Matthew Lord
2023.03.12
IT FDN 110 A Wi 23: Foundations of Programming: Python
Assignment 07
https://github.com/mlord000
https://mlord000.github.io/IntroToProg-Python-Mod07/

Demonstrating Exception Handling and "Pickling" Data

Introduction

In this assignment we will demonstrate the use of error handling with the try and except code blocks of Python. We will also demonstrate using the pickle module to create and store data in a binary file. In preparation of creating the script we reviewed the assignment material as well as several sources of information online. Included in the header of this assignment script are links to several online resources used for reference (Code Block 1).

```
#         * regarding exception handling *
#         https://docs.python.org/3/tutorial/errors.html
#             Examples directly from the python documentation, there are
#             very detailed examples and descriptions to be found here.
#         https://www.programiz.com/python-programming/exception-handling
#             The above webpage was useful because it demonstrated examples
#             which stepped through some of the levels of exception handling.
#             For example, incorporating an "else:" code block or using the
#             "finally:" code block after a try:/exception:
#         https://www.geeksforgeeks.org/python-exception-handling/
#             The above webpage was helpful because it listed and described
#             different types of exceptions that can be used within the
#             exception block.
#
#         * regarding pickling *
#         https://www.geeksforgeeks.org/understanding-python-pickling-example/
#             I feel this is good at explaining the subject because it gives
#             a straightforward example of storing and retrieving the data.
#         https://www.digitalocean.com/community/tutorials/python-pickle-example
#             I appreciated this example because it demonstrates collecting
#             user-entered data and picking this into a file (writing of
#             pickle.dump()) as well as opening (pickle.load()) and
#             displaying the data back to the user.
#         https://wiki.python.org/moin/UsingPickle
#             From the python wiki, the most simple pickle example I have
#             come across.
```

Code Block 1: references to several online resources used to help with this assignment

Exception Handling Example

The setup for this script began with focusing on a simple example of exception handling. The code was written in the main body of the file to get started. Versions of the script were not created at the time so there are no longer references to what that code looked like. (This can be a reminder to create versions as we are writing code in the future so we can reference the changes during and after the creation process.) While testing the code by running it in PyCharm, I divided the code using separation of concerns to keep it organized and begin identifying sections that can be broken down into functions.

An example of this is that after writing and refining the printed user instructions for the try/except example, rather than include this directly in the main body of code, it was separated into a function. Rather than writing the following in the main body (Code Block 2) it was incorporated into a function (Code Block 3).

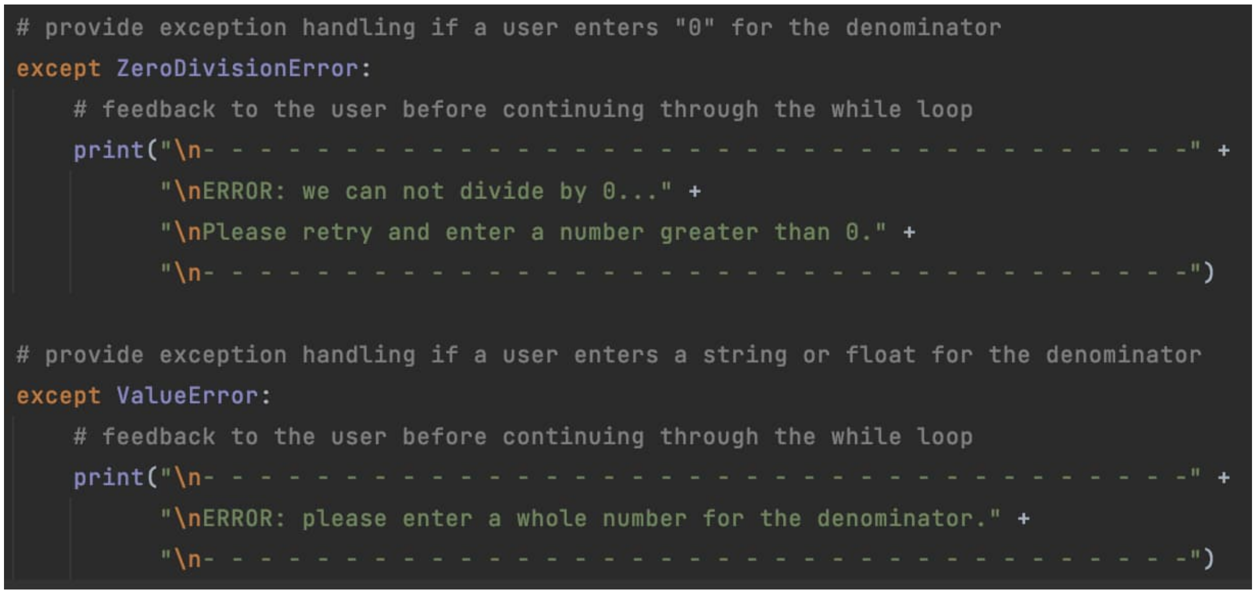
```
# inform the user of what this portion of the program will do
print("\nWe will collect a number from you and apply it to the\n"
    + "denominator of a fraction, then return the quotient.")
print("\nType 'exit' to move on to the pickling portion of the assignment...")
print("\n*** (To force an exception, enter '0', a float number, or a string other than
'exit'.) ***")
```

