Sanghee Kang
February 15, 2022
IT FND 110 Foundations of Programming: Python
Assignment 05
GitHub URL: https://github.com/bsb218218/IntroToProg-Python

To Do List Python Script

1. Introduction

In this paper, I will describe how I created a To Do List Python script, which shows the content in an existing text file, displays a menu of choices to the user (five options), shows the current items in the text file (option 1), adds a new item to the table by capturing user's data (option 2), removes an existing item from the table (option 3), saves tasks to the text file (option 4), and exits the program (option 5), using PyCharm. In the following section, I will describe each step that I took to complete the assignment.

2. Creating the To Do List Python Script

2.1. Creating a sub-folder

Following the directions in the assignment, I created a folder (Assignment05_SangheeKang) inside of the "_PythonClass" folder.

2.2. Creating a new Project in PyCharm

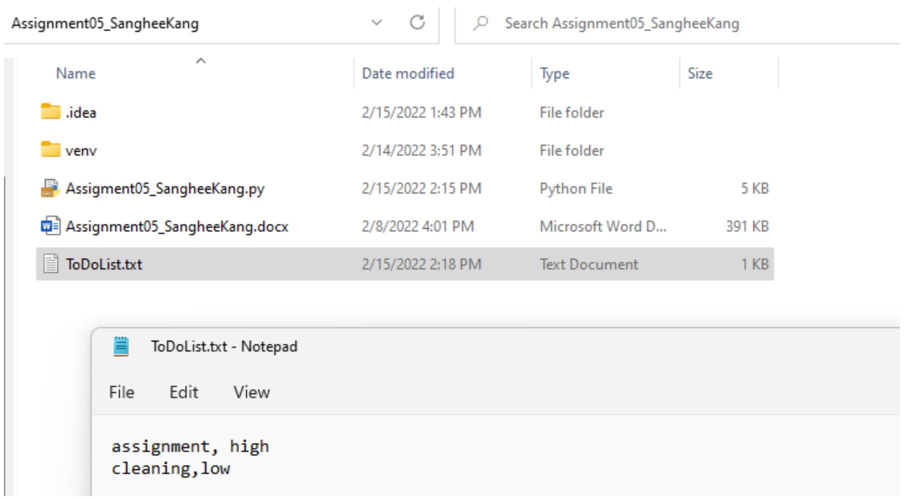
I created a new Project in Pycharm (File-New Project). For the location, I used "C:_PythonClass\Assignment05_SangheeKang" and selected "New environment using Virtualenv".

2.3. Adding the starter Python file (Assignment05_Starter.py) and writing code to the script

In the new project, I added the starter Python file and renamed it as "ToDoList_Assignment05_SangheeKang". In the new file, I modified the script header, specifically the last line by adding my name and the date.

2.3.1. # Step 1: Loading a text file when the program starts

In the processing section, the first step was to load an existing text file and to convert the content in the text file to a python list of dictionaries rows. So, I created a text file with the content (assignment, high; cleaning, low; see Figure 1). To read the text file, I used the open(), 'r' (read), and the close() methods. Because there were multiple rows in the file, I used the for loop. In addition, I added codes in order to change the strings from the text file to a list (lstRow = row.split (",") and then to dictionaries rows (dicRow = {"Task": lstRow[0], "Priority": lstRow[1].strip()}). To add a row to the existing list, I used the append() method. See Figure 2 for the code for Step 1.



