Emmanuel Ihejirika
May 24, 2023
IT FDN 100 A
Assignment 06
Github repository: https://github.com/Emmanuelihej/IntroToProg-Python

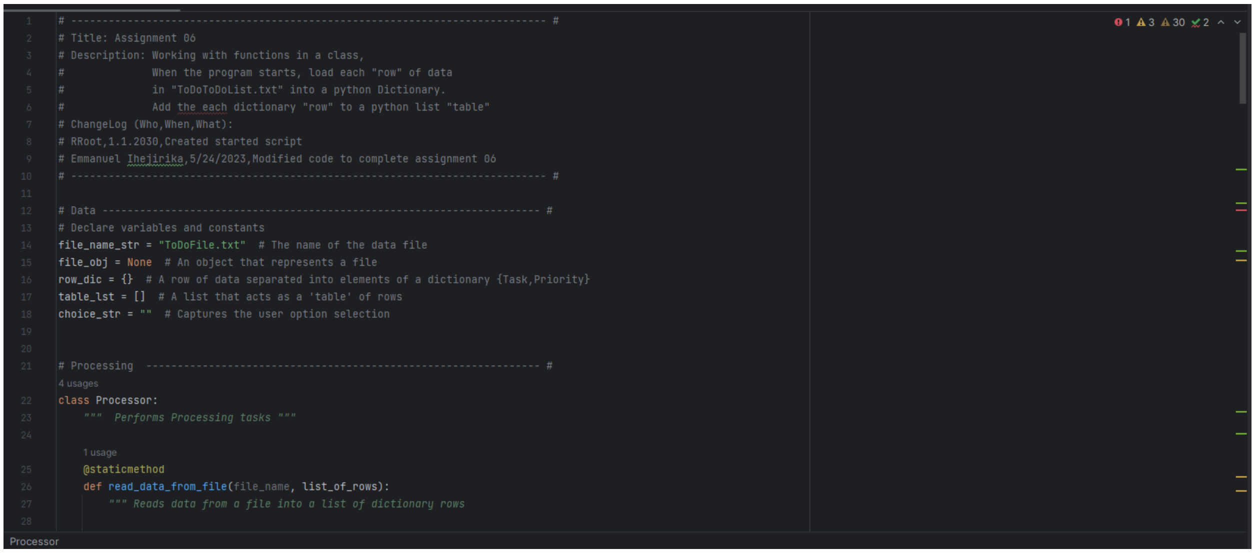
INTRO TO PROGRAMMING

Intro

In my sixth week of the Foundations of Programming: Python course, we learned how to create our own functions and use them to create our program. With this new knowledge we were supposed to implement the last hw assignment into this current one, using the def function tool. We also applied what we learned the previous week to this assignment. The assignment attached to this lecture asks us to create a program that will display a menu to the user, asking the user to select from the list: display tasks entered, add a task and its priority level, choose which row do they want to delete, save the information entered to the .txt file, or close the program, all wrapped up in a def function

