Jessica Chen

November 26, 2022

Foundations Of Programming: Python

Assignment 07

GitHub URL: https://github.com/jesscUW/IntroToProg-Python-Mod07

GitHub Webpage: https://jesscuw.github.io/IntroToProg-Python-Mod07/

Create a script that demonstrates how Pickling and Structured error handling work

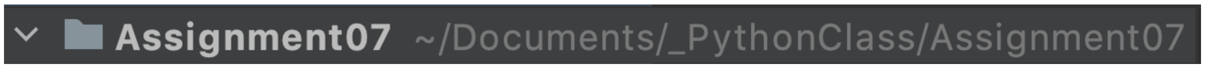
Introduction

In this assignment, I will document how I created a Python script that demonstrates how pickling and structured error handling work. This program asks a user to input a name of a household item and its value, and then both pieces of data are stored in a binary file. The pickle module was used in my script for storing and retrieving the data in the binary file. In addition, I used try and except to handle the exception that might be raised in the process.

Creating the Script

Creating a New Project in PyCharm

To begin with this assignment, I created a new project in PyCharm with the folder location "_PythonClass/Assignment07" (Figure 1).

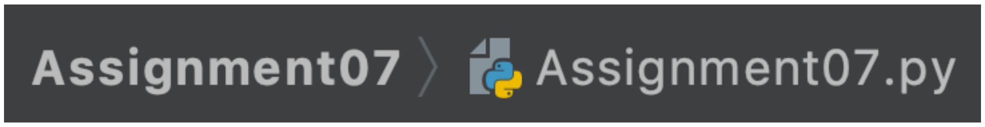


Assignment07 ~/Documents/_PythonClass/Assignment07

Figure 1: Creating a new project in PyCharm

Creating a Python Script

I created a script file named "Assignment07.py" within my project (Figure 2).

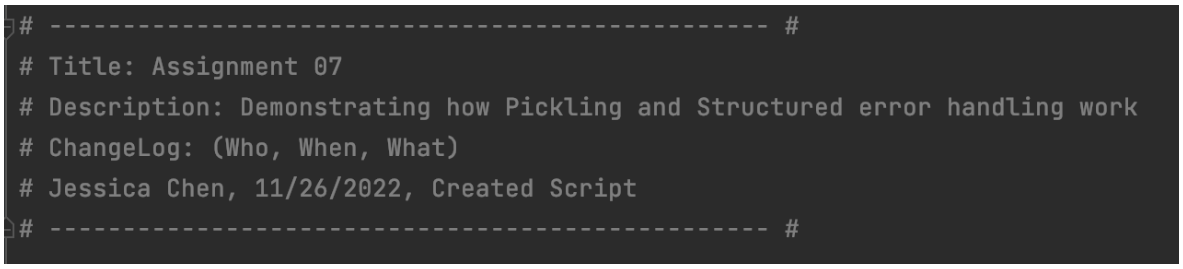


Assignment07 〉 Assignment07.py

Figure 2: Creating a new Python script in the project

Script Header

Before adding code to my script, I put a header at the top to provide some basic information. (Figure 3).

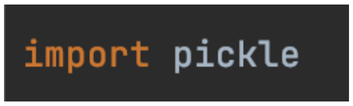


```
# ---------------------------------------------- #
# Title: Assignment 07
# Description: Demonstrating how Pickling and Structured error handling work
# ChangeLog: (Who, When, What)
# Jessica Chen, 11/26/2022, Created Script
# ---------------------------------------------- #
```

Figure 3: Screenshot of the header in my script

Pickling

The first thing I did was import the pickle module (Figure 4).

