Maggie Valencia
February 18, 2023
IT FDN 110 A
Assignment05

Assignment 05 - Task List Program

Introduction

This document demonstrates how to create a short program to add tasks to a list. The intent of this module is to learn to work with dictionaries including being able to create dictionary rows that can then be added to a Python List to create a Table. Other important topics include reading and writing to and from text files.

Challenges

This particular assignment had a higher degree of difficulty. For starters working with someone else's code is challenging to understand intent and purpose. For the most part creating the first few options were not hard, such as seeing content and adding items to a dictionary. The portion that I found most challenging was removing items from a dictionary. I used the book reading, online research and the class videos to help me with this section.

Displaying content of a file

The **Try** and **Except** functions are used to test code and handle errors. This is helpful for example if you want to open a file that might not exist – it is a good way to let the user know of an error without stopping the program (Figure 1)

```
# -- Processing -- #
# Step 1 - When the program starts, load any data you have
# in a text file called ToDoList.txt into a python list of dictionaries rows (like Lab

try:
    objFile = open("ToDoList.txt", "r")  # open  file
    print("|=============================================|")
    print("|These are the current tasks you have saved: |")
    print("|=============================================|")
    for row in objFile:  # find rows
        lstRow = row.split(',')  # split items
        dicRow = {"Task": lstRow[0], "Priority": lstRow[1]}  # create dic with items
        print('\t' + dicRow["Task"] + ' | ' + dicRow["Priority"].strip())  # print
    objFile.close()  # close
except:  # error handling
    print("File not found")  # if file not found
```

Figure 1 - Checking for a file

Displaying content in a table

In figure 2, this is straightforward enough, we want to see what tasks are saved to the text file before adding additional tasks. When a user selects '1' the text file is opened and the content is read to a dictionary row. We then print the dictionary row to the user by referencing the keys: "Item" and "Priority" in the dictionary. When done the file is closed.

```
# Step 3 - Show the current items in the table
#   TODO: Add Code Here
if strChoice.strip() == '1':  # when 1 is selected
    print("Current tasks saved to file:")  # print
    for row in objFile:# finds rows
        lstRow = row.split(',')  # splits values  ','
        dicRow = {"Item": lstRow[0].strip(), "Priority": lstRow[1].strip()}  # dictionary row
        print('\t' + dicRow["Item"] + ' | ' + dicRow["Priority"])  # prints
    objFile.close()  # close file
    continue# continues program
```

Figure 2 - Printing content of a file

Adding content to List/Table

In figure 3, we are collecting additional tasks from a user and adding it to the table. First by asking for a task and the priority of the task. This is saved to a dictionary by using the curly brackets, assigning key:values for each item entered. The append function keeps adding items to the table in memory. This continues as long as the user selects option 2. The challenge in this section was learning to print dictionaries. I was initially treating it like a list and index but from the class lab and online, I finally figured out that we refer to the keys to print out the dictionary row. This is a mistake I repeated multiple times.

```
# Step 4 - Add a new item to the list/Table
# TODO: Add Code Here
elif strChoice.strip() == '2':  # validates user selection
    dicName = input("Enter a new task name: ")  # user input
    dicRank = input("Enter its priority ['Low', 'Medium', 'High']:")  # user input
    dicRow = {"Item": dicName.strip(), "Priority": dicRank.strip()}  # create dictionary
    lstTable.append(dicRow)# add dictionary rows to table
    print("|============================================|")
    print("| So far you have the following new tasks:   |")
    print("|============================================|")
    print("Task", "|", "Priority")  # headings
    for row in lstTable:  # all rows are printed
        print(row["Item"] + ' | ' + row["Priority"])# print values for each key
    continue# move on
```