Assignment05_SangheeKang

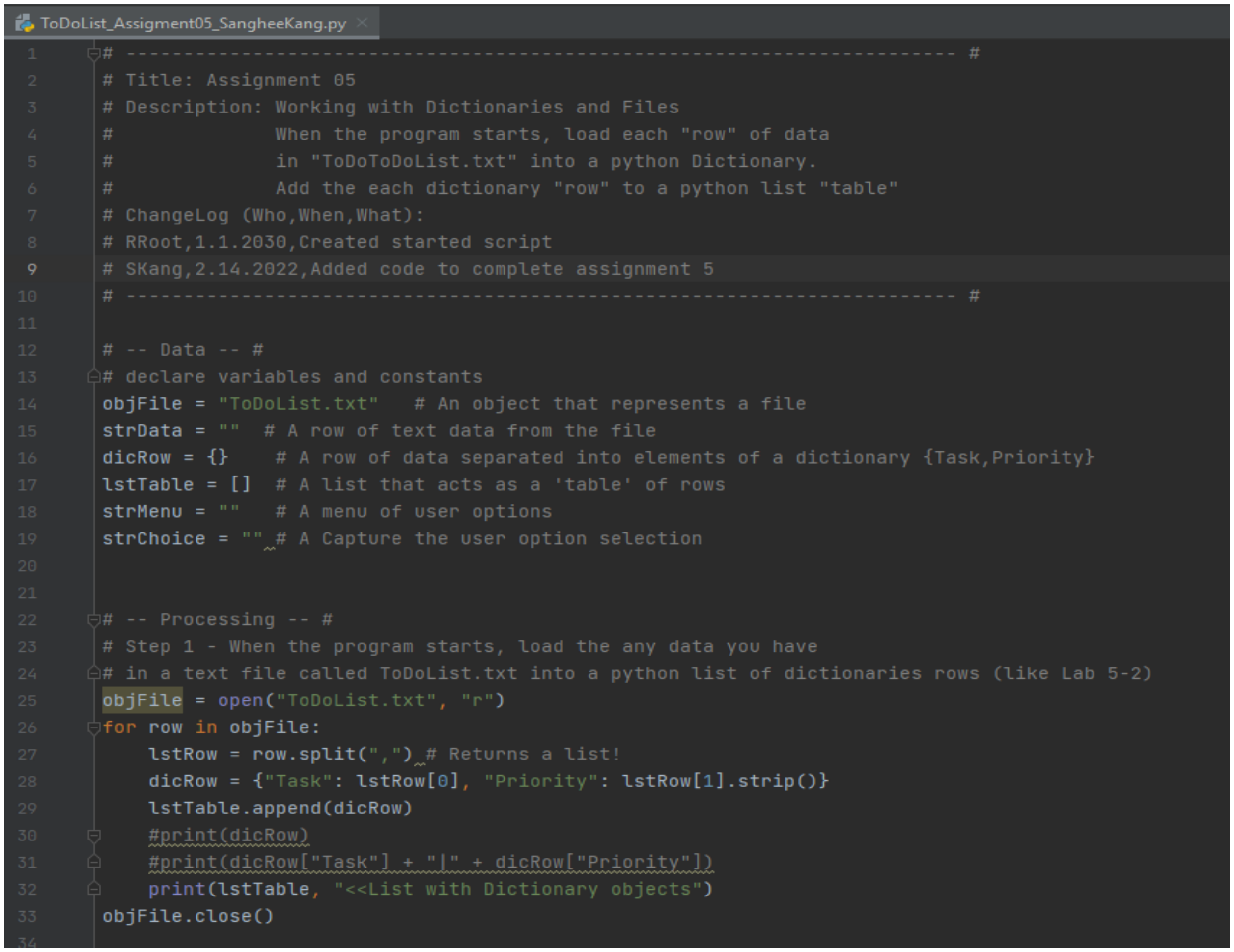
Search Assignment05_SangheeKang

Name	Date modified	Type	Size
.idea	2/15/2022 1:43 PM	File folder	
venv	2/14/2022 3:51 PM	File folder	
Assigment05_SangheeKang.py	2/15/2022 2:15 PM	Python File	5 KB
Assignment05_SangheeKang.docx	2/8/2022 4:01 PM	Microsoft Word D...	391 KB
ToDoList.txt	2/15/2022 2:18 PM	Text Document	1 KB

ToDoList.txt - Notepad

```
assignment, high
cleaning,low
```

Figure 1. A screenshot of the content of an existing ToDoList text file

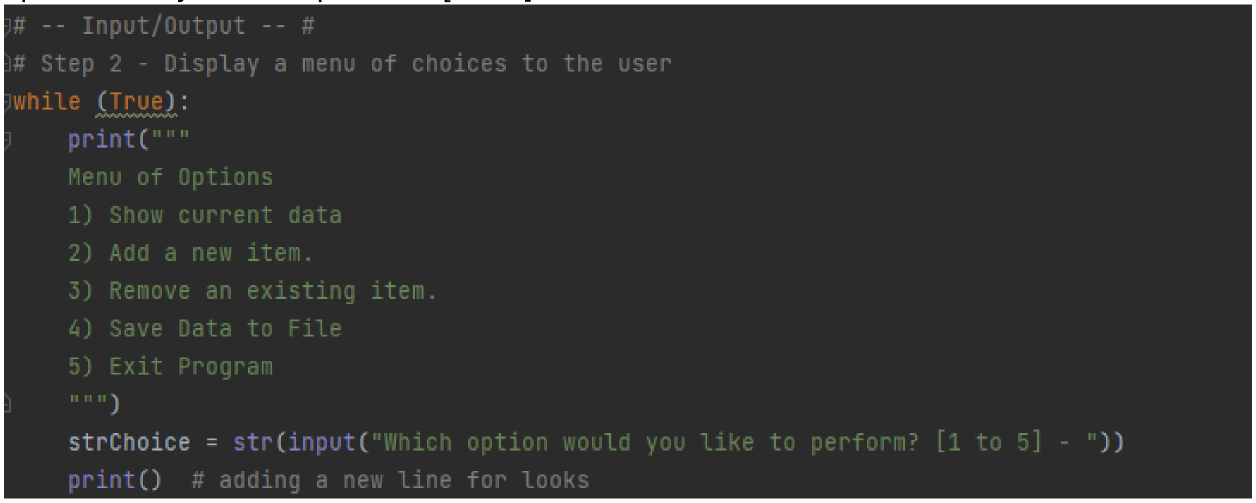


ToDoList_Assigment05_SangheeKang.py

```python
# ------------------------------------------------------------------------ #
# 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
# SKang,2.14.2022,Added code to complete assignment 5
# ------------------------------------------------------------------------ #

# -- Data -- #
# declare variables and constants
objFile = "ToDoList.txt"   # 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


# -- 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)
objFile = open("ToDoList.txt", "r")
for row in objFile:
    lstRow = row.split(",") # Returns a list!
    dicRow = {"Task": lstRow[0], "Priority": lstRow[1].strip()}
    lstTable.append(dicRow)
    #print(dicRow)
    #print(dicRow["Task"] + "|" + dicRow["Priority"])
    print(lstTable, "<<List with Dictionary objects")
objFile.close()
```

Figure 2. The To Do List Python Code (The Script Header, Data section, Step 1) in PyCharm

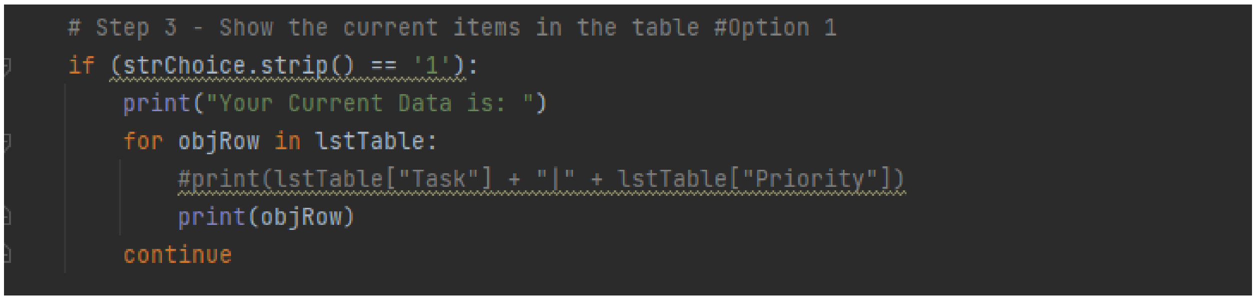
2.3.2. # Step 2: Display a menu of choices to the user (Figure 3)
In Step 2, in order to display a menu of choices to the user, I used the while loop and “Menu of Options” (which had three options) was added using the print() function, so that “Menu of Options” can appear repeatedly once a task is done. Following the menu, a prompt “Which option would you like to perform? [1 to 5]: ” was added to elicit the user’s choice.



```
# -- 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
```

