Ethan Morgan

08/31/2021

IT FDN 110 A, Foundations of Programming (Python)

Assignment 08

GitHub Link: <https://github.com/ethan-morgan/IntroToProg-Python-Mod08> (External Link)

Intro to Programming (Python)

# Introduction

Assignment 08 overall goal was to give experience using classes and objects within Python. Classes and objects are used to help organize functions and data. Classes code can be used either directly or indirectly.

# Product and Price Script

Figure 1 shows the script for the ‘Assignment08.py’ script. The purpose of this code was to take in user input using classes and functions. The script is broken into the typical 4 sections of Data, Processing, Presentation, and Main Script. Three classes were used: Product, File Processing, and Input-Output. This was done to organize the script for clarity.

# ------------------------------------------------------------------------ #  
# Title: Assignment 08  
# Description: Working with classes  
  
# ChangeLog (Who,When,What):  
# RRoot,1.1.2030,Created started script  
# RRoot,1.1.2030,Added pseudo-code to start assignment 8  
# Ethan Morgan,8/31/2021,Modified code to complete assignment 8  
# ------------------------------------------------------------------------ #  
  
  
# Data Start ------------------------------------------------------------- #  
strFileName = 'products\_name\_price.txt'  
ListTable = []  
  
class Product:  
 *"""Stores data about a product:  
  
 properties:  
 product\_name: (string) with the products's name  
 product\_price: (float) with the products's standard price  
 methods:  
 changelog: (When,Who,What)  
 RRoot,1.1.2030,Created Class  
 Ethan Morgan,08/31/2021,Modified code to complete assignment 8  
 """* def \_\_init\_\_(self, product\_name: str, product\_price: float):  
 self.\_\_product\_name = ""  
 self.\_\_product\_price = ""  
 try:  
 self.product\_name = str(product\_name)  
 self.product\_price = float(product\_price)  
 except Exception as e:  
 raise Exception("Error")  
  
 def \_\_str\_\_(self):  
 return self.product\_name+","+str(self.product\_price)  
  
 @property # Setter for Product Name  
 def product\_name(self):  
 return str(self.\_\_product\_name)  
  
 @product\_name.setter # Getter for Product Name  
 def product\_name(self, value: str):  
 self.\_\_product\_name = str(value)  
  
 @property # Setter for Product Price  
 def product\_price(self):  
 return float(self.\_\_product\_price)  
  
 @product\_price.setter # Getter for Product Price, Warning if the User doesn't enter a float value  
 def product\_price(self, value: float):  
 try:  
 self.\_\_product\_price = float(value)  
 except ValueError:  
 raise Exception("User must enter a number for the price.")  
# Data End ---------------------------------------------------------------- #  
  
  
# Processing Start -------------------------------------------------------- #  
class FileProcessor:  
 *"""Processes data to and from a file and a list of product objects:  
  
 methods:  
 save\_data\_to\_file(file\_name, list\_of\_product\_objects):  
  
 read\_data\_from\_file(file\_name): -> (a list of product objects)  
  
 changelog: (When,Who,What)  
 RRoot,1.1.2030,Created Class  
 Ethan Morgan,08/31/2021,Modified code to complete assignment 8  
 """* @staticmethod  
 def file\_read(FileName: str):  
 List\_Table = []  
 file = open(FileName, "r")  
 for row in file:  
 ListData = row.split(",")  
 ListRow = Product(ListData[0], ListData[1])  
 List\_Table.append(ListRow)  
 file.close()  
 print()  
 print("Previous data file exists and data is uploaded.")  
 return List\_Table  
  
 def file\_save(FileName: str, ListRows: list):  
 file = open(FileName,"w")  
 for row in ListRows:  
 file.write(row.\_\_str\_\_() + "\n")  
 print()  
 print("File is saved.")  
  
 # Attempted to create a file but was getting an error when there wasn't  
 # a file already existing with data in it. This program requires a  
 # text file with the proper name in order to run.  
 #def file\_create(FileName: str):  
 # List\_Table = []  
 # ListRow = [0,0]  
 # file = open(FileName,"w")  
 # #ListRow = Product(ListTable[0]=="", ListTable[1]=="")  
 # List\_Table.append(ListRow)  
 # file.close()  
 # print()  
 # print("File is created.")  
# Processing End----------------------------------------------------------- #  
  
  
# Presentation (Input/Output) Start --------------------------------------- #  
class IO:  
 @staticmethod  
 def print\_menu():  
 print("""  
 Product Name/Price Menu:  
 [1] - Show Existing/Current Data  
 [2] - Add Product Name/Price to Existing Data  
 [3] - Save Current Data  
 [4] - Quit The Program  
 """)  
  
 @staticmethod  
 def input\_menu():  
 MenuOption = str(input("Menu #: ").strip())  
 return MenuOption  
  
 @staticmethod  
 def print\_list(input\_list: list):  
 if len(input\_list) != 0:  
 print("The Current List Has: ")  
 for row in input\_list:  
 print(row.product\_name+","+str(row.product\_price))  
 else:  
 print()  
 print("Please add some data using Option [2]!")  
  