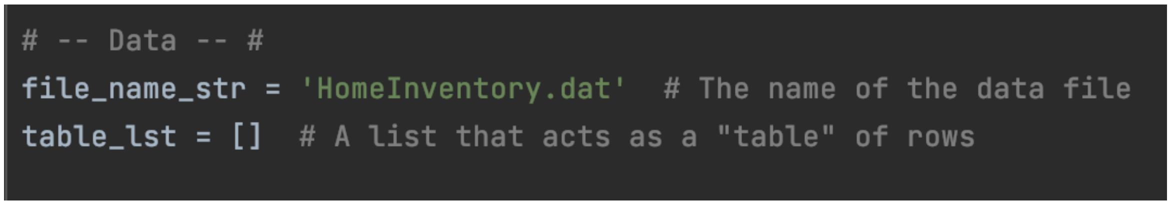


```
import pickle
```

Figure 4: Screenshot of importing the pickle module

Declaring variables

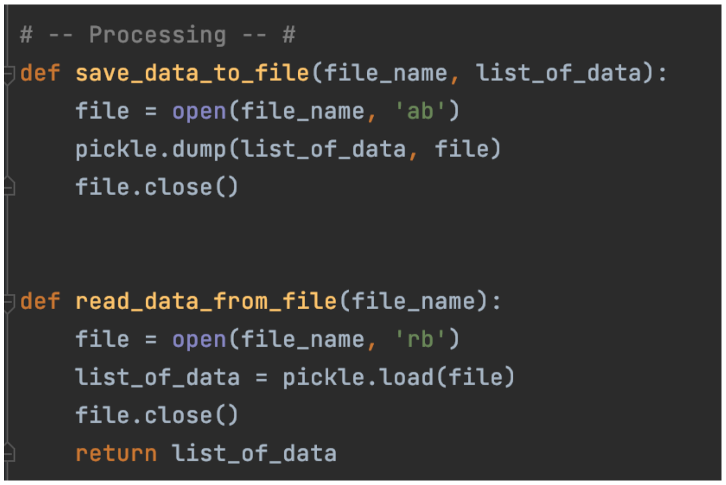
In Data section, I declared two variables: “file_name_str” and “table_lst”. “file_name_str” is the data storage file “HomeInventory.dat”; “table_lst” is a list that contains rows of data (Figure 5).



```
# -- Data -- #
file_name_str = 'HomeInventory.dat'  # The name of the data file
table_lst = []  # A list that acts as a "table" of rows
```

Figure 5: Declaring the variables

Processing
Then I defined two functions in the Processing section. In the first function "save_data_to_file()", a new binary file would be opened for appending data with the file access mode "ab". I used the pickle.dump() function in the statement to pickle and store the data. The second function defined was "read_data_from_file()". I used the pickle.load() function here for retrieving and unpickle the data (Figure 6).



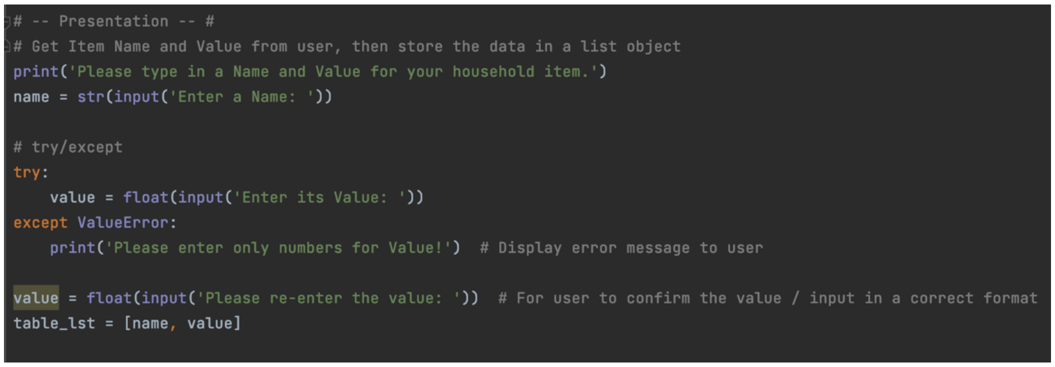
```python
# -- Processing -- #
def save_data_to_file(file_name, list_of_data):
    file = open(file_name, 'ab')
    pickle.dump(list_of_data, file)
    file.close()


def read_data_from_file(file_name):
    file = open(file_name, 'rb')
    list_of_data = pickle.load(file)
    file.close()
    return list_of_data
```

