Daniel Medrano Figueora

05/22/2023

IT FDN 110 A

Assignment 06

Operators, Expression, and Conditional Statements

Intro

This week we learned about global and local variables. Global variables in Python are defined outside of any function and can be accessed from anywhere in the program. They have a global scope and can be modified within functions. Local variables, on the other hand, are defined within a function and can only be accessed within that function. They have a local scope and their values are discarded once the function execution is complete. Classes in Python are blueprints for creating objects that encapsulate data and functionality, while functions are blocks of code that perform specific tasks and can be reused throughout the program. The debugger tool became useful as well to overlook the code and understand where critical issues are.

Background

This week's assignment is to modify the Assignment06_Starter file. Using the methods learned this week, the program is supposed to have a menu of options to add tasks, remove tasks/view them or to save the file. This is all done using classes to call and do the operations.

Intro

I first modified the read data from the file. This is where the code is able to read from the file and is able to add the words task and priority so that it may save more data. The word task and priority gets added to the file to each one the user inputs and can be brought up at any time.

```python
class Processor:
    """  Performs Processing tasks """

    @staticmethod
    def read_data_from_file(file_name, list_of_rows):
        """ 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
        """
        list_of_rows.clear()  # clear current data
        file = open(file_name, "r")
        for line in file:
            task, priority = line.split(",")
            row = {"Task": task.strip(), "Priority": priority.strip()}
            list_of_rows.append(row)
        file.close()
        return list_of_rows
```

The class in the image below adds the data that uses inputs into the list.

```python
    @staticmethod
    def add_data_to_list(task, priority, list_of_rows):
        """ Adds data to a list of dictionary rows

        :param task: (string) with name of task:
        :param priority: (string) with name of priority:
        :param list_of_rows: (list) you want to add more data to:
        :return: (list) of dictionary rows
        """
        row = {"Task": str(task).strip(), "Priority": str(priority).strip()}
        list_of_rows.append(row)
        return list_of_rows
```

The class below removes the tasks and priorities.

```python
    @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
        """
        for row in list_of_rows:
            if row["Task"].lower() == task.lower():
                list_of_rows.remove(row)
                break
        return list_of_rows
```

The class below writes data to the file

```python
    @staticmethod
    def write_data_to_file(file_name, list_of_rows):
        """ Writes data from a list of dictionary rows to a File

        :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, "w")
        for row in list_of_rows:
            file.write(row["Task"] + "," + row["Priority"] + "\n")
        file.close()
        return list_of_rows
```

Each of these classes are called out throughout the code and help keep the code organized in knowing where specific operations are.
Moving onto the IO portion where the menu is operated.

```python
    @staticmethod
    def input_menu_choice():
        """ Gets the menu choice from a user

        :return: string
        """
        choice = str(input("Which option would you like to perform? [1 to 4] - ")).strip()
        print()  # Add an extra line for looks
        return choice

    @staticmethod
    def output_current_tasks_in_list(list_of_rows):
        """ Shows the current Tasks in the list of dictionaries rows

        :param list_of_rows: (list) of rows you want to display
        :return: nothing
        """
        print("******* The current tasks ToDo are: *******")
        for row in list_of_rows:
            print(row["Task"] + " (" + row["Priority"] + ")")
        print("*****************************************")
        print()  # Add an extra line for looks
```

The last two IO sections are displayed in the image below. The first input, gets the task and priority values and adds it to the list which returns a string and its respected task and priority. The second input task asks the user which task to remove from the list and does so by looking up the task.

```python
    @staticmethod
    def input_new_task_and_priority():
        """  Gets task and priority values to be added to the list

        :return: (string, string) with task and priority
        """
        task = input("Enter a new task: ")
        priority = input("Enter the priority for the task: ")
        return task, priority

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

        :return: (string) with task
        """
        task = input("Enter the task to remove: ")
        return task
```

Lastly, the main body of the code is where most of the commands get processed. The previous code shown is more of temporary data storage and user inputs. The main body is now less code as the work gets broken up into the previous two sections which makes it a lot easier to understand for anyone working on the code.

```python
# Step 1 - When the program starts, Load data from ToDoFile.txt.
Processor.read_data_from_file(file_name=file_name_str, list_of_rows=table_lst)  # read file data

# Step 2 - Display a menu of choices to the user
while True:
    # Step 3 Show current data
    IO.output_current_tasks_in_list(list_of_rows=table_lst)  # Show current data in the list/table
    IO.output_menu_tasks()  # Shows menu
    choice_str = IO.input_menu_choice()  # Get menu option

    # Step 4 - Process user's menu choice
    if choice_str.strip() == '1':  # Add a new Task
        task, priority = IO.input_new_task_and_priority()
        table_lst = Processor.add_data_to_list(task=task, priority=priority, list_of_rows=table_lst)
        continue  # to show the menu

    elif choice_str == '2':  # Remove an existing Task
        task = IO.input_task_to_remove()
        table_lst = Processor.remove_data_from_list(task=task, list_of_rows=table_lst)
        continue  # to show the menu

    elif choice_str == '3':  # Save Data to File
        Processor.write_data_to_file(file_name=file_name_str, list_of_rows=table_lst)
        print("Data Saved!")
        continue  # to show the menu

    elif choice_str == '4':  # Exit Program
        print("Goodbye!")
        break  # by exiting loop
```

Running the code within PyCharm, asking to create a new task of “cleaning” and the priority to be “high”. Then Saving the code and exiting the program.

```
Assigment06_Starter

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

Enter a new task: Cleaning
Enter the priority for the task: High
******* The current tasks ToDo are: *******
Cleaning (High)
*******************************************


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


Which option would you like to perform? [1 to 4] - 3

Data Saved!
******* The current tasks ToDo are: *******
Cleaning (High)
*******************************************
```

Using the terminal window to remove the current tasks.

```
Terminal:  Local
PS C:\Users\danie\Google Drive\PythonClass\Assignment06> python.exe "C
******* The current tasks ToDo are: *******
Cleaning (High)
*******************************************


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


Which option would you like to perform? [1 to 4] - 2

Enter the task to remove: cleaning
******* The current tasks ToDo are: *******
*******************************************
```

The task of cleaning was removed successfully and was able to save and exit from the terminal window.

Summary

In the end, global variables are defined outside of any function and can be accessed from anywhere in the program, while local variables are defined within a function and can only be accessed within that function. Classes are used to create objects that contain both data and functionality, providing a blueprint for their structure and behavior. Functions, on the other hand, are blocks of code that perform specific tasks and can be reused throughout the program. Understanding the difference between global and local variables, as well as the concept of classes and functions, is crucial for effective programming in Python.