Code Block 2: User instructions from the main body, later placed into a function

```
# provide instructions to user
TryExceptInstructions()
```

Code Block 3: Applying a function in the while loop instead of a large amount of text instructions

I used a basic example of asking the user for the denominator of a fraction and returning the quotient of this and a randomly created nominator value. This allowed me to identify to common errors that I could catch with exception handling. These two errors are the ZeroDivisionError and ValueError (Figure 1).



```
# provide exception handling if a user enters "0" for the denominator
except ZeroDivisionError:
    # feedback to the user before continuing through the while loop
    print("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -" +
          "\nERROR: we can not divide by 0..." +
          "\nPlease retry and enter a number greater than 0." +
          "\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -")

# provide exception handling if a user enters a string or float for the denominator
except ValueError:
    # feedback to the user before continuing through the while loop
    print("\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -" +
          "\nERROR: please enter a whole number for the denominator." +
          "\n- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -")
```

Figure 1: two identifiable errors when collecting user input to return a quotient

Saving and Loading Data Using Pickle

For the example using the pickle module I tried to keep it as simple as possible by creating a list object that a user can add words to and opt for saving the list to a binary file. I began with the code found on one of the websites referenced during my online research (https://wiki.python.org/moin/UsingPickle). The other major reference I used is from the menu selection code from our Assignment 05. This presented code easily adaptable for this use case. Below is the menu selection code of the while loop from this section of the assignment (Code Block 4):

```
    # Add a new word/string to the list
    if (menu_choice.strip() == '1'):
        # prompt user for a task to add to the list/table
        user_word = input("Enter a word: ")
        # add the word to a list
        session_list.append(user_word)
        # inform the user what action has taken place
        print("We have added the word '" + user_word + "' to the current session list.")
        continue  # continue to the while/menu loop

    elif (menu_choice.strip() == '2'):
        # print the current list of words
        print("The current list of words in this session is:")
        print(session_list)

    elif (menu_choice.strip() == '3'):
        # verify the user wants to overwrite
        user_input = input("Are you sure you want to overwrite the saved data? (Y or N): ")
        # check value of user input
        if user_input.lower() == "y":
            # assign the current list to a list to be saved
            pickle_list = session_list
            # save the list as a binary file using pickle
            pickle.dump(pickle_list, open(file_name, "wb"))
            # inform the user of what action has taken place
            print("You have overwritten the saved data with the current list of words.")
        elif user_input.lower() == "n":
            # inform the user
            print("You have chosen NOT to overwrite the saved data.")
        else:
            print("\nPlease use only 'Y' or 'N'")
            print("Select a menu option:")
        continue  # continue to the while/menu loop

    elif (menu_choice.strip() == '4'):
        if os.path.isfile(file_name):
            # open and load the saved list using pickle
            pickle_list = pickle.load(open(file_name, "rb"))
        else:
            # inform the user that the file doesn't exist
            print("There is no data file saved... we will create one.")
            # create the file
            pickle.dump(pickle_list, open(file_name, "wb"))
            # inform the user that the file has been created
            print("The file has been created. It is currently empty.")
        # display the list to the user
        print("The saved list of data is: ")
        print(pickle_list)

    elif (menu_choice.strip() == '5'):
        # verify the user would like to exit the program
        user_input = input("Would you like to exit the program? (Y or N): ")
        # check value of user input
        if user_input.lower() == "y":
            # inform user they have exited the program
            print("You have exited the program.")
            break  # and Exit the program
        elif user_input.lower() == "n":
            # inform the user they have chosen NOT to exit the program
            print("You have chosen NOT to exit the program.")
            continue  # continue to the while/menu loop
        else:
            print("\nPlease use only 'Y' or 'N'")
            print("Select a menu option:")
            continue  # continue to the while/menu loop
```

Code Block 4: menu selection code for pickle example from assignment 07 showing similarities to the menu code in Assignment 5.

All of this seemed to go together in a fairly straightforward manner. The only issue I ran into was an error that occurred if the binary file was not already created. I wanted to be able to load a file if it had already been created or create a file if it had not been created. If I needed to create a file, I did not want to write anything to it unless the user made that selection. I researched online and found a module that can be imported and used to test if a file exists (os.path) and used this in the code block for menu selection “4”.

It seems to me while working on this that there is still room to refine this code into additional functions, or perhaps even a set of class objects. Additionally, I was curious about whether one could present the binary data as characters to the user in via the command line. It seems like this could be a nice option to have, though I don’t know how practical it is.

Conclusion

In this assignment I created a script which demonstrates exception handling and pickling data using the Python programming language. To do so I relied on information provided by the course, various resources online, and pulled already created code from previous assignments to develop these examples.