Name: Ian Basques-Jellison
Date: August 13, 2025
Course: IT FDN 110 A Su 25: Foundations of Programming: Python
Assignment: Assignment05
GitHubURL: https://github.com/ianjelly/IntroToProg-Python-Mod05

Using Dictionaries and Error Handling

Contents

Introduction

This document describes the steps performed in the creation of a Python program, "Assignment05.py", that is similar to "Assignment04.py", except that it additionally uses dictionaries and exception handling. Most of the program's functionality and acceptance criteria is already accomplished by the starter file, "Assignment05-Starter.py". This document describes how the starter file is expanded upon to accomplish the remaining acceptance criteria.

Acceptance Criteria

Acceptance criteria for the program are defined in the “Mod05-Assignment.docx” file for the Introduction to Python Course IT FDN 110 A. The program must include the following components:

- File Name: Assignment05.py
- Script Header (Title, Program Description, and Change Log)
- The constants: **MENU** and **FILE_NAME**
- The variables: **student_first_name**, **student_last_name**, **course_name**, **file**, **menu_choice**, **student_data** and **students**
- The input() function
- A print() statement
- A while loop
- File processing functions: open(), load() and close()
- Error handling

When the program starts, the contents of the “Enrollments.json” are automatically read into the **students** two-dimensional list of lists (table).

The program uses a four-option menu, and does the following for each menu choice:

Menu Choice (1)	prompts the user to enter the student’s first and last name, and the course name
Menu Choice (2)	prints a comma-separated strings composed of the first name, last name, course name for all the data in the **students** variable, including any data initially in the file
Menu Choice (3)	writes the contents of the **students** variable to the “Enrollments.json” and displays what was written to the file
Menu Choice (4)	ends the program

Much of the program’s functionality and acceptance criteria is already accomplished by the starter file, “Assignment05-Starter.py”. The acceptance criteria that are <u>not</u> accomplished by the starter file, are satisfied by the steps described in this document.

Program Construction

The program consists of a script file organized into three sections: a Header, Setup code, and the Main Body. These sections, their subcomponents (define constants, define variables, etc.), and the majority of the functionality of the program are specified by the starter file, “Assignment05-Starter.py”. The starter file is shown in Figure 1.

```
# ------------------------------------------------------------------------------------------ #
# Title: Assignment05
# Desc: This assignment demonstrates using dictionaries, files, and exception handling
# Change Log: (Who, When, What)
#   RRoot,1/1/2030,Created Script
#   <Your Name Here>,<Date>, <Activity>
# ------------------------------------------------------------------------------------------ #

# 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.csv"

# 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: list = []  # one row of student data (TODO: Change this to a Dictionary)
students: list = []  # a table of student data
csv_data: str = ''  # Holds combined CSV data. Note: Remove later since it is NOT needed with the JSON
File
file = None  # Holds a reference to an opened file.
menu_choice: str  # Hold the choice made by the user.


# When the program starts, read the file data into a list of lists (table)
# Extract the data from the file
file = open(FILE_NAME, "r")
for row in file.readlines():
    # Transform the data from the file
    student_data = row.split(',')
    student_data = [student_data[0], student_data[1], student_data[2].strip()]
    # Load it into our collection (list of lists)
    students.append(student_data)
file.close()

# 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!
        student_first_name = input("Enter the student's first name: ")
        student_last_name = input("Enter the student's last name: ")
        course_name = input("Please enter the name of the course: ")
        student_data = [student_first_name,student_last_name,course_name]
        students.append(student_data)
        print(f"You have registered {student_first_name} {student_last_name} for {course_name}.")
        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[0]} {student[1]} is enrolled in {student[2]}")
        print("-"*50)
        continue

    # Save the data to a file
    elif menu_choice == "3":
        file = open(FILE_NAME, "w")
        for student in students:
            csv_data = f"{student[0]},{student[1]},{student[2]}\n"
            file.write(csv_data)
        file.close()
        print("The following data was saved to file!")
        for student in students:
            print(f"Student {student[0]} {student[1]} is enrolled in {student[2]}")
        continue

    # Stop the loop
    elif menu_choice == "4":
        break  # out of the loop
    else:
        print("Please only choose option 1, 2, 3, or 4")

print("Program Ended")
```

