DUW
May 31, 2022
Foundations of Programming, Python
Assignment 07

Error Handling & Pickling

Introduction

In this assignment I will explain step-by-step how I created a script that saves to a binary file and how I defined a few examples of error handling. Assignment 07 is meant to prove understanding of the Python pickling concept and error handling situations. Key sources I used to learn more about pickling and error handling:

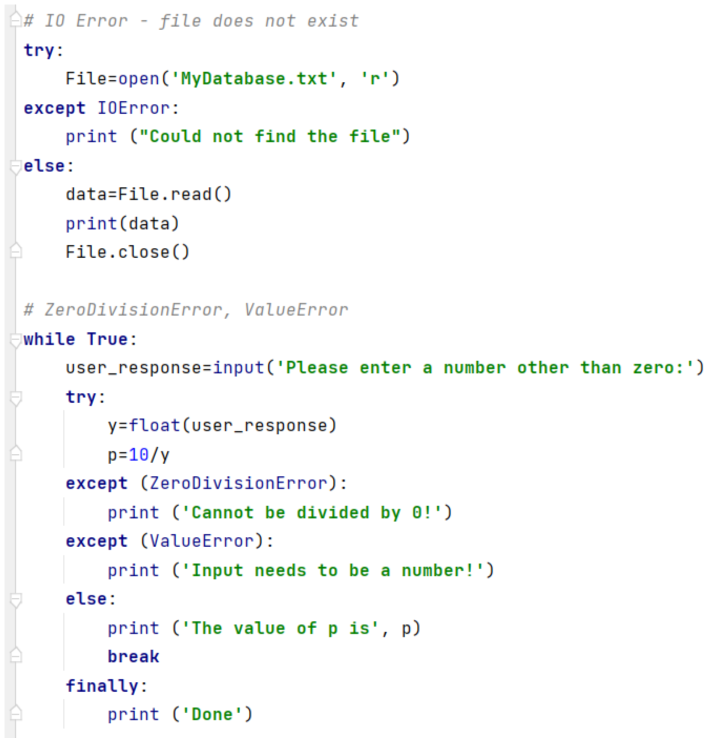
- Pickle – Python object serialization (Python docs)
- Errors and Exceptions (Python docs)
- edX class Introduction to Programming Using Python

Creating the Program

The steps I followed to develop and run the program:

1. **Created a new script file with the .py extension**
2. **Updated the script header including inline comments in the script file**
    a. Updated the change log (who, when, what).
3. **Added some error handling examples:**
    a. **IO Error: file does not exist using the syntax try-except-else:**
        i. I added code for the program to open and read a file that does not exist.
        ii. I added the exception for the input/output error and included that the program should print that it could not find the file if it doesn't find it.
        iii. I added an 'else' statement that the program should read the file if it finds it.
    b. **Zero division error and value error showcasing how to use multiple except statements**:
        i. Added a while loop so that the program continues running until the user adds the right input
        ii. Added an input statement asking the user to enter a number other than zero.
        iii. Converted the user input into a float
        iv. Added a new variable p that is calculated based on the user's input.
        v. Introduced the ***Zero division error*** and told the program to print an error message if the user inputs 0.
        vi. Introduced the Value Error if the user provides something else than a number like a text response.
        vii. Added an 'else' statement to print the value of the number calculated based on user input.
        viii. Added a 'break' to break the while loop at this point.
        ix. Added a 'finally' statement to print 'Done' when the program ends.
    c. ***Figure 1*** shows my code with the error handling examples.
4. **Demonstrated the pickling concept by saving to a binary file**.
    a. Imported the pickle
    b. Defined my variables

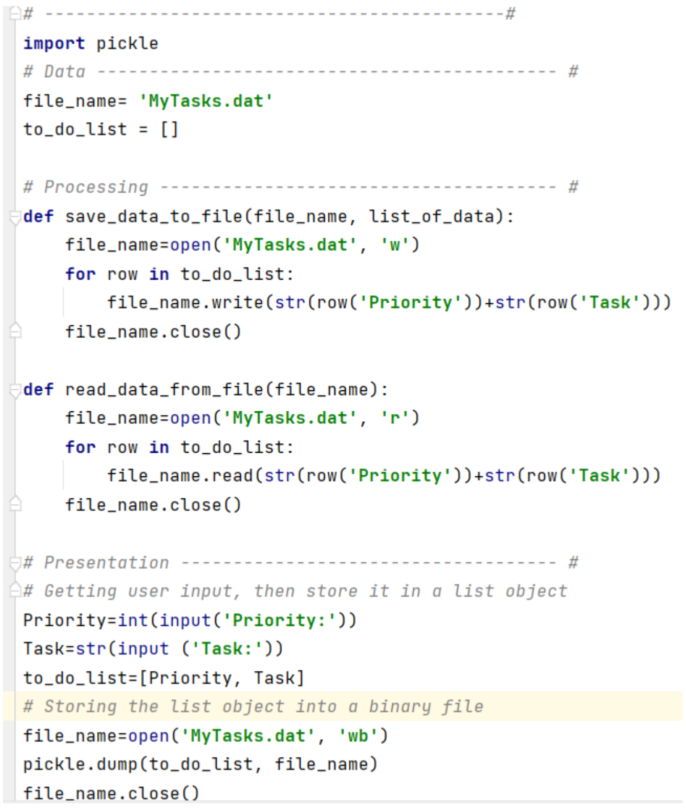
c. Processed the data by defining my functions to save data to the file and then to read data from the file.
d. Added code for the presentation section:
    i. Added two input statements to collect data from the user
    ii. Stored the input into a list
    iii. Stored the list into a binary file by opening the file in writing 'wb' mode and using the 'pickle.dump' expression. Closed the file.
    iv. Read the file into a new list and displayed the content. Opened the file and used the 'pickle.load' expression to read the file. Closed the file.
    v. Added a print statement to print the data in the file.
e. ***Figure 2*** shows my code with the pickling script.



```
# IO Error - file does not exist
try:
    File=open('MyDatabase.txt', 'r')
except IOError:
    print ("Could not find the file")
else:
    data=File.read()
    print(data)
    File.close()

# ZeroDivisionError, ValueError
while True:
    user_response=input('Please enter a number other than zero:')
    try:
        y=float(user_response)
        p=10/y
    except (ZeroDivisionError):
        print ('Cannot be divided by 0!')
    except (ValueError):
        print ('Input needs to be a number!')
    else:
        print ('The value of p is', p)
        break
    finally:
        print ('Done')
```