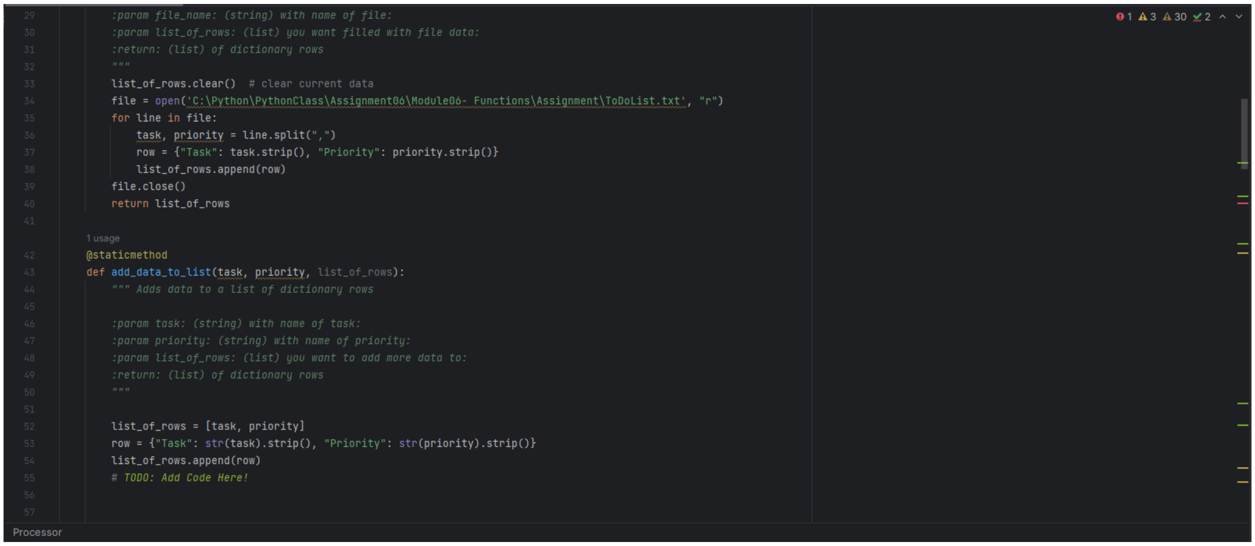
Creating the Program

After reading the assignment prompt, I understood what needed to be done and began to work out how I wanted each function to look like. I took the professor's advice and tackled the code one section at a time. Which means that I would complete one menu function at a time. For this assignment I did struggle with sectioning the program and at the end of it was not able to get the program to run completely. In Figures 1-7 you will see what I was able to complete regarding the python programming.



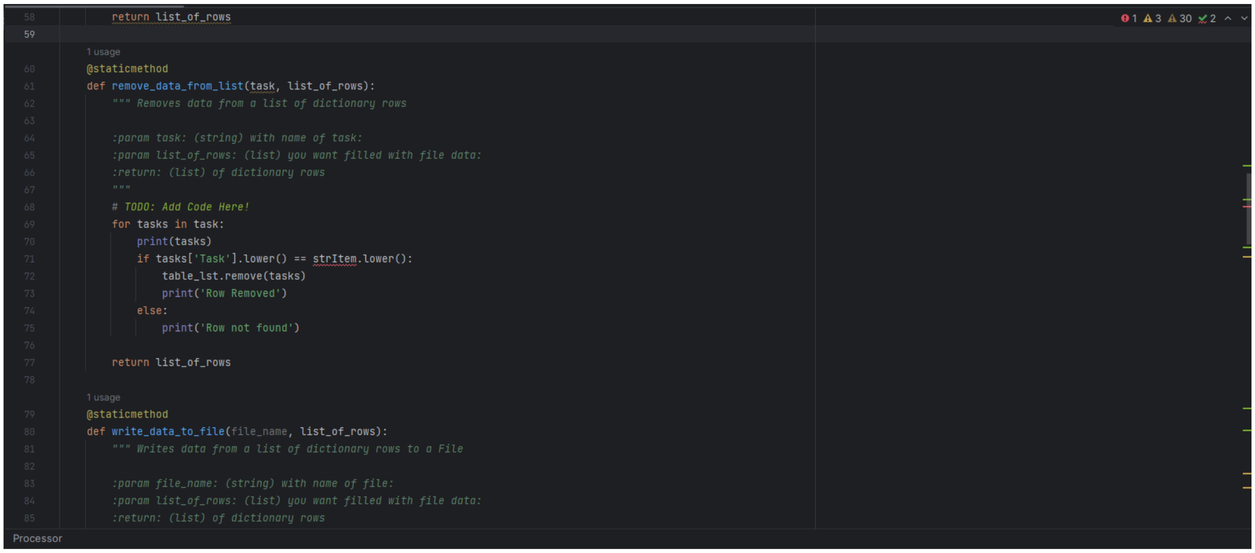
```
# ---------------------------------------------------------------------------- #
# Title: Assignment 06
# Description: Working with functions in a class,
#              When the program starts, load each "row" of data
#              in "ToDoToDoList.txt" into a python Dictionary.
#              Add the each dictionary "row" to a python list "table"
# ChangeLog (Who,When,What):
# RRoot,1.1.2030,Created started script
# Emmanuel Ihejirika,5/24/2023,Modified code to complete assignment 06
# ---------------------------------------------------------------------------- #

# Data ------------------------------------------------------------------------ #
# Declare variables and constants
file_name_str = "ToDoFile.txt"  # The name of the data file
file_obj = None  # An object that represents a file
row_dic = {}  # A row of data separated into elements of a dictionary {Task,Priority}
table_lst = []  # A list that acts as a 'table' of rows
choice_str = ""  # Captures the user option selection


# Processing  ----------------------------------------------------------------- #
4 usages
class Processor:
    """  Performs Processing tasks """

    1 usage
    @staticmethod
    def read_data_from_file(file_name, list_of_rows):
        """ Reads data from a file into a list of dictionary rows

