Sopheaktra Danh
8/24/2020
Foundations of Programming: Python
Assignment 07
https://github.com/SDanh/IntroToProg-Python-Mod07

07: Files & Exceptions

Introduction

This week covered working with Binary Files and Exceptions. The former being an overview of 'Pickling' & 'Unpickling' which is Python's variation of Serialization in other languages while the latter was an overview of Python's Error/Exception hierarchy. For the weekly assignment we were to also research both topics ourselves and to use that knowledge to create a new script to show how pickling and exception handling worked in python.

Research

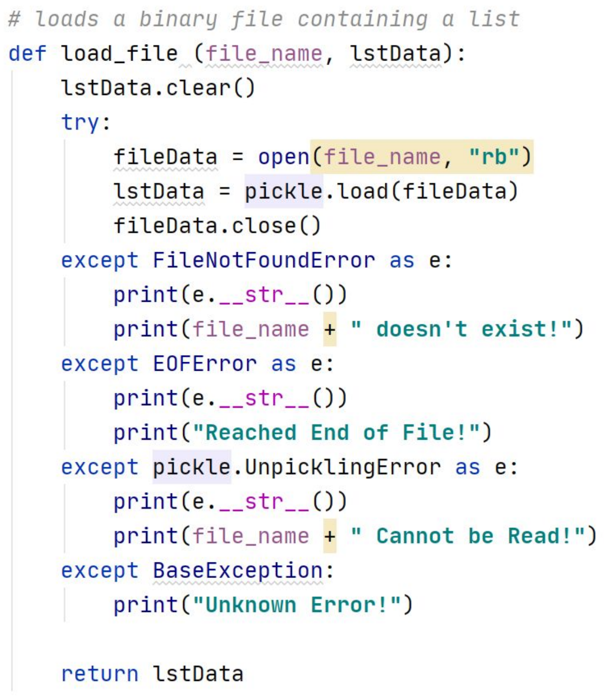
I searched for multiple sources for Python Exception Handling & Pickling and found two useful links for each topic below (Figure 1).

Exception Handling Resources
https://www.programiz.com/python-programming/exception-handling http://cs.carleton.edu/cs_comps/1213/pylearn/final_results/encyclopedia/
Python Pickling Resources
https://www.tutorialspoint.com/python-pickling https://docs.python.org/3/library/pickle.html

Figure 1: Research Resources

Basic Implementation

Primarily most of the work was on the basic implementation of pickling and unpickling. I reasoned that exceptions could be done within the reading and writing to files. This was done with a Try-Catch block that would catch errors such as non-existent files, end-of-files, and files that could not be pickled/unpickled from (Figure 2).

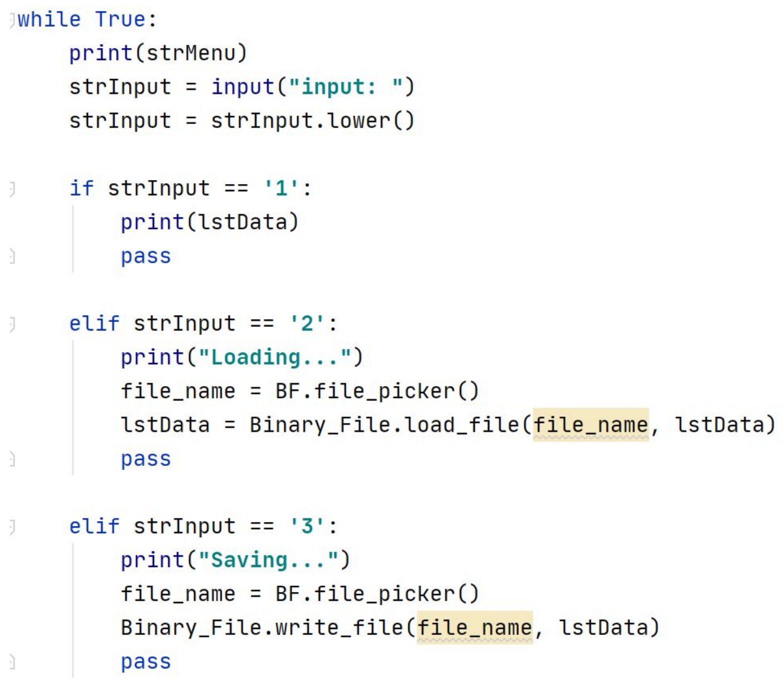


```python
# loads a binary file containing a list
def load_file (file_name, lstData):
    lstData.clear()
    try:
        fileData = open(file_name, "rb")
        lstData = pickle.load(fileData)
        fileData.close()
    except FileNotFoundError as e:
        print(e.__str__())
        print(file_name + " doesn't exist!")
    except EOFError as e:
        print(e.__str__())
        print("Reached End of File!")
    except pickle.UnpicklingError as e:
        print(e.__str__())
        print(file_name + " Cannot be Read!")
    except BaseException:
        print("Unknown Error!")

    return lstData
```