Figure 3 - Adding more task to a table

Remove items from a table

This is where things get complex (figure 4), simply using functions like **remove** or **del** do not work on their own. it is required to go through all the items in the table by means of the index to find the right item to delete. Using the **while** loop, a comparison is made across each item until it finds the right item to delete. If the item is found, it is deleted and a message confirms this. If no task is found, a message confirms no deletion has taken place. This is by far the most difficult step in the program that required online research and looking at the example from class, figure 4.

```
# Step 5 - Remove a new item from the list/Table
elif (strChoice.strip() == '3'):   # validates user selection
    # TODO: Add Code Here
    TskDlt = input("Which task would you like to remove? ")  # input
    bIntItemRemoved = False  # boolean flag
    RowNumb = 0 #counter
    while (RowNumb < len(lstTable)):  # loop
        if (TskDlt.lower() == str(list(dict(lstTable[RowNumb]).values())[0])):  # comparison
            del lstTable[RowNumb]  # delete
            bIntItemRemoved = True  # if deleted then True
        RowNumb += 1 # adds as go through table
    if (bIntItemRemoved == True):  # if item found confirms
        print("You task has been removed:", TskDlt)
    else: # if item not found
        print("Value", TskDlt, "could not be found")  # not found
    print("Your remaining tasks are:")  # prints remaining items
    for row in lstTable:
        print(row["Item"] + ' | ' + row["Priority"])
        continue  # move on
```

Figure 4 - Remove from table

Saving tasks and exit

In figure 5, the user selects to save tasks. I did not think it was necessary to ask the user if they want to save since option 4 is exactly that, “save the task to file”. It seemed redundant to do that. If a user selects to exit then no tasks would be saved.

```
# Step 6 - Save tasks to the ToDoToDoList.txt file
elif (strChoice.strip() == '4'):
    # TODO: Add Code Here
    print("|======================|")
    print("| Your saved tasks are: |")
    print("|======================|")
    for row in lstTable:  # Finds rows in table
        print(row["Item"] + ' | ' + row["Priority"])  # Print
    objFile = open("ToDoList.txt", "w")  # Open file
    for row in lstTable:  # Rows in the table
        objFile.write(row["Item"] + ', ' + row["Priority"] + "\n")  # Save to file
    objFile.close()  # Close
    continue
# Step 7 - Exit program
elif (strChoice.strip() == '5'):  # Validate user selection
    # TODO: Add Code Here
    print("You have ended the program...goodbye!")  # Print
    break  # Exit the program
```

