Alex Jehle

August 8th, 2020

IT FDN 110 A

Assignment 05

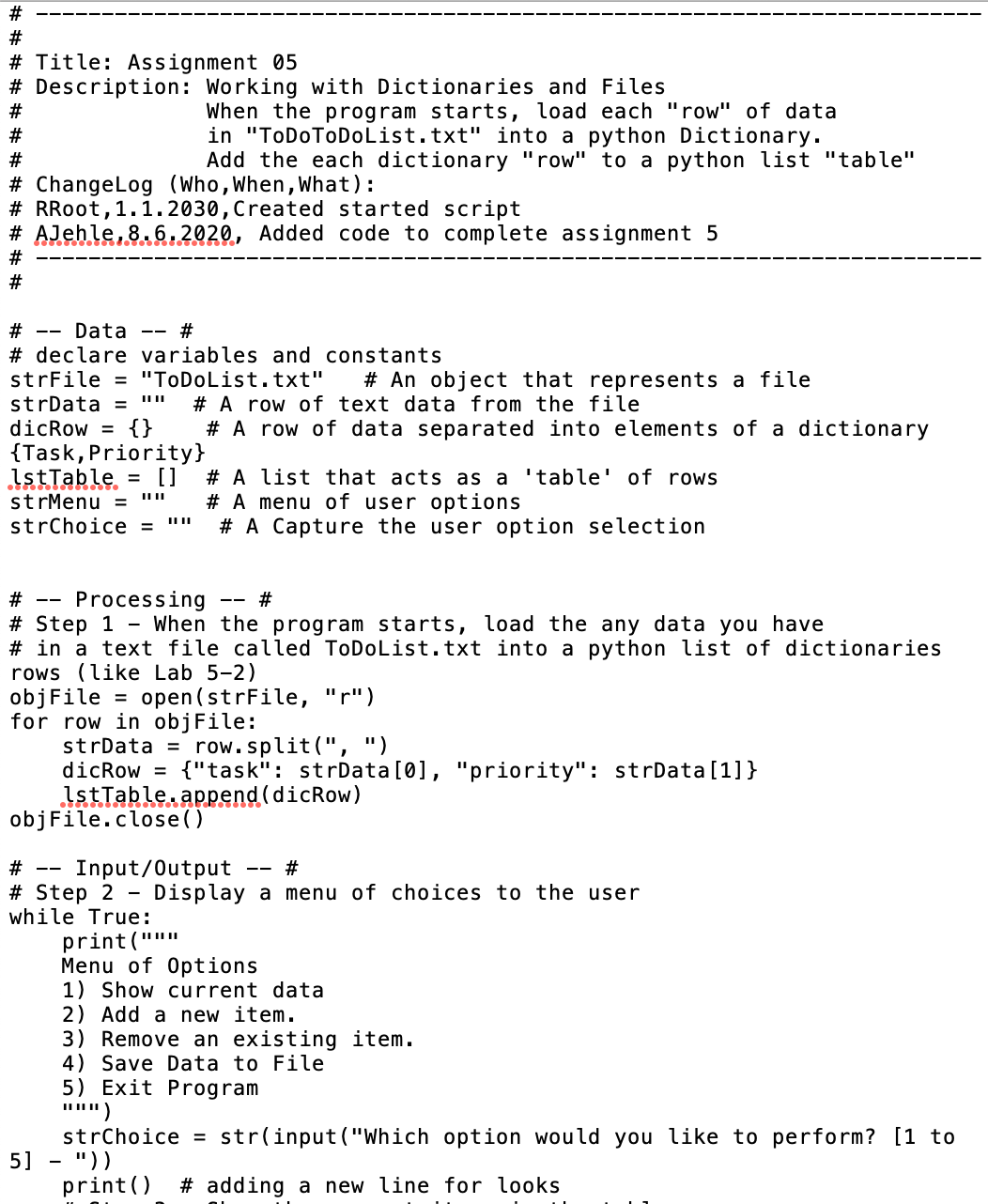
