Ethan Morgan

08/10/2021

IT FDN 110 A, Foundations of Programming (Python)

Assignment 05

Intro to Programming (Python)

# Introduction

Assignment 05 was a continuation of the skills learned in Assignment 04, like writing to a text file, with some new things to expand upon with respect to lists and their use in the script. An additional added element was using a starter code that had the skeleton of the final script we were to use. This assignment gave us a look into how data is saved: be it in memory of the computer, temporarily saved without being saved on the hard drive, or saved to the hard drive via a text file.

# To Do List Script

Figure 1 shows the code used for the To Do List Script for Assignment 05. There are many aspects to this script that should be noted. From the start, the class was provided a script template that included a particular format. This is done to keep the code clear and consistent so that if anyone else were to pick up the code and try to modify it, the editor could easily see where things are (e.g. Title Block, Data, Processing, and Input/Output).

Another thing new was there was a skeleton of code that provided a guide of where to put the different aspects of the required code inside the script. Personally, it was difficult at first because I am used to writing all the code I use so it took me some time to go through what was done in order for me to move forward.

Next, the code that was written beyond what was provided in the module zip file was in the Processing section of the script. Here a try/except was used to either read an existing file or post that there is no existing file and to direct the user how to input data into a file. In this section, if there is a text file with the name “ToDoList.txt” then the data will be read in and saved to a list.

The following Section is the Input/Output part of the script where the user is required to input values and the script will post back responses to said inputs. There are several if statements, ‘.lower()’ and ‘.scrap()’ operators so that the user input does not error out the script. Note that when the data is saved that the text file is overwritten and not appended to.

# ------------------------------------------------------------------------ #  
# Title: Assignment 05  
# Description: Working with Dictionaries and Files  
# When the program starts, load each "row" of data  
# in "ToDoToDoList.txt" into a python Dictionary.  
# Add the each dictionary "row" to a python list "table"  
# ChangeLog (Who,When,What):  
# RRoot,1.1.2030,Created started script  
# Ethan Morgan, 08/09/2021, Added code to complete assignment 5  
# ------------------------------------------------------------------------ #  
  
  
# -- Data -- #  
# declare variables and constants  
strFile = "ToDoList.txt"  
objFile = "" # An object that represents a file  
strData = "" # A row of text data from the file  
dicRow = {} # A row of data separated into elements of a dictionary {Task,Priority}  
lstTable = [] # A list that acts as a 'table' of rows  
strMenu = "" # A menu of user options  
strChoice = "" # A Capture the user option selection  
strTask = "" # A variable for User Input at Step 4 for the Task  
strPriority = "" # A variable for User Input at Step 4 for the Priority  
strRemoveTask = "" # A variable for User Input to Remove a Task from the List in Step 5  
  
  
# -- Processing -- #  
# Step 1 - When the program starts, load the any data you have  
# in a text file called ToDoList.txt into a python list of dictionaries rows (like Lab 5-2)  
try:  
 objFile = open(strFile , "r")  
 for row in objFile:  
 strData = row.split(",")  
 dicRow = {"Task" : strData[0] , "Priority" : strData[1].strip()}  
 lstTable.append(dicRow)  
 objFile.close()  
 print()  
 print("There An Existing File, Data From File Is Saved To Memory")  
except:  
 print()  
 print("No Available Existing File, List Will Start A New")  
 print("Continue With The Menu Of Options")  
  
  
# -- Input/Output -- #  
# Step 2 - Display a menu of choices to the user  
while (True):  
 print("""  
 Menu of Options  
 1) Show current data  
 2) Add a new item.  
 3) Remove an existing item.  
 4) Save Data to File  
 5) Exit Program  
 """)  
 strChoice = str(input("Which option would you like to perform? [1 to 5] - "))  
 print() # adding a new line for looks  
  
 # Step 3 - Show the current items in the table  
 if (strChoice.strip() == '1'):  
 try:  
 print("Task -- Priority")  
 print("=" \* 30)  
 for row in lstTable:  
 print(row["Task"] + " -- " + row["Priority"])  
 except:  
 print("No Current Items In The List")  
 print("Please Select Option 2 To Add Items To The List")  
 continue  
  
 # Step 4 - Add a new item to the list/Table  
 elif (strChoice.strip() == '2'):  
 strTask = input("Enter A Task: ").strip()  
 strPriority = input("Enter A Priority (High, Low): ").strip()  
 if strPriority.lower() == "high" or strPriority.lower() == "low":  
 dicRow = {"Task" : strTask , "Priority" : strPriority}  
 lstTable.append(dicRow)  
 print(strTask + " Has Been Added To The List")  
 else:  
 print("Please Enter Either 'High' or 'Low', '" + strPriority + "' Isn't A Valid Input")  
 continue  
  