 @staticmethod  
 def input\_data():  
 ProductName\_Input = str(input("Product Name: ").strip())  
 while True:  
 try:  
 ProductPrice\_Input = float(input("Product Price: ").strip())  
 if type(ProductPrice\_Input) == float:  
 Product\_Name\_Price = Product(product\_name=ProductName\_Input,product\_price=ProductPrice\_Input)  
 return Product\_Name\_Price  
 else:  
 continue  
 except ValueError:  
 print("Please enter a numeric value for the price.")  
# Presentation (Input/Output) End ----------------------------------------- #  
  
# Main Body of Script Start ----------------------------------------------- #  
# Load data from file into a list of product objects when script starts  
# Show user a menu of options  
# Get user's menu option choice  
 # Show user current data in the list of product objects  
 # Let user add data to the list of product objects  
 # let user save current data to file and exit program  
  
# Refer to file\_create fucntion definition for issue description  
#try:  
ListTable = FileProcessor.file\_read(strFileName) # Command to Read Existing File  
#except:  
# print()  
# print("File does not exist, therefore a file will be created.")  
# ListTable = FileProcessor.file\_create(strFileName) # Command to Create File if it Doesn't Exist  
  
while True: # Use 'while' Loop to Continue Asking for User Input  
 IO.print\_menu() # Print Menu for Selection  
 UserInput = IO.input\_menu() # Take in User Input from Menu Selection  
  
 if UserInput == "1": # Show Existing/Current Data  
 try:  
 IO.print\_list(ListTable) # Print the data to screen  
 except:  
 print()  
 print("There is nothing in the list yet, please use Option [2] to add things to the list.")  
 continue  
  
 elif UserInput == "2": # Add Product Name/Price to Existing Data  
 ListTable.append(IO.input\_data())  
  
 elif UserInput == "3": # Save Current Data  
 try:  
 FileProcessor.file\_save(strFileName, ListTable)  
 except:  
 print()  
 print("No data was added please add something first.")  
 continue  
  
 elif UserInput == "4": # Quit the Program  
 print()  
 print("You are exiting the program.")  
 break  
  
 else:  
 print()  
 print("Please enter Option [1], [2], [3], or [4]")  
# Main Body of Script End ------------------------------------------------- #

Figure ‘Assignment08.py’ Script

One thing that I had an issue with in this script was the ability to run the script without a text file with data in the folder location to begin with. I created a function to just create a file, but an error of “NoneType” kept coming up. I believe this has something to do with there being no data in the file. I tried to add data to the file via the code but it just wasn’t working so instead in order to run this code you must have a text file with the correct name and data inside.

Figure 2 and Figure 2 show the script from Assignment 08 working in both PyCharm and Command OS/Shell respectively. Note that not all functionality of the code is shown in these two figures. Please refer to running the code to see the other capabilities of the script.