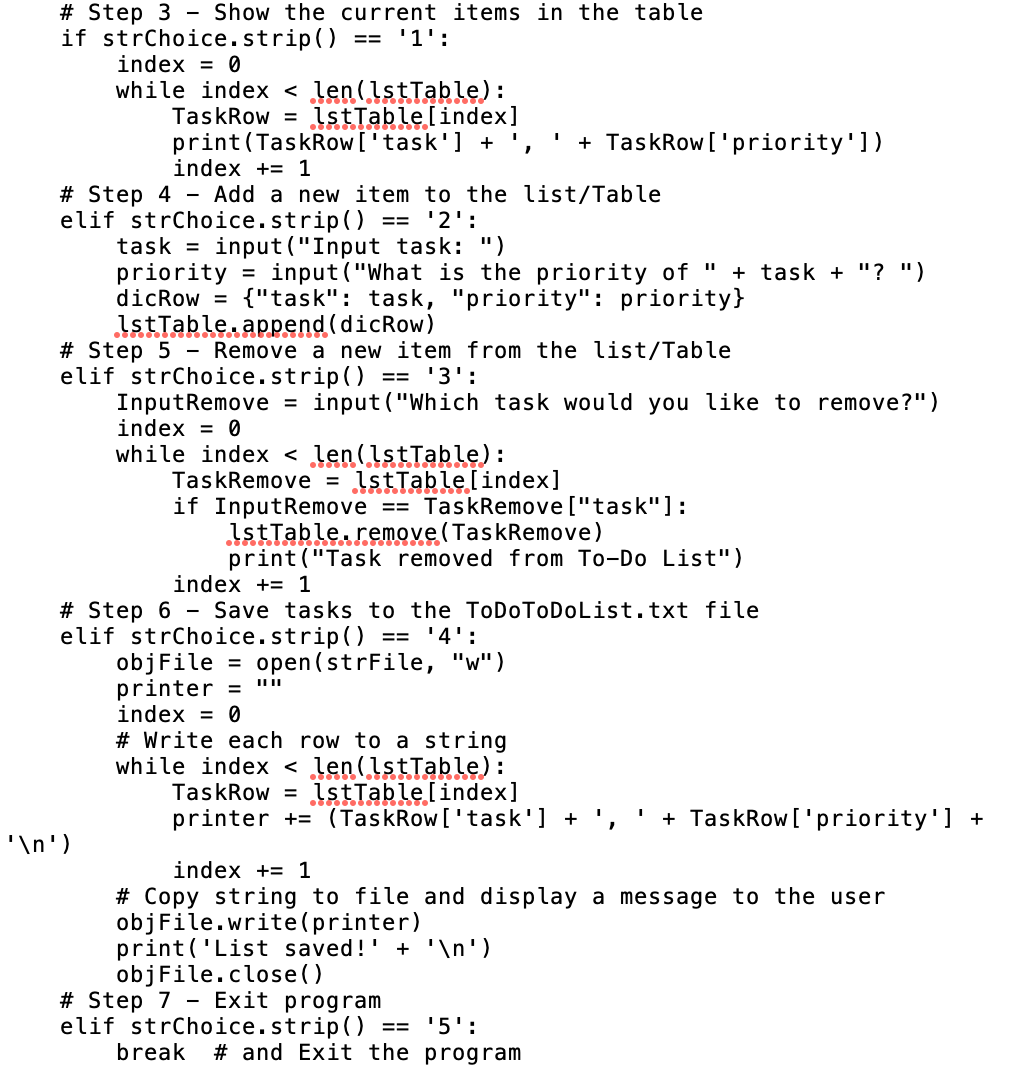
To Do List Script