```

Figure 1. Starting lines and comments of my program



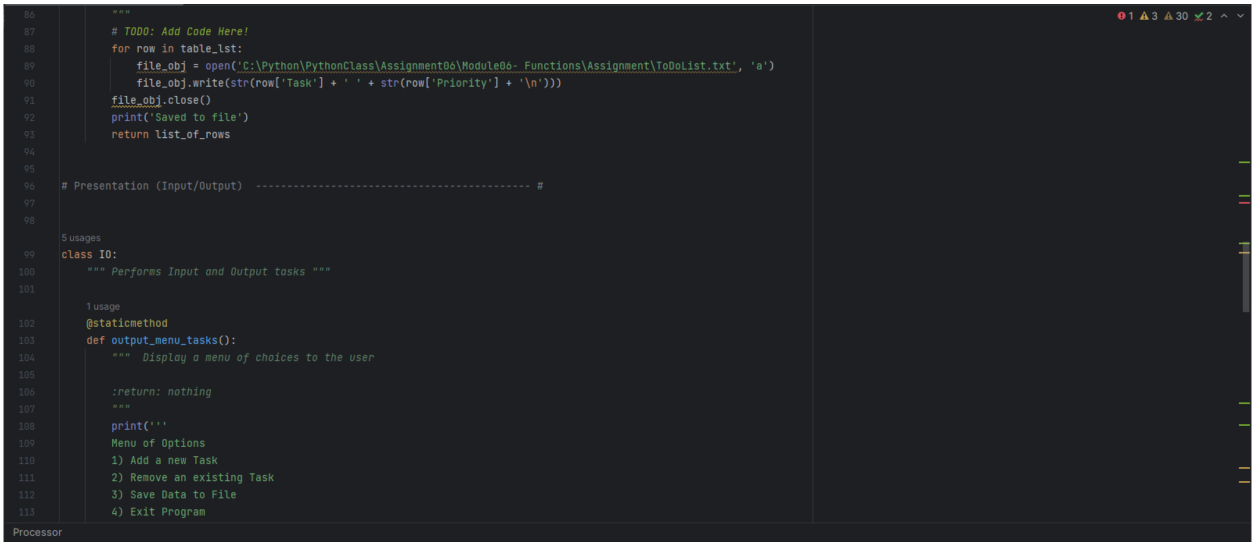
```
        :param file_name: (string) with name of file:
        :param list_of_rows: (list) you want filled with file data:
        :return: (list) of dictionary rows
        """
        list_of_rows.clear()  # clear current data
        file = open('C:\Python\PythonClass\Assignment06\Module06- Functions\Assignment\ToDoList.txt', "r")
        for line in file:
            task, priority = line.split(",")
            row = {"Task": task.strip(), "Priority": priority.strip()}
            list_of_rows.append(row)
        file.close()
        return list_of_rows

    @staticmethod
    def add_data_to_list(task, priority, list_of_rows):
        """ Adds data to a list of dictionary rows

        :param task: (string) with name of task:
        :param priority: (string) with name of priority:
        :param list_of_rows: (list) you want to add more data to:
        :return: (list) of dictionary rows
        """

        list_of_rows = [task, priority]
        row = {"Task": str(task).strip(), "Priority": str(priority).strip()}
        list_of_rows.append(row)
        # TODO: Add Code Here!