Figure 3. The To Do List Python Code (Step 2) in PyCharm

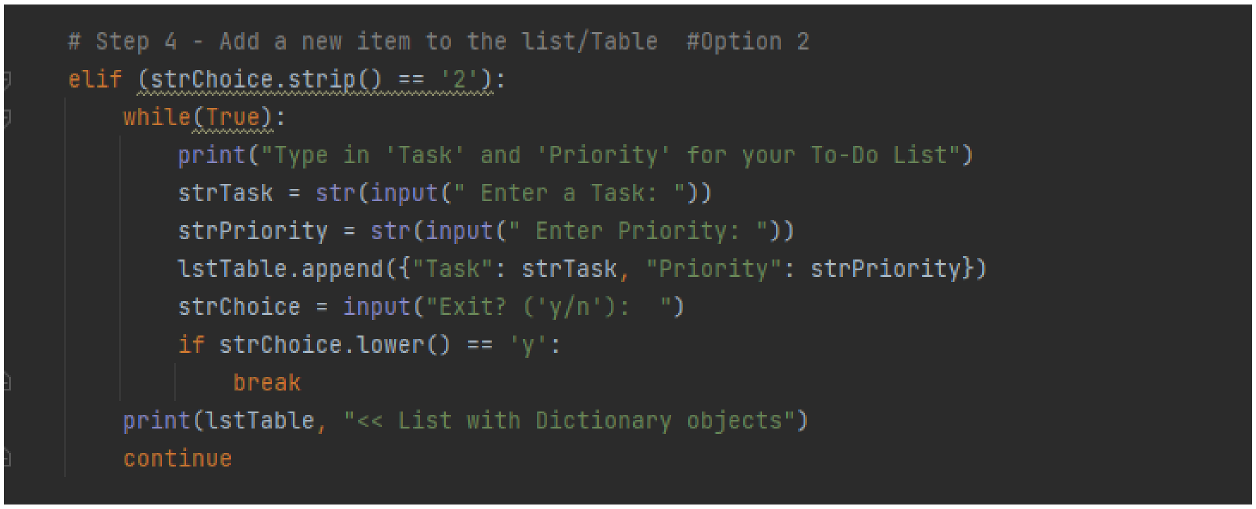
2.3.3. # Step 3: Show the current items in the table (Option 1; Figure 4)
For Steps 3-7, I used the if-elif-else statements. In Step 3, the if statement was used for Option 1, which was prompted when the user asked to show the current data saved in the text file. To display multiple rows, the for loop was used so that all data could be shown.



```
    # Step 3 - Show the current items in the table #Option 1
    if (strChoice.strip() == '1'):
        print("Your Current Data is: ")
        for objRow in lstTable:
            #print(lstTable["Task"] + "|" + lstTable["Priority"])
            print(objRow)
        continue
```

Figure 4. The To Do List Python Code (Step 3) in PyCharm

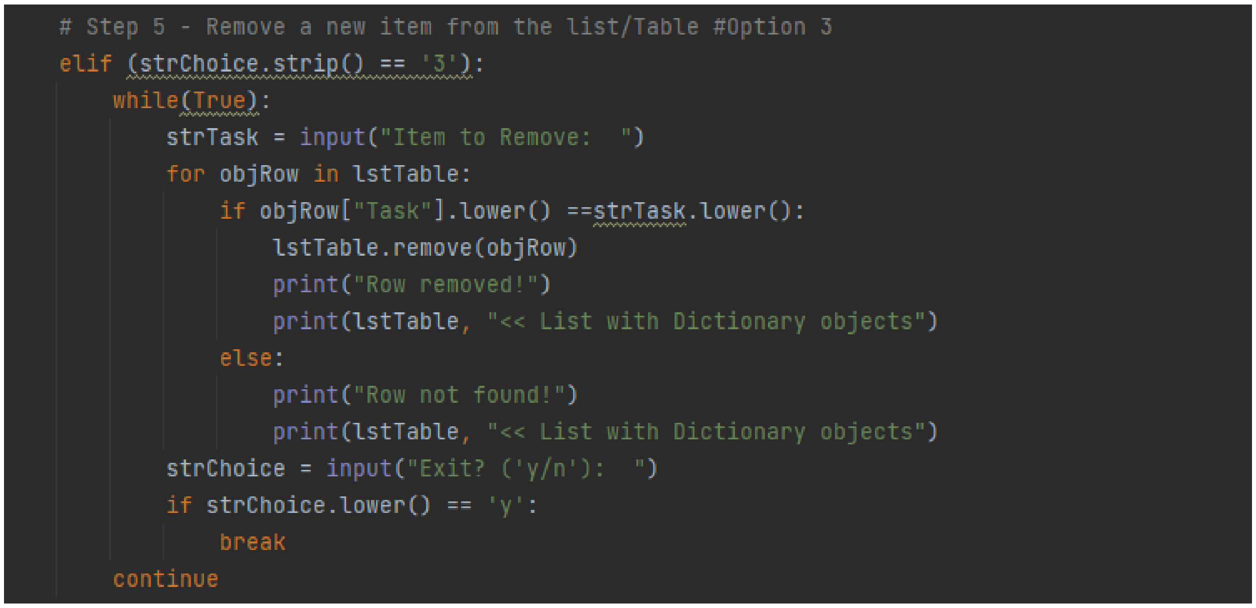
2.3.4. # Step 4: Add a new item to the list/Table (Option 2; Figure 5)
In Step 4, in order to add a new item to the list whenever the user chooses that option, directions—Type in a ‘Task’ and ‘Priority’ for your To-Do List; Enter a Task: ; Enter Priority: —were added using the print(), the str(), and the input() functions. Also, to add data to the existing dictionaries rows, the append() method was used. The while loop was used so that the user can add multiple items to the dictionaries rows. By adding strChoice = input(“Exit? (‘y/n’): ”) and the if statement, the code allowed the user to choose to exit or continue to add items.



```
# Step 4 - Add a new item to the list/Table  #Option 2
elif (strChoice.strip() == '2'):
    while(True):
        print("Type in 'Task' and 'Priority' for your To-Do List")
        strTask = str(input(" Enter a Task: "))
        strPriority = str(input(" Enter Priority: "))
        lstTable.append({"Task": strTask, "Priority": strPriority})
        strChoice = input("Exit? ('y/n'):  ")
        if strChoice.lower() == 'y':
            break
    print(lstTable, "<< List with Dictionary objects")
    continue
```

Figure 5. The To Do List Python Code (Step 4) in PyCharm

2.3.5. # Step 5: Remove an existing item from the list/Table (Option 3; Figure 6)
For Step 5, because the program needs to check all rows in the table to locate the item that the user wants to remove, the for loop was used. Also, in order for the program to check whether the item that the user entered matched any item in the list, the if-else statements were used. When the task is accomplished, “Row removed” is shown and when no item is found, “Row not found” appeared. Again, strChoice = input(“Exit? (‘y/n’): ”) and the if statement were used to elicit the user to choose to exit or continue to remove items.