Figure 6: Defining two functions for adding and extracting data in a binary file

Presentation
I used the print() function to print out the message telling the user what they are going to do. Two variables "name" and "value" were named and assigned the user's input. Then I added "value" again for the user to re-enter the item's value. This statement was added because of the try-except block, which I will explain later. "table_lst" was assigned "[name, value]".

try/except
As it happened to me in a previous assignment that I entered a word instead of a number and the program crashed while testing on my code, I added a try-except block for handling this error. The first thing I did was ask the user for the item's value. A string would return and then be converted to a floating-point number. Then, I wrote an except clause with a statement to be executed if an exception is raised. I specified the exception type "ValueError" to be handled. If the user enters an unconvertible string, the exception is caught and the user will be informed "Please enter only numbers for Value!".

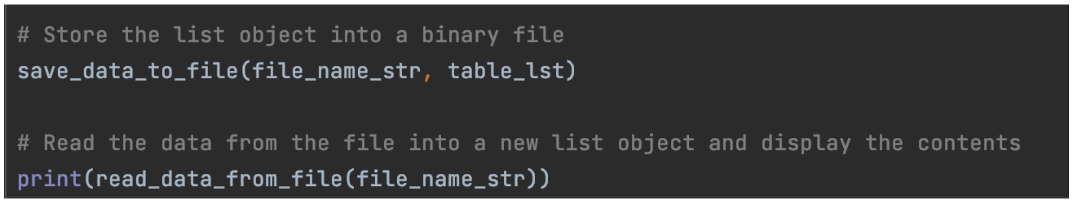
To continue with the code to the end (saving and retrieving the item and its value), I redeclared “value” here to request the user to enter the value again. The user can have another chance to confirm that they are entering a correct answer at this point (Figure 7).



```
# -- Presentation -- #
# Get Item Name and Value from user, then store the data in a list object
print('Please type in a Name and Value for your household item.')
name = str(input('Enter a Name: '))

# try/except
try:
    value = float(input('Enter its Value: '))
except ValueError:
    print('Please enter only numbers for Value!')  # Display error message to user

value = float(input('Please re-enter the value: '))  # For user to confirm the value / input in a correct format
table_lst = [name, value]
```

Figure 7: Getting user’s input and using a try statement with an except clause

The final step was calling the functions. I called “save_data_to_file()” to save the data in the binary file “HomeInventory.dat”; then I called the “read_data_from_file()” function in a print statement to read the data from the binary file and display the content (Figure 8).