Figure 5 - Saving and Exit

Running the program in PC

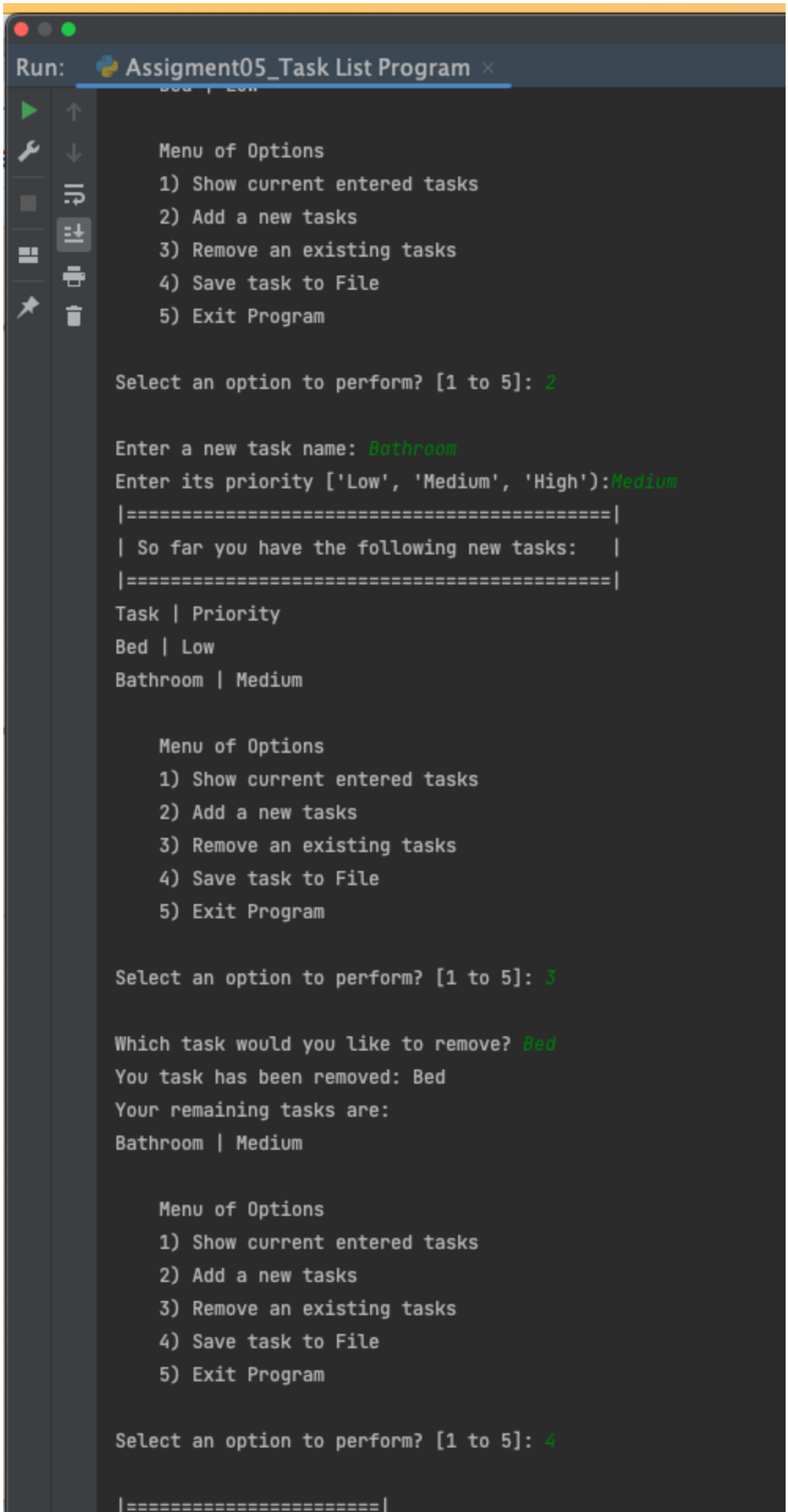
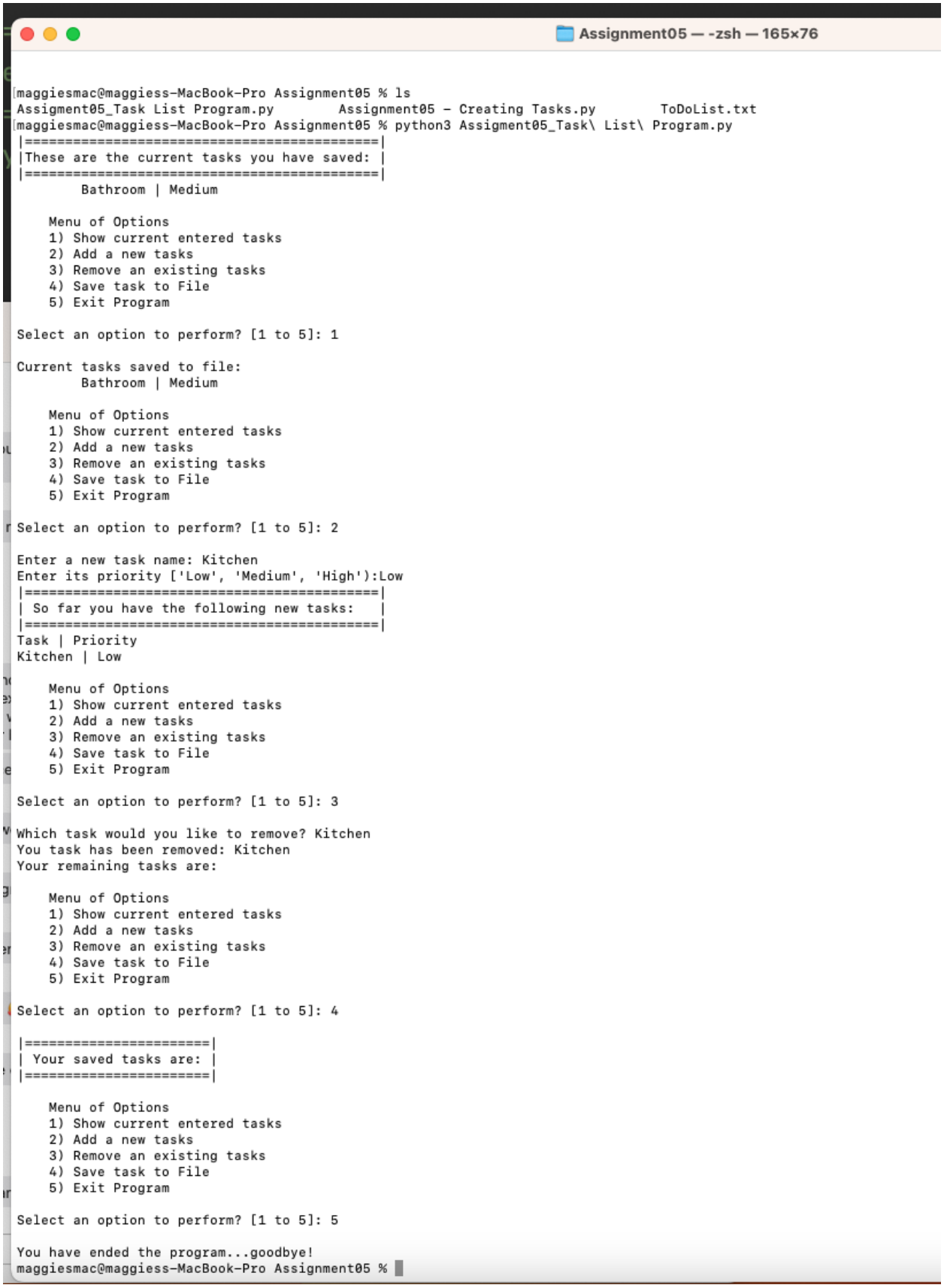

