```
# ------------------------------------------------------------------------
----- #
# Title: Assignment 06_Natta
# Description: Working with functions in a class,
#              When the program starts, load each "row" of data
#              in "ToDoFile.txt" into a python Dictionary.
#              Add each dictionary "row" to a python list "table"
# ChangeLog (Who,When,What):(Natta Panapornsirikun , 05/24/2023, Filling
the missing code)
# RRoot,1.1.2030,Created started script
# <Natta Panapornsirikun>,<05/24/2023>,Modified code to complete
assignment 06
# ------------------------------------------------------------------------
----- #

# Data ------------------------------------------------------------------
---- #
# Declare variables and constants
ToDoFile_str = "ToDoFile.txt"  # The name of the data 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
        try:
            file = open(file_name, "r")  # load each "row" of data
            for line in file:
                task, priority = line.strip().split(",")
                row = {"Task": task, "Priority": priority}
                list_of_rows.append(row)
            file.close()
        except FileNotFoundError:
            print("File not found. Creating a new file.")
        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
        """
        row = {"Task": str(task).strip(), "Priority":
str(priority).strip()}
        list_of_rows.append(row)
        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
        """
        list_of_rows = [row for row in list_of_rows if
row["Task"].lower() != task.lower()]
        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
        """
        file = open(file_name, "w")  # load each "row" of data
        for row in list_of_rows:
            task = row["Task"]
            priority = row["Priority"]
            file.write(f"{task},{priority}\n")
        file.close()
        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) Please add a new Task
```

```
        2) Please remove an existing Task
        3) Please Save Data to File
        4) Exit Program
        ''')
        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
    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 = str(input("Enter Task? - ")).strip()
        priority = str(input("Enter priority? - ")).strip()
        return task, priority

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

        :return: (string) with task
        """
        task = str(input("Enter Task you want to delete: - ")).strip()
        return task


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

# Step 1 - When the program starts, Load data from ToDoFile_str.
```

```
Processor.read_data_from_file(ToDoFile_str, table_lst)  # read file data

# Step 2 - Display a menu of choices to the user
while True:
    # 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=ToDoFile_str,
list_of_rows=table_lst)
        print("Data Saved!")
        continue  # to show the menu

    elif choice_str == '4':  # Exit Program
        print("Goodbye!")
```