Ethan Morgan

08/17/2021

< IT FDN 110 A, Foundations of Programming (Python)

Assignment 06

GitHub Link: <https://github.com/ethan-morgan/IntroToProg-Python-Mod6>

Intro to Programming (Python)

# Introduction

Assignment 06 was a continuation of the skills learned in Assignment 05, but with including functions rather than putting code in sequence/in line of which you would like it to run. Another thing that was introduced was the uses of classes for functions. Some aspects similar to Assignment 05 are writing to a text file and using someone else’s code as a starting point.

# To Do List with Functions

Figure 1 shows the script used for Assignment 06. Similar to the Assignment 05 script, the code is organized into different sections: Data, Processing, Presentation (Input/Output), and Main Body. Different from Assignment 05 is the use of functions to better separate the code for better organization.

For me, working with functions was harder to understand how the code was working but after staring at it for a while it started to make sense. I definitely think more in a ‘linear’ type way where if I’m trying to do something its just defined as I need it. I know that’s not how things are in the real world in coding because things can be hidden but it definitely is a change.

# ---------------------------------------------------------------------------- #  
# Title: Assignment 06  
# Description: Working with functions in a class,  
# 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  
# RRoot,1.1.2030,Added code to complete assignment 5  
# Ethan Morgan, 08/17/2021, Modified code to complete assignment 6  
# ---------------------------------------------------------------------------- #  
  
# Data ---------------------------------------------------------------------- #  
# Declare variables and constants  
strFileName = "ToDoFile.txt" # The name of the data file  
objFile = None # An object that represents a file  
dicRow = {} # A row of data separated into elements of a dictionary {Task,Priority}  
lstTable = [] # A list that acts as a 'table' of rows  
strChoice = "" # Captures the user option selection  
strTask = "" # Captures the user task data  
strPriority = "" # Captures the user priority data  
strStatus = "" # Captures the status of an processing functions  
list\_table = []  
count = ""  
  
# Processing --------------------------------------------------------------- #  
class Processor:  
 *""" Performs Processing tasks """* @staticmethod  
 def read\_data\_from\_file(file\_name, list\_of\_rows):  
 *"""  
 Reads data from a file into a list of dictionary rows.* ***:param*** *file\_name: (string) with name of file:* ***:param*** *list\_of\_rows: (list) you want filled with file data:* ***:return****: (list) of dictionary rows  
 """* list\_of\_rows.clear() # clear current data  
 file = open(file\_name, "r")  
 for line in file:  
 task, priority = line.split(",")  
 dicRow = {"Task": task.strip(), "Priority": priority.strip()}  
 list\_of\_rows.append(dicRow)  
 file.close()  
 return list\_of\_rows, 'Success'  
  
 @staticmethod  
 def add\_data\_to\_list(task, priority, list\_of\_rows):  
 *"""  
 Adds user input data into Dictionary Row.  
 """* dicRow = {"Task" : task, "Priority" : priority}  
 list\_of\_rows.append(dicRow)  
 return list\_of\_rows, 'Success'  
  
 @staticmethod  
 def remove\_data\_from\_list(task, list\_of\_rows):  
 *"""  
 Removes a row from the list of rows.  
 """* for dicRow in list\_of\_rows:  
 if dicRow["Task"].lower() == task.lower():  
 list\_of\_rows.remove(dicRow)  
 return list\_of\_rows, 'Success'  
  
 @staticmethod  
 def write\_data\_to\_file(file\_name, list\_of\_rows):  
 *"""  
 Writes the data to the file.  
 """* objFile = open(file\_name, "a")  
 for dicRow in list\_of\_rows: # Write each row of data to the file  
 objFile.write(dicRow["Task"] + "," + dicRow["Priority"] + "\n")  
 objFile.close()  
 #return list\_of\_rows, 'Success'  
 return 'Success'  
  