Figure 1: Start file Assignment05-Starter.py

Header

The script header in the Python program uses the same format as defined in the acceptance criteria from the assignment document file, except with the Change Log updated with my name and date. Figure 2 shows the script header for “Assignment05.py”.

```
# ------------------------------------------------------------------------------------------ #
# Title: Assignment05
# Desc: This assignment demonstrates using dictionaries, files, and exception handling
# Change Log: (Who, When, What)
#   Ian Basques-Jellison, 8/10/2025, Created Script
#   <Your Name Here>, <Date>, <Activity>
# ------------------------------------------------------------------------------------------ #
```

Figure 2: Header from Assignment05.py

Setup Code

The starter file, “Assignment05-Starter.py”, defines all the necessary constants and variables, except the **FILE_NAME: str** constant must be changed from “Enrollments.csv” to “Enrollments.json”, and the **student_data** variable must be changed from type “list” to “dict”. Figure 3 shows the Setup code from “Assignment05.py”.

```
# 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.csv"  # Change to .json later
# 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 = {}  # one row of student data
students: list = []  # a table of student data
csv_data: str = ''  # Holds combined CSV data. Note: Remove later since it is NOT needed with
the JSON File
file = None  # Holds a reference to an opened file.
menu_choice: str  # Hold the choice made by the user.
```

Figure 3: Setup Code from Assignment05.py

Main Body

The main body of the Python program implements the four menu choices as specified in the acceptance criteria. Much of the functionality for the program is already defined in the starter file, “Assignment05-Starter.py”. The changes that are required to satisfy the rest of the acceptance criteria are:

1. When the program starts, read the contents of the “Enrollments.csv” into the **student_data** variable as a dictionary instead of a list
2. For menu choice 1, store the user inputs for **student_first_name**, **student_last_name**, and **course_name**, in the **student_data** variable as a dictionary instead of a list
3. For menu choice 2, call the data from the dictionary instead of a list
4. For menu choice 3, call the data from the dictionary instead of a list
5. Convert all the file handling to use the “Enrollments.json” JSON file instead of the “Enrollments.csv” CSV file
6. Add exception handling

Reading “Enrollments.csv” into the student_data variable as a dictionary

For the first part, we simply adjust the assignment of the **student_data** variable to use a dictionary instead of a list. The code is shown in Figure 4, with the changes highlighted.

```
# When the program starts, read the file data into a list of lists (table)
# Extract the data from the file
file = open(FILE_NAME, "r")
for row in file.readlines():
    # Transform the data from the file
    student_data = row.strip().split(',')
    if student_data != ['']:
        student_data = {"FirstName": student_data[0],
                        "LastName": student_data[1],
                        "CourseName": student_data[2]}
        # Load it into our collection (list of lists)
        students.append(student_data)
        continue
    else:
        continue
file.close()
```

Figure 4: Reading “Enrollments.csv” into the student_data variable using a dictionary instead of a list

Menu Choice 1: Input user data

The next step, is to edit the code for menu choice 1 to store the user provided data in the **student_data** variable as a dictionary instead of a list. Figure 5 shows menu choice 1 updated with this one line of code, highlighted in the figure.

```
    # Input user data
    if menu_choice == "1":  # This will not work if it is an integer!
        student_first_name = input("Enter the student's first name: ")
        student_last_name = input("Enter the student's last name: ")
        course_name = input("Please enter the name of the 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}.")
        continue
```

Figure 5: Menu Choice 1

Menu Choice 2: Present the current data

Similar to menu choice 1, the code for menu choice 2, is edited to call the data from **student** in **students** as a dictionary instead of a list. Figure 6 shows menu choice 2 updated with the code necessary to perform this.

```
    # 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"]} "\
                    f"{student["LastName"]} is enrolled in "\
                    f"{student["CourseName"]}")
        print("-"*50)
        continue
```

