Wyatt Parks
24May2023
FOP: Python
Assignment 06
https://github.com/wyparks/IntroToProg-Python

Task List with Functions

Intro

This assignment is similar to last week's in modifying a prewritten code for the purpose of creating a script to record tasks and their priorities. This week we are adding an additional level by using classes and functions to better organize our code.

Functions

Functions in Python are reusable blocks of code that perform specific tasks. They help in organizing and structuring a program by breaking it into smaller, more manageable pieces. Functions are defined using the "def" keyword, followed by the function name and a set of parentheses containing any input parameters. These parameters are optional, and they allow the function to receive values from the calling code. Within a function, you can write the instructions to be executed, and you can also include a return statement to provide a result back to the caller. Functions are called by using their name followed by parentheses, and the input arguments can be provided within those parentheses. Python functions enhance code modularity, reusability, and readability, making it easier to maintain and understand complex programs[1].

Writing the script

The code that was given to me had two classes, one for processing data and the other for displaying and collecting information, to and from the user (figure 1). The processor class contained functions that that operated using values from the user and from a txt file to update a list of dictionaries. Similar to the

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

    1 usage
    @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:
```

Figure 1. The Processor class.

previous assignment, a function was written to add data from a list. This is done by using parameters to pass values to the function which were the user inputs for the task and its priority and the list we are updating with the function. The function then feeds the user inputs into a dictionary and appends the list, then returns the list (figure 2).

Wyatt Parks
24May2023
FOP: Python
Assignment 06
https://github.com/wyparks/IntroToProg-Python

```
1 usage
@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)  # appending the list with the new dictionary that was generated

    return list_of_rows
```

Figure 2. Add to list function.

The 'IO' class consisted of functions that processed user inputs into variables and displayed data for the user. Under the IO class, a function was written to prompt the user to input a new task and its priority and store those values as variables. The variables were then passed back out of the function (figure 3).

```
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_new = input('Please enter the task you wish to add: ')  # getting task from user
    priority_new = input(f'Enter the priority of {task_new}: ')  # getting priority
    return task_new, priority_new
```

Figure 3. I/O function for collecting new tasks and priorities.

In the main body of code functions were called upon in conjunction to interact with the user and interact with the data. First, using the IO class function for a new task (previously discussed in reference to figure

```
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
```

Figure 4. Calling functions in the main code to add new data to the list.

3), user inputs were returned into the variables 'task' and 'priority'. Those variables were then passed into the class Processor function for adding a new item to the list. The function then returned the updated list into the variable table_lst.

Wyatt Parks
24May2023
FOP: Python
Assignment 06
https://github.com/wyparks/IntroToProg-Python

Summary

In this assignment, we expanded upon the previous task by incorporating classes and functions to create a script for recording tasks and their priorities. Functions, reusable blocks of code in Python, played a crucial role in structuring the program and enhancing code modularity. The provided code included two classes: one for data processing and another for user interaction. The processor class contained functions to add data to a list by creating dictionaries from user inputs, while the IO class focused on collecting user inputs and displaying information. By leveraging classes and functions, we achieved improved code organization, reusability, and readability, enabling efficient management of tasks and priorities[1].

Citations

1. OpenAI ChatGPT, May. 2023, chat.openai.com/chat: Aspects of this assignment were informed and created by queries I submitted to the ChatGPT.