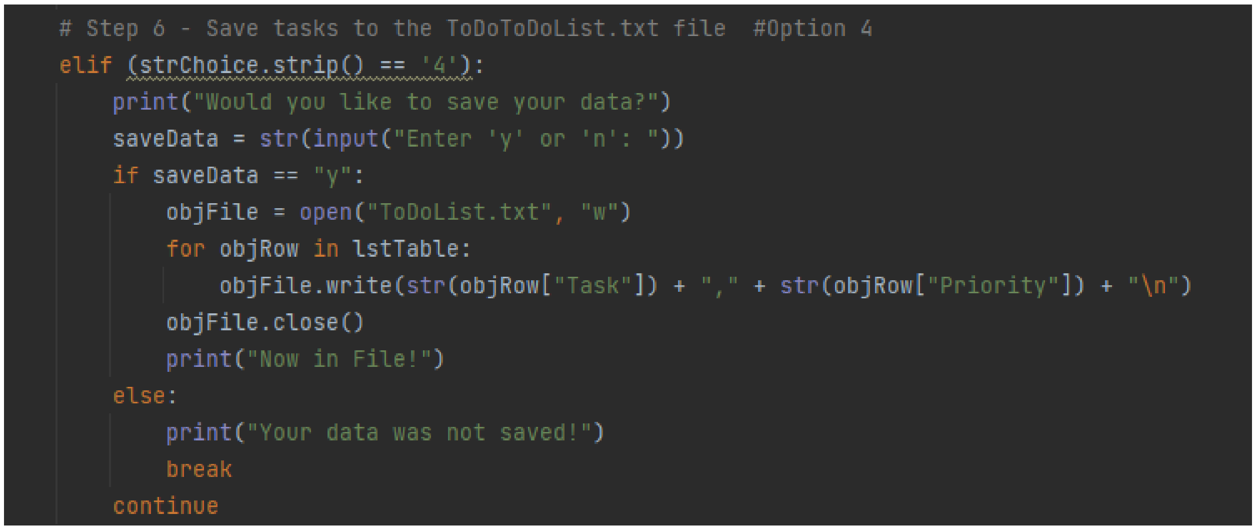


```
# Step 5 - Remove a new item from the list/Table #Option 3
elif (strChoice.strip() == '3'):
    while(True):
        strTask = input("Item to Remove:  ")
        for objRow in lstTable:
            if objRow["Task"].lower() ==strTask.lower():
                lstTable.remove(objRow)
                print("Row removed!")
                print(lstTable, "<< List with Dictionary objects")
            else:
                print("Row not found!")
                print(lstTable, "<< List with Dictionary objects")
        strChoice = input("Exit? ('y/n'):  ")
        if strChoice.lower() == 'y':
            break
    continue
```

Figure 6. The To Do List Python Code (Step 5) in PyCharm

2.3.6. # Step 6: Save tasks to the ToDoToDoList.txt file (Option 4; Figure 7)

To save the data when the user wants to do so, the elif statement (for Option 4), the print(), the input(), and str() functions were first used to provide directions ("Would you like to save your data?"; Enter 'y' or 'n':). As the program needs to perform a task based on the user's choice (i.e., 'y' or 'n'), the if and else statements were used. When the user selects saving the data (entered in 'y'), the program saved the data using functions such as open() to open a connection to a file, objF.write() to write the data into the text file, and objF.close() in order to close the connection to the file. Also, as there were multiple rows, the for loop was used. If the user chooses 'n', the program does not save the data into a file and shows the message "Your data was not saved!" in the else statement.

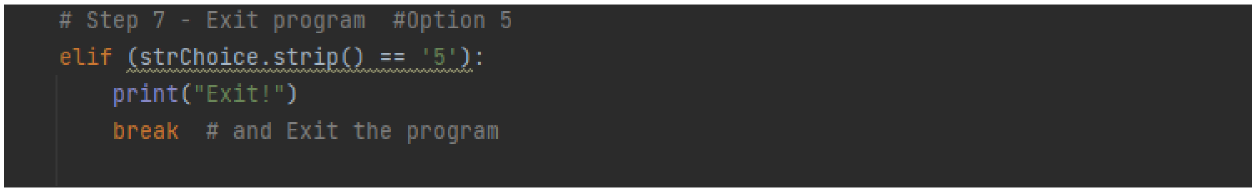


```
    # Step 6 - Save tasks to the ToDoToDoList.txt file  #Option 4
    elif (strChoice.strip() == '4'):
        print("Would you like to save your data?")
        saveData = str(input("Enter 'y' or 'n': "))
        if saveData == "y":
            objFile = open("ToDoList.txt", "w")
            for objRow in lstTable:
                objFile.write(str(objRow["Task"]) + "," + str(objRow["Priority"]) + "\n")
            objFile.close()
            print("Now in File!")
        else:
            print("Your data was not saved!")
            break
        continue
```

Figure 7. The To Do List Python Code (Step 6) in PyCharm

2.3.7. # Step 7: Exit program (Option 5; Figure 8)

In the last step, the elif statement was used. If the user chooses 5, then the user exits from the program and the program shows "Exit!".



