Name: Axel Adams
Date: Feb, 21, 2023
Course: Intro to Programming (PYTHON)
https://github.com/H-imperialis/Intro-to-Programming---Module-06/blob/main/Class%20Docs/Assignment_06

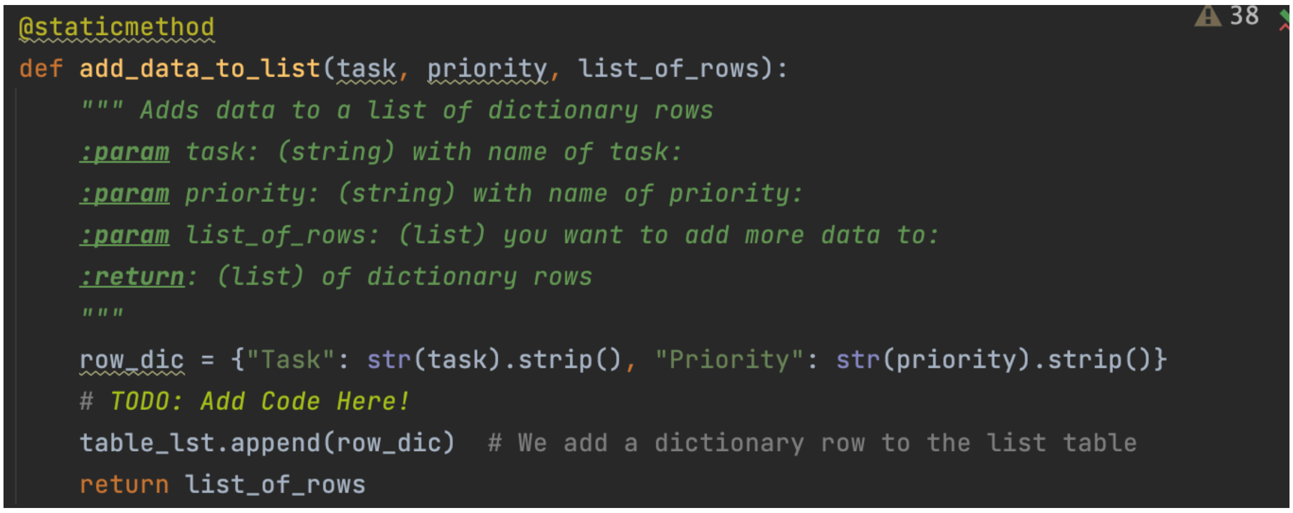
Assignment 04 - To Do List Utilizing Functions

Introduction:
Functions are a useful tool that allows the coder to define a set of operations that can then be repeated throughout the script. This makes it unnecessary for the cover to repeat the code for each process of the code. This becomes advantageous as scripts become longer and more complicated. It also limits the number of variables that need to be defined within the script. In this assignment, we use the To Do List script from Assignment 05 and modify it such that we use functions.

Topic: Pseudo-coding Assignment 04
The pseudo-code was provided for us in the script prompt.

Topic: Converting Processor Processes to Function
The first processing function was provided for us - in terms of reading and writing the text file.

