Name: Amruth D

Date: 17thMay2023

Course: IT FDN 110 A

Assignment 05

The 'To DO List' Script (Python)

Introduction

In this assignment, we will understand, what is the difference between 'Lists' and 'Dictionary'. And, finally we will go through a python script using Lists and Dictionaries' to create an interactive TO-DO List. Also, we will learn how to add and remove from a list and finally update the saved version to the text file.

Lists, Dictionaries and their differences

Lists:

A list is an ordered collection of items enclosed in square brackets []. It can contain elements of different types such as integers, strings, or even other lists. Lists are mutable, meaning you can change, add, or remove elements once they are created.

```
my_list = [1, "apple", True, 3.14]  # a List are defined using Square Brackets
```

Fig 1.0 Lists Example

Dictionary:

A dictionary is an unordered collection of key-value pairs enclosed in curly braces {} or created using the **dict()** constructor. Each key-value pair in a dictionary is separated by a colon ':', and keys are unique within a dictionary. Dictionaries are also mutable.

```
person = {
    'name': 'Amruth',
    'age': 30,
    'city': 'Bengaluru'
}
```

Fig 2.0 Dictionary Example

Difference between List and Dictionary

Structure:

A list is an ordered collection of elements. Each element in a list is assigned a unique index based on its position in the list. Lists are enclosed in square brackets [].

A dictionary is an unordered collection of key-value pairs. Each key is unique within a dictionary, and it is used to access its corresponding value. Dictionaries are enclosed in curly braces {} or created using the dict() constructor.

Data Organization:

Elements in a list are accessed by their index, starting from 0. The order of elements in a list is preserved, and you can access, modify, add, or remove elements using their index.

Values in a dictionary are accessed by their keys. The keys serve as unique identifiers for the values. The order of elements in a dictionary is not preserved, and you can access, modify, add, or remove values using their keys.

Mutable vs. Immutable:

Lists are mutable, which means you can change the values of individual elements, add new elements, or remove existing elements.

Dictionaries are also mutable. You can modify the values associated with keys, add new key-value pairs, or remove existing key-value pairs.

Use Cases:

Lists are suitable for storing ordered collections of data where the order matters. For example, a list can be used to store a sequence of numbers, a list of names, or a series of steps in a process.

Dictionaries are ideal when you want to associate values with specific keys or when you need to retrieve values based on their identifiers. Dictionaries are commonly used for tasks such as storing user information, mapping data, or representing real-world objects with attributes.

Writing The TO-DO Python Script

In this assignment, we will be working with a partially created script. This was exciting as well as challenging. Though the pre written script is an advantage, it was also little tricky to start with.

```
# ------------------------------------------------------------------------ #
# Title: Assignment 05
# Description: Working with Dictionaries and Files
#              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
# Amruth D,17thMay2023,Added code to complete assignment 5
# ------------------------------------------------------------------------ #
```

Fig 3.0 Title Block in PyCharm

A "Title Block" typically refers to a section of comments at the beginning of the script that provides information about the script, such as its purpose, author, and date of creation.

The title block should be placed at the beginning of the script, before any code is executed.

It can include any information that is relevant to the script and its intended use. This information can be useful for other developers who may need to work with or modify the script in the future.

```
# -- Data -- #
# declare variables and constants
objFile = "ToDoList.txt"   # An object that represents a file
strData = ""  # A row of text data from the file
dicRow = {}    # A row of data separated into elements of a dictionary {Task,Priority}
lstTable = []  # A list that acts as a 'table' of rows
strMenu = ""   # A menu of user options
strChoice = ""# A Capture the user option selection
```

Fig 4.0 Declaring Variables and Constants

The code begins by declaring variables and constants such as objFile (the file name), strData (a string to store data from the file), dicRow (a dictionary to hold task and priority data), lstTable (a list of dictionaries), strMenu (a string for the menu options), and strChoice (a string to capture the user's choice).

```
# -- Processing -- #
# Step 1 - When the program starts, load any data you have
# in a text file called ToDoList.txt into a python list of dictionaries rows (like Lab 5-2)
try:
    with open(objFile, "r") as file:
        for line in file:
            task, priority = line.strip().split(",")
            dicRow = {"Task": task, "Priority": priority}D
            lstTable.append(dicRow)
except FileNotFoundError:
    print("No existing data file found. Starting with an empty list.")
```

Fig 5.0 Loading Data from File

Here the script attempts to open the file specified by objFile in read mode ("r"). It then reads each line from the file, splits it into task and priority using the split() method, creates a dictionary dicRow with keys "Task" and "Priority" and corresponding values, and appends this dictionary to the lstTable. If the file is not found, it prints a message indicating 'No existing data file is found'.

```
# -- Input/Output -- #
# Step 2 - Display a menu of choices to the user
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
    """)
    strChoice = str(input("Which option would you like to perform? [1 to 5] - "))
    print()  # adding a new line for looks
```