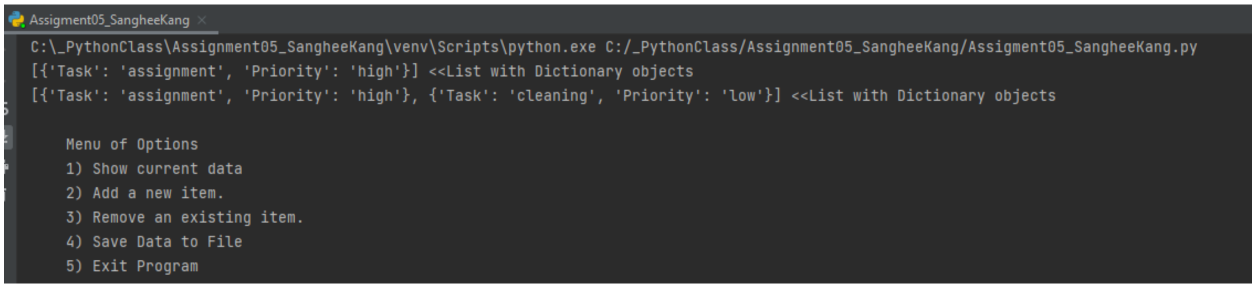
```
    # Step 7 - Exit program  #Option 5
    elif (strChoice.strip() == '5'):
        print("Exit!")
        break  # and Exit the program
```

Figure 8. The To Do List Python Code (Step 7) in PyCharm

2.4. Running the ToDoList Python script and verifying that it worked

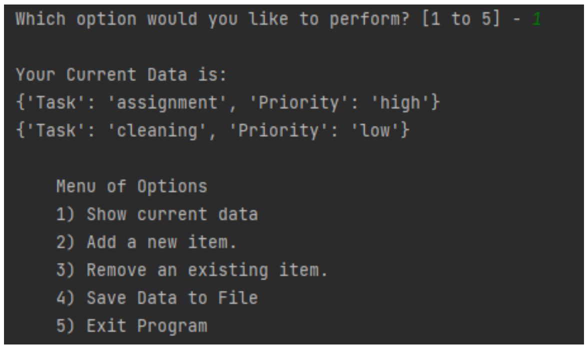
To run my Python script in PyCharm, I right-clicked and selected "Run Assignment05_SangheeKang." I chose all five options one by one to see if all of them perform its task successfully. In the existing text file, I had two tasks (assignment: high; cleaning: low) and added two tasks (ordering a dresser: mid; doing laundry: mid) using the program and removed the task "assignment". When I am done with running all functions in the code, I checked output (Figures 9-14) as well as the text file saved in the same folder. As shown in

Figure 15, the file only contained the three remaining tasks. Therefore, my python code performed its task successfully.



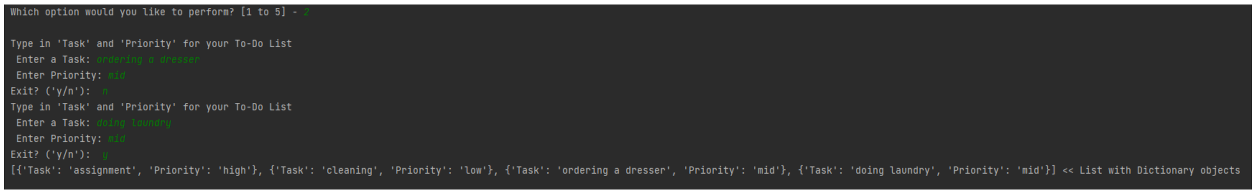
```
Assigment05_SangheeKang
C:\_PythonClass\Assignment05_SangheeKang\venv\Scripts\python.exe C:/_PythonClass/Assignment05_SangheeKang/Assigment05_SangheeKang.py
[{'Task': 'assignment', 'Priority': 'high'}] <<List with Dictionary objects
[{'Task': 'assignment', 'Priority': 'high'}, {'Task': 'cleaning', 'Priority': 'low'}] <<List with Dictionary objects

    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
```

Figure 9. Final output of my To Do List Python script in PyCharm (Steps 1, 2)



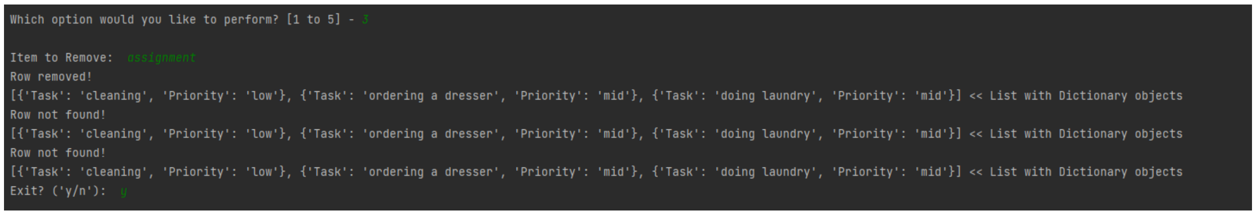
```
Which option would you like to perform? [1 to 5] - 1

Your Current Data is:
{'Task': 'assignment', 'Priority': 'high'}
{'Task': 'cleaning', 'Priority': 'low'}

    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
```

Figure 10. Final output of my To Do List Python script in PyCharm (Step 3-Option 1)



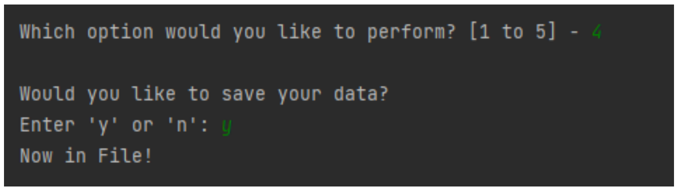
```
Which option would you like to perform? [1 to 5] - 2

Type in 'Task' and 'Priority' for your To-Do List
 Enter a Task: ordering a dresser
 Enter Priority: mid
Exit? ('y/n'):  n
Type in 'Task' and 'Priority' for your To-Do List
 Enter a Task: doing laundry
 Enter Priority: mid
Exit? ('y/n'):  y
[{'Task': 'assignment', 'Priority': 'high'}, {'Task': 'cleaning', 'Priority': 'low'}, {'Task': 'ordering a dresser', 'Priority': 'mid'}, {'Task': 'doing laundry', 'Priority': 'mid'}] << List with Dictionary objects
```