Figure 1: Error handling



```
# --------------------------------------------#
import pickle
# Data -------------------------------------------- #
file_name= 'MyTasks.dat'
to_do_list = []

# Processing -------------------------------------- #
def save_data_to_file(file_name, list_of_data):
    file_name=open('MyTasks.dat', 'w')
    for row in to_do_list:
        file_name.write(str(row('Priority'))+str(row('Task')))
    file_name.close()

def read_data_from_file(file_name):
    file_name=open('MyTasks.dat', 'r')
    for row in to_do_list:
        file_name.read(str(row('Priority'))+str(row('Task')))
    file_name.close()

# Presentation ------------------------------------ #
# Getting user input, then store it in a list object
Priority=int(input('Priority:'))
Task=str(input ('Task:'))
to_do_list=[Priority, Task]
# Storing the list object into a binary file
file_name=open('MyTasks.dat', 'wb')
pickle.dump(to_do_list, file_name)
file_name.close()
```

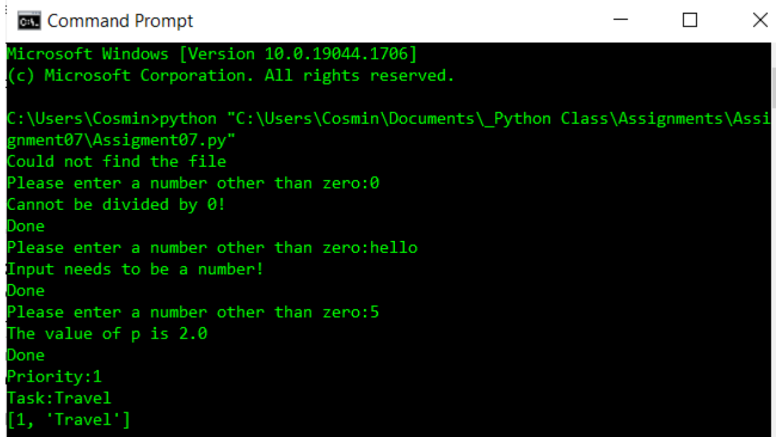
```
# Reading the data from the file into a new list object and display the contents
file_name=open('MyTasks.dat', 'rb')
file_name_data=pickle.load(file_name)
file_name.close()

print(file_name_data)
```

Figure 2: Functions in the class Input/Output

5. **Ran the program through CMD**
    a. Opened CMD and added the path file for “Assignment05”

b. ***Figure 3*** shows the output in CMD

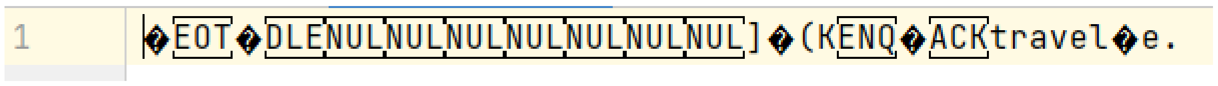


```
Microsoft Windows [Version 10.0.19044.1706]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Cosmin>python "C:\Users\Cosmin\Documents\_Python Class\Assignments\Assi
gnment07\Assigment07.py"
Could not find the file
Please enter a number other than zero:0
Cannot be divided by 0!
Done
Please enter a number other than zero:hello
Input needs to be a number!
Done
Please enter a number other than zero:5
The value of p is 2.0
Done
Priority:1
Task:Travel
[1, 'Travel']
```

Figure 3: Output in the CMD

6. **Opened the binary file in a text editor**
   a. Located the text file and opened it to verify that it displays the user input
   b. ***Figure 4*** shows the output in the text editor



```
�EOT�DLENULNULNULNULNULNULNUL]�(KENQ�ACKtravel�e.
```

Figure 4: Output in the binary file

Summary

Creating and running Assignment 07 allowed me to work with error handling types and saving the file into a binary format.