Fig 6.0 Displaying a menu of choice to the user

Then the script enters a while loop to display a menu of options to the user. The user's choice is stored in strChoice. Based on the choice, the corresponding actions are performed.

```
# Step 3 - Show the current items in the table
if (strChoice.strip() == '1'):
    if not lstTable:
        print("No data available.")
    else:
        print("Current Data:")
        for row in lstTable:
            print(f"Task: {row['Task']}, Priority: {row['Priority']}")
    continue
```

Fig 7.0 Shows the current items in the table

If the user chooses option '1', the code checks if **lstTable** is empty. If it is, it prints a message indicating no data is available. Otherwise, it iterates over each dictionary in **lstTable** and prints the task and priority.

```
# Step 4 - Add a new item to the list/Table
elif (strChoice.strip() == '2'):
    task = input("Enter the task: ")
    priority = input("Enter the priority: ")
    dicRow = {"Task": task, "Priority": priority}
    lstTable.append(dicRow)
    print("Task added.")
    continue
```

Fig 8.0 Adds new item to the list

If the user chooses option '2', the script prompts the user to enter a task and priority. It creates a new dictionary **dicRow** with keys "Task" and "Priority" and the entered values. This dictionary is then appended to **lstTable**, and a message is printed to indicate the task was added.

```
# Step 5 - Remove a new item from the list/Table
elif (strChoice.strip() == '3'):
    if not lstTable:
        print("No data available.")
    else:
        task = input("Enter the task to remove: ")
        found = False
        for row in lstTable:
            if row["Task"].lower() == task.lower():
                lstTable.remove(row)
                found = True
                break
        if found:
            print("Task removed.")
        else:
            print("Task not found.")
    continue
```

Fig 9.0 Removes new item to the list

If the user chooses option '3', the script checks if **lstTable** is empty. If it is, it prints a message indicating no data is available. Otherwise, it prompts the user to enter a task to remove. It iterates over each dictionary in **lstTable** and compares the entered task with the task in each dictionary. If a match is found, the corresponding dictionary is removed from **lstTable**. If no match is found, it prints a message indicating the task was not found.

```
# Step 6 - Save tasks to the ToDoToDoList.txt file
elif (strChoice.strip() == '4'):
    with open(objFile, "w") as file:
        for row in lstTable:
            file.write(f"{row['Task']},{row['Priority']}\n")
    print("Data saved to file.")
    continue
```

Fig 10.0 Saves data to text file

If the user chooses option '4', the script opens the file specified by **objFile** in write mode (**"w"**). It iterates over each dictionary in **lstTable** and writes the task and priority to the file. Each entry is written on a new line. After saving, it prints a message indicating the data was saved to the file.

```
    # Step 7 - Exit program
    elif (strChoice.strip() == '5'):
        input('Press Enter to exit .... !') #Confirming before exiting
        break  # and Exit the program
```

Fig 11.0 Exits the program

If the user chooses option '5', the script prompts for confirmation before exiting the program.

Executing the Python Script

Now that the script is written, the next step is to execute the script to check the correctness.

```
C:\_PythonClass\Assignment05\venv\Scripts\python.exe "C:/_PythonClass/Assignment05/Assigment05_ToDo List.py"

    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

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

Current Data:
Task: Birthday, Priority: 1
Task: Go For Walk, Priority: 1
Task: 1, Priority: 4

    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

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

Data saved to file.

    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
```

```
    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

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

Press Enter to exit .... !

Process finished with exit code 0
```

Fig 12.0 Output of script in PyCharm

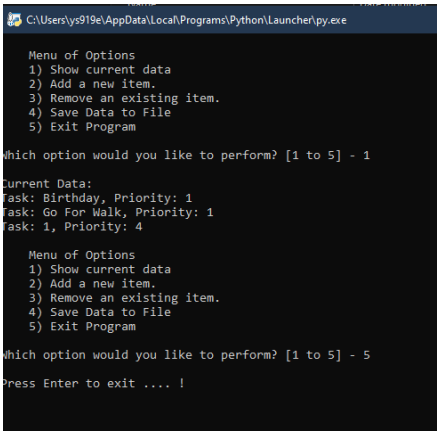

Fig 10.0 Output of script in Command Prompt

Summary

In the fifth week, we learnt about using List, Dictionaries and difference between Lists and Dictionaries.

Also, I used a partially written script to start and completed the code to perform various actions like Data Initialization, Loading data from file, showing menu and interacting with it, displaying data, Adding and Removing Items, Saving data to file. Overall, it was an exciting week of learning.