# Introduction

The purpose of this program is to prioritize and update a to do list tasks based off of a menu selection and input from the user.

# Program

The program Assigment05\_starter.py (Figure 1) consists of one continuous script with no function breaks or additional files. It begins by opening the to do text file, ‘ToDoList.txt’. The program then enters into a while loop for the menu, prompting the user for one of five choices: displaying data, adding a new item, removing an existing item, saving data to the file, or exiting.

***Figure 1: ‘Assigment05\_starter.py’ Script***

## Displaying Data

If the user selects the option to display the data, each of the tasks on the to-do list is printed to the terminal as seen in Figure 2. This includes existing tasks and any changes the user has made since launching the program. The program then returns to the initial menu option.

## 

***Figure 2: Output of displayed data in PyCharm and the terminal***

## Adding Data

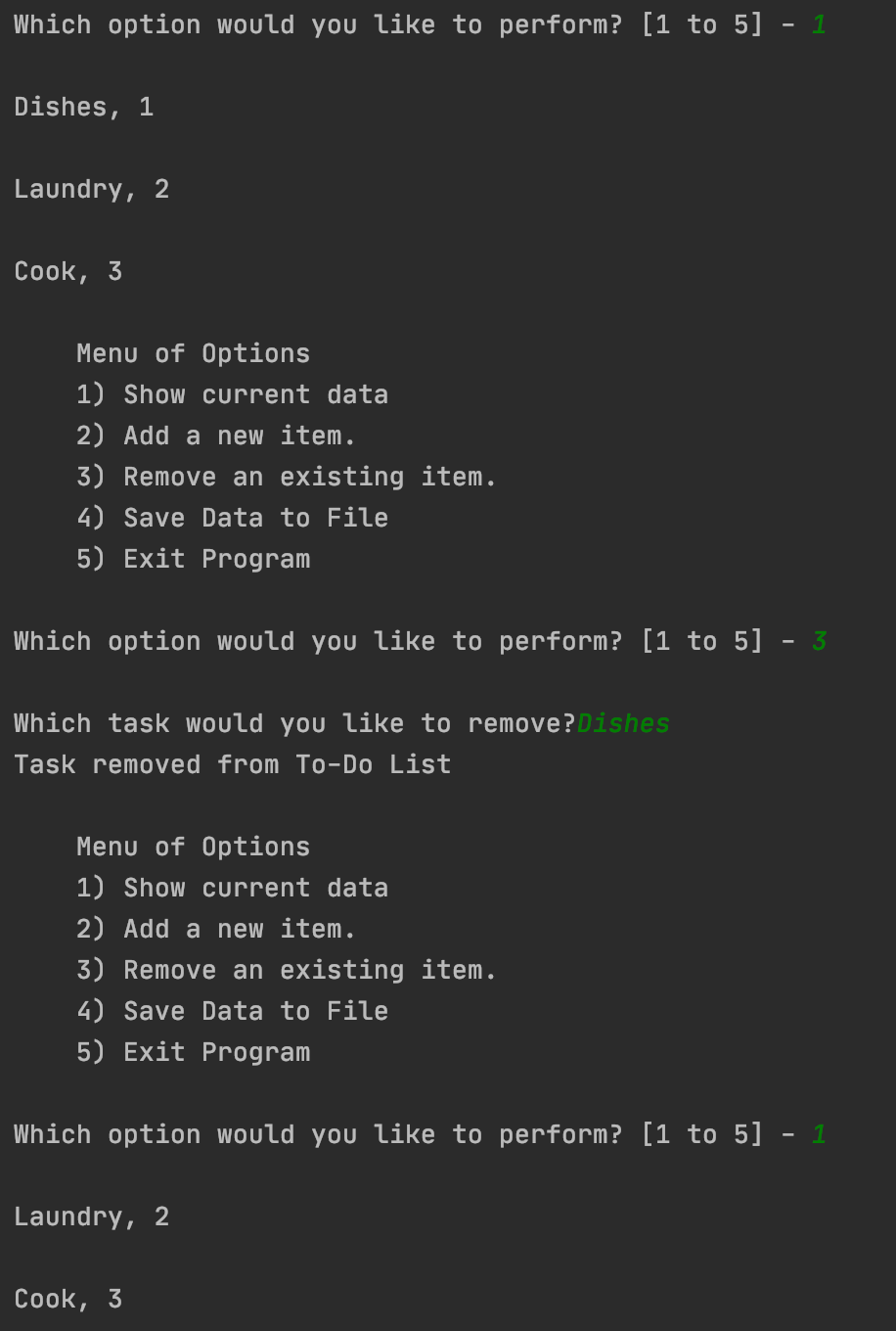
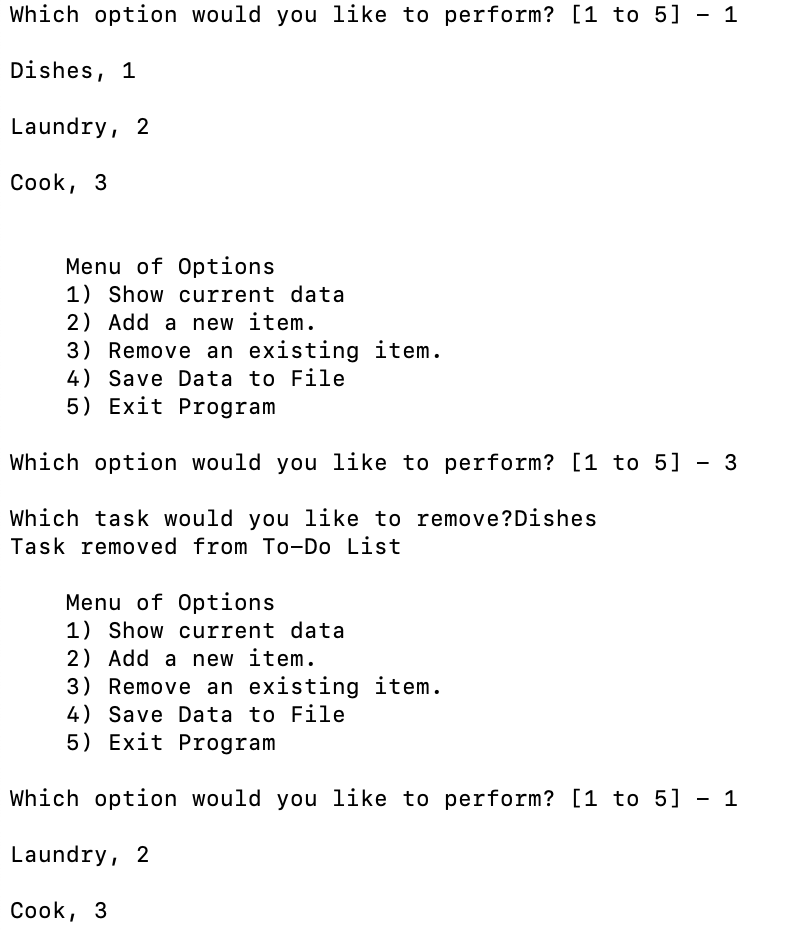
If the user selects the option to add a new item the program prompts the user to enter the task they wish to add, and then prompts for the priority. The program appends these items to the to-do list. The program then returns to the initial menu option. This is shown in Figure 3.