# Presentation (Input/Output) -------------------------------------------- #  
class IO:  
 *""" Performs Input and Output tasks """* @staticmethod  
 def print\_menu\_Tasks():  
 *""" Display a menu of choices to the user* ***:return****: nothing  
 """* print('''  
 Menu of Options  
 1) Add a new Task  
 2) Remove an existing Task  
 3) Save Data to File   
 4) Reload Data from File  
 5) Exit Program  
 ''')  
 print() # Add an extra line for looks  
  
 @staticmethod  
 def input\_menu\_choice():  
 *""" Gets the menu choice from a user* ***:return****: string  
 """* choice = str(input("Which option would you like to perform? [1 to 5] - ")).strip()  
 print() # Add an extra line for looks  
 return choice  
  
 @staticmethod  
 def print\_current\_Tasks\_in\_list(list\_of\_rows):  
 *""" Shows the current Tasks in the list of dictionaries rows* ***:param*** *list\_of\_rows: (list) of rows you want to display* ***:return****: nothing  
 """* print("\*\*\*\*\*\*\* The current Tasks ToDo are: \*\*\*\*\*\*\*")  
 for row in list\_of\_rows:  
 print(row["Task"] + " (" + row["Priority"] + ")")  
 print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")  
 print() # Add an extra line for looks  
  
 @staticmethod  
 def input\_yes\_no\_choice(message):  
 *""" Gets a yes or no choice from the user* ***:return****: string  
 """* return str(input(message)).strip().lower()  
  
 @staticmethod  
 def input\_press\_to\_continue(optional\_message=''):  
 *""" Pause program and show a message before continuing* ***:param*** *optional\_message: An optional message you want to display* ***:return****: nothing  
 """* print(optional\_message)  
 input('Press the [Enter] key to continue.')  
  
 @staticmethod  
 def input\_new\_task\_and\_priority():  
 *"""  
 Take a task and priority from user to add.  
 """* task = input("Input a Task to Add: ")  
 priority = input("Input a Priority to Add: ")  
 return task, priority  
  
 @staticmethod  
 def input\_task\_to\_remove():  
 *"""  
 Take a task and corresponding priority to remove.  
 """* task = input("Input a Task to Remove: ")  
 return task  
  
# Main Body of Script ------------------------------------------------------ #  
  
# Step 1 - When the program starts, either Load data from ToDoFile.txt or  
# create ToDoFile.txt file.  
try:  
 Processor.read\_data\_from\_file(strFileName, lstTable) # read file data  
except:  
 objFile = open(strFileName,"w")  
 objFile.close()  
  
# Step 2 - Display a menu of choices to the user  
while(True):  
 # Step 3 Show current data  
 IO.print\_current\_Tasks\_in\_list(lstTable) # Show current data in the list/table  
 IO.print\_menu\_Tasks() # Shows menu  
 strChoice = IO.input\_menu\_choice() # Get menu option  
   
 # Step 4 - Process user's menu choice  
 if strChoice.strip() == '1': # Add a new Task  
 task, priority = IO.input\_new\_task\_and\_priority()  
 list\_table, status = Processor.add\_data\_to\_list(task, priority, list\_table)  
 IO.print\_current\_Tasks\_in\_list(list\_table)  
 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, list\_table)  
 if count == "":  
 IO.input\_press\_to\_continue("Task is Deleted")  
 else:  
 IO.input\_press\_to\_continue("Task is Not Found, Input a Task Name")  
 IO.print\_current\_Tasks\_in\_list(list\_table)  
 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, list\_table)  
 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, list\_table)  
 IO.input\_press\_to\_continue(strStatus)  
 else:  
 IO.input\_press\_to\_continue("File Reload Cancelled!")  
 continue # to show the menu  
  
 elif strChoice == '5': # Exit Program  
 print("Goodbye!")  
 break # and Exit

Figure To Do List Script for Assignment 06

Figure 2 shows a screenshot of the ToDoList script working in PyCharm. Only a portion of the working code was shown in the figure below for clarity.