Figure 2: Pickling with Try-Catch blocks.

Usability

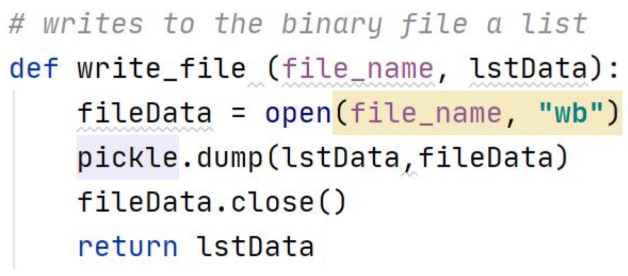
A menu interface similar to the previous assignments was implemented to showcase the script's ability to pickly, unpickle, and handle exceptions. Unlike the previous versions the script asks for which file it should save and load from. (Figure 3).



```python
while True:
    print(strMenu)
    strInput = input("input: ")
    strInput = strInput.lower()

    if strInput == '1':
        print(lstData)
        pass

    elif strInput == '2':
        print("Loading...")
        file_name = BF.file_picker()
        lstData = Binary_File.load_file(file_name, lstData)
        pass

    elif strInput == '3':
        print("Saving...")
        file_name = BF.file_picker()
        Binary_File.write_file(file_name, lstData)
        pass
```

Figure 3: Menu Interface

To clean code for readability, the reading and writing to binary file, the 'pickling', was moved to it's own class: Binary_File(). The functions comprised of a reading & writing to binary file function and a basic helper function to capture the file name for use with the class (Figures 2 & 4).

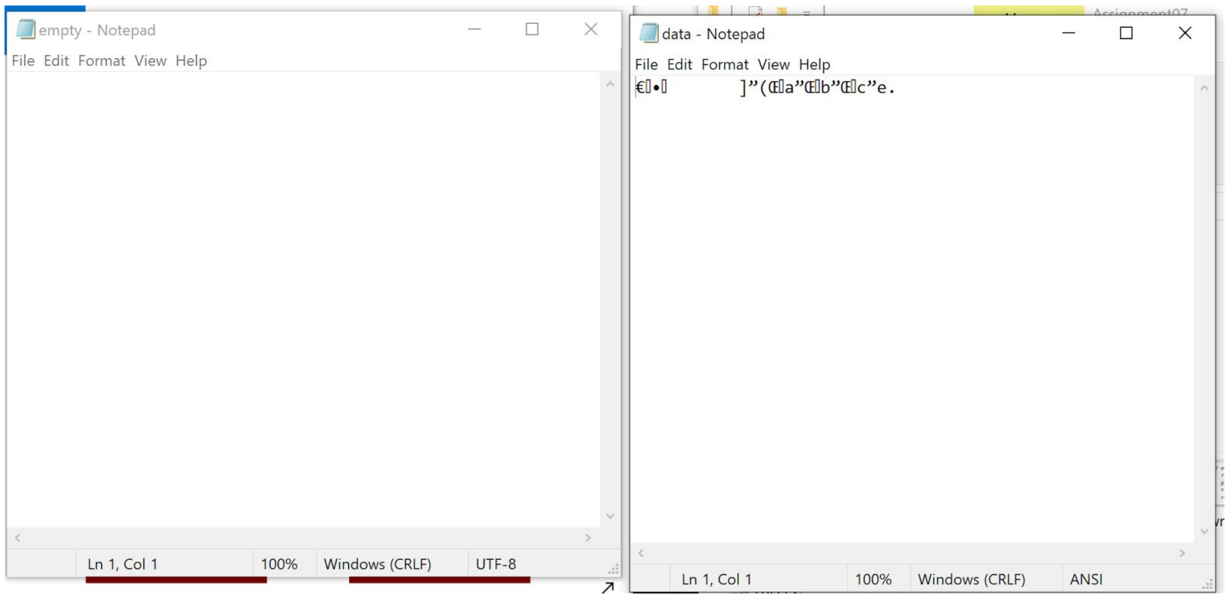


```
# writes to the binary file a list
def write_file (file_name, lstData):
    fileData = open(file_name, "wb")
    pickle.dump(lstData,fileData)
    fileData.close()
    return lstData
```