```

Figure 2. Continuation of my program



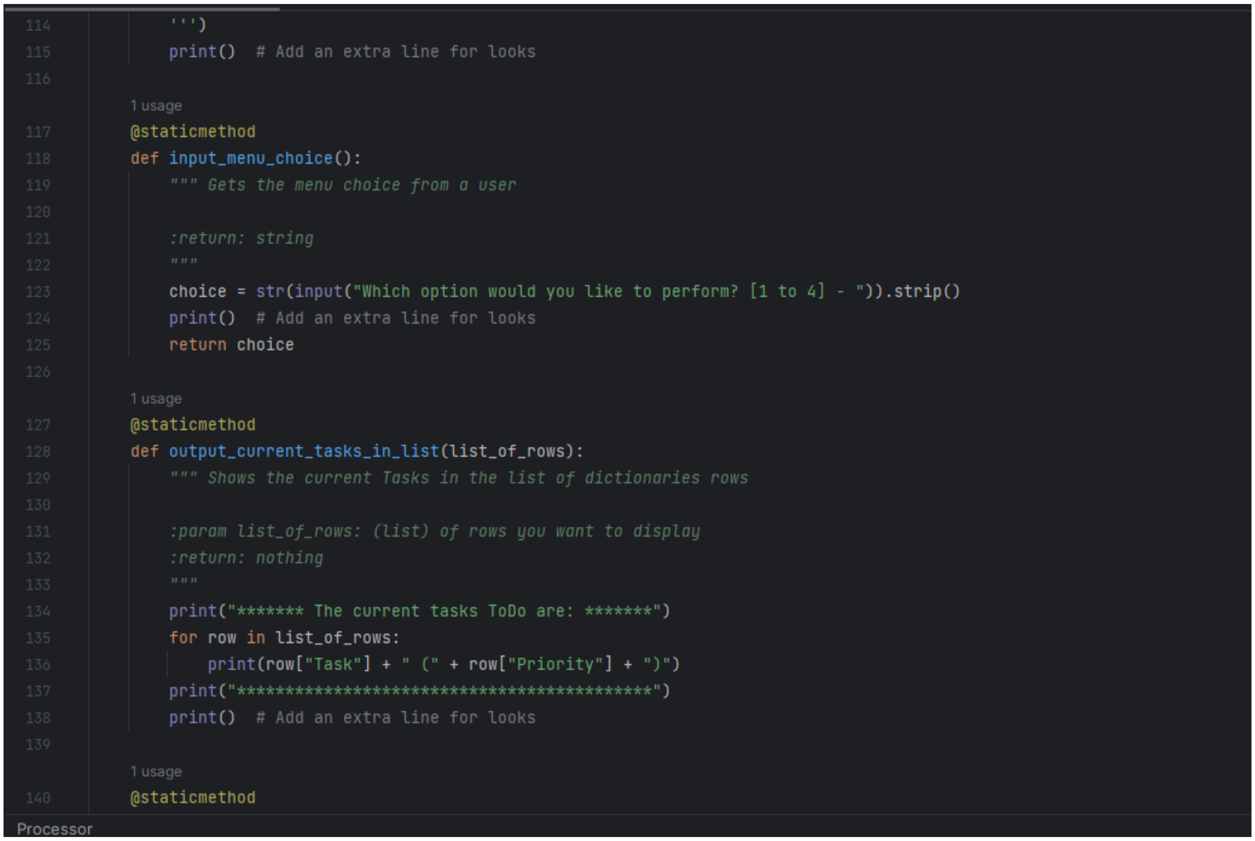
```
        return list_of_rows

    @staticmethod
    def remove_data_from_list(task, list_of_rows):
        """ Removes data from a list of dictionary rows

        :param task: (string) with name of task:
        :param list_of_rows: (list) you want filled with file data:
        :return: (list) of dictionary rows
        """
        # TODO: Add Code Here!
        for tasks in task:
            print(tasks)
            if tasks['Task'].lower() == strItem.lower():
                table_lst.remove(tasks)
                print('Row Removed')
            else:
                print('Row not found')

        return list_of_rows

    @staticmethod
    def write_data_to_file(file_name, list_of_rows):
        """ Writes data from a list of dictionary rows to a File

        :param file_name: (string) with name of file:
        :param list_of_rows: (list) you want filled with file data:
        :return: (list) of dictionary rows
```

