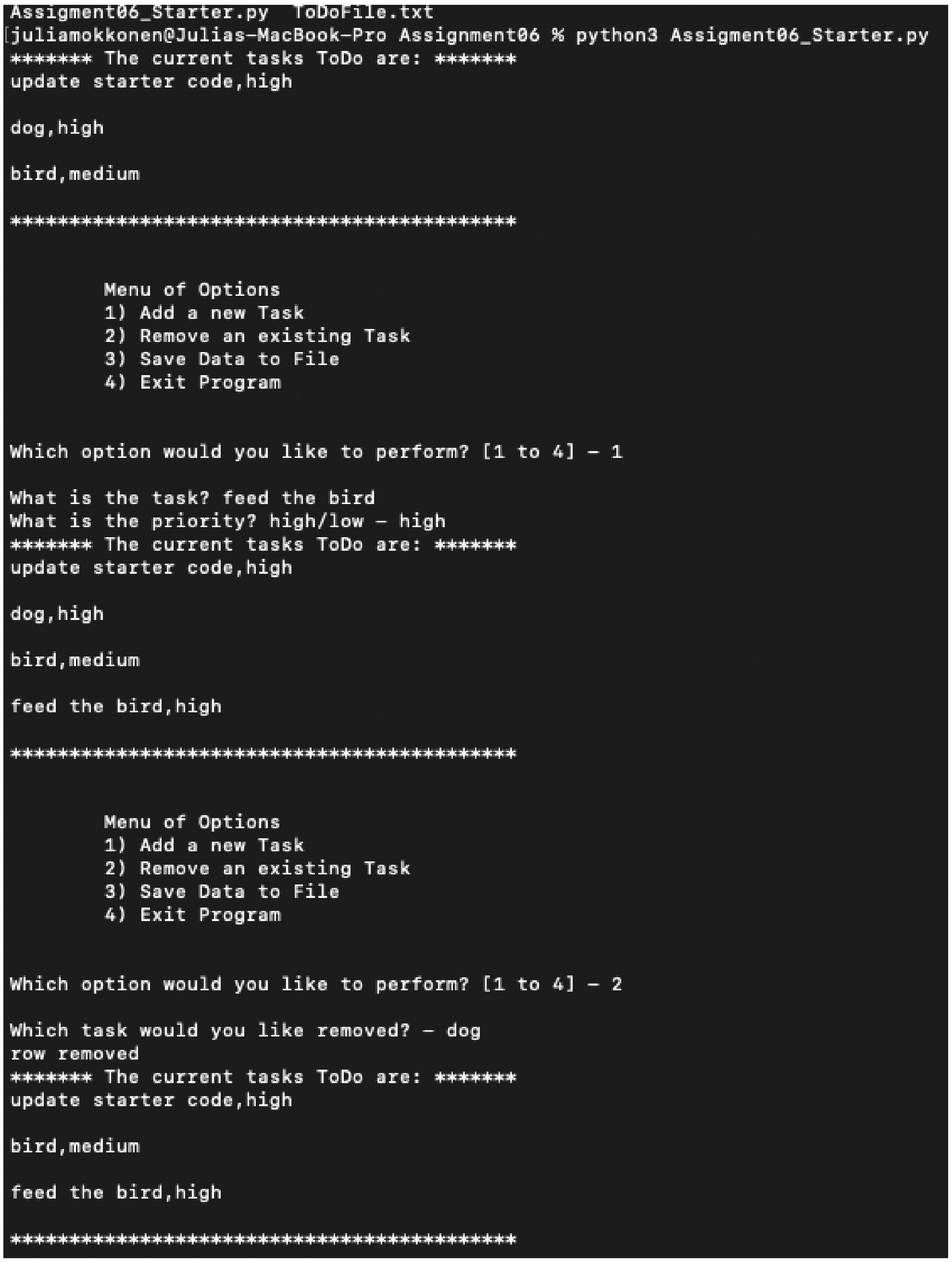
Julia Moekkoenen
August 14, 2023
IT FDN 110 A
Assignment 06
https://github.com/juliamoe/IntroToProg-Python