Figure 4: Write/Pickling Function

Testing

Testing was done in PyCharm and standard Windows Command Prompt shell. As like the previous python assignments the real test was done in Shell. Included in the Assignment07 folder are three test files: 'data.txt', 'empty.txt', & 'image.png'. These test files were used to test pickling and exception handling (Figures 5 & 6).



```
€•        ]”(Œa”Œb”Œc”e.
```

Figure 5: empty.txt & data.txt

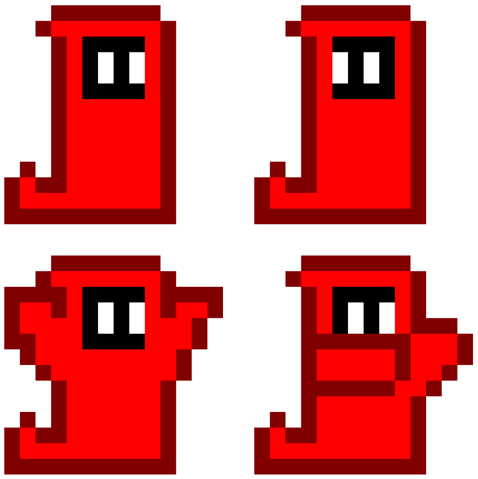
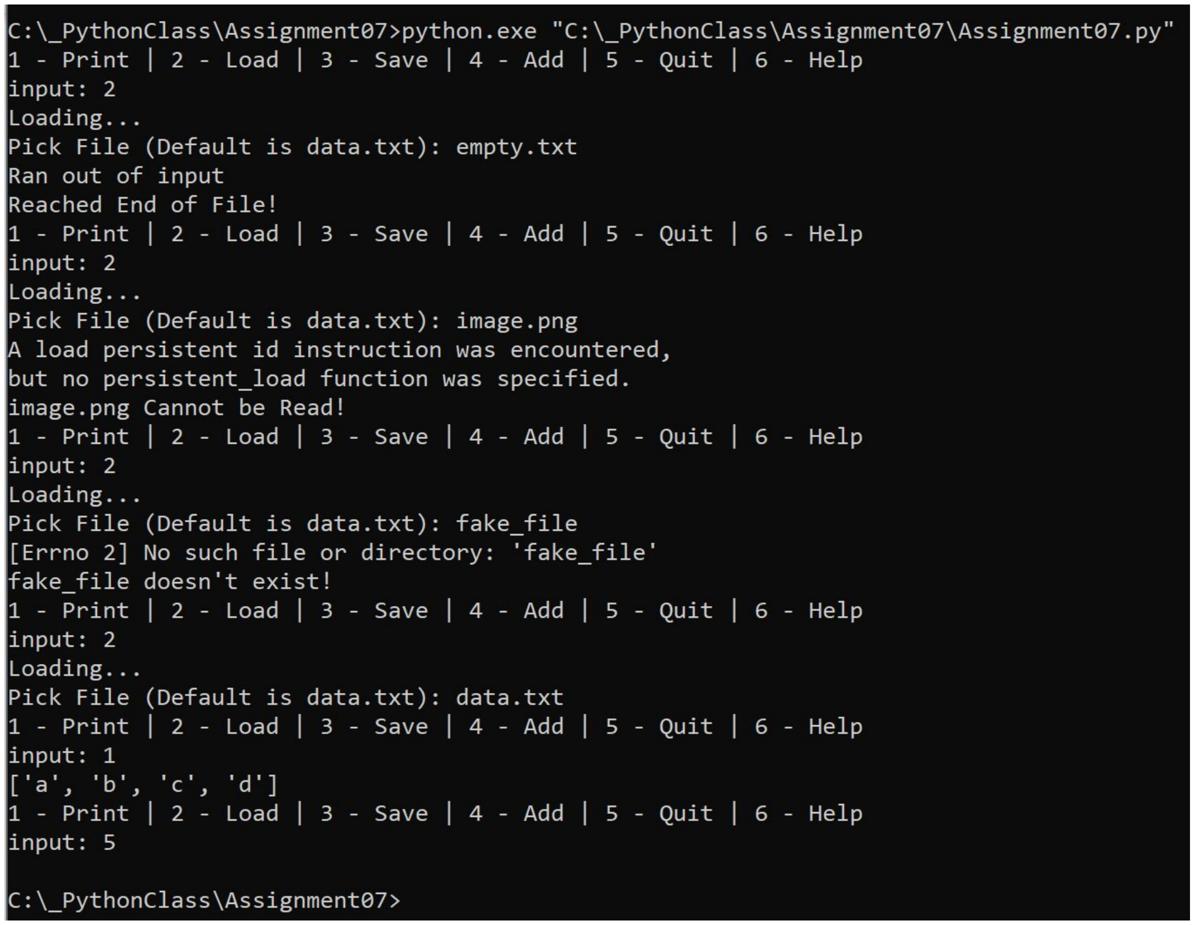