Figure 3. Continuation of my program



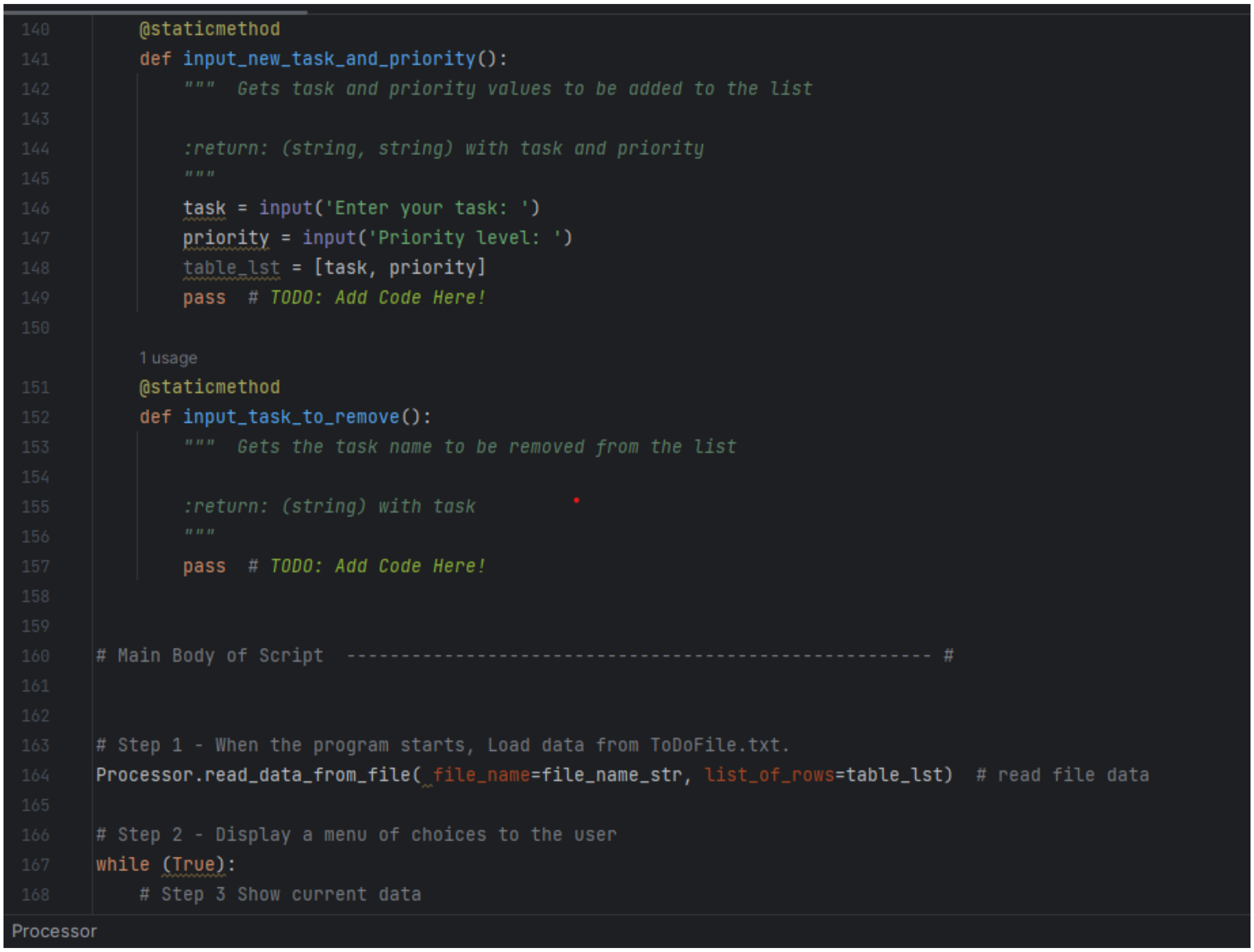
```python
        """
        # TODO: Add Code Here!
        for row in table_lst:
            file_obj = open('C:\Python\PythonClass\Assignment06\Module06- Functions\Assignment\ToDoList.txt', 'a')
            file_obj.write(str(row['Task'] + ' ' + str(row['Priority'] + '\n')))
        file_obj.close()
        print('Saved to file')
        return list_of_rows


# Presentation (Input/Output)  -------------------------------------------- #


class IO:
    """ Performs Input and Output tasks """

    @staticmethod
    def output_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) Exit Program
```

Figure 4.