## 

***Figure 3: Output of added data in PyCharm and the terminal***

## Removing Data

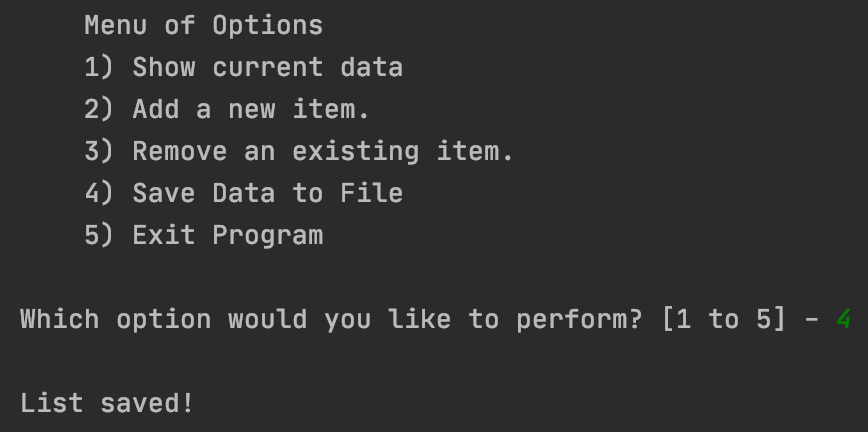
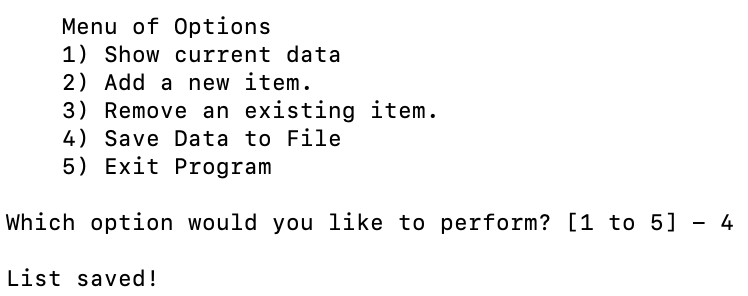
If the user selects the option to remove an item, the program prompts the user to enter the task they wish to delete. The program iterates through the program, removes that task from the to-do list, and displays a confirmation, as seen in Figure 4. The program then returns to the initial menu option.

