Trace Kershaw

Assignment 7

IT FDN 100

**Pickles and Exceptions**

The purpose of this assignment was to learn about two basic components of python programming. Pickling and Error Exception Handling. Pickling is useful for data storage and extraction of complex data, so it is a handy thing to know for largely back-end and storage purposes. Error Exception Handling is more relevant for front-end purposes. It allows one to create custom error messages for end users (or programmers) that are more user friendly than what is provided by python coding. Both are important concepts to managing the back and front end of one’s program. For this assignment, we were asked to do some research on both pickling and exception handling in python.

**Researching Pickling**

* We were first asked to do some research on pickling in python. Through that research, I found that pickling is used to store and unpack related data (often done for dictionaries, lists, tuples) such that all of the information is stored together, and can be used for later referencing or unpacking in a program. Some of the main components of pickling are saving sets of objects to a binary data file (pickled data goes to binary and not text data files) using the dump function, reading and unpickling data used the load function.
* During my research I also found there is a lot of talk about the security of pickling, and a lot of warnings in the python community that pickled data is not secure, and can contain malicious programs, and that only pickled data from trusted sources should be unpickled.

**Researching Error Exception Handling**

* Error exception handling allows programmers to make standard python error messages more user friendly, and create their own custom error messages or message classes (types of errors and error messages that may be specific to a specific program). This gives programmers a lot of flexibility on how a user navigates and interacts with their program, and the type of information they give the user to help make their end using experience a pleasant one.
* Python comes with many types and classes of error messages. To program Error Exception Handling, it is primarily done through the Try and Except codes. Try is used before the primary code section, and except is used if there is an error within the code section. If just Except is put, any error would trigger the action after except (usually a print statement stating what went wrong), or specific errors can be named so custom error messages can be conveyed.