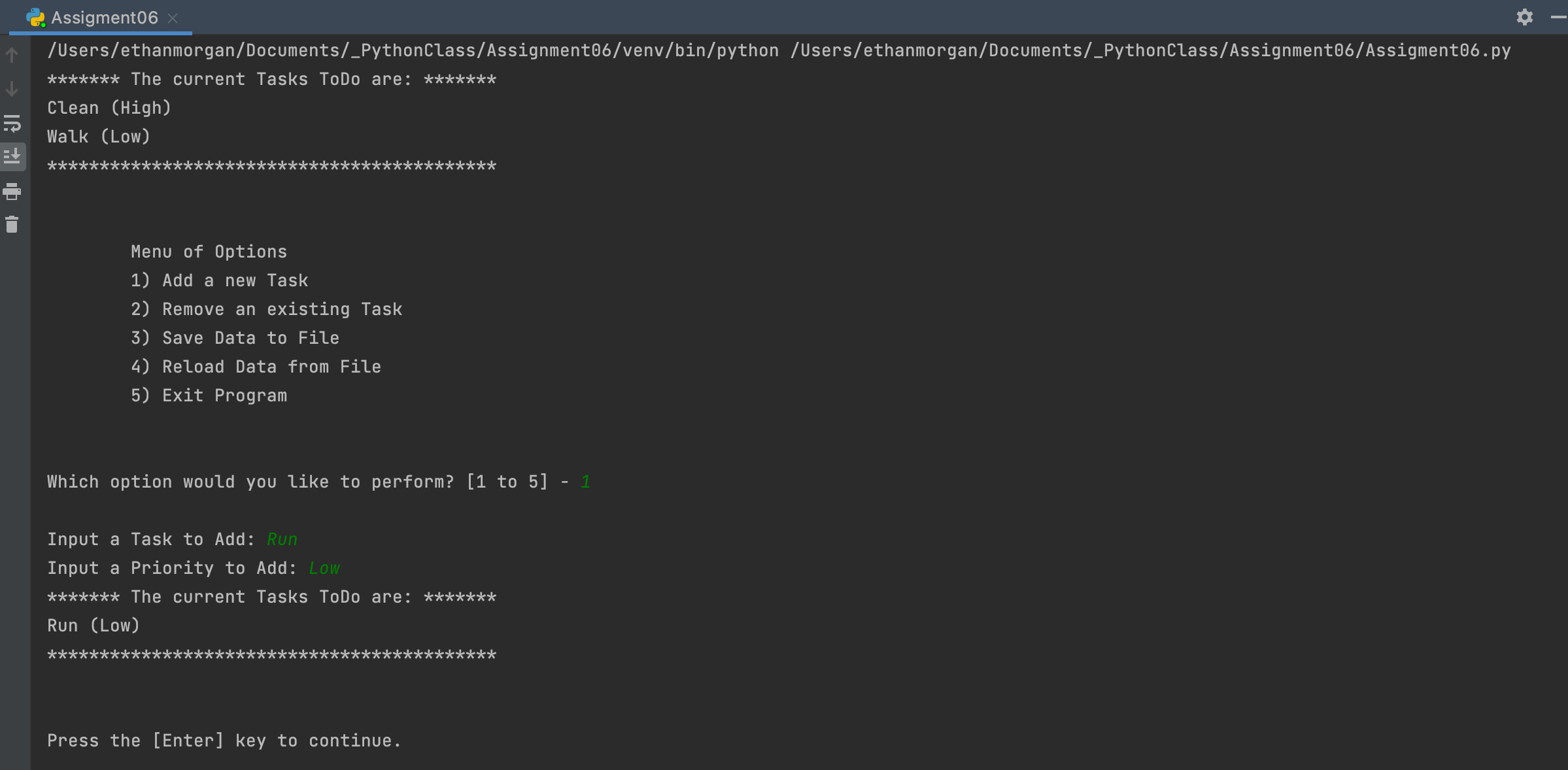


Figure Screenshot of the Script Working in PyCharm

Figure 3 shows a screenshot of the ToDoList script working in the Command OS/Shell. Again, only a portion of the working code was shown in the figure below for clarity.