***Figure 4: Output of removing data in PyCharm and the terminal***

## Saving to the File

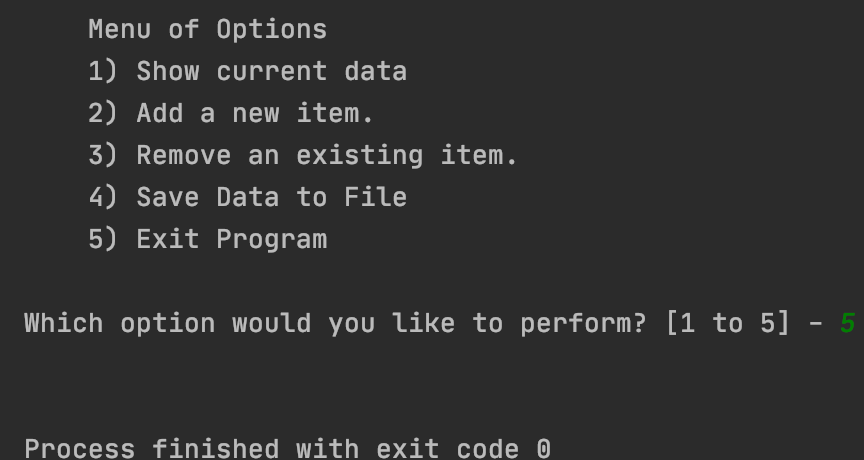
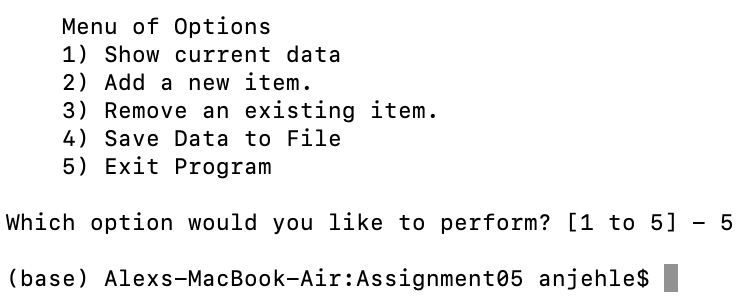
If the user selects the option to save then the program overwrites the current ‘ToDoList.txt’ with a copy of the data list read in at the beginning of the program and any changes applied. The program then displays a message that the list was saved and returns to the main menu. This process is shown in Figure 5.

***Figure 5: Output after the data is saved in PyCharm and the terminal***

## Exiting the Program

If the user selects the option to exit then the program ends, as seen in Figure 6.