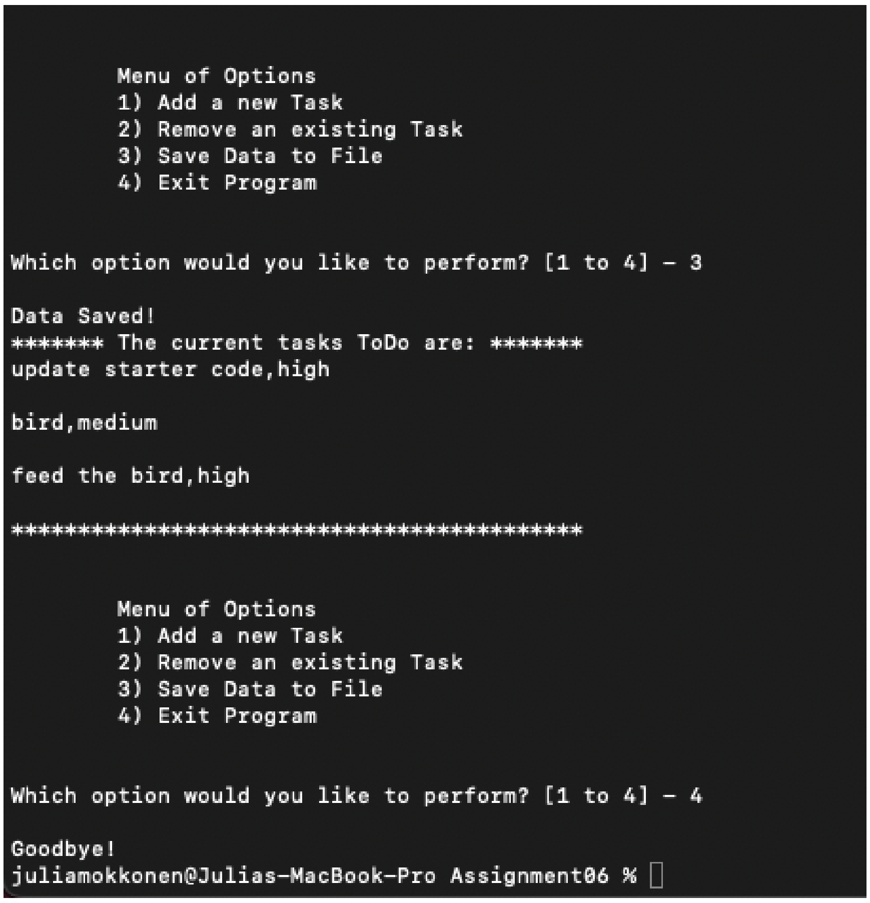
# Creating a To Do List

## Introduction

The purpose of the code was to create a To Do List by using functions and utilizing an existing code that had not been finished. functionalities of the script were to write, remove, and save data into a .txt file located in the same folder as where the program was saved. The user would choose which action they wanted to accomplish by choosing it from the starter menu. The working code is presented in the figure 1.1 and the before and after editing .txt file is presented in the figures 1.2 and 1.3.

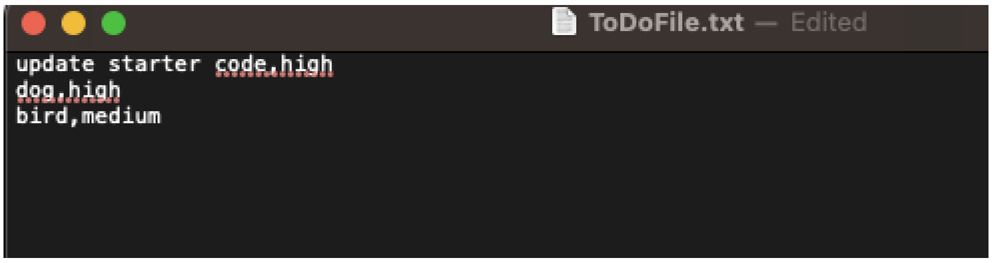


```
Assigment06_Starter.py  ToDoFile.txt
[juliamokkonen@Julias-MacBook-Pro Assignment06 % python3 Assigment06_Starter.py
******* The current tasks ToDo are: *******
update starter code,high

dog,high

bird,medium

*******************************************


        Menu of Options
        1) Add a new Task
        2) Remove an existing Task
        3) Save Data to File
        4) Exit Program


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

What is the task? feed the bird
What is the priority? high/low - high
******* The current tasks ToDo are: *******
update starter code,high

dog,high

bird,medium

feed the bird,high

*******************************************


        Menu of Options
        1) Add a new Task
        2) Remove an existing Task
        3) Save Data to File
        4) Exit Program


Which option would you like to perform? [1 to 4] - 2

Which task would you like removed? - dog
row removed
******* The current tasks ToDo are: *******
update starter code,high

bird,medium

feed the bird,high

*******************************************
```



