# Morgan Farmer

# August 6nd, 2021

# Python Course

# Assignment05

[https://github.com/morganfarmer/IntroToProg-python/blob/main/Assigment05\_Starter.py](#_Assignment05)

Task & Priority Program

Introductory

In this program I will be explaining how I was able to edit the assignment starter file provided to perform the functions that were noted in pseudo code.

Header and Declaring Variables and Constants

With the header it is very straight forward, I put In my name and the date. Declaring the variables and constants were also straight forwards because they were already in listed in the program.

objFile = "ToDoList.txt"

dicRow = {'Task': 'Dishes', 'Priority': 'Pay Rent'}

lstTable = ['Tasks, Priority']

strMenu = "Menu of Choices"

strChoice = ""

Processing

When the program starts, I open the text file that I want the table and dictionary I want to save to, to do that you use the open function. “w” stands for write, writing to the file.

objFile = open('ToDoList.txt','w')

Input/Output

First the program will display the menu to user using a print function and if statement. Starting the program this will be the first thing the user sees.

while (True):  
 print("""  
 Menu of Options  
 1) Show current data  
 2) Add a new item.  
 3) Remove an existing item.  
 4) Save Data to File  
 5) Exit Program  
 """)

If the user decides to go with option number one. I print an if statement, use the sting choice statement and strip the out any unwanted spaces. Next open the to do list text file and note the will be reading the current data. I used a print function for flair just to state that hey, I’m showing you a preview of your current items. Last for this step I put in the dictionary row function to display the tasks and priorities and close the file. That is the end of the first option.

if (strChoice.strip() == '1'):  
 # *TODO: Add Code Here* objFile = open('ToDoList.txt', 'r')  
 print('Previewing Current Items...')  
 dicRow = {'Task': 'Dishes', 'Priority': 'Pay Rent'}  
 print(dicRow)  
 objFile.close()  
 continue

Option number two, I opened the file once again but instead of reading it is writing so I use a ‘w’. I’m going to be honest, I’m not sure if all the lines I used were in correct order, but I know that when I run the program the option works. I based the function order by lab 5-2.

elif (strChoice.strip() == '2'):  
 # *TODO: Add Code Here* objFile = open('ToDoList.txt', 'w')  
 dicRow = {'Task': 'Dishes', 'Priority': 'Pay Rent'}  
 objFile.write(dicRow['Task'] + ',' + dicRow['Priority'] + '\n')  
 lstRow = ['Task','Priority']  
 print('Enter new task or priority', '\n')  
 objFile.write(input())  
 print('New data saved!')  
 objFile.close()  
 continue

Option number three, remove an item or list. Remember we declared the file variable at the beginning, so we won’t have to do it again down here at the bottom of the program. Open the file to write into. List your “list”. I used the delete function and the closed the text file. For extra flair I put in a print function that states the items are removed.

elif (strChoice.strip() == '3'):  
 # *TODO: Add Code Here* strFile = 'ToDoList.txt'  
 objFile = open(strFile, 'w')  
 lstRow = ['Dishes','Pay rent']  
 del lstRow[1]  
 objFile.close()  
 print('Items Removed')  
 continue

Option number four, save tasks to the text file. Repeat the first step of option two. Open the text file list the row that needs to be saved, write the list to the file and then add a message to the user stating that the tasks are saved.

elif (strChoice.strip() == '4'):  
 # *TODO: Add Code Here* strFile = 'ToDoList.txt'  
 objFile = open(strFile, 'w')  
 lstRow = ['Dishes']  
 objFile.write(lstRow[0])  
 objFile.close()  
 print('Tasks Saved')  
 continue

Option number five was simple. Add print statement saying that the program has ended. This option ends with a break instead of a continue so it will automatically end the program when running.

elif (strChoice.strip() == '5'):  
 # *TODO: Add Code Here* print('Program has ended!')  
 break

Conclusion

Filling in the blanks of a code program helped me stay on the right path, but there is a good side and a down side to everything. Good, would be the fact that without the pseudo code I wouldn’t know what the intentions of some things that were put into the program. The pseudo code helped me figure exactly was missing and which allowed me to piece everything together like a puzzle. Downside would be the fact that if I wanted to change things around, I wouldn’t know how to. As I am getting further into this course, I am realizing that python has a style to it. All code does not have to be the same. Other than that, that is the only downside.