Figure 1: Adding Data Operation



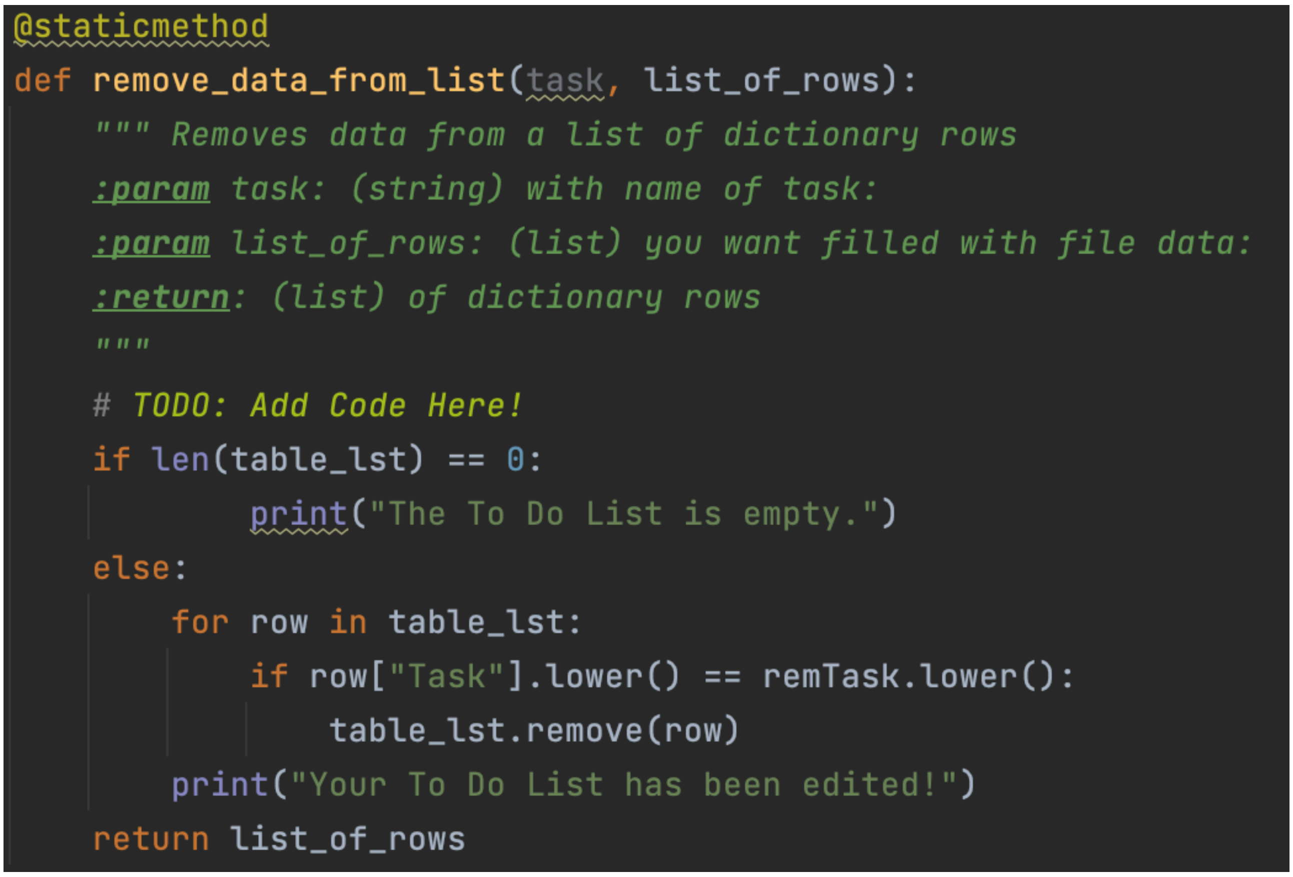
```
@staticmethod
def add_data_to_list(task, priority, list_of_rows):
    """ Adds data to a list of dictionary rows
    :param task: (string) with name of task:
    :param priority: (string) with name of priority:
    :param list_of_rows: (list) you want to add more data to:
    :return: (list) of dictionary rows
    """
    row_dic = {"Task": str(task).strip(), "Priority": str(priority).strip()}
    # TODO: Add Code Here!
    table_lst.append(row_dic)  # We add a dictionary row to the list table
    return list_of_rows
```

Adding New To Do Item: The main process added to this function is appending the new dictionary row to the table list that is read from the text file.

Removing New To Do Item: In this function, I used the length function based upon a empty to do list to let the user know the to do list is empty. If the list is occupied, we query the rows for the string form of the remove task generated by the IO function for remove - “remTask.”

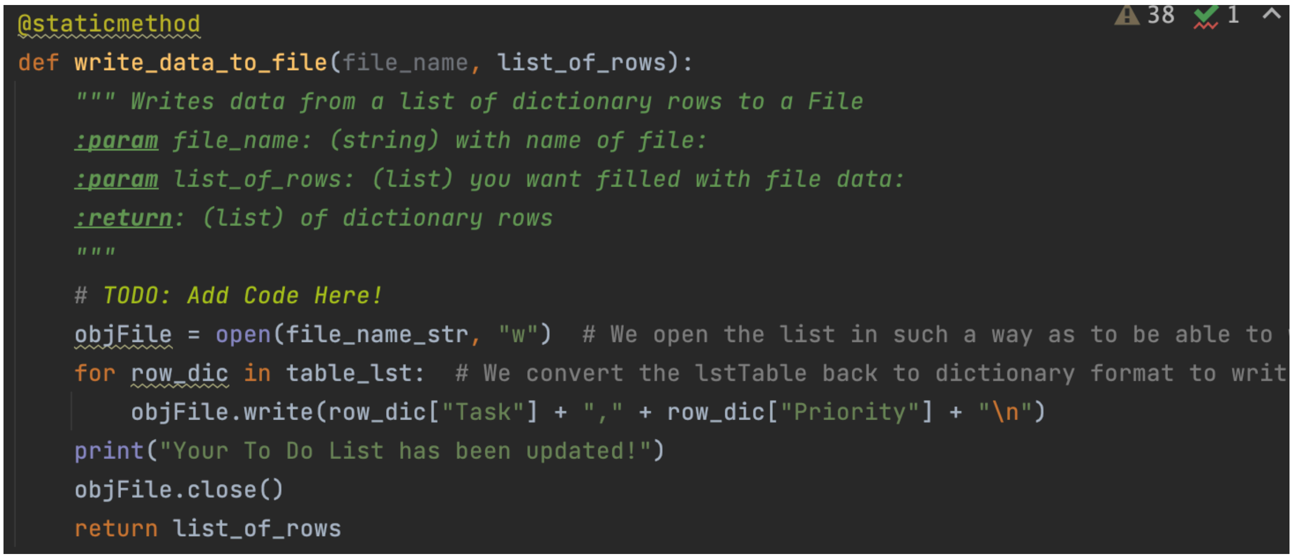
Writing Data to File: First we open the file in a writable format. Then each row in the list table, is converted to dictionary style, and this is written to file. A confirmation statement is provided to the user.