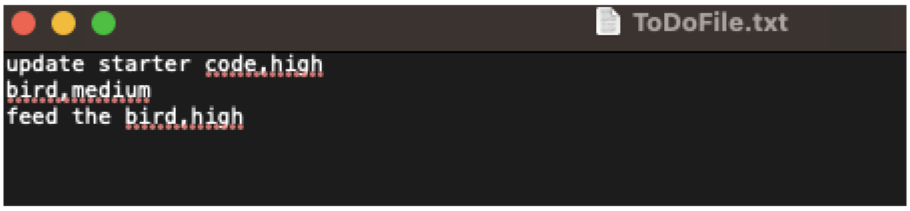
```
        Menu of Options
        1) Add a new Task
        2) Remove an existing Task
        3) Save Data to File
        4) Exit Program


Which option would you like to perform? [1 to 4] - 3

Data Saved!
******* The current tasks ToDo are: *******
update starter code,high

bird,medium

feed the bird,high

*******************************************


        Menu of Options
        1) Add a new Task
        2) Remove an existing Task
        3) Save Data to File
        4) Exit Program


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

Goodbye!
juliamokkonen@Julias-MacBook-Pro Assignment06 % 
```

***Figure 1.1: Working code in the Mac Terminal***



```
update starter code,high
dog,high
bird,medium
```

***Figure 1.2: Output printed into a text file before editing***

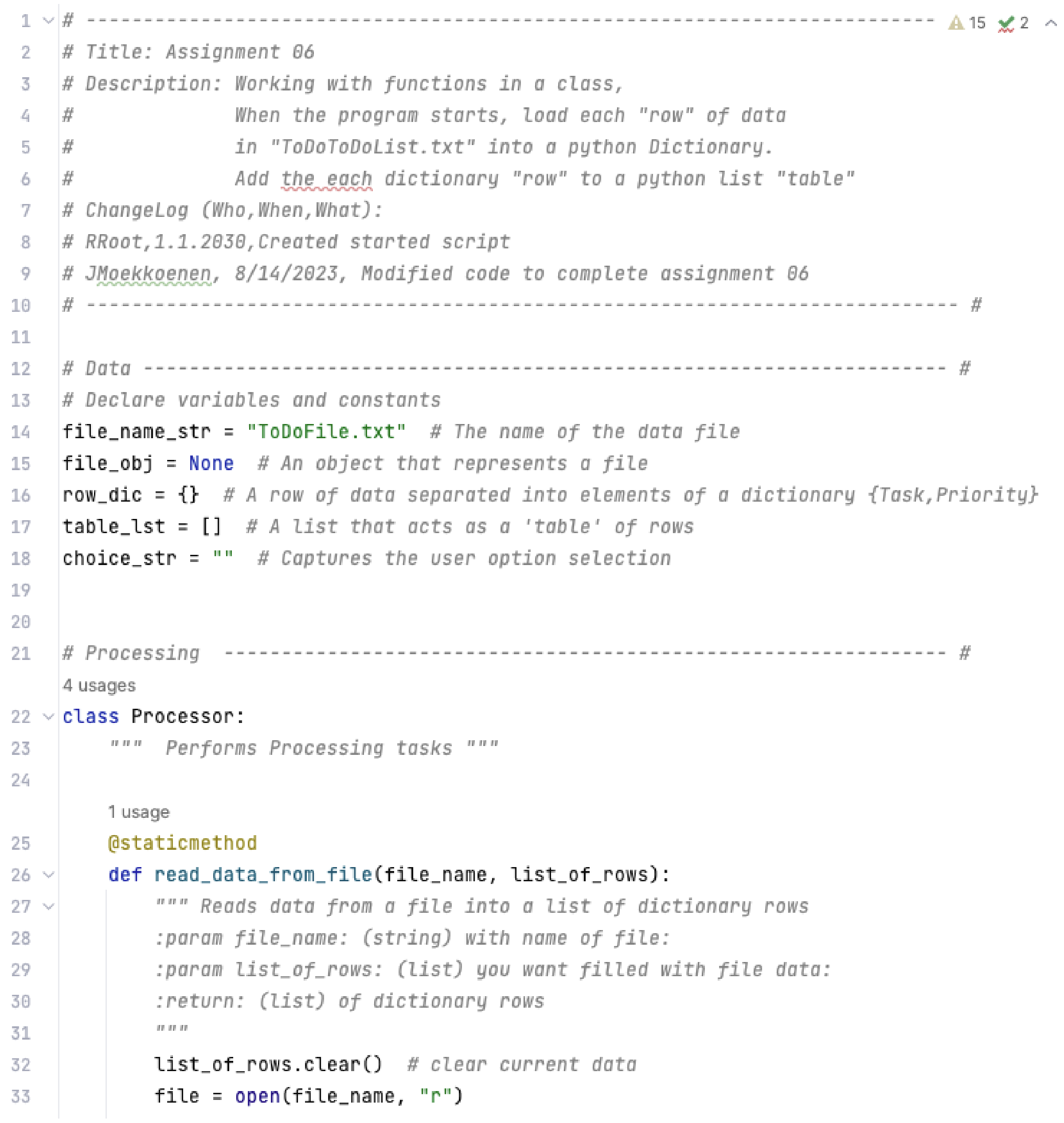


```
update starter code,high
bird,medium
feed the bird,high
```