Figure 11. Final output of my To Do List Python script in PyCharm (Step 4-Option 2)



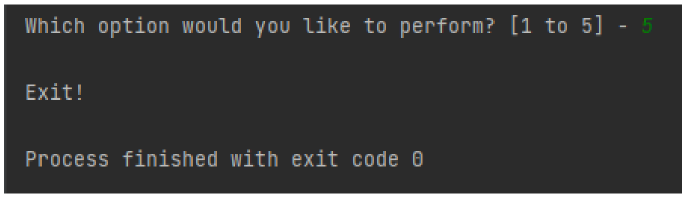
```
Which option would you like to perform? [1 to 5] - 3

Item to Remove:  assignment
Row removed!
[{'Task': 'cleaning', 'Priority': 'low'}, {'Task': 'ordering a dresser', 'Priority': 'mid'}, {'Task': 'doing laundry', 'Priority': 'mid'}] << List with Dictionary objects
Row not found!
[{'Task': 'cleaning', 'Priority': 'low'}, {'Task': 'ordering a dresser', 'Priority': 'mid'}, {'Task': 'doing laundry', 'Priority': 'mid'}] << List with Dictionary objects
Row not found!
[{'Task': 'cleaning', 'Priority': 'low'}, {'Task': 'ordering a dresser', 'Priority': 'mid'}, {'Task': 'doing laundry', 'Priority': 'mid'}] << List with Dictionary objects
Exit? ('y/n'):  y
```

Figure 12. Final output of my To Do List Python script in PyCharm (Step 5-Option 3)



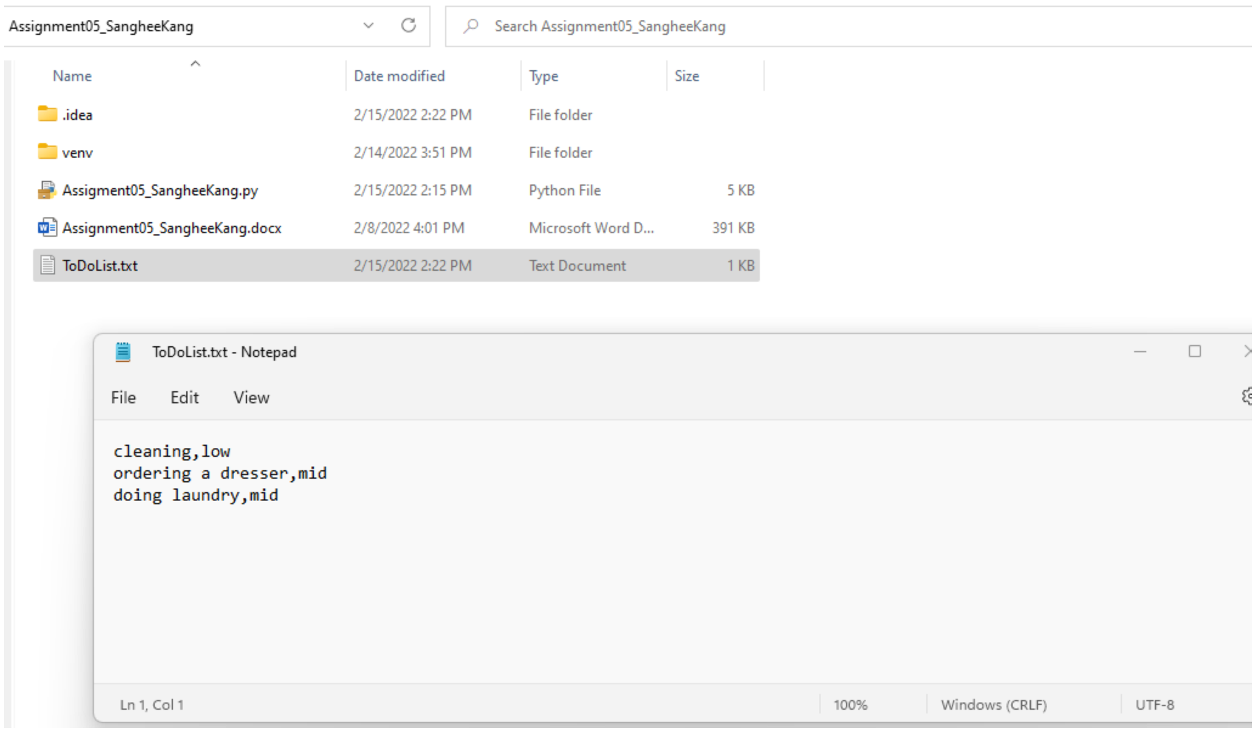
```
Which option would you like to perform? [1 to 5] - 4

Would you like to save your data?
Enter 'y' or 'n': y
Now in File!
```

Figure 13. Final output of my To Do List Python script in PyCharm (Step 6-Option 4)



```
Which option would you like to perform? [1 to 5] - 5

Exit!

Process finished with exit code 0
```

Figure 14. Final output of my To Do List Python script in PyCharm (Step 7-Option 5)