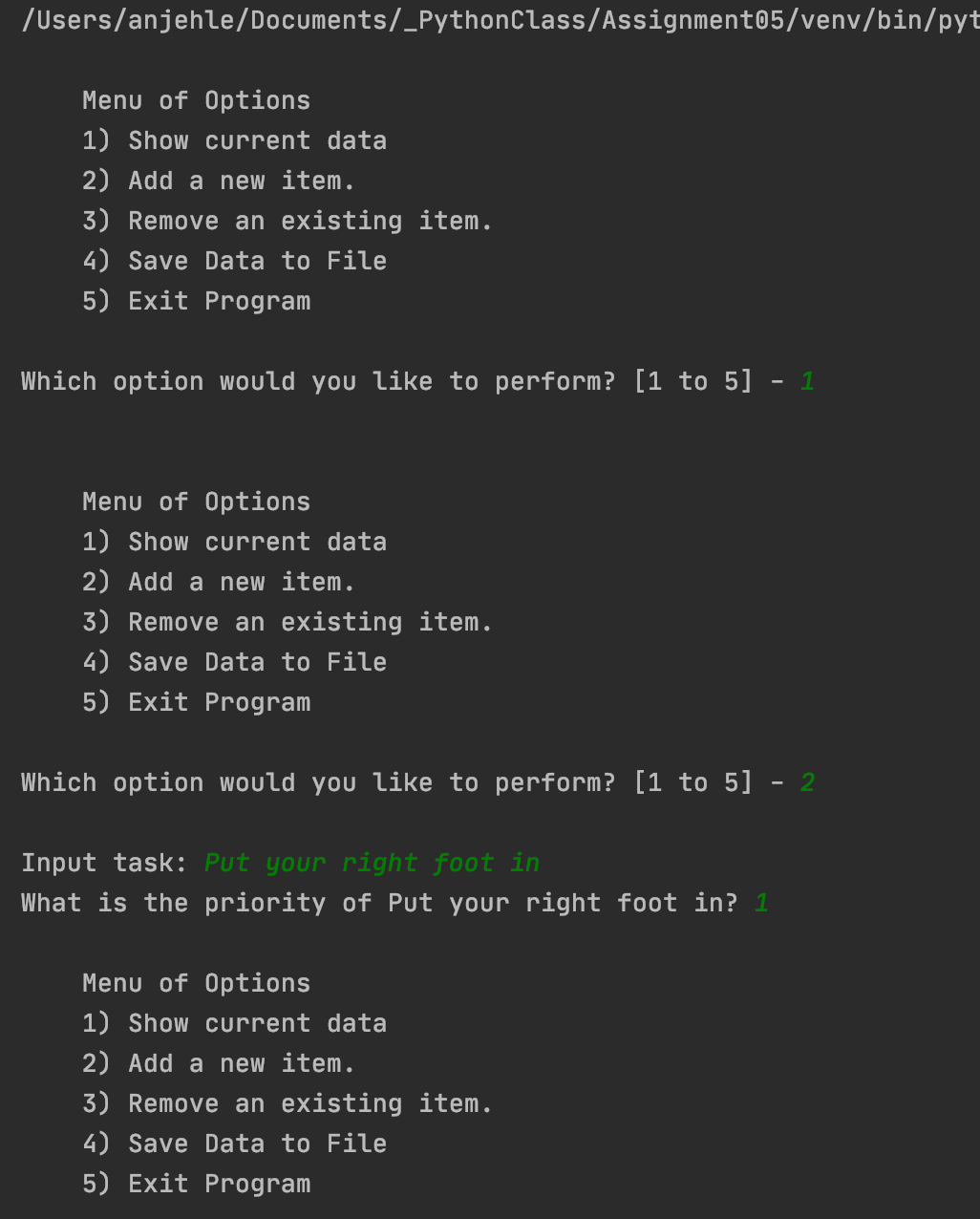
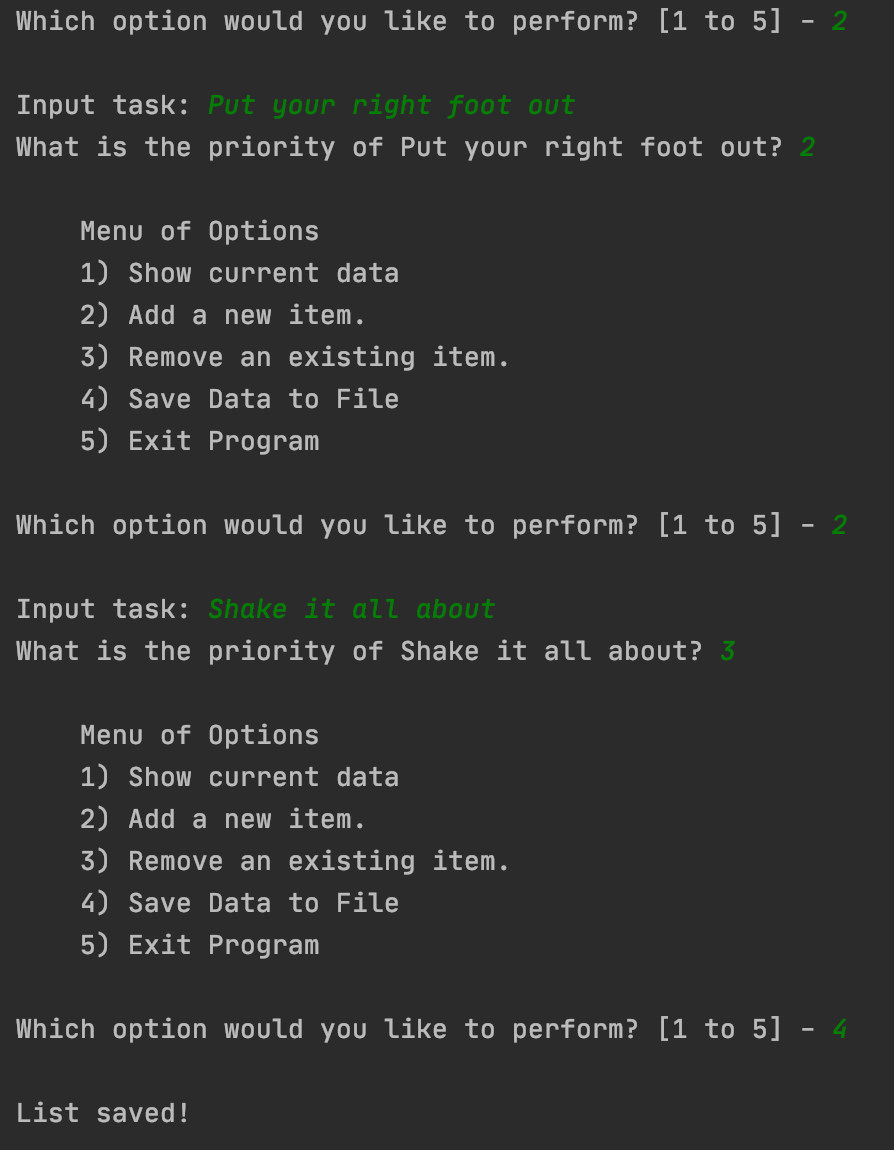
 