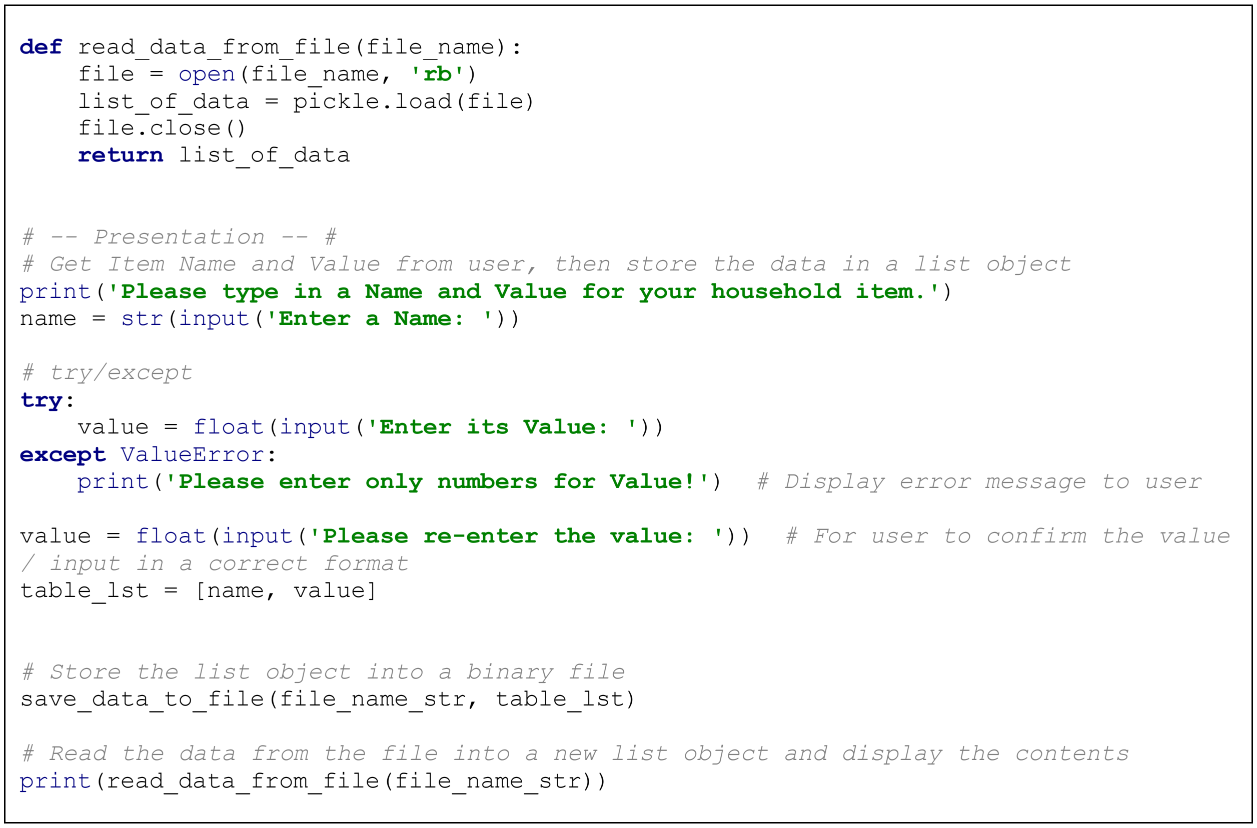
```
# Store the list object into a binary file
save_data_to_file(file_name_str, table_lst)

# Read the data from the file into a new list object and display the contents
print(read_data_from_file(file_name_str))
```

Figure 8: Calling functions

Comments

To help other people understand my code and also be able to remind myself my original intentions in the future, I added some notes. My script is shown as below (Figure 9):

```
# -------------------------------------------------- #
# Title: Assignment 07
# Description: Demonstrating how Pickling and Structured error handling work
# ChangeLog: (Who, When, What)
# Jessica Chen, 11/26/2022, Created Script
# -------------------------------------------------- #

import pickle

# -- Data -- #
file_name_str = 'HomeInventory.dat'  # The name of the data file
table_lst = []  # A list that acts as a "table" of rows


# -- Processing -- #
def save_data_to_file(file_name, list_of_data):
    file = open(file_name, 'ab')
    pickle.dump(list_of_data, file)
    file.close()
```

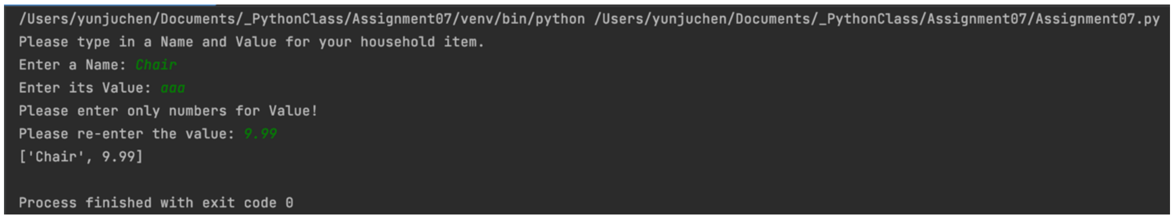


```python
def read_data_from_file(file_name):
    file = open(file_name, 'rb')
    list_of_data = pickle.load(file)
    file.close()
    return list_of_data


# -- Presentation -- #
# Get Item Name and Value from user, then store the data in a list object
print('Please type in a Name and Value for your household item.')
name = str(input('Enter a Name: '))

# try/except
try:
    value = float(input('Enter its Value: '))
except ValueError:
    print('Please enter only numbers for Value!')  # Display error message to user

value = float(input('Please re-enter the value: '))  # For user to confirm the value
/ input in a correct format
table_lst = [name, value]


# Store the list object into a binary file
save_data_to_file(file_name_str, table_lst)

# Read the data from the file into a new list object and display the contents
print(read_data_from_file(file_name_str))
```

