_name = input("Please enter the name of the course: ")
            student_data = {"first_name":student_first_name,"last_name":student_last_name,"course_name":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("User entered invalid information. Please try again.")
```

**Figure 2. Error handling.** The above image displays an example of error handling I added to the script. I used ValueError to display a message to the user that they inputted the incorrect values into the name field. The line "if not student_first_name.isalpha():" helps to read the user's input and determine whether or not to raise the error. The command isalpha() examines the entry to see if it includes non-alphabetic characters.

## Running the Script and GitHub

I was able to run the script successfully in the Terminal, IDLE, and PyCharm. I struggled with PyCharm for this assignment as I was still using IDLE for a large majority of my work. However, it is helpful to use both IDEs as it gives me exposure to how each one works. Also, here is a link to my GitHub repository: https://github.com/chreddy1020/IntroToProg-Python.

## Reflection and Summary

I enjoyed working on this assignment. It felt like a comfortable next step from the previous assignment. However, it simply took me some time to complete. I wish I would have completed this assignment in time to receive feedback from the instructors. However, the companion video for this assignment helped a great deal with sorting out any errors in my script. I was able to troubleshoot and follow examples of how someone else worked on this assignment.