Gibran D. Washington

Professor Randal Root

ITN FDN 100: Assignment05

6 November 2016

User Program, *Voodoo ToDo*, Updateable ToDo List Text File

The program has four primary parts that allows a user to create a dictionary based task list in a text file. There were a few challenging parts that provided a number of issues with completing the outcomes required by the program. Below are the four primary sections of the program:

1. Read line/s from a text file and place the 2-dimensional string data in a dictionary.
2. Provide the user with a menu selection for the program.
3. Save a new tasks to dictionaries and save to original text file
4. Allow user to delete a task from dictionary

Initially I thought it would be best to build from the bottom up instead of from the top down but I actually had more code and extras that were not necessary. My first section reading lines from the text file didn’t work well with a ***for in loop*** statement but when I included a ***readlines()*** line from an index it seemed to work sufficiently. I also opened an empty list and an empty dictionary variable to hold my new comma separated Task and Priority. In Figure 1, I include the code for part 1 along with the text file operations.

TDL = open("C:\\\_PythonClass.\\ToDo1.txt", "r")  
#-- Data --#  
# declare variables and constants  
taskLog =[]  
dictList = {}  
  
**for** line **in** TDL:  
 print(TDL.readlines()[0])  
 dictList = {"Pay Bills":"high"}  
 **for** key, value **in** dictList.items():  
 print("\nThis is a sample task in the Voodoo ToDo Program:", key)  
 print("\nThis is the corresponding level of priority for the task:", value)  
 **continue**# close file from read status  
TDL.close()

Figure 1. Intro/section 1 reading one string line from text and printing the corresponding line for user to see data.

I decided not to iterate over each line in the file or use two statements based on some inconsistencies in what was being retrieved despite being aware of the 0 starting point. I think that putting all the strings for both of the sample Task/Priority data on the same line would have made easier to accomplish. The output from the above code is shown in Figure 2 along with the print statement output for the title name and description of the program.

In Figure 2, the output code also includes the part 2 of the Voodoo ToDo program with the user menu for allowing the dictionary to be appended or key value pairs to be deleted from variable **dictList.**  The menu has all the choices required with a **try… except** block to capture any errand input from user input. Initially in an earlier version, the **try... except** included the entire block of code along with the print statements for the menu.

Voodoo ToDo

This Program is designed to help end the bad habits of not writing down tasks or completing

task based on the level of importance you assign. It stores your task list in a file for portability.

##############################################################################

Pay Bills, high

This is a sample task in the Voodoo ToDo Program: Pay Bills

This is the corresponding level of priority for the task: high

Voodoo ToDo Menu Options:

Press 0 - Exit Without Saving

Press 1 - to Add task

Press 2 - to Remove task

Press 3 - to Save all tasks to the ToDo1.txt file and exit!

Please Make a Selection:

Figure 2. Output from first part of the program and assignment for Module 5.

As a user friendly gesture, I allowed the except statement to give the user a second chance to input the proper value to continue in program rather than end program immediately.

# setup for list dictionary array  
taskHeaders = ["Task", "Priority"]  
taskLog.insert(0,taskHeaders)  
taskLog = [dictList]  
  
#-- Processing --#  
# perform task for user menu selection  
select = **None  
while** select != "0":  
  
 print(  
 '''  
 Voodoo ToDo Menu Options:  
  
 Press 0 - Exit Without Saving  
 Press 1 - to Add task  
 Press 2 - to Remove task  
 Press 3 - to Save all tasks to the ToDo1.txt file and exit!  
 '''  
 )  
# -- Presentation (I/O) --#  
# get user menu option  
 **try**:  
 select = int(input("Please Make a Selection: "))  
 **except**:  
 print("You did not enter a valid #0 to 3 from menu\n")  
 select = int(input("Please Make a Selection: "))  
 # Exit Program Without Saving  
 **if** select == "0":  
 print("Exiting the Program!")  
 **break**

Figure 3. Part 2 of program code that displays menu and accepts user input for four choices.

