Kevin Bangen

11/24/2020

Foundations of Programming: Python

Assignment 06

GitHub Link:

Functions

# Introduction

This paper will discuss the steps taken in completing Assignment 06. Assignment 06 was built using “starter code” and elements of Assignment 05. The solution utilized a series of custom functions in addition to a series of “for” and “while” loops. This code read data from .txt file, processed (modified) that data, and wrote the data to a .txt file.

# Processing

Three functions in the “Processor” class within the “starter code” required completion/updating. Once updated these functions added data to the list, removed data from the list, and wrote data to the file. These lines of code were more-or-less identical to elements of Assignment 05. The script for the completed functions is shown here:

@staticmethod

def add\_data\_to\_list(task\_input, priority\_input, list\_of\_rows):

dicRow = {"Task": task\_input, "Priority": priority\_input.strip()}

list\_of\_rows.append(dicRow)

return list\_of\_rows, 'Success'

@staticmethod

def remove\_data\_from\_list(delete\_input, list\_of\_rows):

for row in lstTable:

if (row["Task"] == delete\_input):

list\_of\_rows.remove(row)

return list\_of\_rows, 'Success'

@staticmethod

def write\_data\_to\_file(file\_name, list\_of\_rows):

objFile = open(file\_name, "w")

for row in list\_of\_rows:

objFile.write(row["Task"] + ", " + row["Priority"] + '\n')

objFile.close()

return list\_of\_rows, 'Success'

# Input/Output

Two functions in the “Presentation” class within the starter code required completion/updating. These two functions collect inputs from the user and are used in adding data to the list or removing a task from the list. Similar to the Processing functions, these lines of code were more-or-less identical to elements of Assignment 05. The script for the completed functions is shown here:

@staticmethod

def input\_new\_task\_and\_priority():

strTask = str(input("Please provide a task:"))

strPriority = str(input("Please provide a priority:"))

return strTask, strPriority

@staticmethod

def input\_task\_to\_remove():

strTask = str(input("Please provide a task:"))

return strTask

# Main Body of Script

The main body of the script utilized the custom functions (discussed above) in addition to a few additional lines of code to complete a variety of tasks. The script used in completing the tasks associated with user inputs 1,2,3 and 4 required updates/completion. Once completed, the tasks performed by the script are very similar to those from Assignment 05. The completed script is shown here:

# Step 4 - Process user's menu choice

if strChoice.strip() == '1': # Add a new Task

task, priority = IO.input\_new\_task\_and\_priority()

Processor.add\_data\_to\_list(task,priority,lstTable)

IO.input\_press\_to\_continue(strStatus)

continue # to show the menu

elif strChoice == '2': # Remove an existing Task

task = IO.input\_task\_to\_remove()

Processor.remove\_data\_from\_list(task,lstTable)

IO.input\_press\_to\_continue(strStatus)

continue # to show the menu

elif strChoice == '3': # Save Data to File

strChoice = IO.input\_yes\_no\_choice("Save this data to file? (y/n) - ")

if strChoice.lower() == "y":

Processor.write\_data\_to\_file(strFileName,lstTable)

IO.input\_press\_to\_continue(strStatus)

else:

IO.input\_press\_to\_continue("Save Cancelled!")

continue # to show the menu

elif strChoice == '4': # Reload Data from File

print("Warning: Unsaved Data Will Be Lost!")

strChoice = IO.input\_yes\_no\_choice("Are you sure you want to reload data from file? (y/n) - ")

if strChoice.lower() == 'y':

Processor.read\_data\_from\_file(strFileName,lstTable)

IO.input\_press\_to\_continue(strStatus)

else:

IO.input\_press\_to\_continue("File Reload Cancelled!")

continue # to show the menu

# Difficulties + Completed Tests

The primary challenges faced were keeping all of variables organized/consistent and unpacking the data from the custom functions.

Using functions led to their being a much larger number of variables / assigned values. Keeping all of these values organized was difficult. The incorrect value was used at times and this led to errors that required de-bugging.

When first written, the code did not correctly unpack the data from the custom functions. The code called on the values within the input/output functions and did not assign values as the tuple was unpacked. The code was eventually updated to correct this issue.

Successful tests in PyCharm and the Command Prompt are shown here:

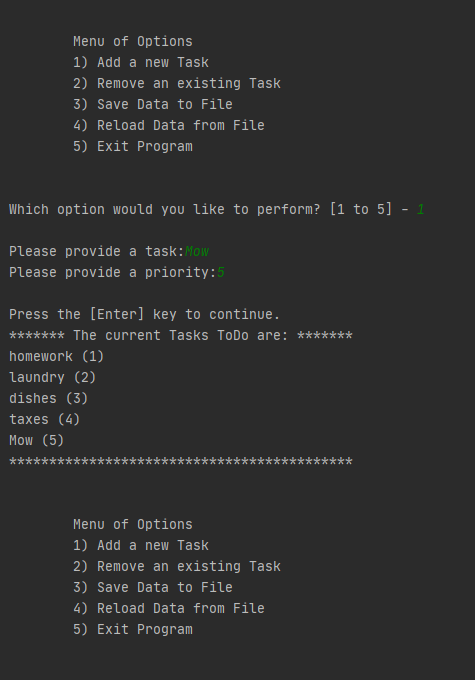


Figure 1: Successful test run in PyCharm.

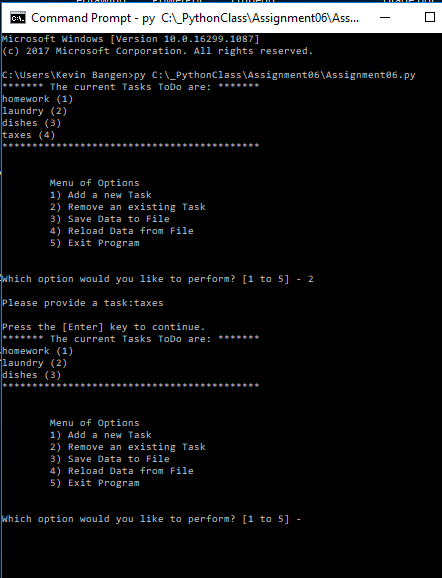


Figure 2: Successful test run in Command Prompt.

# Conclusion

In summary, this paper discussed the steps taken in completing Assignment 06. Assignment 06 was built using “starter code” and elements of Assignment 05. The solution utilized a series of custom functions in addition to a series of “for” and “while” loops. This code read data from .txt file, processed (modified) that data, and wrote the data to a .txt file.