 # Step 5 - Remove a new item from the list/Table  
 elif (strChoice.strip() == '3'):  
 strRemoveTask = input("Please Enter A Task To Remove: ").strip()  
 Count = 1  
 for row in lstTable:  
 if row['Task'].lower() == strRemoveTask.lower():  
 lstTable.remove(row)  
 print(strRemoveTask + " Has Been Removed From The List")  
 Count = Count + 1  
 if Count == 1:  
 print(strRemoveTask + " Is Not In The List")  
 continue  
  
 # Step 6 - Save tasks to the ToDoList.txt file  
 elif (strChoice.strip() == '4'):  
 objFile = open(strFile , "w")  
 for row in lstTable:  
 objFile.write(row["Task"] + "," + row["Priority"] + "\n")  
 objFile.close()  
 print("Data Is Saved To The Text File")  
 continue  
  
 # Step 7 - Exit program  
 elif (strChoice.strip() == '5'):  
 print("Thanks For Using The Program!")  
  
 break # and Exit the program

Figure To Do List Script for Assignment 05

Figure 2 shows the script running in PyCharm for the “Add a new item” option. Please note that all other facets of the script work and only Option 2 was shown for clarity.

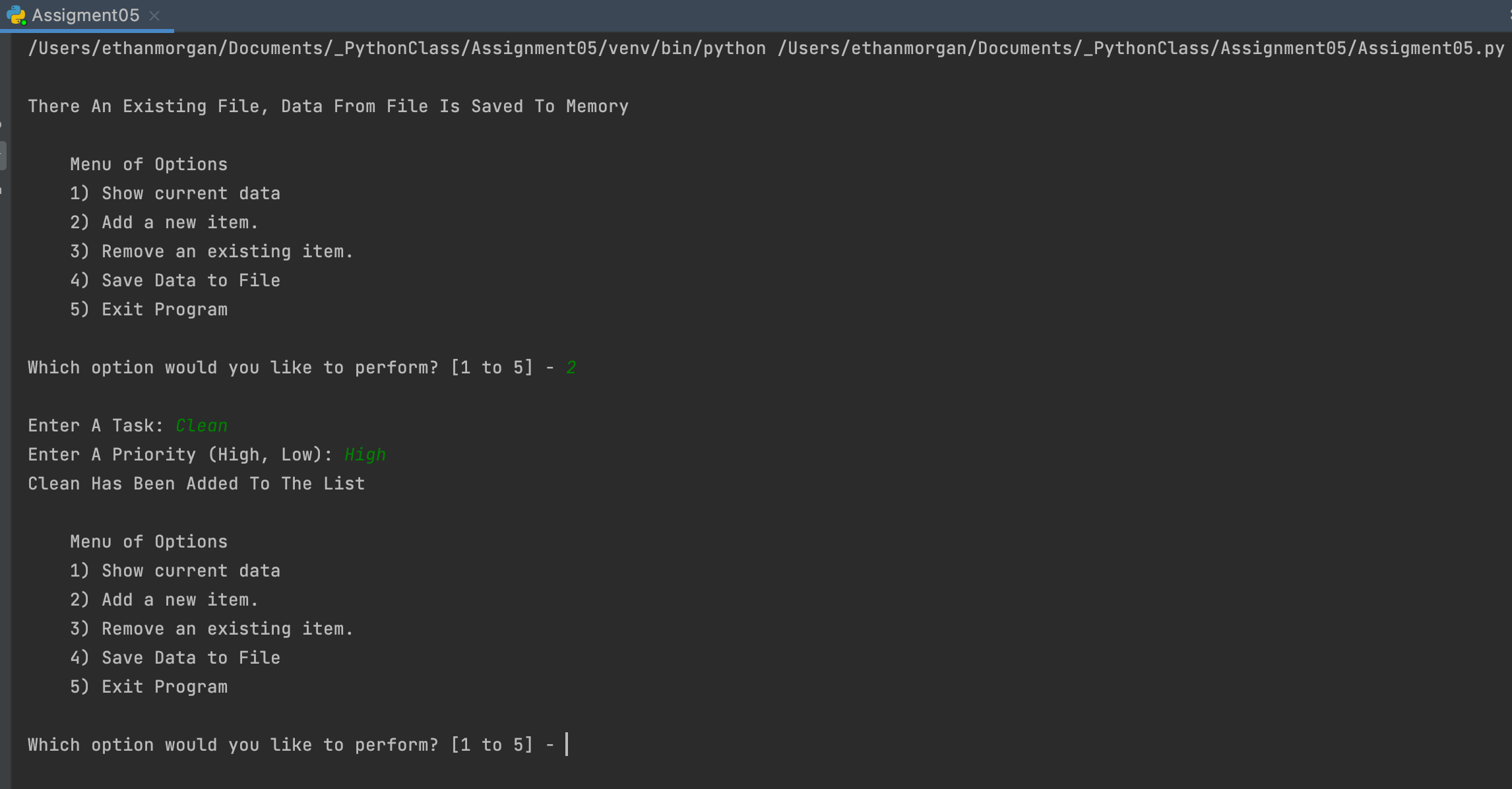


Figure PyCharm Screenshot of Script Working

Figure 3 shows the script working in the Command OS/Shell for Options 3 and 1. Please note that all other facets of the script work.

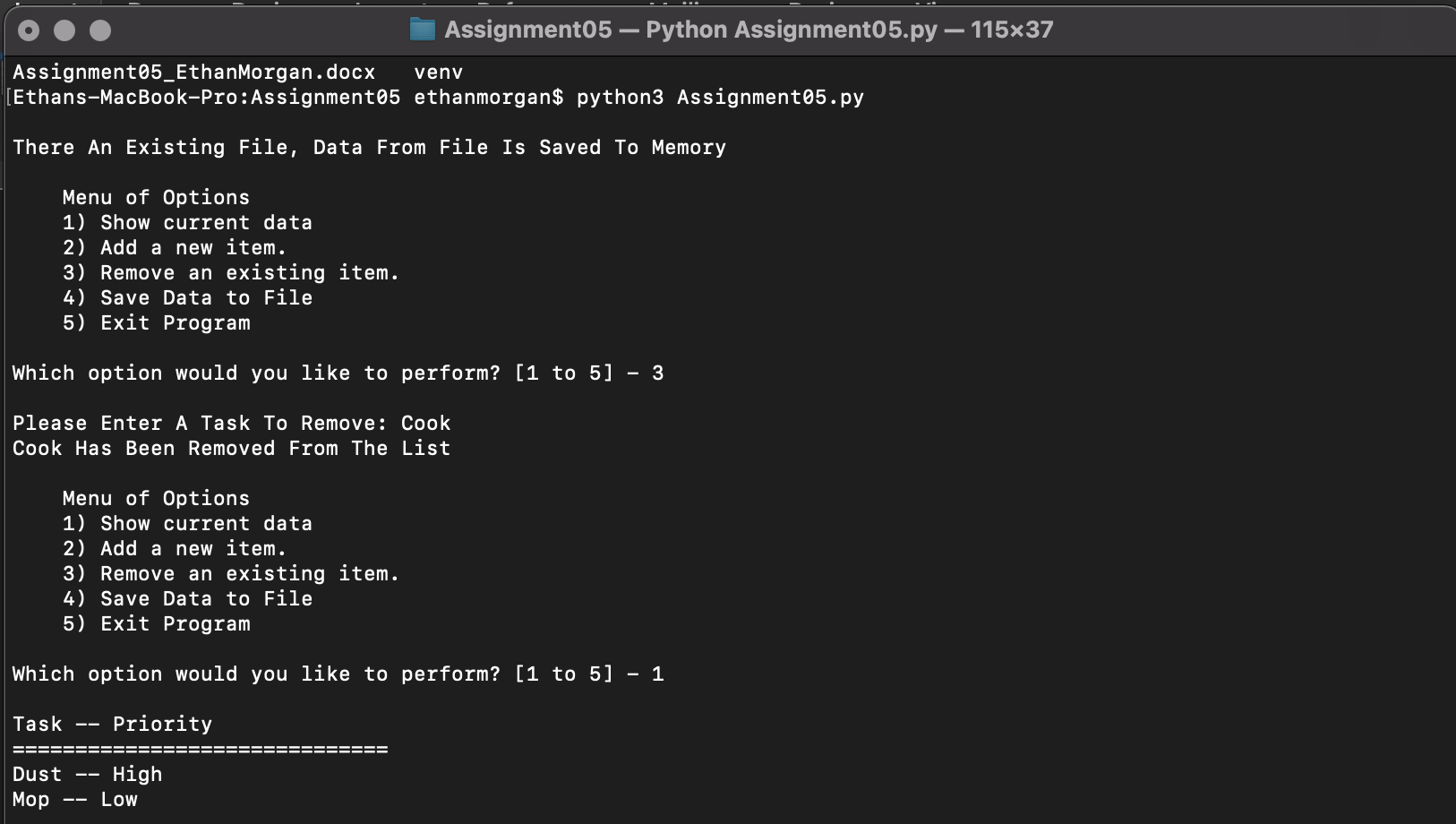


Figure Command OS/Shell Screenshot of Script Working

Finally Figure 4 shows the screenshot of the text file created by the script when entering Dust, Mop, and Cook for High, Low, and High Priorities respectively.

