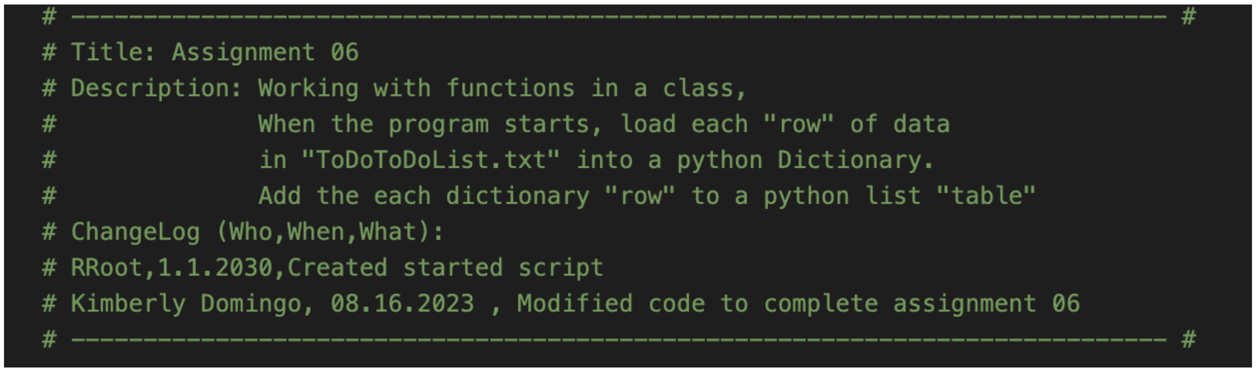
Kimberly Domingo
Date: 8/16/2023
Class: IT FDN 110A
Assignment 06

Introduction:

This week's topic is functions. Assignment06 deals with our To Do List program that we have created with last week's assignment. To clean up our program, we utilize the concept of functions and classes. I will go through the changes that I have made to the starter code to finish this assignment.

Writing/Editing the Program:

We were luckily given a starter code to work with for this assignment. We needed to modify the



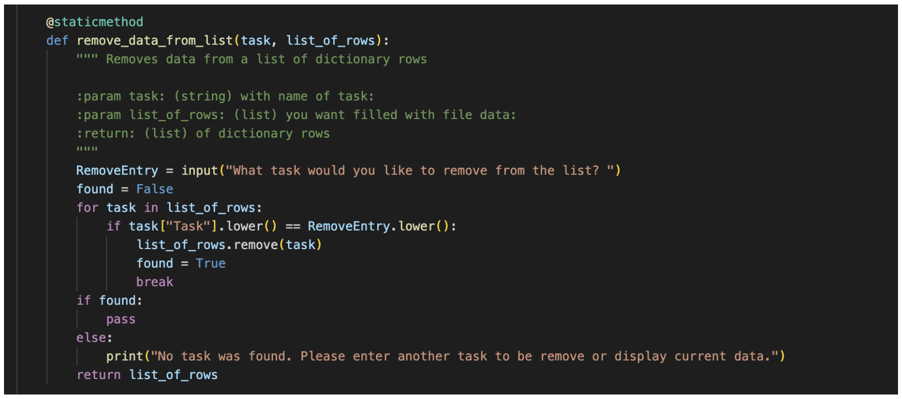
```
# ---------------------------------------------------------------------------- #
# Title: Assignment 06
# Description: Working with functions in a class,
#              When the program starts, load each "row" of data
#              in "ToDoToDoList.txt" into a python Dictionary.
#              Add the each dictionary "row" to a python list "table"
# ChangeLog (Who,When,What):
# RRoot,1.1.2030,Created started script
# Kimberly Domingo, 08.16.2023 , Modified code to complete assignment 06
# ---------------------------------------------------------------------------- #
```

Figure 1: Starter code for Assignment06

Program to make it functional. In comparison to Assignment05, we see that the sections were already broken down into different classes, processing and presentation. I would classify them as our front end and back end of the program to make it easier for myself to understand the two different classes.

It is clearly marked on which part of the program that is missing codes. I started with the processing class. I have used what I did in the last assignment to fill in the missing codes. Most were the same.

However, I ran into a little bit of a weird error when I was writing my code for removing data.