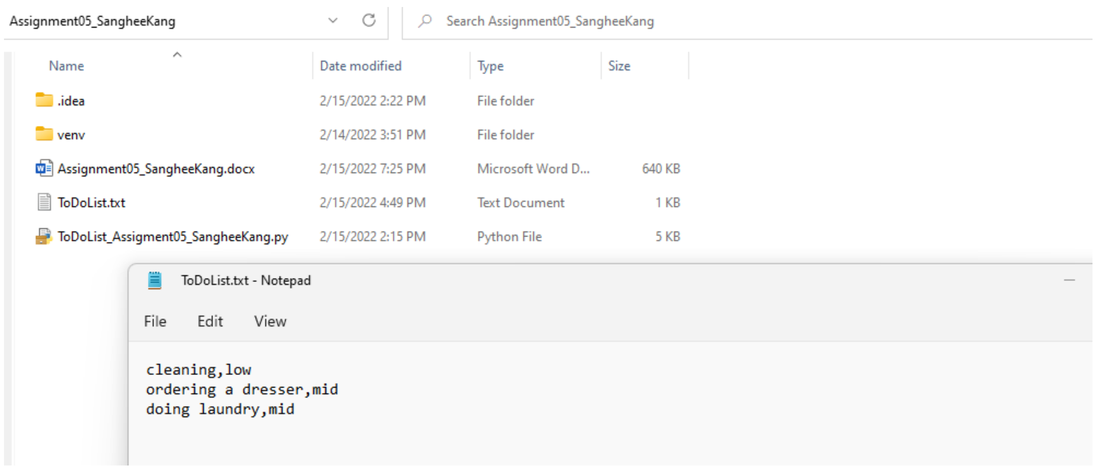


Name	Date modified	Type	Size
.idea	2/15/2022 2:22 PM	File folder	
venv	2/14/2022 3:51 PM	File folder	
Assigment05_SangheeKang.py	2/15/2022 2:15 PM	Python File	5 KB
Assignment05_SangheeKang.docx	2/8/2022 4:01 PM	Microsoft Word D...	391 KB
ToDoList.txt	2/15/2022 2:22 PM	Text Document	1 KB

```
cleaning,low
ordering a dresser,mid
doing laundry,mid
```

Figure 15. A screenshot of the content of ToDoList.txt (final output) and its location

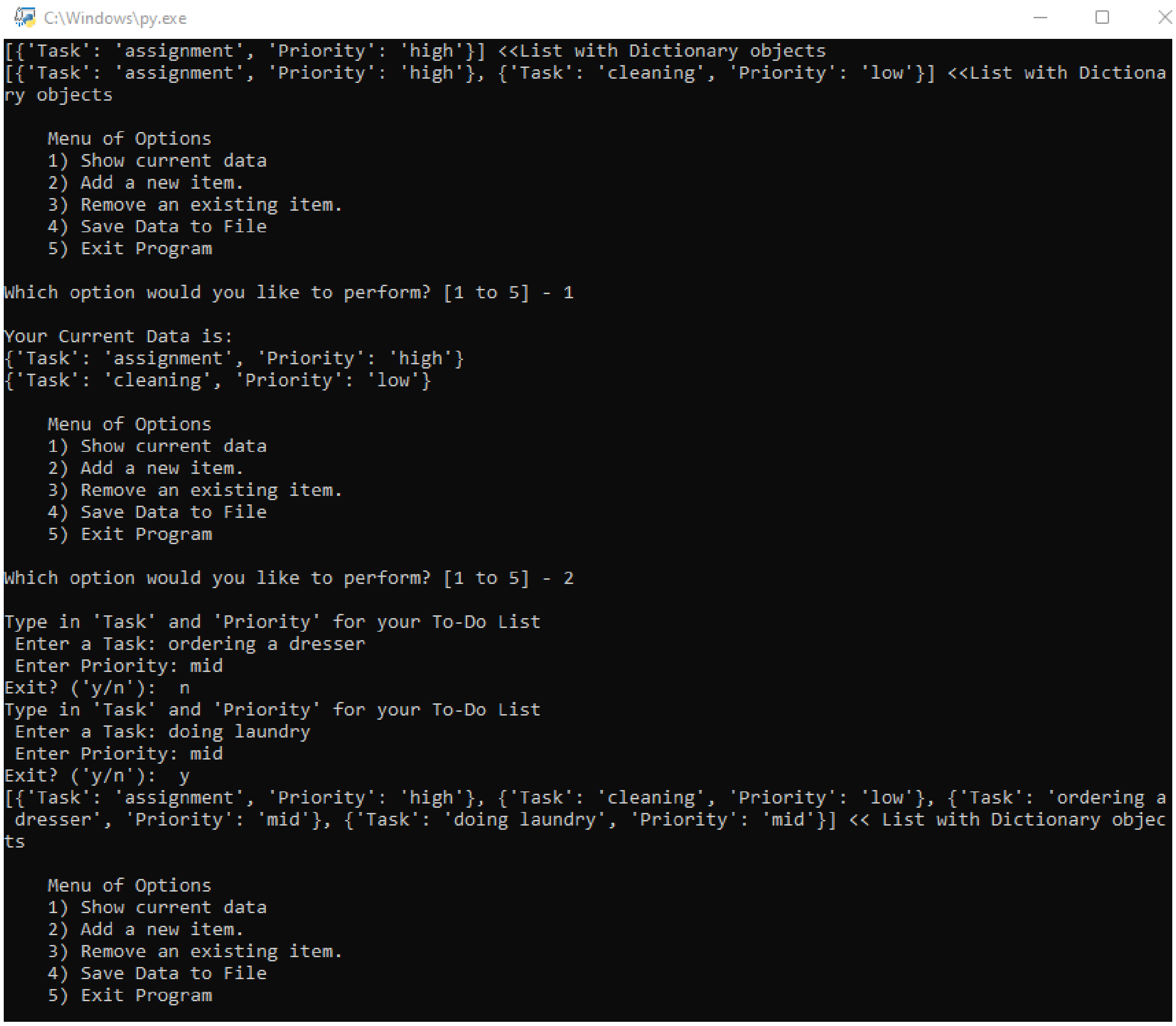
I also ran the Python code in a command shell. I followed the same procedures as I did in the PyCharm and the same output was yielded including the content of the text file. (see Figures 16 and 17).