Figure 6: image.png

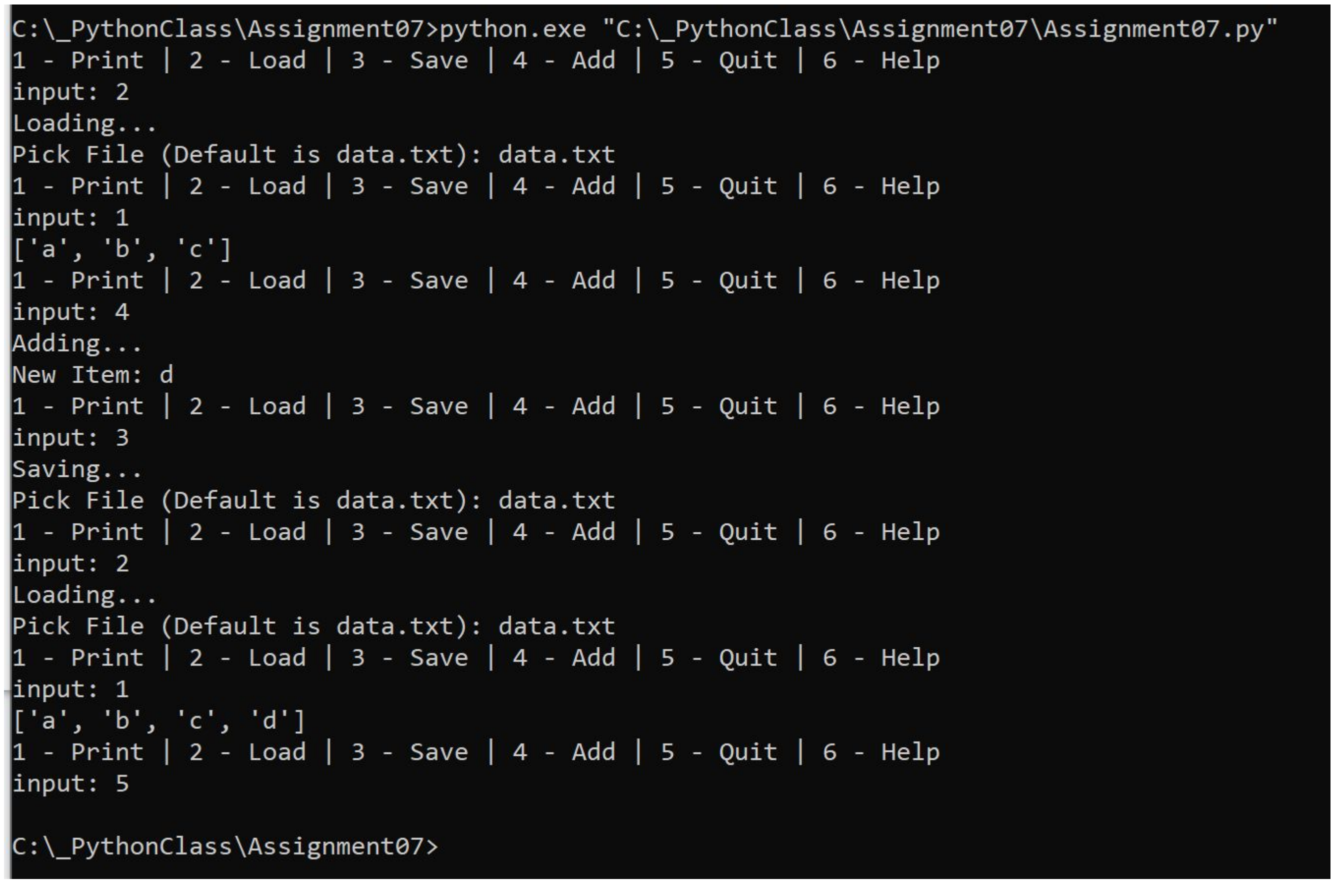
To help with the understanding process I implemented a help option in the menu. ‘empty.txt’ would throw a End Of File error as the program runs around using a single list that is used in the pickling process to load and dump from. As the file is empty there is nothing to be read. ‘image.png’ in turn would throw an Unpickling error because png files cannot be pickled or unpickled as-is. ‘data.txt’ is the only one of the three files to work as it already contains readable binary. Any other inputted file name would throw a Files Does Not Exist error for self-explanatory reasons (Figure 7).



```
C:\_PythonClass\Assignment07>python.exe "C:\_PythonClass\Assignment07\Assignment07.py"
1 - Print | 2 - Load | 3 - Save | 4 - Add | 5 - Quit | 6 - Help
input: 2
Loading...
Pick File (Default is data.txt): empty.txt
Ran out of input
Reached End of File!
1 - Print | 2 - Load | 3 - Save | 4 - Add | 5 - Quit | 6 - Help
input: 2
Loading...
Pick File (Default is data.txt): image.png
A load persistent id instruction was encountered,
but no persistent_load function was specified.
image.png Cannot be Read!
1 - Print | 2 - Load | 3 - Save | 4 - Add | 5 - Quit | 6 - Help
input: 2
Loading...
Pick File (Default is data.txt): fake_file
[Errno 2] No such file or directory: 'fake_file'
fake_file doesn't exist!
1 - Print | 2 - Load | 3 - Save | 4 - Add | 5 - Quit | 6 - Help
input: 2
Loading...
Pick File (Default is data.txt): data.txt
1 - Print | 2 - Load | 3 - Save | 4 - Add | 5 - Quit | 6 - Help
input: 1
['a', 'b', 'c', 'd']
1 - Print | 2 - Load | 3 - Save | 4 - Add | 5 - Quit | 6 - Help
input: 5

C:\_PythonClass\Assignment07>
```