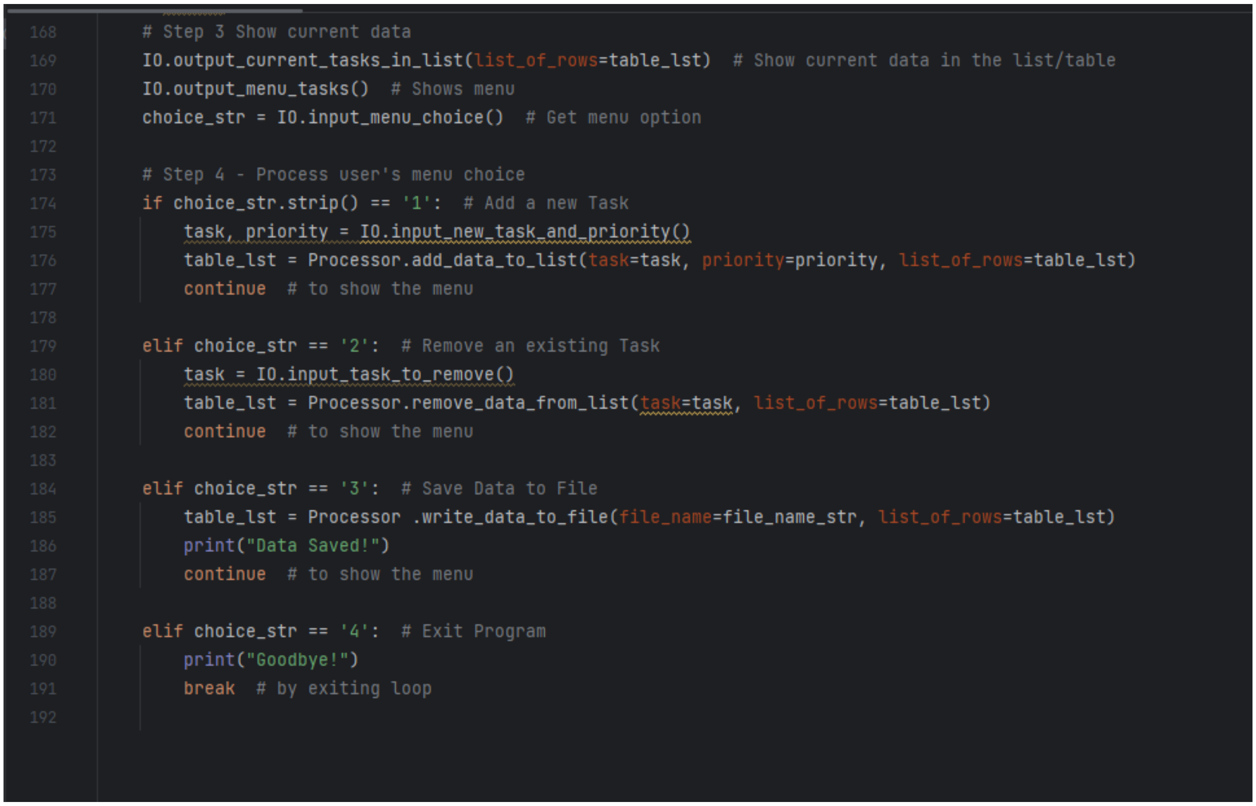
```python
        ''')
        print()  # Add an extra line for looks

    @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 4] - ")).strip()
        print()  # Add an extra line for looks
        return choice

    @staticmethod
    def output_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
```

Figure 5.



```
    @staticmethod
    def input_new_task_and_priority():
        """  Gets task and priority values to be added to the list

        :return: (string, string) with task and priority
        """
        task = input('Enter your task: ')
        priority = input('Priority level: ')
        table_lst = [task, priority]
        pass  # TODO: Add Code Here!

    1 usage
    @staticmethod
    def input_task_to_remove():
        """  Gets the task name to be removed from the list

        :return: (string) with task
        """
        pass  # TODO: Add Code Here!


# Main Body of Script  ------------------------------------------------------ #


# Step 1 - When the program starts, Load data from ToDoFile.txt.
Processor.read_data_from_file( file_name=file_name_str, list_of_rows=table_lst)  # read file data

# Step 2 - Display a menu of choices to the user
while (True):
    # Step 3 Show current data
```

Figure 6.