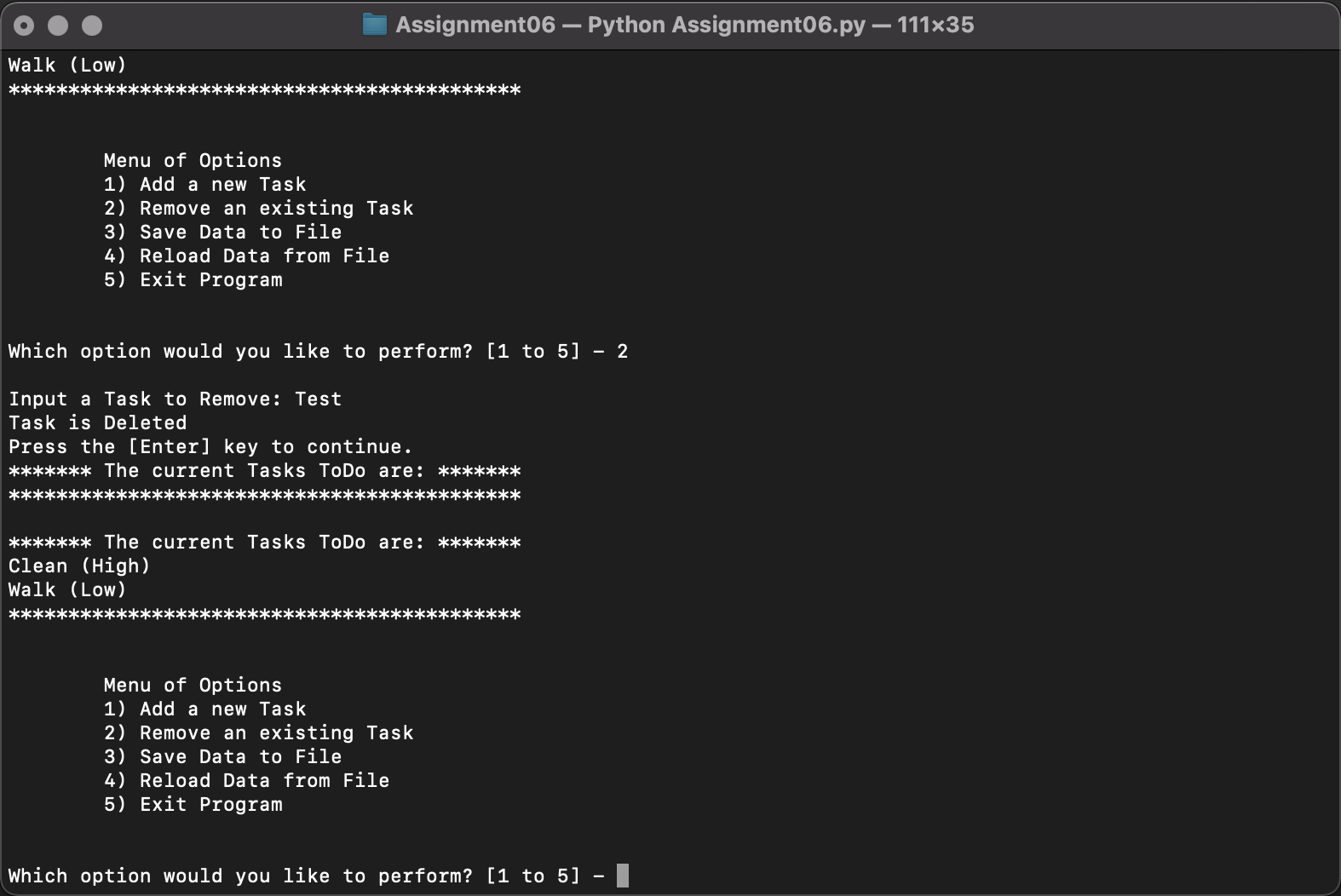


Figure Screenshot of the Script Working in the Command OS/Shell

Figure 4 shows a screenshot the text file output from the ToDoList script.