***Figure 1.3: Output printed into a text file after removing the task row "dog" and adding the task row "feed the bird"***

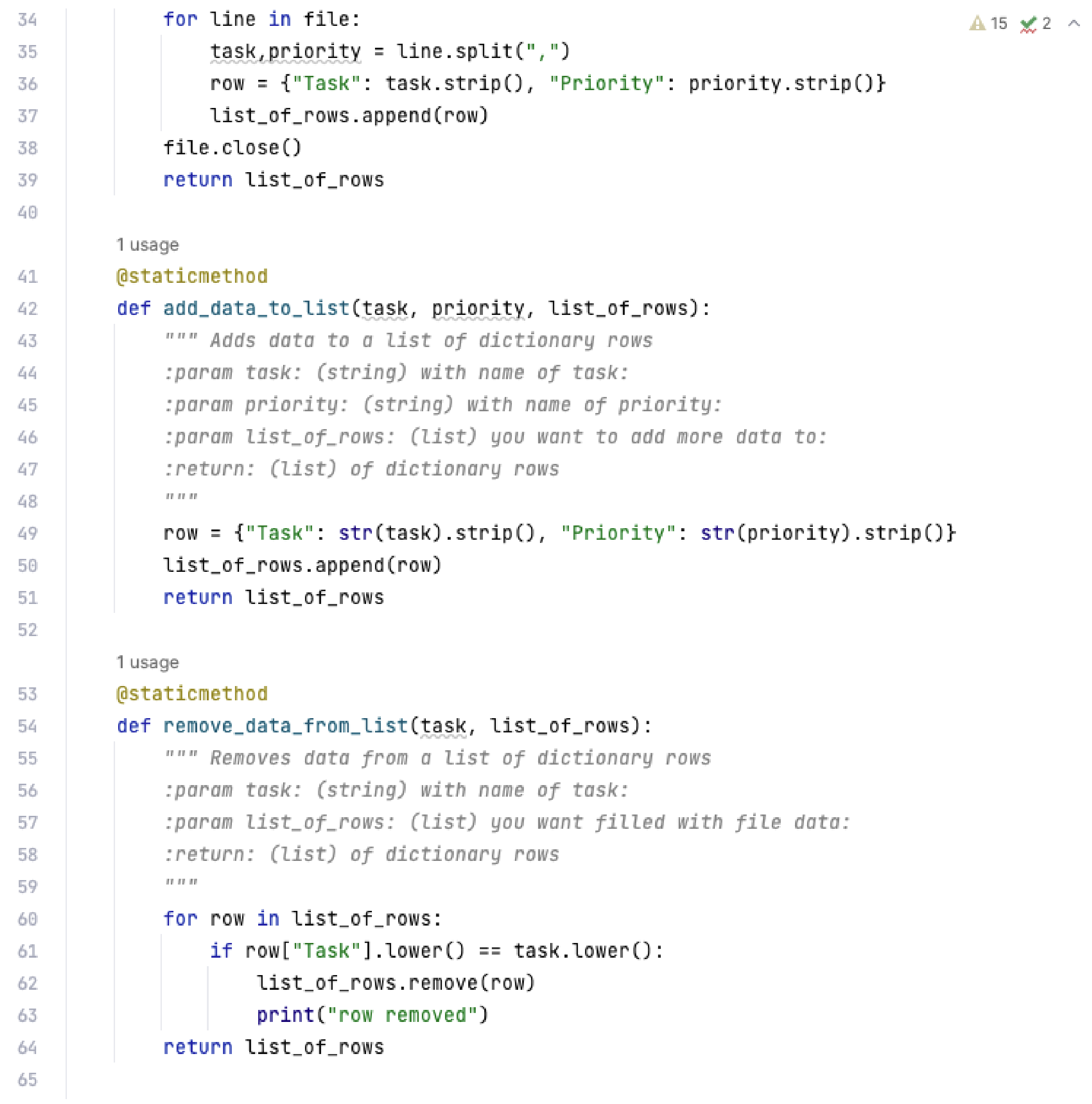
## Use of Functions

The code was divided into three sections which were presentation layer, data layer, and processing layer. The script heavily relied on using functions which were called from the presentation section of the code. The user input data was prompted and provided in the data layer, and actual processing of the data happened in the processing layer. Using functions makes it easier to make edits into the script and if the same processes are done in multiple different parts of the program, calling the same function more than once avoids unnecessary copy and paste on the code. The working code can be seen in figures 1.4a-1.4e.