Figure 9: Screenshot of my script

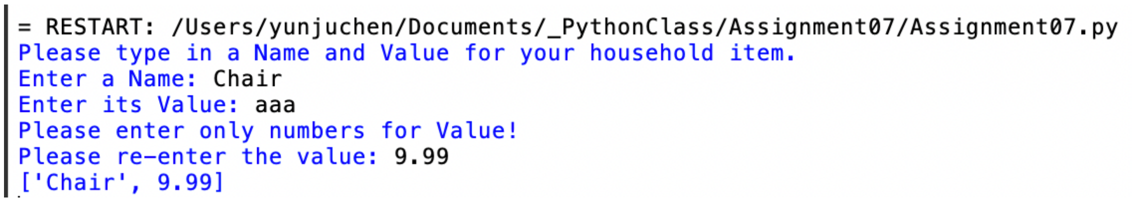
Running the Script

I ran Assignment07.py in both PyCharm and Shell. Following the prompt, I entered the item name, and then I typed in “aaa” for the Value to test if try-except block worked. The error message “Please enter only numbers for Value!” showed up. Then the program asked me to enter the value again. This time I correctly input a number “9.99”, and the program displayed my data “chair” and “9.99”. The script ran as expected (Figure 10 and 11); also, a binary file “HomeInventory.dat” was created and the data was written into the file (Figure 12).



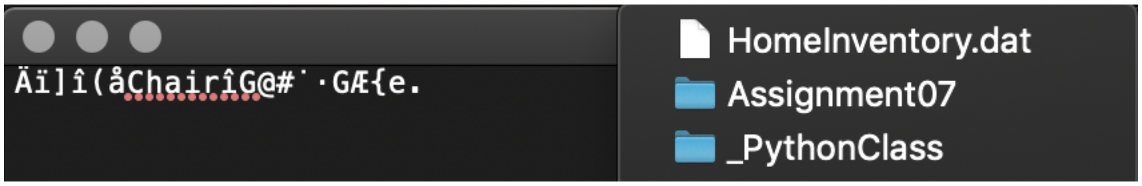
```
/Users/yunjuchen/Documents/_PythonClass/Assignment07/venv/bin/python /Users/yunjuchen/Documents/_PythonClass/Assignment07/Assignment07.py
Please type in a Name and Value for your household item.
Enter a Name: Chair
Enter its Value: aaa
Please enter only numbers for Value!
Please re-enter the value: 9.99
['Chair', 9.99]

Process finished with exit code 0
```

Figure 10: Screenshot of Assignment07.py running in PyCharm



```
= RESTART: /Users/yunjuchen/Documents/_PythonClass/Assignment07/Assignment07.py
Please type in a Name and Value for your household item.
Enter a Name: Chair
Enter its Value: aaa
Please enter only numbers for Value!
Please re-enter the value: 9.99
['Chair', 9.99]
```

Figure 11: Screenshot of Assignment07.py running in Shell



```
Äï]î(åChairîG@#˙·GÆ{e.
```

Figure 12: Screenshot of the data written in the binary file

Summary

Through Module 07, I learned about Python's pickle module and exception handling. When searching for the pages explaining these subjects, I found that "Python Pickling: https://www.tutorialspoint.com/python-pickling (external site)" and "Python Error Handling: https://www.w3schools.com/python/gloss_python_error_handling.asp (external site)" were helpful because they provided several simple and straightforward examples. After reading and watching the materials, I created a script demonstrating how pickling and structured error handling work.