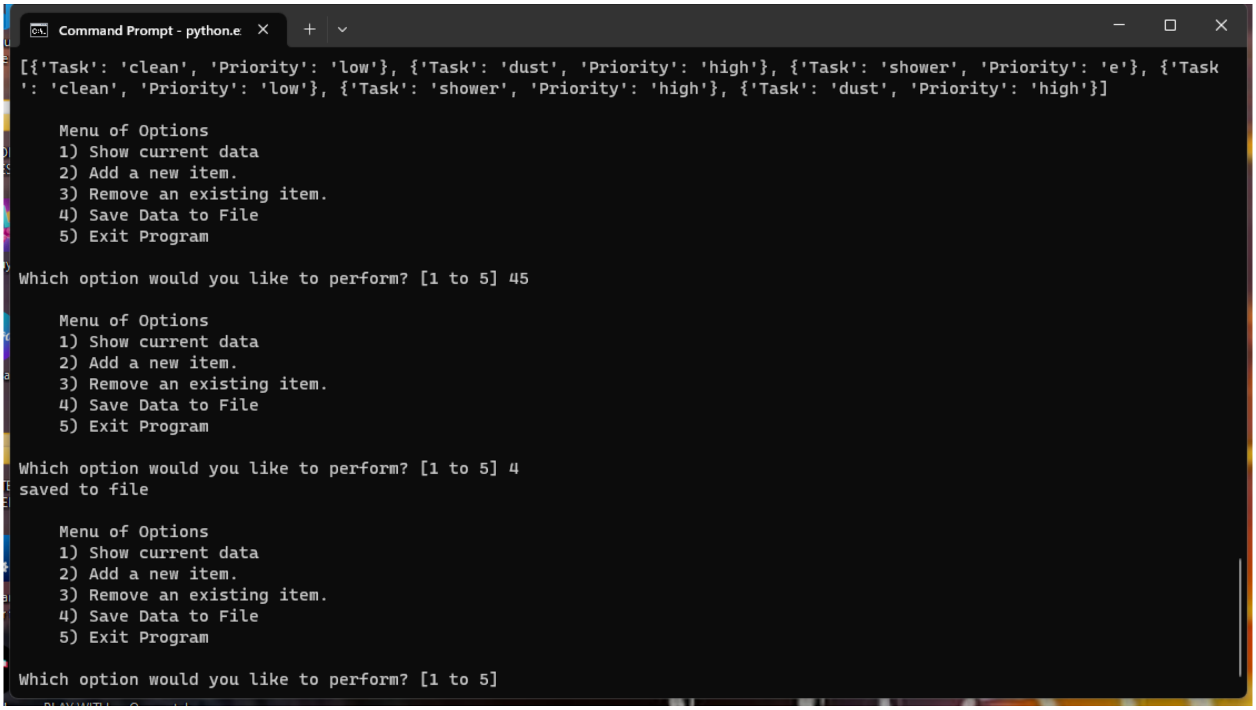
```
    # Step 3 Show current data
    IO.output_current_tasks_in_list(list_of_rows=table_lst)  # Show current data in the list/table
    IO.output_menu_tasks()  # Shows menu
    choice_str = IO.input_menu_choice()  # Get menu option

    # Step 4 - Process user's menu choice
    if choice_str.strip() == '1':  # Add a new Task
        task, priority = IO.input_new_task_and_priority()
        table_lst = Processor.add_data_to_list(task=task, priority=priority, list_of_rows=table_lst)
        continue  # to show the menu

    elif choice_str == '2':  # Remove an existing Task
        task = IO.input_task_to_remove()
        table_lst = Processor.remove_data_from_list(task=task, list_of_rows=table_lst)
        continue  # to show the menu

    elif choice_str == '3':  # Save Data to File
        table_lst = Processor .write_data_to_file(file_name=file_name_str, list_of_rows=table_lst)
        print("Data Saved!")
        continue  # to show the menu

    elif choice_str == '4':  # Exit Program
        print("Goodbye!")
        break  # by exiting loop

```

Figure 7.

