

Jake Murphy

November 13, 2019

IT FDN 100

Assignment06

# Functions & Classes

## Contents

Introduction .....	2
Outline .....	2
Operation .....	3
Menu:.....	3
Show current task list:.....	3
Add a new task.....	3
Remove an existing task: .....	4
Save Data to File: .....	4
Reload Data from File: .....	5
Exit Program:.....	6
Summary .....	6
Script .....	7

## Introduction

This intent of this paper is to describe the procedures used in successfully performing this assignment.

## Outline

Given a “Starter-file” provided by the instructor, and building off the last five assignments, we (the students) are to create a new project in PyCharm that manages a “To Do File”. The file manages a user inputted table containing a task and priority column. We (the students) were to modify the starter-file in such a manner that would provide a clear and error free program.

The program presents the user with five options:

1. Show current data
2. Add a new item
3. Remove an existing item
4. Save data to file
5. Reload Data from File
6. Exit program

One of the main differences between Assignment05 and Assignment06 is the use of functions and classes. The goal was to create multiple functions to be called within different classes. The following is a walkthrough of the operation of the program.

# Operation

Menu:

When the program launches, the user is prompted with a menu with five options.

```
Menu of Options
1) Show current data
2) Add a new item.
3) Remove an existing item.
4) Save Data to File
5) Reload Data from File
6) Exit Program

Which option would you like to perform? [1 to 6] -
```

Show current task list:

If the user selects option 1, they are presented with the list data. If the list is empty, the list will display empty.

```
Which option would you like to perform? [1 to 6] - 1

***** The current items ToDo are: *****
aaa (low)
bbb (low)
*****

Menu of Options
1) Show current data
2) Add a new item.
3) Remove an existing item.
4) Save Data to File
5) Reload Data from File
6) Exit Program

Which option would you like to perform? [1 to 6] -
```

Add a new task

If the user wishes to add a new task to the to-do-list, they are presented with the following

```
Which option would you like to perform? [1 to 6] - 2

What is the task? - ccc
What is the priority? [high|low] - high
***** The current items ToDo are: *****
aaa (low)
bbb (low)
ccc (high)
*****

Menu of Options
1) Show current data
2) Add a new item.
3) Remove an existing item.
4) Save Data to File
5) Reload Data from File
6) Exit Program

Which option would you like to perform? [1 to 6] -
```

### Remove an existing task:

Here I have selected option 3 to remove an existing item. The user is asked “which task would you like removed”. I have chosen task “ccc”. The user is then prompted with “The task was removed” and the current items ToDo list is displayed along with the main menu.

```
Which option would you like to perform? [1 to 6] - 3
```

```
Which TASK would you like removed? - ccc  
The task was removed.
```

```
***** The current items ToDo are: *****  
aaa (low)  
bbb (low)  
*****
```

```
Menu of Options  
1) Show current data  
2) Add a new item.  
3) Remove an existing item.  
4) Save Data to File  
5) Reload Data from File  
6) Exit Program
```

### Save Data to File:

For this example, I have reloaded the task “ccc” with “High” priority to the list. I have then selected option 4 “Save data to file”.

Once option 4 is selected, the current items ToDo list is displayed once more. The user is then prompted to save this data with an input of y or n.

If the user selects y, they are greeted with “Data saved to file!” and asked for an input to return to the main menu.

```
Which option would you like to perform? [1 to 6] - 2
```

```
What is the task? - ccc  
What is the priority? [high|low] - high  
***** The current items ToDo are: *****  
aaa (low)  
bbb (low)  
ccc (high)  
*****
```

```
Menu of Options  
1) Show current data  
2) Add a new item.  
3) Remove an existing item.  
4) Save Data to File  
5) Reload Data from File  
6) Exit Program
```

```
Which option would you like to perform? [1 to 6] - 4
```

```
***** The current items ToDo are: *****  
aaa (low)  
bbb (low)  
ccc (high)  
*****
```

```
Save this data to file? (y/n) - y  
Data saved to file! Press the [Enter] key to return to menu.
```

Had the user inputted ‘n’, they would be greeted with the following message and input option:

```
Save this data to file? (y/n) - n  
New data was NOT Saved, but previous data still exists! Press the [Enter] key to return to menu.
```

## Reload Data from File:

In the event the user wishes to reload the file data, they would select option 4. For this example, I have added another task “ddd” with a high priority.

I did not save this new data to file, so when I select option 5 to “Reload Data from File” I am prompted with a warning

“Warning: This will replace all unsaved changes. Data loss may occur!”

I am then given the option to reload my data via user input y or n.

I selected ‘y’ and as you can see on the right, the list no longer has the task “ddd” and its associated priority level.