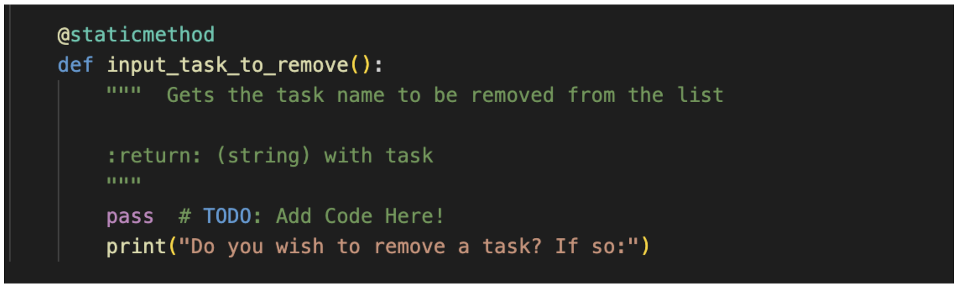


```
@staticmethod
def remove_data_from_list(task, list_of_rows):
    """ Removes data from a list of dictionary rows

    :param task: (string) with name of task:
    :param list_of_rows: (list) you want filled with file data:
    :return: (list) of dictionary rows
    """
    RemoveEntry = input("What task would you like to remove from the list? ")
    found = False
    for task in list_of_rows:
        if task["Task"].lower() == RemoveEntry.lower():
            list_of_rows.remove(task)
            found = True
            break
    if found:
        pass
    else:
        print("No task was found. Please enter another task to be remove or display current data.")
    return list_of_rows
```

Figure 2: Remove data function

Inside my remove data function, I initially added in the input function to check what the user wants to remove. However, there's already an input in the remove data function inside the Presentation class. Now, the program is asking the users to input the task they wanted to remove twice. If I remove the input function inside the remove data function in the Processing class, it will cause an error. To get around this, I ended up adding a print statement in the remove data function in the Presentation class.

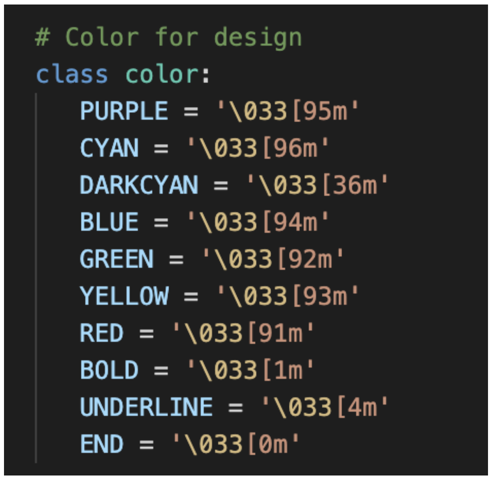


```
@staticmethod
def input_task_to_remove():
    """  Gets the task name to be removed from the list

    :return: (string) with task
    """
    pass  # TODO: Add Code Here!
    print("Do you wish to remove a task? If so:")
```

Figure 3: Print function

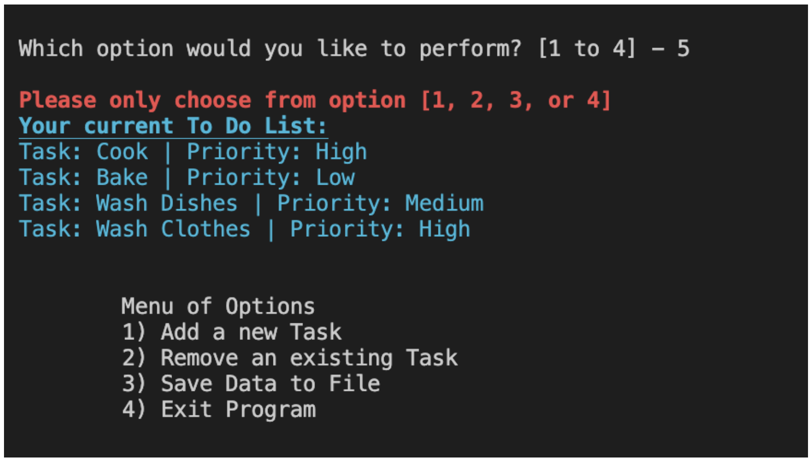
I have decided to further optimize the program by adding some other features. I added a class that deals with changing the colors and formatting of the output. The codes for these can be



```
# Color for design
class color:
   PURPLE = '\033[95m'
   CYAN = '\033[96m'
   DARKCYAN = '\033[36m'
   BLUE = '\033[94m'
   GREEN = '\033[92m'
   YELLOW = '\033[93m'
   RED = '\033[91m'
   BOLD = '\033[1m'
   UNDERLINE = '\033[4m'
   END = '\033[0m'
```

Figure 4: Text format and color codes

Taken from google. Since they are inside a class. To use these codes, you would need to type in **color.PURPLE + “String” + color.END.** Like inside our while loop, to call the functions inside a specific class, we need to call our class name then the function we want. Notice that you need the color.END at the end of the text you want to format.



```
Which option would you like to perform? [1 to 4] - 5

Please only choose from option [1, 2, 3, or 4]
Your current To Do List:
Task: Cook | Priority: High
Task: Bake | Priority: Low
Task: Wash Dishes | Priority: Medium
Task: Wash Clothes | Priority: High


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

Figure 5: Output of texts with color format

Conclusion:

In this assignment, we went over how to use functions and classes to clean up our To Do List program. We can see that these are a great tool in making our programs more organized and cleaner. This is very helpful especially when other people are reading our code.