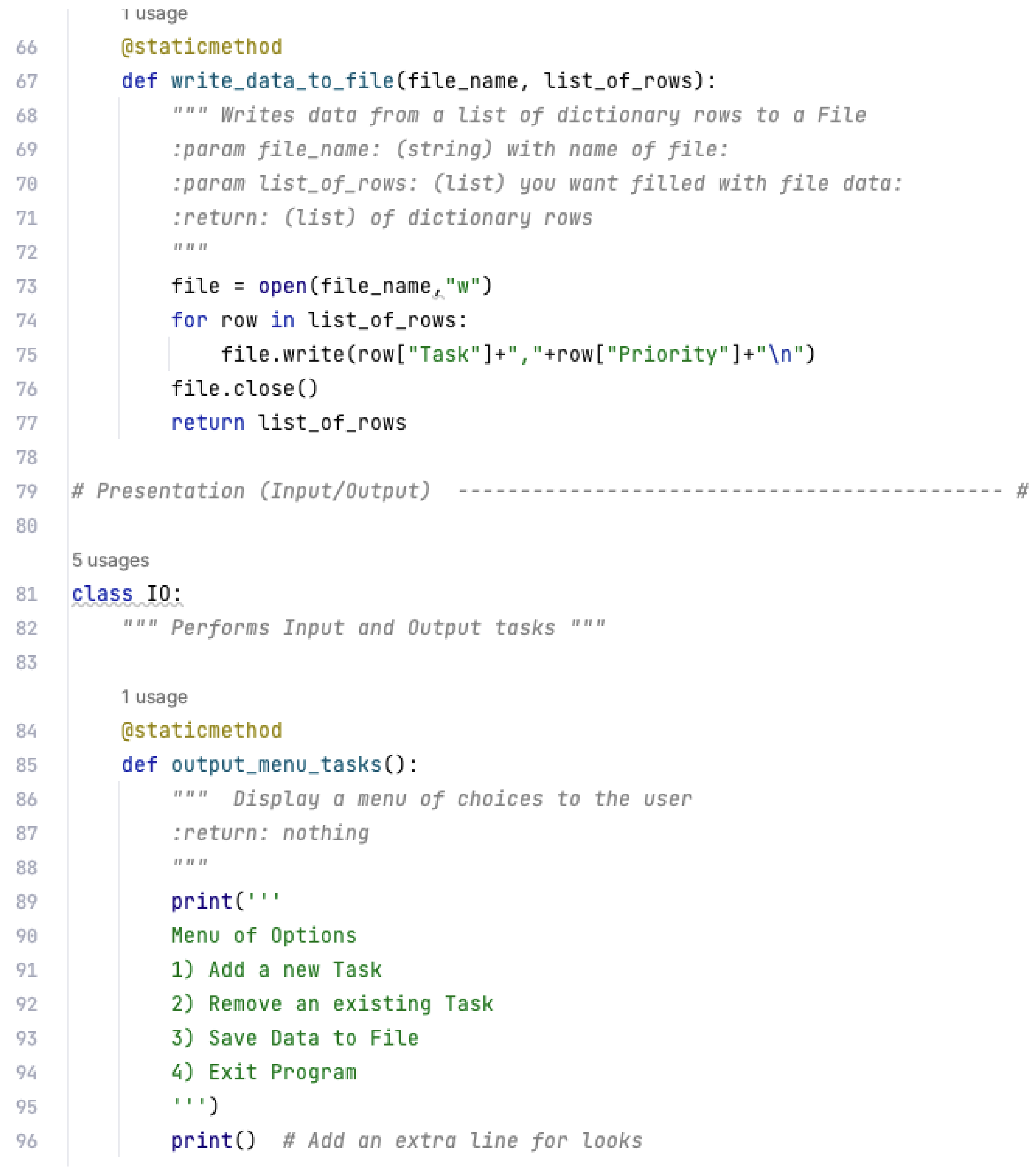
```python
# ------------------------------------------------------------------------------
# Title: Assignment 06
# Description: Working with functions in a class,
#              When the program starts, load each "row" of data
#              in "ToDoToDoList.txt" into a python Dictionary.
#              Add the each dictionary "row" to a python list "table"
# ChangeLog (Who,When,What):
# RRoot,1.1.2030,Created started script
# JMoekkoenen, 8/14/2023, 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  --------------------------------------------------------------- #
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")
```