I am then sent back to the main menu.

```
Which option would you like to perform? [1 to 6] - 2

What is the task? - ddd
What is the priority? [high|low] - high
***** The current items ToDo are: *****
aaa (low)
bbb (low)
ccc (high)
ddd (high)
*****

Menu of Options
1) Show current data
2) Add a new item.
3) Remove an existing item.
4) Save Data to File
5) Reload Data from File
6) Exit Program

Which option would you like to perform? [1 to 6] - 5

Warning: This will replace all unsaved changes. Data loss may occur!
Reload file data without saving? [y/n] - y
***** The current items ToDo are: *****
aaa (low)
bbb (low)
ccc (high)
*****

Menu of Options
1) Show current data
2) Add a new item.
3) Remove an existing item.
4) Save Data to File
5) Reload Data from File
6) Exit Program

Which option would you like to perform? [1 to 6] -
```

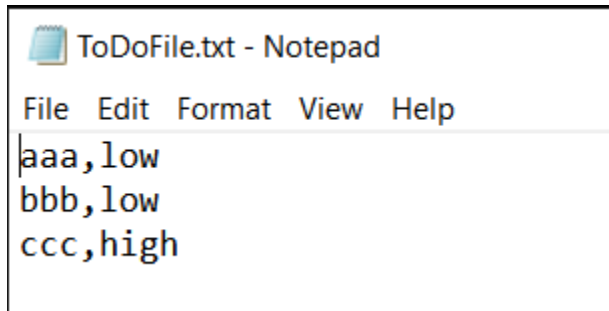
Had the user inputted ‘n’ which tells the program to not load the last data list, they would see a screen as such:

```
Warning: This will replace all unsaved changes. Data loss may occur!
Reload file data without saving? [y/n] - n
File data was NOT reloaded! Press the [Enter] key to return to menu.
***** The current items ToDo are: *****
aaa (low)
bbb (low)
ccc (high)
ddd (high)
*****
```

Exit Program:

When the user chooses the last option to exit the program, the program immediately closes. Note: any unsaved information will be lost.

Since I did not save the last list item “ddd” before exiting, my ToDoFile.txt only displays the following:



```
ToDoFile.txt - Notepad
File Edit Format View Help
aaa,low
bbb,low
ccc,high
```

## Summary

Classes and functions may come naturally to some, but it takes patience. The most important lesson I have learned throughout the process has been variables within classes, locally or globally and their effect on functions. Make sure you map out your variables accordingly, otherwise it can become time consuming. The option to run in debug mode through Pycharm, helped elevate some of the issues that would have prevented me from completing and submitting this assignment on time.

## Script

```
!..
strFileName = "ToDoFile.txt" # The name of the data file
objFile = None # An object that represents a file
dicRow = {} # A row of data separated into elements of a dictionary {Task,Priority}
lstTable = [] # A dictionary that acts as a 'table' of rows
strData, strChoice, strTask, strPriority = "", "", "", "" # Capture the user options

# Data ----- #
# Processing ----- #
class FileProcessor:
    """ Processing the data to and from a text file """

    @staticmethod
    def ReadFileDataToList(file_name, list_of_rows):
        """
        Desc - 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
        """
        file = open(file_name, "r")
        for line in file:
            data = line.split(",")
            row = {"Task": data[0].strip(), "Priority": data[1].strip()}
            list_of_rows.append(row)
        file.close()
        return list_of_rows

    @staticmethod
    def WriteListDataToFile(file_name, list_of_rows):
        """
        Desc - Writes data from a file into a List of dictionary rows

        :param file_name: (string) with name of file:
        :param w: writes data to file
        :return: pass
        """
        objFile = open(file_name, "w")
        for dicRow in list_of_rows: # Write each row of data to the file
            objFile.write(dicRow["Task"] + "," + dicRow["Priority"] + "\n")
        objFile.close()

pass
```

```

@staticmethod
def ReloadDataToFile(strFileName, lstTable):
    """
    Desc - Reloads Last saved List

    :param strFileName: (string) with name of file:
    :param lstTable: (string) saved file data
    :return: main menu
    """

    lstTable.clear() # Added to fix bug 1.1.2030
    FileProcessor.ReadFileDataToList(strFileName, lstTable) # Replace the current List data with file data
    IO.ShowCurrentItemsInList(lstTable) # Show current data in the List/table


@staticmethod
def RemoveData(strKeyToRemove):
    """
    Desc - User selects key to remove from List

    :param strKeyToRemove: (string) Name of key
    :return: main menu
    """

    intRowNumber = 0 #sets counter to zero to identify key to be removed
    blnItemRemoved = False #creates a boolean flag for Loop
    while intRowNumber < len(lstTable): #searches through Lsttable for key
        if strKeyToRemove == str(list(dict(lstTable[intRowNumber]).values())[0]): # Search current row column 0
            del lstTable[intRowNumber] # Delete the row if a match is found
            blnItemRemoved = True # Set the flag so the Loop stops
            intRowNumber += 1 # Increase counter to get next row

    if blnItemRemoved:
        print("The task was removed.")
    else:
        print("I'm sorry, but I could not find that task.")
    print() # Add an extra Line for Looks


# Processing ----- #