***Figure 6: Output after the exit option is selected in PyCharm and the terminal***

# Results

## Execution from PyCharm

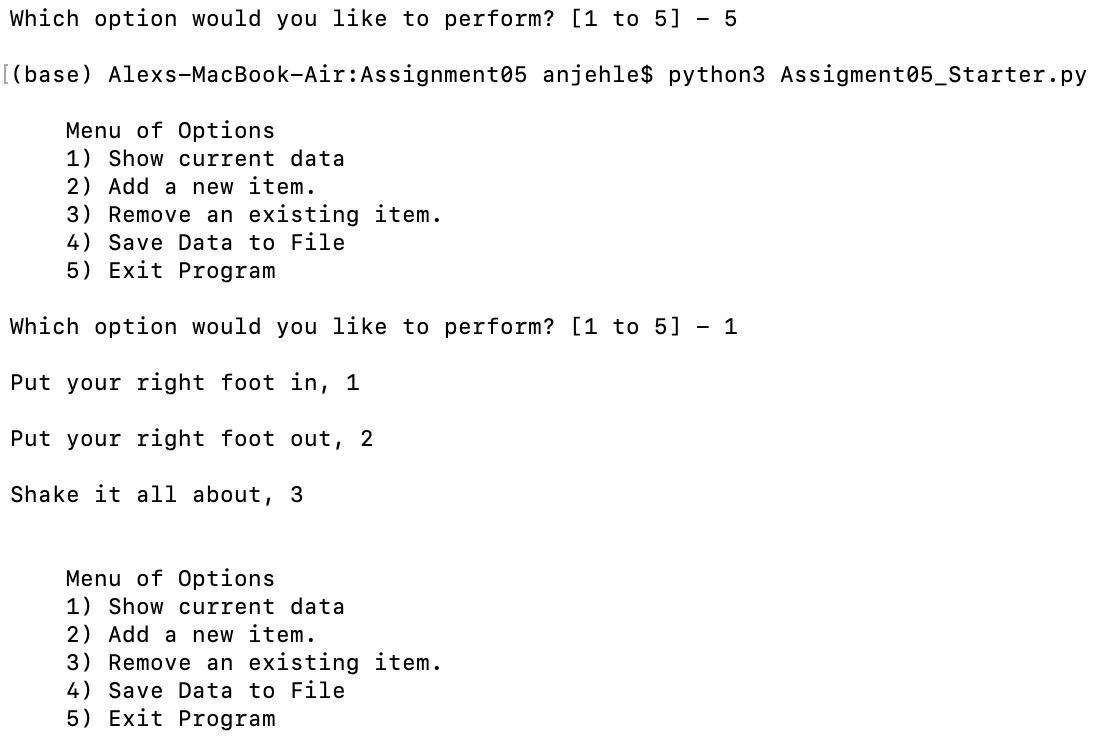
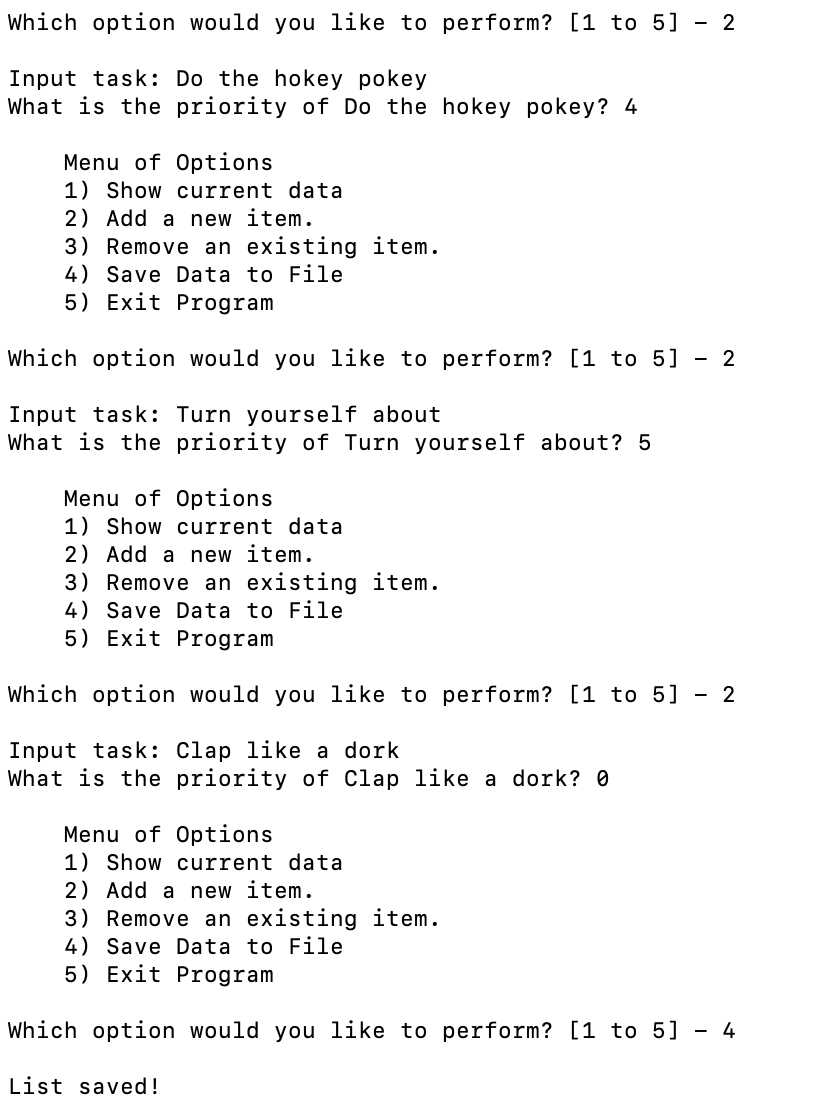
The program Assigment05\_Starter.py executed successfully on its first attempt from PyCharm. Three items were added and in each case, both inputs were taken in as strings and the proper processing and output was observed (Figure 7). This program did not take a measurable amount of time to execute.