Figure 6: Menu Choice 2

Menu Choice 3: Save the data to a file

For menu choice 3, we use the same code as we added for menu choice 2, except this time we also edit the code in the write() function. Figure 7 shows the final code for menu choice 3.

```
    # Save the data to a file
    elif menu_choice == "3":
        file = open(FILE_NAME, "w")
        for student in students:
            # csv_data = f"{student[0]},{student[1]},{student[2]}\n"  # Remove later

file.write(f'{student["FirstName"]},{student["LastName"]},{student["CourseName"]}\n')
        file.close()
        print("The following data was saved to file!")
        for student in students:
            print(f"Student {student["FirstName"]} "\
                    f"{student["LastName"]} is enrolled in "\
                    f"{student["CourseName"]}")
        continue
```

Figure 7: Menu Choice 3

Using JSON file instead of CSV file

At this stage, we convert all the file handling to use the “Enrollments.json” instead of the “Enrollments.csv”. This is done by first importing the json module, then replacing the file readline() and write() functions with the appropriate load() and dump() functions for json files. Figure 7 shows the script header for “Assignment05.py” with the line of code used to import the json module.

```
# ------------------------------------------------------------------------------------------ #
# Title: Assignment05
# Desc: This assignment demonstrates using dictionaries, files, and exception handling
# Change Log: (Who, When, What)
#   Ian Basques-Jellison, 8/10/2025, Created Script
#   <Your Name Here>, <Date>, <Activity>
# ------------------------------------------------------------------------------------------ #
import json  # import code from Python's JSON module
```

Figure 8: Header from Assignment05.py with json module imported

Extracting the data from the JSON file simply becomes a single line using the load() function. Figure 9 shows the script with the line of code used to extract the data from the JSON file.

```
# When the program starts, read the file data into a list of lists (table)
# Extract the data from the file
file = open(FILE_NAME, "r")
students = json.load(file)
file.close()
```

Figure 9: Extracting data from the JSON file

Likewise, for menu choice 3, writing the data to the JSON file simply becomes a single line using the dump() function. Figure 10 shows the script with the line of code used to extract the data from the JSON file.

```
    # Save the data to a file
    elif menu_choice == "3":
        file = open(FILE_NAME, "w")
        json.dump(students, file, indent=2)
        file.close()
        print("The following data was saved to file!")
        for student in students:
            print(f"Student {student["FirstName"]} "\
                    f"{student["LastName"]} is enrolled in "\
                    f"{student["CourseName"]}")
        continue
```

Figure 10: Writing data to the JSON file (menu choice 3)

Exception Handling

The final step is to implement the exception handling. Per the acceptance criteria, exception handling is added in three places: when the data is first extracted from the file, when the user enters a first and last name, and when the data is written to the file.

Figure 11 shows the exception handling for when the file is read into the list of dictionary rows.

```
# 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("Enrollments.json file not found.\n")
    print("-- Technical Error Message --")
    print(e, e.__doc__, type(e), sep='\n')
except Exception as e:
    print("There was a non-specific error!\n")
    print("-- Technical Error Message --")
    print(e, e.__doc__, type(e), sep='\n')
finally:
    if file.closed == False:
        file.close()
```

Figure 11: Exception handling for reading the file

Figure 12 shows the exception handling for when the user enters a first and last name. This part of the code checks if the names contain numbers.

```
    # 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("The first name must not contain numbers.")
            student_last_name = input("Enter the student's last name: ")
            if not student_last_name.isalpha():
                raise ValueError("The last name must not contain numbers.")
            course_name = input("Please enter the name of the course: ")
            student_data = {"FirstName": student_first_name,
                            "LastName": student_last_name,
                            "CourseName": course_name}
            students.append(student_data)
            print(f"You have registered "\
                  f"{student_first_name} "\
                  f"{student_last_name} "\
                  f"for {course_name}.")
        except ValueError as e:
            print(e)  # Prints the custom message
            print("-- Technical Error Message --")
            print(e.__doc__)
            print(e.__str__())
        except Exception as e:
            print("There was a non-specific error!\n")
            print("-- Technical Error Message --")
            print(e, e.__doc__, type(e), sep='\n')
        continue
```