***Figure 1.4a: Code for To Do list***



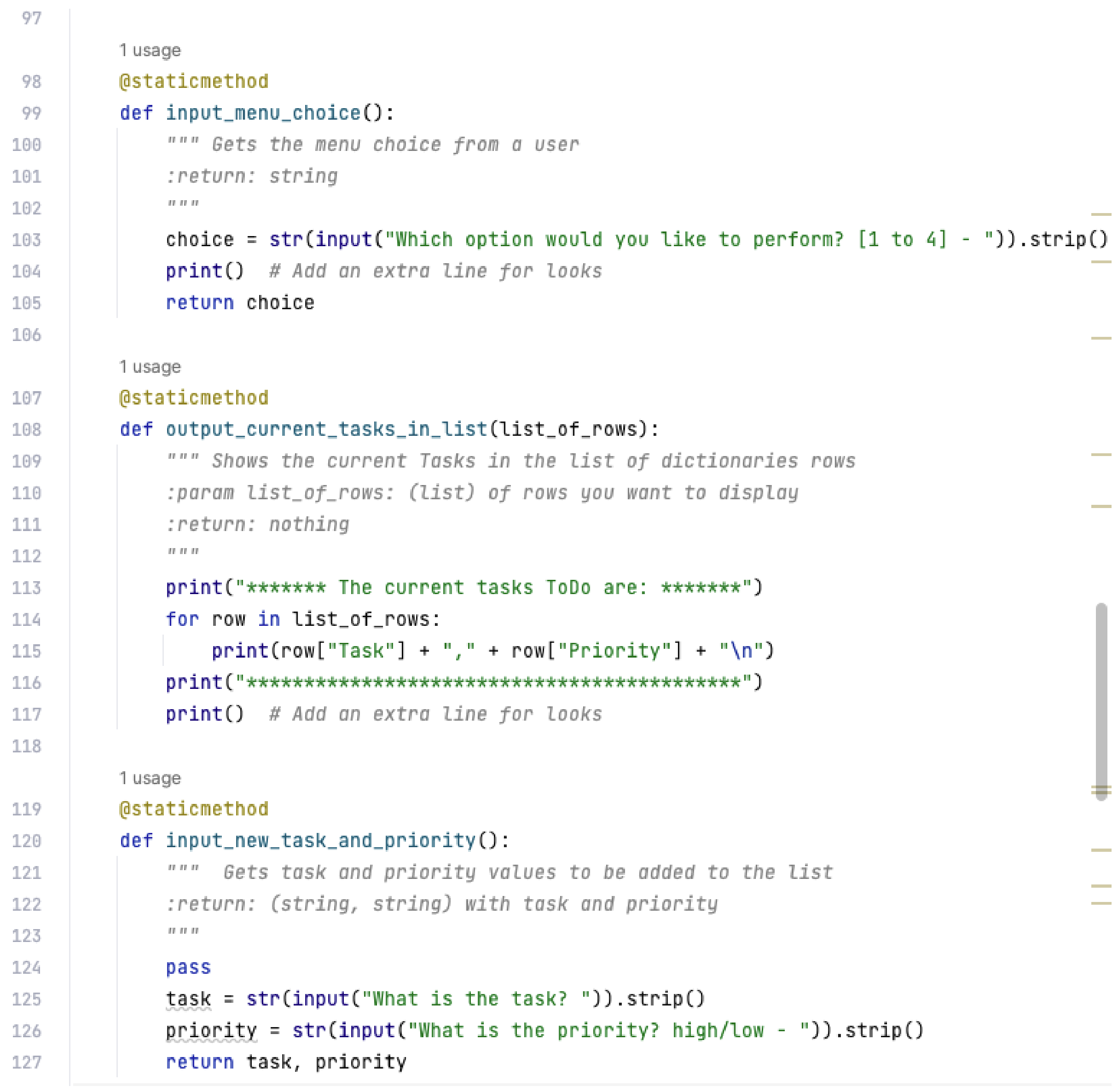
```python
        for line in file:
            task,priority = line.split(",")
            row = {"Task": task.strip(), "Priority": priority.strip()}
            list_of_rows.append(row)
        file.close()
        return list_of_rows

    1 usage
    @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
        """
        row = {"Task": str(task).strip(), "Priority": str(priority).strip()}
        list_of_rows.append(row)
        return list_of_rows

    1 usage
    @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
        """
        for row in list_of_rows:
            if row["Task"].lower() == task.lower():
                list_of_rows.remove(row)
                print("row removed")
        return list_of_rows