***Figure 7: Execution of ‘Assigment05\_Starter.py’ in PyCharm***

## Execution from the terminal

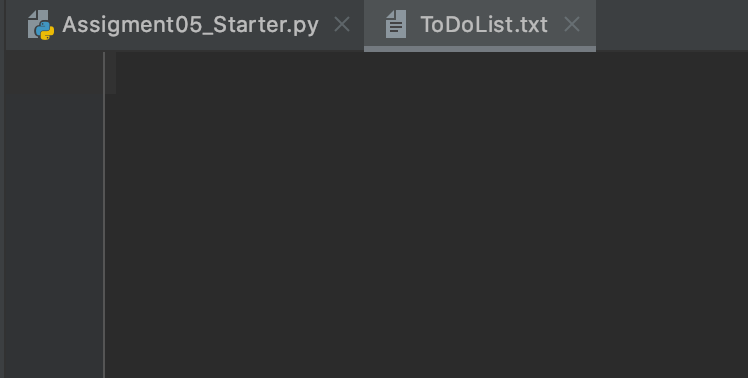
The program Assigment05\_Starter.py executed successfully on its first attempt from the terminal. Three items were added and in each case, each task was taken in as a string and the proper processing and output was observed (Figure 8). This program did not take a measurable amount of time to execute.

***Figure 8: Execution of ‘Assigment05\_Starter.py’ in the terminal***

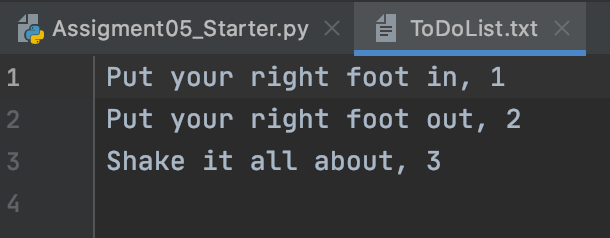
## Results

The program was launched in both cases with a blank text file ‘ToDoList.txt’, as seen in Figure 9.

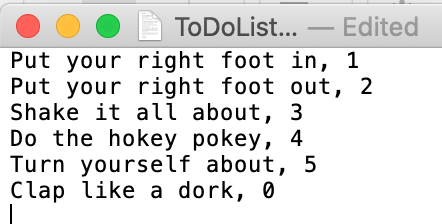
***Figure 9: ’ToDoList.txt’ before program execution***

The program ran successfully through PyCharm first. The program opened the file ‘ToDoList.txt’ and added three tasks and their priorities, as seen in Figure 10.



***Figure 10: Results stored in ‘ToDoList.txt’ after PyCharm execution***

The program ran successfully through the terminal second. The program opened the file ‘ToDoList.txt’ and appended three items and their values to the end, as seen in Figure 11.



***Figure 11: Results stored in ‘ToDoList.txt’ after terminal execution***

# Summary

The program consists of 86 lines including comments and headers. The intended result to update the to-do list in one living file was achieved with no unintended consequences.