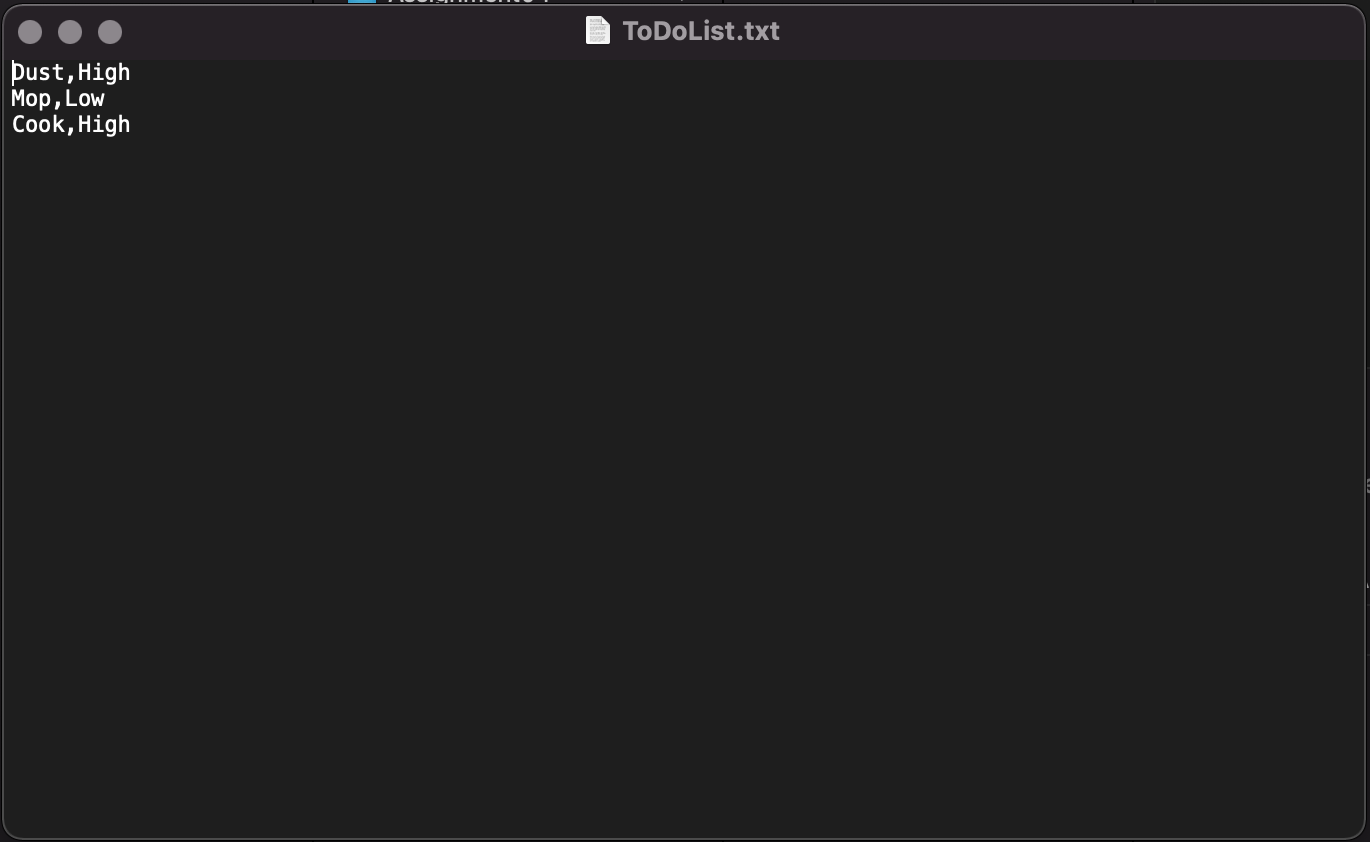


Figure Screenshot of Text File

# Summary

Assignment 05 provided a deeper look into using Python for programming. It was difficult to take someone else’s script skeleton/shell, but once you took the time to understand what was done it made it easy to drop in the code where you needed to. Additionally, this makes it easier for a common approach to code let potential editors know what is going on quickly. It was also helpful to understand how memory is used, via temporary or persistant (e.g. a text file).