```



---

```

# Presentation (Input/Output) ----- #
class IO:
    """ A class for perform Input and Output """

    @staticmethod
    def OutputMenuItems():
        """ Display a menu of choices to the user
        :return: nothing
        """
        print('''
        Menu of Options
        1) Show current data
        2) Add a new item.
        3) Remove an existing item.
        4) Save Data to File
        5) Reload Data from File
        6) Exit Program
        ''')
        print() # Add an extra Line for Looks

    @staticmethod
    def InputMenuChoice():
        """ Gets the menu choice from a user
        :return: string
        """
        choice = str(input("Which option would you like to perform? [1 to 6] - ")).strip()
        print() # Add an extra Line for Looks
        return choice

    @staticmethod
    def ShowCurrentItemsInList(list_of_rows):
        """ Shows the current items in the List of dictionaries rows

        :param list_of_rows: (List) of rows you want to display
        :return: nothing
        """
        print("***** The current items ToDo are: *****")
        for row in list_of_rows:
            print(row["Task"] + " (" + row["Priority"] + ")")
        print("*****")
        print() # Add an extra Line for Looks

```

```

@staticmethod
def AddRowToList(task, priority, list_of_row):
    """
    Desc- User inputs new item to add to List

    :param list_of_row: (List) of rows you want to display
    :return: nothing
    """
    task = str(input("What is the task? - ")).strip() # Get task from user
    priority = str(input("What is the priority? [high|low] - ")).strip() # Get priority from user
    dicRow = {"Task": task, "Priority": priority} # Create a new dictionary row
    list_of_row.append(dicRow) # Add the new row to the List/tab
    #-----Called Current Items function from above-----#
    IO.ShowCurrentItemsInList(lstTable)

    # TODO: Create more functions that perform various IO tasks as needed

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

# Main Body of Script ----- #

# Step 1 - When the program starts, Load data from ToDoFile.txt.
FileProcessor.ReadFileDataToList(strFileName, lstTable) # read file data

# Step 2 - Display a menu of choices to the user
while (True):
    IO.OutputMenuItems() # Shows menu
    strChoice = IO.InputMenuChoice() # Get menu option

    # Step 3 - Process user's menu choice
    # Step 3.1 Show current data
    if strChoice.strip() == '1':
        IO.ShowCurrentItemsInList(lstTable) # Show current data in the List/table
        continue # to show the menu

    # Step 3.2 - Add a new item to the List/Table
    elif strChoice.strip() == '2':
        print() # Add an extra line for looks
        # ***** Converted to Function *****
        IO.AddRowToList(strTask, strPriority, lstTable)
        print() # Add an extra line for looks
        continue # to show the menu

    # Step 3.3 - Remove a new item to the List/Table
    elif strChoice == '3':

        # Step 3.3.a - Ask user for item and prepare searching while loop
        strKeyToRemove = input("Which TASK would you like removed? - ") # get task user wants deleted
        # ***** Converted to Function *****
        FileProcessor.RemoveData(strKeyToRemove)
        IO.ShowCurrentItemsInList(lstTable) # Show current data in the List/table
        continue # to show the menu

```

```

# ***** Converted to Function
FileProcessor.RemoveData(strKeyToRemove)
IO.ShowCurrentItemsInList(lstTable) # Show current data in the List/table
continue # to show the menu

# Step 3.4 - Save tasks to the ToDoFile.txt file
elif strChoice == '4':

    # Step 3.4.a - Show the current items in the table
    IO.ShowCurrentItemsInList(lstTable) # Show current data in the List/table

    # Step 3.4.b - Ask if user if they want save that data
    if "y" == str(input("Save this data to file? (y/n) - ")).strip().lower(): # Double-check with user

# ***** Converted to Function
FileProcessor.WriteListDataToFile(strFileName, lstTable)
input("Data saved to file! Press the [Enter] key to return to menu.")
else: # Let the user know the data was not saved
    input("New data was NOT Saved, but previous data still exists! Press the [Enter] key to return to menu.")
    continue # to show the menu

# Step 3.5 - Reload data from the ToDoFile.txt file (clears the current data from the List/table)
elif strChoice == '5':
    print("Warning: This will replace all unsaved changes. Data loss may occur!") # Warn user of data Loss
    strYesOrNo = input("Reload file data without saving? [y/n] - ") # Double-check with user
    if strYesOrNo.lower() == 'y':

# ***** Converted to Function
FileProcessor.ReloadDataToFile(strFileName, lstTable)
else:
    input("File data was NOT reloaded! Press the [Enter] key to return to menu.")
    IO.ShowCurrentItemsInList(lstTable) # Show current data in the List/table
    continue # to show the menu

# Step 3.6 - Exit the program
elif (strChoice == '6'):
    break # and Exit

# Main Body of Script ----- #

```