```

***Figure 1.4b: Code for To Do list***



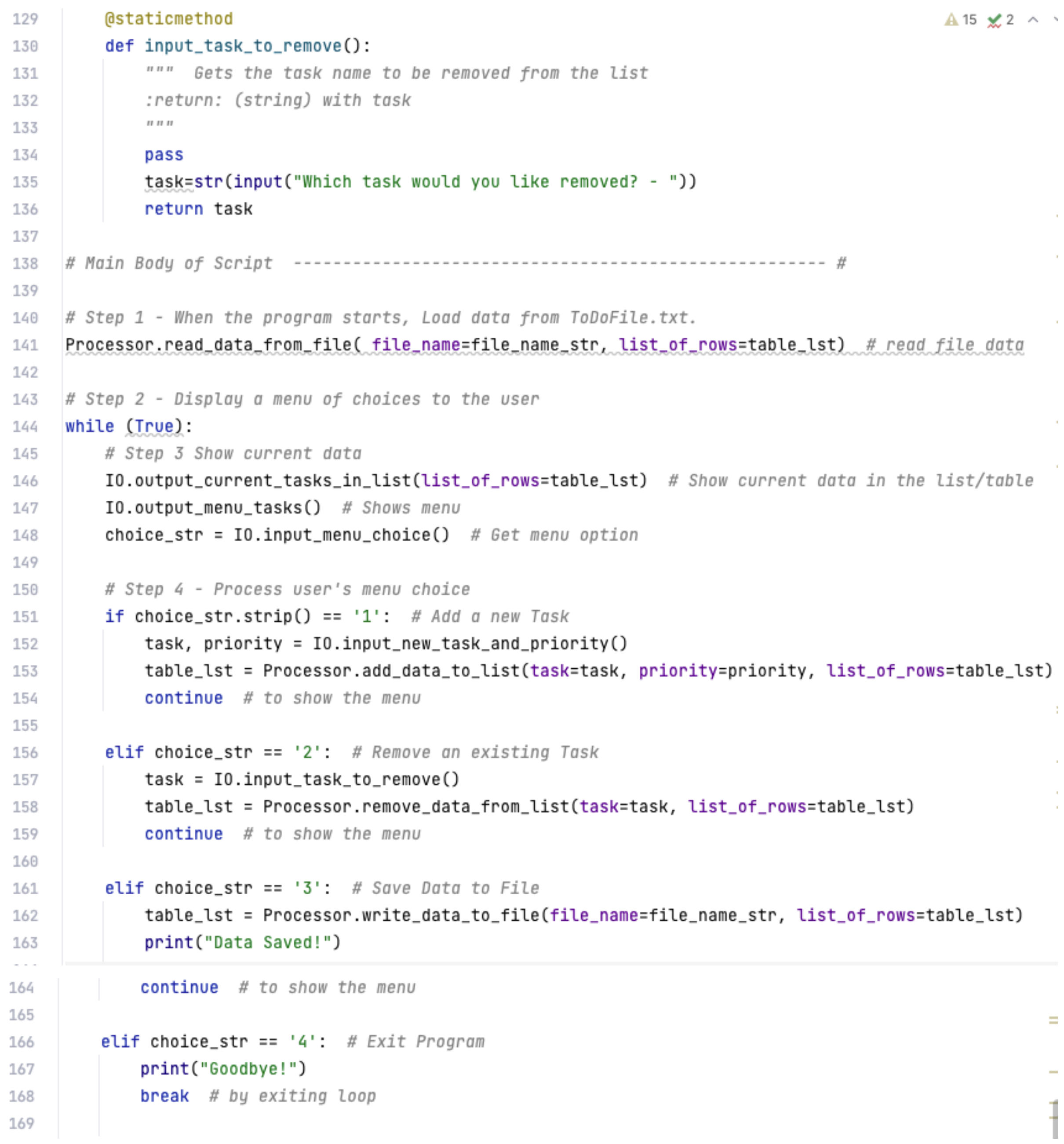
```
    1 usage
    @staticmethod
    def write_data_to_file(file_name, list_of_rows):
        """ Writes data from a list of dictionary rows to a File
        :param file_name: (string) with name of file:
        :param list_of_rows: (list) you want filled with file data:
        :return: (list) of dictionary rows
        """
        file = open(file_name,"w")
        for row in list_of_rows:
            file.write(row["Task"]+","+row["Priority"]+"\n")
        file.close()
        return list_of_rows

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

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

    1 usage
    @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
        ''')
        print()  # Add an extra line for looks
```