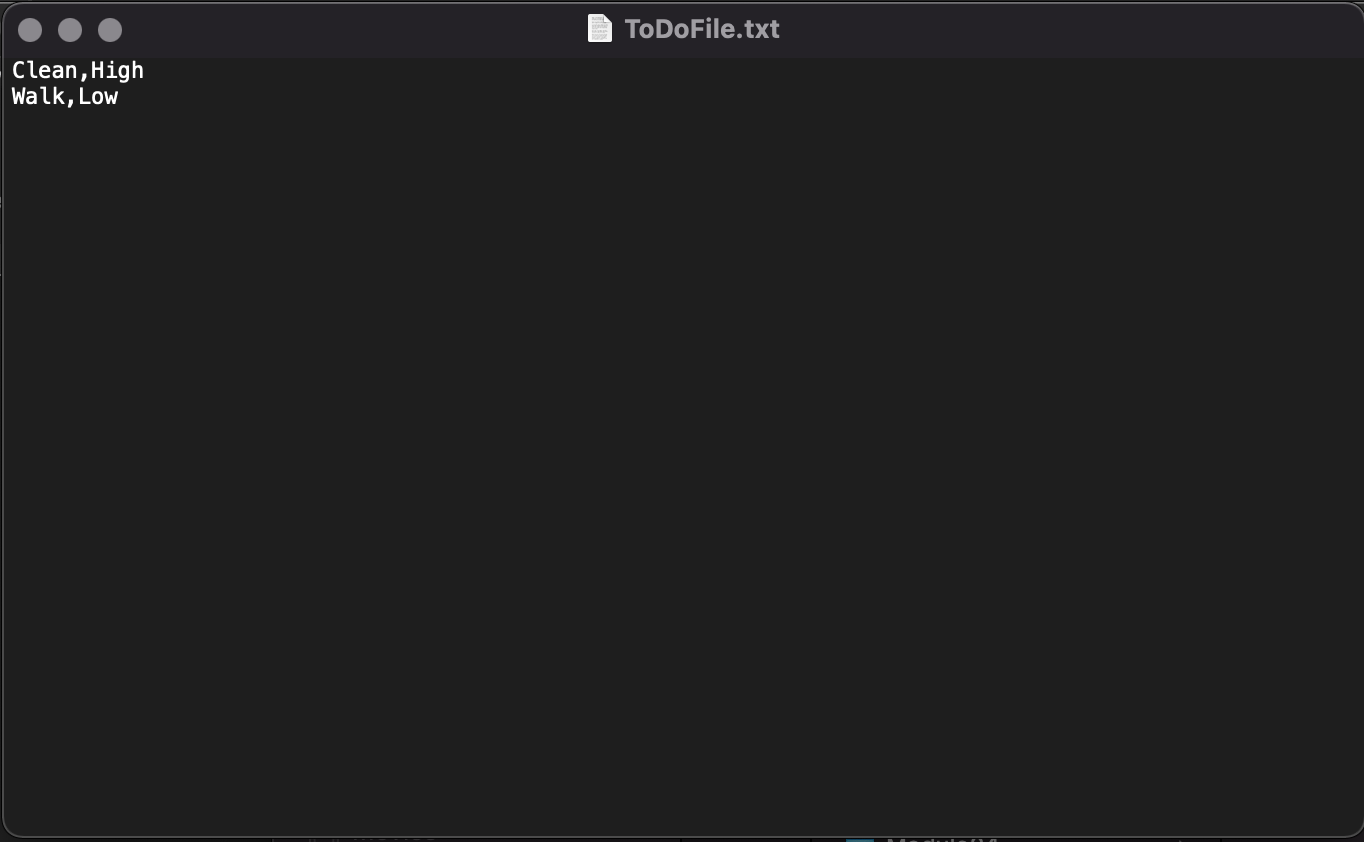


Figure Text File Output of To Do Task List

# Summary

Assignment 06 provided a deeper look into using Python for programming using functions. 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. However, I will say for me it was easier in Assignment 05 rather than 06, mostly because everything was in a linear approach.