Figure 12: Exception handling for user input of first and last name

Finally, Figure 13 shows the exception handling for when data is written to the file. This part of the code checks if the data is in a valid JSON format.

```
    # Save the data to a file
    elif menu_choice == "3":
        try:
            file = open(FILE_NAME, "w")
            json.dump(students, file, indent=2)
            file.close()
            print("The following data was saved to file!")
            for student in students:
                print(f"Student {student["FirstName"]} "\
                        f"{student["LastName"]} is enrolled in "\
                        f"{student["CourseName"]}")
        except TypeError as e:
            print("Please check the data is a valid JSON format\n")
            print("-- Technical Error Message --")
            print(e, e.__doc__, type(e), sep='\n')
        except Exception as e:
            print("-- Technical Error Message --")
            print("Built-In Python error info: ")
            print(e, e.__doc__, type(e), sep='\n')
        finally:
            if file.closed == False:
                file.close()
        continue
```

Figure 13: Exception handling for writing the file

Testing

Once the script is written, the program is tested in PyCharm and from the Windows console.

Summary

This document describes the steps performed in the creation of a Python program, “Assignment05.py”, that is similar to “Assignment04.py”, except that it additionally uses dictionaries and exception handling. Much of the program’s functionality and acceptance criteria is already accomplished by the starter file, “Assignment05-Starter.py”. The program satisfies the acceptance criteria by expanding upon the starter file to satisfy the remaining acceptance criteria.

The final completed Python script is shown in Figure 14.

```
# ------------------------------------------------------------------------------------------ #
# Title: Assignment05
# Desc: This assignment demonstrates using dictionaries, files, and exception handling
# Change Log: (Who, When, What)
#   Ian Basques-Jellison, 8/10/2025, Created Script
#   <Your Name Here>, <Date>, <Activity>
# ------------------------------------------------------------------------------------------ #
import json  # import code from Python's JSON module

# 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 = {}  # one row of student data
students: list = []  # a table of student data
csv_data: str = ''  # Holds combined CSV data. Note: Remove later since it is NOT needed with the JSON
File
file = None  # Holds a reference to an opened file.
menu_choice: str  # Hold the choice made by the user.


# 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("Enrollments.json file not found.\n")
    print("-- Technical Error Message --")
    print(e, e.__doc__, type(e), sep='\n')
except Exception as e:
    print("There was a non-specific error!\n")
    print("-- Technical Error Message --")
    print(e, e.__doc__, type(e), sep='\n')
finally:
    if file.closed == False:
        file.close()

# 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!
        try:
            student_first_name = input("Enter the student's first name: ")
            if not student_first_name.isalpha():
                raise ValueError("The first name must not contain numbers.")
            student_last_name = input("Enter the student's last name: ")
            if not student_last_name.isalpha():
                raise ValueError("The last name must not contain numbers.")
            course_name = input("Please enter the name of the course: ")
            student_data = {"FirstName": student_first_name,
                            "LastName": student_last_name,
                            "CourseName": course_name}
            students.append(student_data)
            print(f"You have registered "\
                  f"{student_first_name} "\
                  f"{student_last_name} "\
                  f"for {course_name}.")
        except ValueError as e:
            print(e)  # Prints the custom message
            print("-- Technical Error Message --")
            print(e.__doc__)
            print(e.__str__())
        except Exception as e:
            print("There was a non-specific error!\n")
            print("-- Technical Error Message --")
            print(e, e.__doc__, type(e), sep='\n')
        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"]} "\
                    f"{student["LastName"]} is enrolled in "\
                    f"{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, indent=2)
            file.close()
            print("The following data was saved to file!")
            for student in students:
                print(f"Student {student["FirstName"]} "\
                        f"{student["LastName"]} is enrolled in "\
                        f"{student["CourseName"]}")
        except TypeError as e:
            print("Please check the data is a valid JSON format\n")
            print("-- Technical Error Message --")
            print(e, e.__doc__, type(e), sep='\n')
        except Exception as e:
            print("-- Technical Error Message --")
            print("Built-In Python error info: ")
            print(e, e.__doc__, type(e), sep='\n')
        finally:
            if file.closed == False:
                file.close()
        continue

    # Stop the loop
    elif menu_choice == "4":
        break  # out of the loop
    else:
        print("Please only choose option 1, 2, 3, or 4")

print("Program Ended")
```

Figure 14: Complete Python Script for Assignment05.py