The program when done correctly will do what assignment 5 does but the blocks of code will be organized into def functions to make things easier. It will display to the user anything that is already in the file then a menu and prompt them to select a number between 1 and 5. These numbers correspond to an action item on the menu, previously stated. Then just as in the example from assignment04 the program will close the .txt file and tell the user that the data has been saved, when the user decides. Telling the user this ensures the data was saved because the program would have failed and not told us, it is a way to confirm that part of the program worked successfully. I followed along with the lecture video and made sure to comprehend each step and attempt them myself.One issue that stuck out to me was my for Tasks function, when using a dictionary but it was solved in office hours.

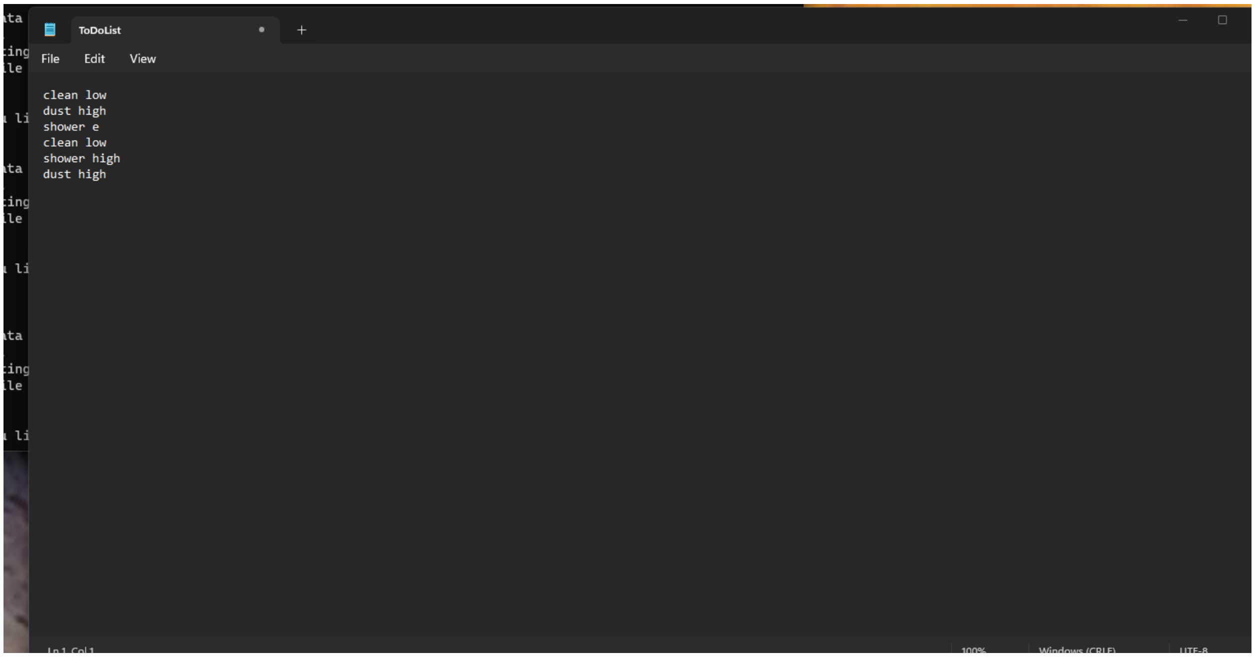
Summary

I attempted to create a program that will display a menu to the user and ask the user what options they want to choose. You will see my program in figures 1-7.What I took from this was how to allow the user to create a task list they may want to keep track of and how to use the .open, .write, dictionary, and .close functions in while true and if statements, and organize the information into processing and user input functions. Although I am a little discouraged at not completing this assignment I will continue to push forward.



```
[{'Task': 'clean', 'Priority': 'low'}, {'Task': 'dust', 'Priority': 'high'}, {'Task': 'shower', 'Priority': 'e'}, {'Task
': 'clean', 'Priority': 'low'}, {'Task': 'shower', 'Priority': 'high'}, {'Task': 'dust', 'Priority': 'high'}]

    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] 45

    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
saved to 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]
```

Figure 5. Script ran to completion



```
clean low
dust high
shower e
clean low
shower high
dust high
```

Figure 6. .txt file of HomeInventory