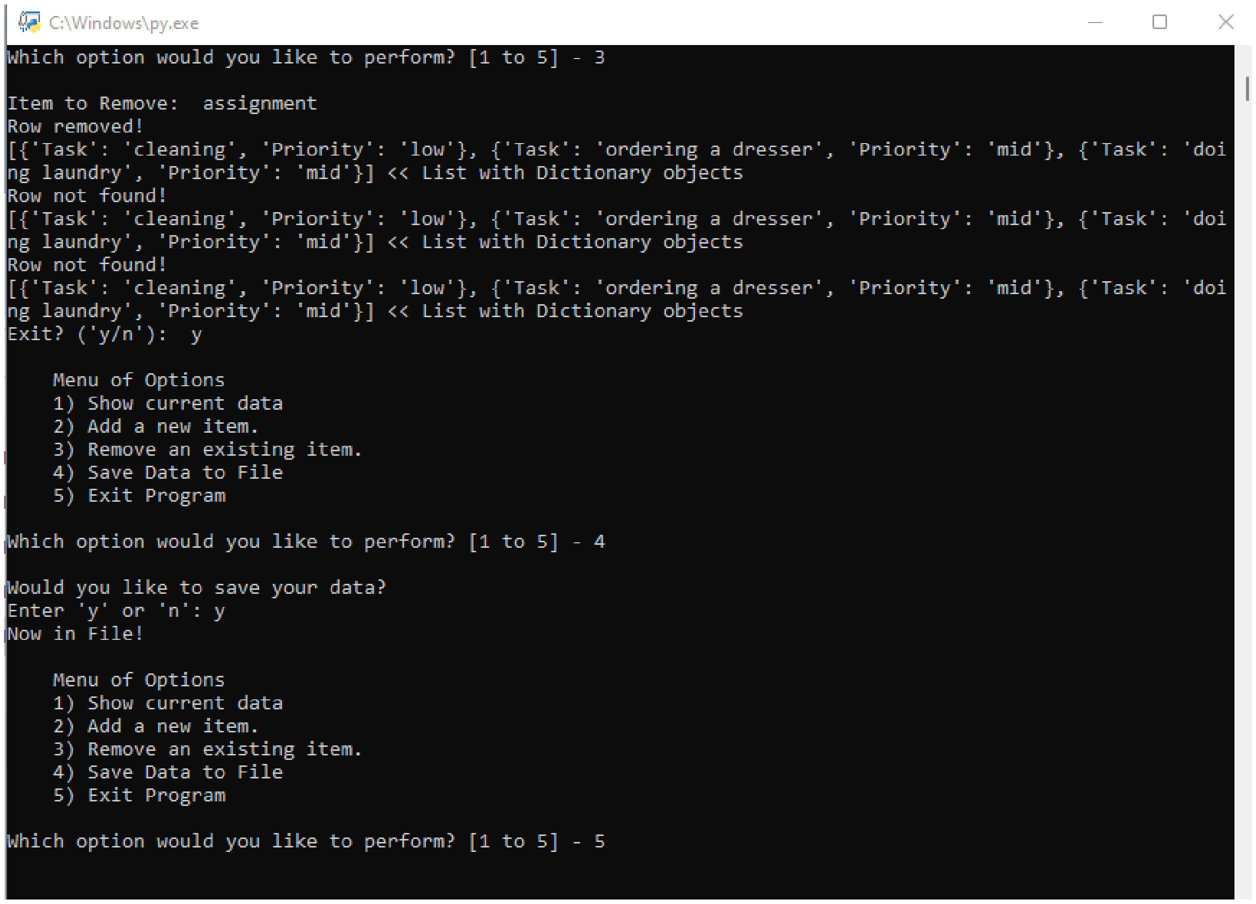
Name	Date modified	Type	Size
.idea	2/15/2022 2:22 PM	File folder	
venv	2/14/2022 3:51 PM	File folder	
Assignment05_SangheeKang.docx	2/15/2022 7:25 PM	Microsoft Word D...	640 KB
ToDoList.txt	2/15/2022 4:49 PM	Text Document	1 KB
ToDoList_Assigment05_SangheeKang.py	2/15/2022 2:15 PM	Python File	5 KB

```
cleaning,low
ordering a dresser,mid
doing laundry,mid
```

Figure 16. A screenshot of the content of ToDoList.txt (final output) and its location after running the code in the command sheel



```
[{'Task': 'assignment', 'Priority': 'high'}] <<List with Dictionary objects
[{'Task': 'assignment', 'Priority': 'high'}, {'Task': 'cleaning', 'Priority': 'low'}] <<List with Dictiona
ry objects

    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

Which option would you like to perform? [1 to 5] - 1

Your Current Data is:
{'Task': 'assignment', 'Priority': 'high'}
{'Task': 'cleaning', 'Priority': 'low'}

    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

Which option would you like to perform? [1 to 5] - 2

Type in 'Task' and 'Priority' for your To-Do List
 Enter a Task: ordering a dresser
 Enter Priority: mid
Exit? ('y/n'):  n
Type in 'Task' and 'Priority' for your To-Do List
 Enter a Task: doing laundry
 Enter Priority: mid
Exit? ('y/n'):  y
[{'Task': 'assignment', 'Priority': 'high'}, {'Task': 'cleaning', 'Priority': 'low'}, {'Task': 'ordering a
 dresser', 'Priority': 'mid'}, {'Task': 'doing laundry', 'Priority': 'mid'}] << List with Dictionary objec
ts

    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
```



```
Which option would you like to perform? [1 to 5] - 3

Item to Remove:  assignment
Row removed!
[{'Task': 'cleaning', 'Priority': 'low'}, {'Task': 'ordering a dresser', 'Priority': 'mid'}, {'Task': 'doi
ng laundry', 'Priority': 'mid'}] << List with Dictionary objects
Row not found!
[{'Task': 'cleaning', 'Priority': 'low'}, {'Task': 'ordering a dresser', 'Priority': 'mid'}, {'Task': 'doi
ng laundry', 'Priority': 'mid'}] << List with Dictionary objects
Row not found!
[{'Task': 'cleaning', 'Priority': 'low'}, {'Task': 'ordering a dresser', 'Priority': 'mid'}, {'Task': 'doi
ng laundry', 'Priority': 'mid'}] << List with Dictionary objects
Exit? ('y/n'):  y

    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

Which option would you like to perform? [1 to 5] - 4

Would you like to save your data?
Enter 'y' or 'n': y
Now in File!

    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

Which option would you like to perform? [1 to 5] - 5
```

Figure 16. Final output of my To Do List Python script in a command shell (top and bottom)

3. Summary

For this assignment, I created a code that can do different tasks (showing the content in an existing text file, displaying a menu of choices to the user, showing the current items in the text file, adding a new item to the table by capturing user's data, removing an existing item from the table, saving tasks to the text file, and exiting the program), using PyCharm. It was another challenging assignment but I kind of enjoyed doing it as it felt that I am creating something practical, which can be actually used in a real life.