Carl Caba

August 17, 2020

IT FDN 110 A

Assignment07

<https://github.com/simiaczar>

Introduction of Binary Files Error Handling and Pickling

# Introduction:

Throughout this module we were tasked with exercising pickling within a program working with data in binary files, and error handling.

## Working with Binary files:

What I learned from this section is that binary files are an alternative to the text(.txt) format. The way that this is accessed is by importing pickles:

**import** pickle

## Structured Error Handling:

Error handling is a way for coders to get around the ‘bug’ problem by trapping the issue before it happens. In order to do this a coder could write:

try:  
 quotient = 5 / 0  
 print(quotient)  
except:  
 print("There was an error! <<< Custom Message!\n")

# Assignment Execution:

During this assignment we were given the task of utilizing pickles, manipulating binary files, and error handling. At the beginning of the script we import pickle, which allows the coder to import code from another code file. Next we establish the data section of the script where we use objFile, strFileName, and lstCustomer, which can be found throughout the script. Next is processing where interacting with the binary file can be found. Here we open the file and save that data to the file; next the program read the file and checks for other entries. The data that the program can store is found from the input that the user makes in the new\_data\_to\_list(): section. The last part of this script is the error handling section where if problems occur the program is conducted to execute the process.

import pickle # This imports code from another code file!  
  
# Data -------------------------------------------- #  
objFile = None  
strFileName = 'AppData.dat'  
lstCustomer = []  
  
# Processing -------------------------------------- #  
def save\_data\_to\_file(file\_name, list\_of\_data):  
 with open(file\_name, 'wb') as file:  
 pickle.dump(list\_of\_data, file)  
 file.close()  
  
def read\_data\_from\_file(file\_name):  
 with open(file\_name, 'rb') as file:  
 list\_of\_data = pickle.load(file)  
 file.close()  
 return list\_of\_data  
  
# Presentation ------------------------------------ #  
def new\_data\_to\_list():  
 CustomerID = int(input('Enter the customer ID: '))  
 strName = str(input('Enter the name: '))  
 lstCustomer = [CustomerID, strName]  
 return lstCustomer  
  
try:  
 fileData = read\_data\_from\_file(strFileName)  
except FileNotFoundError as e:  
 print("File not found, please try again")  
 fileData = lstCustomer  
except pickle.UnpicklingError as e:  
 print("The file you selected is corrupt")  
  
try:  
 lstCustomer.append(new\_data\_to\_list())  
 save\_data\_to\_file(strFileName, lstCustomer)  
except ValueError as e:  
 print(e)  
  
 input("Press Enter to Exit")

# Summary:

Working through this section has allowed me to exercise using the pickle function, interacting with the binary functions, and structured error handling. By implementing these tools into my script I can see first hand just how they work together.