Figure 7: Testing exceptions

Pickling/Saving to ‘data.txt’ was done by adding a new entry ‘d’ and the file was reloaded to indicate that the data persisted (Figure 8).



```
C:\_PythonClass\Assignment07>python.exe "C:\_PythonClass\Assignment07\Assignment07.py"
1 - Print | 2 - Load | 3 - Save | 4 - Add | 5 - Quit | 6 - Help
input: 2
Loading...
Pick File (Default is data.txt): data.txt
1 - Print | 2 - Load | 3 - Save | 4 - Add | 5 - Quit | 6 - Help
input: 1
['a', 'b', 'c']
1 - Print | 2 - Load | 3 - Save | 4 - Add | 5 - Quit | 6 - Help
input: 4
Adding...
New Item: d
1 - Print | 2 - Load | 3 - Save | 4 - Add | 5 - Quit | 6 - Help
input: 3
Saving...
Pick File (Default is data.txt): data.txt
1 - Print | 2 - Load | 3 - Save | 4 - Add | 5 - Quit | 6 - Help
input: 2
Loading...
Pick File (Default is data.txt): data.txt
1 - Print | 2 - Load | 3 - Save | 4 - Add | 5 - Quit | 6 - Help
input: 1
['a', 'b', 'c', 'd']
1 - Print | 2 - Load | 3 - Save | 4 - Add | 5 - Quit | 6 - Help
input: 5

C:\_PythonClass\Assignment07>
```

Figure 8: Pickling/Saving to binary file.

Summary

Examples of pickling and exception handling can be done separately and possibly in a more readable format. Creating a program that can read and write to binary files and handle affiliated errors is in my belief a better real world example. Showing a practical application of two topics is helpful in the learning process.