Run:
Assigment05_Task List Program
Menu of Options
1) Show current entered tasks
2) Add a new tasks
3) Remove an existing tasks
4) Save task to File
5) Exit Program
Select an option to perform? [1 to 5]: 2
Enter a new task name: Bathroom
Enter its priority ['Low', 'Medium', 'High']:Medium
|==========|
| So far you have the following new tasks: |
|==========|
Task | Priority
Bed | Low
Bathroom | Medium
Menu of Options
1) Show current entered tasks
2) Add a new tasks
3) Remove an existing tasks
4) Save task to File
5) Exit Program
Select an option to perform? [1 to 5]: 3
Which task would you like to remove? Bed
You task has been removed: Bed
Your remaining tasks are:
Bathroom | Medium
Menu of Options
1) Show current entered tasks
2) Add a new tasks
3) Remove an existing tasks
4) Save task to File
5) Exit Program
Select an option to perform? [1 to 5]: 4

Figure 6 - Running in PC

Running in terminal


Assignment05 — -zsh — 165×76
maggiesmac@maggiess-MacBook-Pro Assignment05 % ls
Assigment05_Task List Program.py Assignment05 - Creating Tasks.py ToDoList.txt
maggiesmac@maggiess-MacBook-Pro Assignment05 % python3 Assigment05_Task\ List\ Program.py
|==========|
|These are the current tasks you have saved: |
|==========|
Bathroom | Medium
Menu of Options
1) Show current entered tasks
2) Add a new tasks
3) Remove an existing tasks
4) Save task to File
5) Exit Program
Select an option to perform? [1 to 5]: 1
Current tasks saved to file:
Bathroom | Medium
Menu of Options
1) Show current entered tasks
2) Add a new tasks
3) Remove an existing tasks
4) Save task to File
5) Exit Program
Select an option to perform? [1 to 5]: 2
Enter a new task name: Kitchen
Enter its priority ['Low', 'Medium', 'High']:Low
|==========|
| So far you have the following new tasks: |
|==========|
Task | Priority
Kitchen | Low
Menu of Options
1) Show current entered tasks
2) Add a new tasks
3) Remove an existing tasks
4) Save task to File
5) Exit Program
Select an option to perform? [1 to 5]: 3
Which task would you like to remove? Kitchen
You task has been removed: Kitchen
Your remaining tasks are:
Menu of Options
1) Show current entered tasks
2) Add a new tasks
3) Remove an existing tasks
4) Save task to File
5) Exit Program
Select an option to perform? [1 to 5]: 4
|==========|
| Your saved tasks are: |
|==========|
Menu of Options
1) Show current entered tasks
2) Add a new tasks
3) Remove an existing tasks
4) Save task to File
5) Exit Program
Select an option to perform? [1 to 5]: 5
You have ended the program...goodbye!
maggiesmac@maggiess-MacBook-Pro Assignment05 %

Summary

In conclusion, in this module we have learned how to work with Python Lists, Dictionaries, how to print to files and how to read from files. Particularly challenging was removing from keys/values from a dictionary.