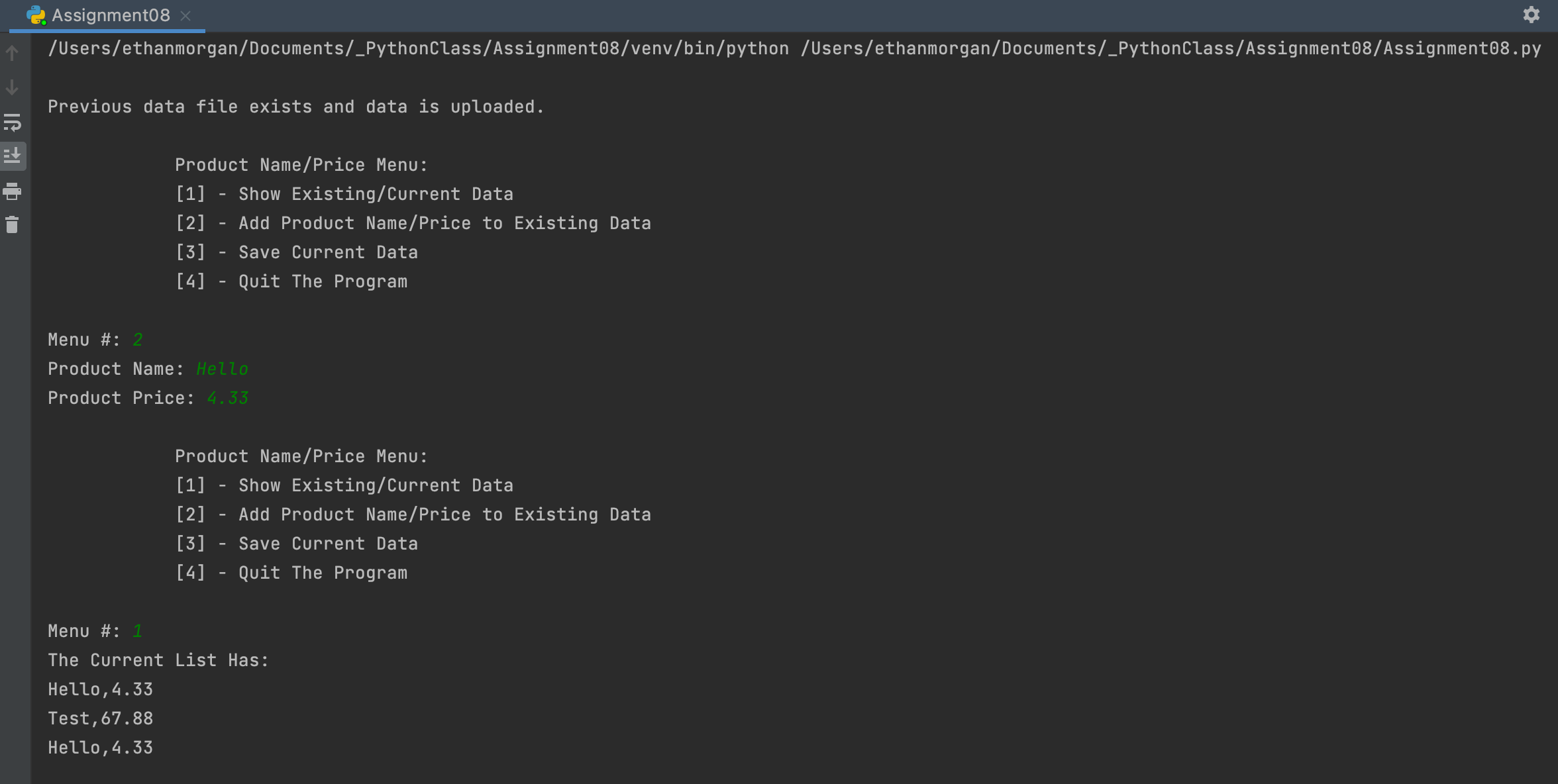


Figure ‘Assignment08.py’ Script Working Screenshot in PyCharm

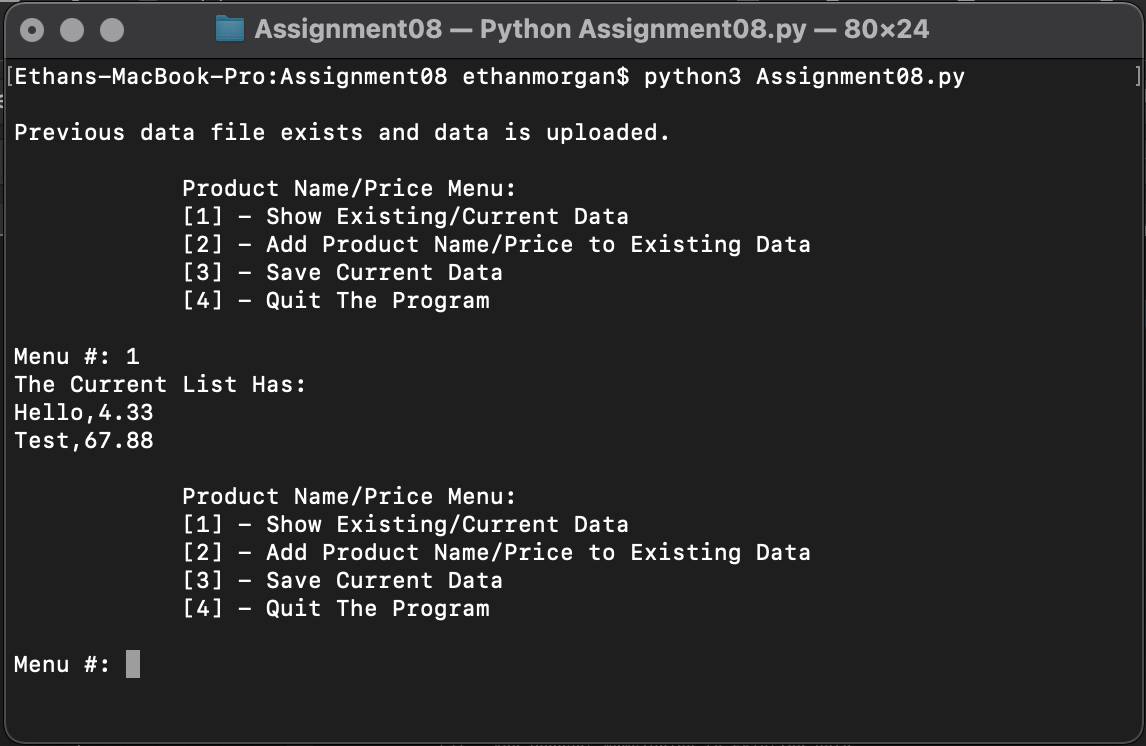


Figure “Assignment08.py’ Script Working Screenshot in Command OS/Shell

# Summary

Assignment 08 provided a continuation of the use of classes in Python. This assignment also provided us a chance of taking being able to write the majority of the script rather than taking someone else’s code, besides for pusedo code. Additionally there was the use of error handling to help provide more user friendly errors to be present than the Python provided errors.