Figure 2: Remove Data from the List



```python
@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!
    if len(table_lst) == 0:
            print("The To Do List is empty.")
    else:
        for row in table_lst:
            if row["Task"].lower() == remTask.lower():
                table_lst.remove(row)
        print("Your To Do List has been edited!")
    return list_of_rows
```

Figure 3: Writing Data to File



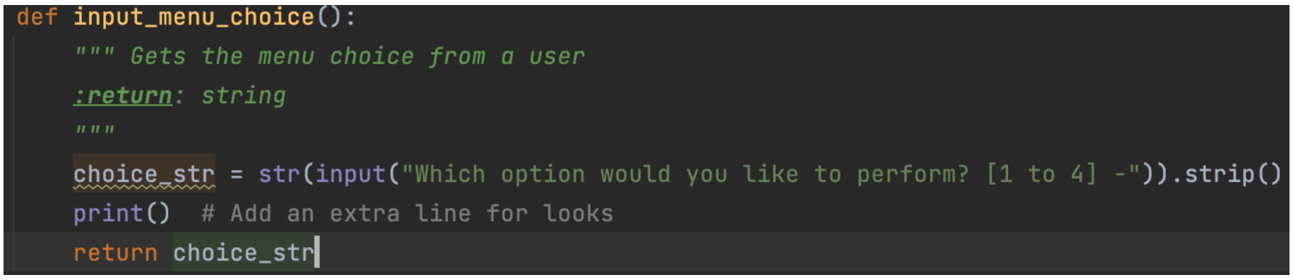
```python
@staticmethod
def write_data_to_file(file_name, list_of_rows):
    """ Writes data from a list of dictionary rows to a File
    :param file_name: (string) with name of file:
    :param list_of_rows: (list) you want filled with file data:
    :return: (list) of dictionary rows
    """
    # TODO: Add Code Here!
    objFile = open(file_name_str, "w")  # We open the list in such a way as to be able to
    for row_dic in table_lst:  # We convert the lstTable back to dictionary format to writ
        objFile.write(row_dic["Task"] + "," + row_dic["Priority"] + "\n")
    print("Your To Do List has been updated!")
    objFile.close()
    return list_of_rows
```

Topic: converting Input/Output Processes to Functions
The first function of printing the menu is provided in the prompt.

Collect user selection from menu: We link the variable “choice_str” to the input of the user with response to the prompt of selecting a menu option. The output of the function is this new variable.

Display current To Do List: This is provided for us in the prompt.

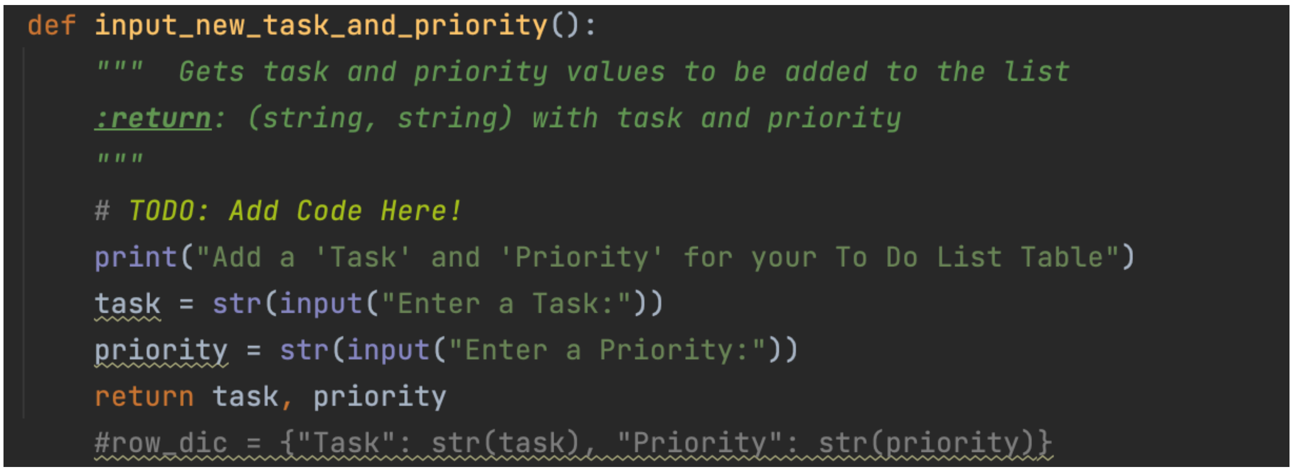
Figure 4: Collecting User Menu Input



```
def input_menu_choice():
    """ Gets the menu choice from a user
    :return: string
    """
    choice_str = str(input("Which option would you like to perform? [1 to 4] -")).strip()
    print()  # Add an extra line for looks
    return choice_str
```