There is not much i/o at this third part of the program, Figure 3, but the last else sequence for user selection 1, 2, and 3 was the most challenging part for me to do in a simple yet logical manner using the code knowledge I have learned thus far in class. I would expect that a far more refined and polished program would provide a few locations to that not only give feedback to the user on their selection but would also include statements to re-route code to other lines or ***method()*** or ***functions()*** that would execute to remove ambiguity or provide further input to move user through the program in clean efficient manner.

# add a new task to the dictionary  
 **if** select == "1":  
 addTask = input("\nWhat is the new task to be added?: ")  
 **if** addTask **not in** dictList:  
# allows user to define priority level of new task  
 addPriority = input("\nPlease Type the Priority level: \nHigh\nMedium\nLow\n")  
 **if** (addPriority.lower() == "y"):  
 dictList[addTask] = addPriority  
 print("\n",addTask,"has been added to the list.\n")  
 **else**:  
 print("\nThe task is already in list! Use different name for the task.")  
  
  
  
# deletes a user selected task from table  
 **elif** select == "2":  
 addTask = input("Which Task Would You Like to Remove?"  
 "You must type the complete task name/description!",dictList,"\n")  
 **if** addTask **in** dictList:  
 **del** dictList[addTask]  
 print("\nAlright, The task--", addTask, "--has been deleted from list")  
 **else**:  
 print("\nSorry, there was a problem--",addTask,"--was not found. Please check spelling.")  
  
  
  
# Saves all task to file and exits program  
 **elif** select == "3":  
 strUserData = input("Press Any Key to Save Task and Exit Program: ")  
# opens text file for writing  
 TDL = open("C:\\\_PythonClass.\\ToDo1.txt", "w")  
 print("\nYour Task Data is Saving to ToDo1.txt")  
# saves the list dictionary with new task appended  
 TDL.write(taskLog, "\n")  
   
# close write file ToDo.txt  
TDL.close()

Figure 4. Last and 4th section of code that appends, deletes, and saves task to text file.

This report does not include output as is the normal and standard discourse for showing results and logic flow for a program. I expect to go back and make some changes to confirm that it runs and also to add additional lines that improve the stability.

Works Cited

Dawson, Michael. Python Programming for the Absolute Beginner, 3rd ed. Boston: Cengage Learning PTR, 2011.

Root, Randal. Python Videos: Python Module 05-01 to 05-13 *YouTube, uploaded by Randal Root,* 25 July 2015. <https://www.youtube.com/playlist>[list=PLfycUyp06LG9OVfidlxwfjz7JYRhq83PN](https://www.youtube.com/playlist?list=PLfycUyp06LG9OVfidlxwfjz7JYRhq83PN#_blank)

Summerfield, Mark. *Programming in Python 3, A complete Introduction to the Python Language*, 2nd ed. Boston Pearson Education Inc., 2010.

James, Joe. *Python: Data Structures - Lists, Tuples, Sets & Dictionaries*

<http://www.youtube.com/watch?v=R8mOaSIHT8U&feature=related>. Retrieved October 20, 2016.

Tutorials Point

<http://www.tutorialspoint.com/python/python_tuples.htm>. Retrieved October 20, 2016

ComputerScienceUK. *12 – Python Coding – Reading and Writing to Text Files using Lists –www ComputerScienceUK com,* <https://www.youtube.com/watch?v=Olq86NN6-Ko>. World Wide Web, Retrieved November 1, 2016.

WikiBooks

<https://en.wikibooks.org/wiki/Non-Programmer%27s_Tutorial_for_Python_2.6/Lists>. World Wide Web, Retrieved November 1, 2016.

BoBo To Bogo

<http://www.bogotobogo.com/python/python_dictionaries_tuples.php>. World Wide Web, Retrieved November 1, 2016.