**Developing a Program Demonstrating Pickling and Error Exception Handling**

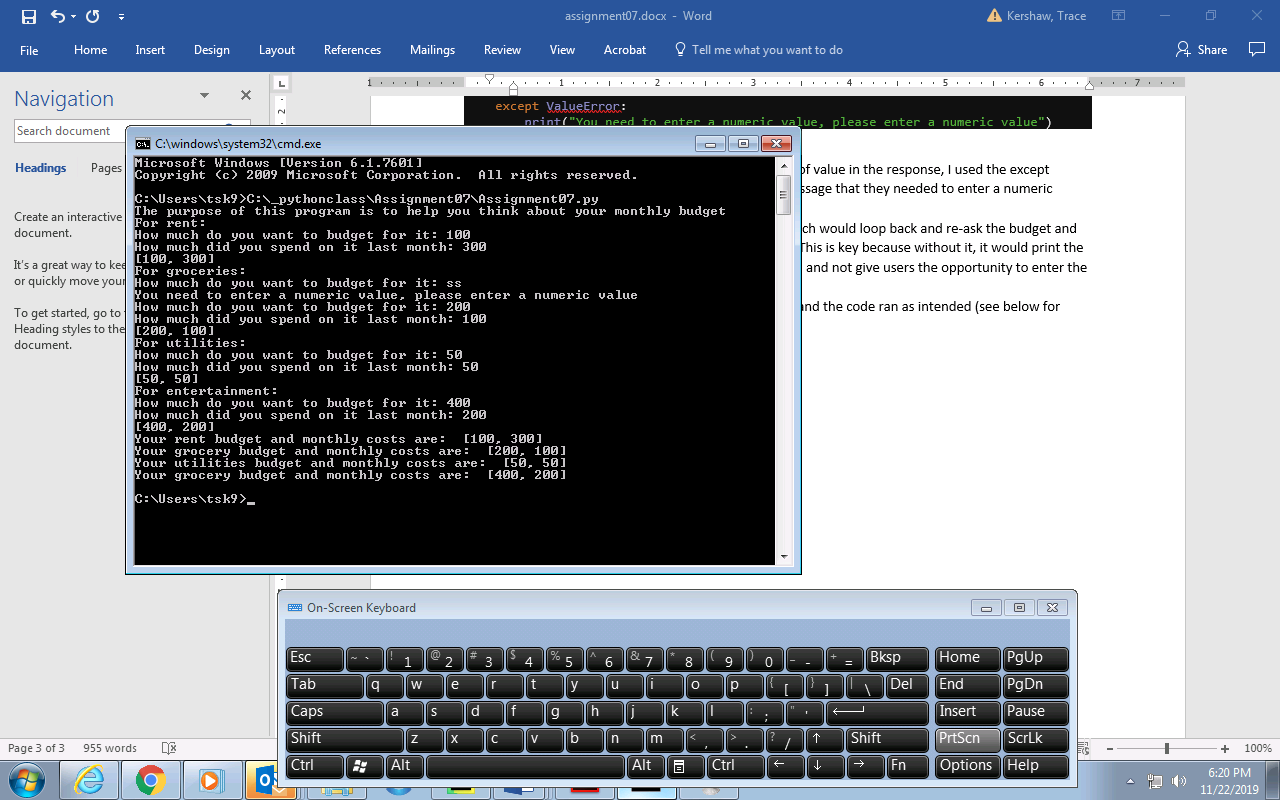
* We were next tasked with writing some code that uses pickling to save data to a file and unpack that data to a file.
* I decided to create a program that asked users to make a budget for four items: rent, groceries, utilities, and entertainment. For each budget item they would propose a monthly budget and how much they spent on that category last month.
* Therefore, I have four categories with two items per category (budget, costs).
* The goal was to pickle those 4 groupings together.
* Further, given that I wanted users to enter in dollar amounts budgeted and spent, I created integer variables for those items. Therefore, I created error exception handling that would provide an error message if they entered a non-numeric value for the budget and cost items.
* First, I created a series of user input codes for each category to get user budget and cost, and then created a list for each set of inputs. I decided to write these all out separately for the clarity of the code instead of doing functions.
* r\_budget=int(input("How much do you want to budget for it: "))  
  r\_cost=int(input("How much did you spend on it last month: "))  
  r\_list=[r\_budget**,**r\_cost]  
  print(r\_list)
* Once I had the set of lists, I pickled the data first by creating a new binary data file using the ab (append a binary file) command.
* objFile=open("budget.dat"**,** "ab")  
  pickle.dump(r\_list**,**objFile)  
  pickle.dump(g\_list**,**objFile)  
  pickle.dump(u\_list**,**objFile)  
  pickle.dump(e\_list**,**objFile)  
  objFile.close()
* And then pickling each of the 4 list objects to that file using the dump command.
* Next, I unpickled the data by opening the file in read mode, and unpickled the data using the load command.

objFile=open("budget.dat"**,**"rb")  
rent=pickle.load(objFile)  
groceries=pickle.load(objFile)  
utilities=pickle.load(objFile)  
entertainment=pickle.load(objFile)  
print ("Your rent budget and monthly costs are: "**,** rent)  
print ("Your grocery budget and monthly costs are: "**,** groceries)  
print ("Your utilities budget and monthly costs are: "**,** utilities)  
print ("Your grocery budget and monthly costs are: "**,** entertainment)  
objFile.close()

* Finally, I printed out the unpickled data using the print command, with language that lets the user know which type of data it is.
* Next, I created error exception handling, so users had to put integer values for the budget and cost questions. To do this, I used the Try and Except commands.

while True:  
 try:  
 g\_budget=int(input("How much do you want to budget for it: "))  
 g\_cost=int(input("How much did you spend on it last month: "))  
 g\_list=[g\_budget**,**g\_cost]  
 print(g\_list)  
 break  
 except ValueError:  
 print("You need to enter a numeric value, please enter a numeric value")

* Since the focus was on putting the correct type of value in the response, I used the except ValueError command, and created a custom message that they needed to enter a numeric value.
* I also used a loop using the while command, which would loop back and re-ask the budget and cost questions if users did not enter an integer. This is key because without it, it would print the error message but move on to the next question and not give users the opportunity to enter the value correctly.
* I tested the code in PyCharm and Windows OS, and the code ran as intended (see below for examples).



* Finally, I loaded my assignment on GitHub