***Figure 1.4c: Code for To Do list***



```
    1 usage
    @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

    1 usage
    @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"] + "\n")
        print("*******************************************")
        print()  # Add an extra line for looks

    1 usage
    @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
        """
        pass
        task = str(input("What is the task? ")).strip()
        priority = str(input("What is the priority? high/low - ")).strip()
        return task, priority
```

***Figure 1.4d: Code for To Do list***



```
129      @staticmethod
130      def input_task_to_remove():
131          """  Gets the task name to be removed from the list
132          :return: (string) with task
133          """
134          pass
135          task=str(input("Which task would you like removed? - "))
136          return task
137
138  # Main Body of Script  ------------------------------------------------------ #
139
140  # Step 1 - When the program starts, Load data from ToDoFile.txt.
141  Processor.read_data_from_file( file_name=file_name_str, list_of_rows=table_lst)  # read file data
142
143  # Step 2 - Display a menu of choices to the user
144  while (True):
145      # Step 3 Show current data
146      IO.output_current_tasks_in_list(list_of_rows=table_lst)  # Show current data in the list/table
147      IO.output_menu_tasks()  # Shows menu
148      choice_str = IO.input_menu_choice()  # Get menu option
149
150      # Step 4 - Process user's menu choice
151      if choice_str.strip() == '1':  # Add a new Task
152          task, priority = IO.input_new_task_and_priority()
153          table_lst = Processor.add_data_to_list(task=task, priority=priority, list_of_rows=table_lst)
154          continue  # to show the menu
155
156      elif choice_str == '2':  # Remove an existing Task
157          task = IO.input_task_to_remove()
158          table_lst = Processor.remove_data_from_list(task=task, list_of_rows=table_lst)
159          continue  # to show the menu
160
161      elif choice_str == '3':  # Save Data to File
162          table_lst = Processor.write_data_to_file(file_name=file_name_str, list_of_rows=table_lst)
163          print("Data Saved!")
164          continue  # to show the menu
165
166      elif choice_str == '4':  # Exit Program
167          print("Goodbye!")
168          break  # by exiting loop
169
```

***Figure 1.4e: Code for To Do list***

## Challenges

The most challenging part of the assignment was to follow the ready-made code and how the original editor had been thinking whenever they wrote it. Some of the variables were named the same which caused some confusion, one example of this was on line 153 where task=task and priority=priority. Person editing the code may find the same naming convention confusing and mix up which one is the returned variable.

## Summary

The assignment was to use an unfinished script of a To Do List and finish it by implementing the use of functions.The script was divided into three sections, presentation, data, and processing. The use of functions makes the code more editable and shortens the overall length of the code because the same function can be called multiple times with different data sets.