Collecting User Task and Priority: A prompt is provided. The user input for task and priority are linked to variables and these variables are defined as the output of the function.

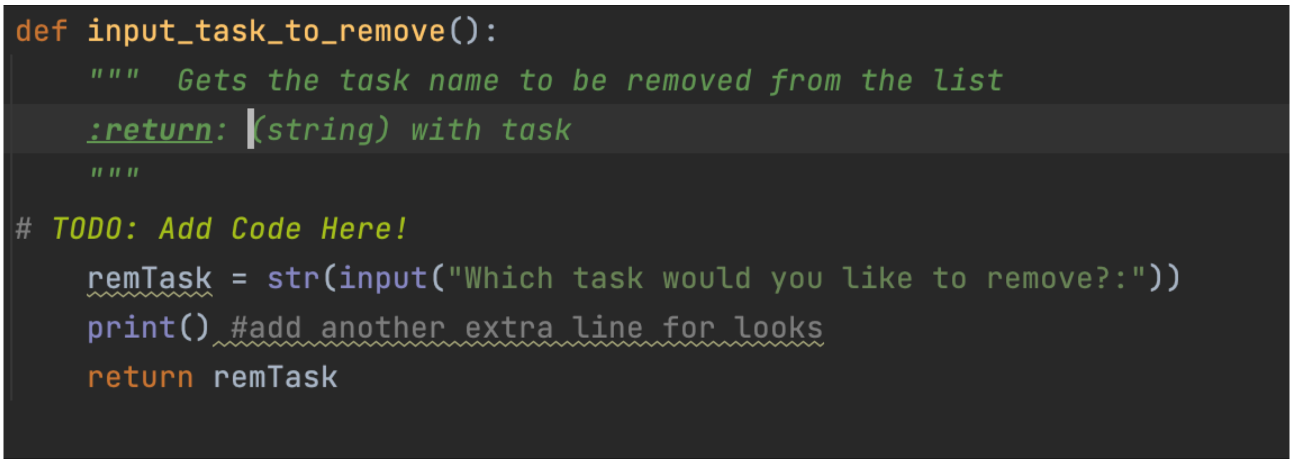
Figure 5: Collecting User Task and Priority



```
def input_new_task_and_priority():
    """  Gets task and priority values to be added to the list
    :return: (string, string) with task and priority
    """
    # TODO: Add Code Here!
    print("Add a 'Task' and 'Priority' for your To Do List Table")
    task = str(input("Enter a Task:"))
    priority = str(input("Enter a Priority:"))
    return task, priority
    #row_dic = {"Task": str(task), "Priority": str(priority)}
```

Collecting User Item to Delete: We link a new variable “remTask” to the string of the input of a prompt for the user to remove an item. The output from the function is this new variable.

Figure 6: Collecting User item to Delete



```
def input_task_to_remove():
    """  Gets the task name to be removed from the list
    :return: (string) with task
    """
# TODO: Add Code Here!
    remTask = str(input("Which task would you like to remove?:"))
    print() #add another extra line for looks
    return remTask
```

Topic: Main Script
The main script is provided by the prompt.

Topic: Debugging
The main issue I encountered in copying the script from the HTML on canvas was removing gremlins introduced by formatting - i.e. removing introduced returns from the comments. The next issue was ensuring that parameters defined in the functions are internally consistent (within the function) and externally consistent (within the main script).

Summary:
In this assignment, we demonstrated a different method for accomplishing the To Do List inventory task using functions. This results in a more concise script and limits the number of variables. The most challenging aspect of this assignment is making sure your parameters are consistent within the defined functions and the