Dustin Strop

Feb 29, 2020

Foundations of Programming: Python

Assignment06

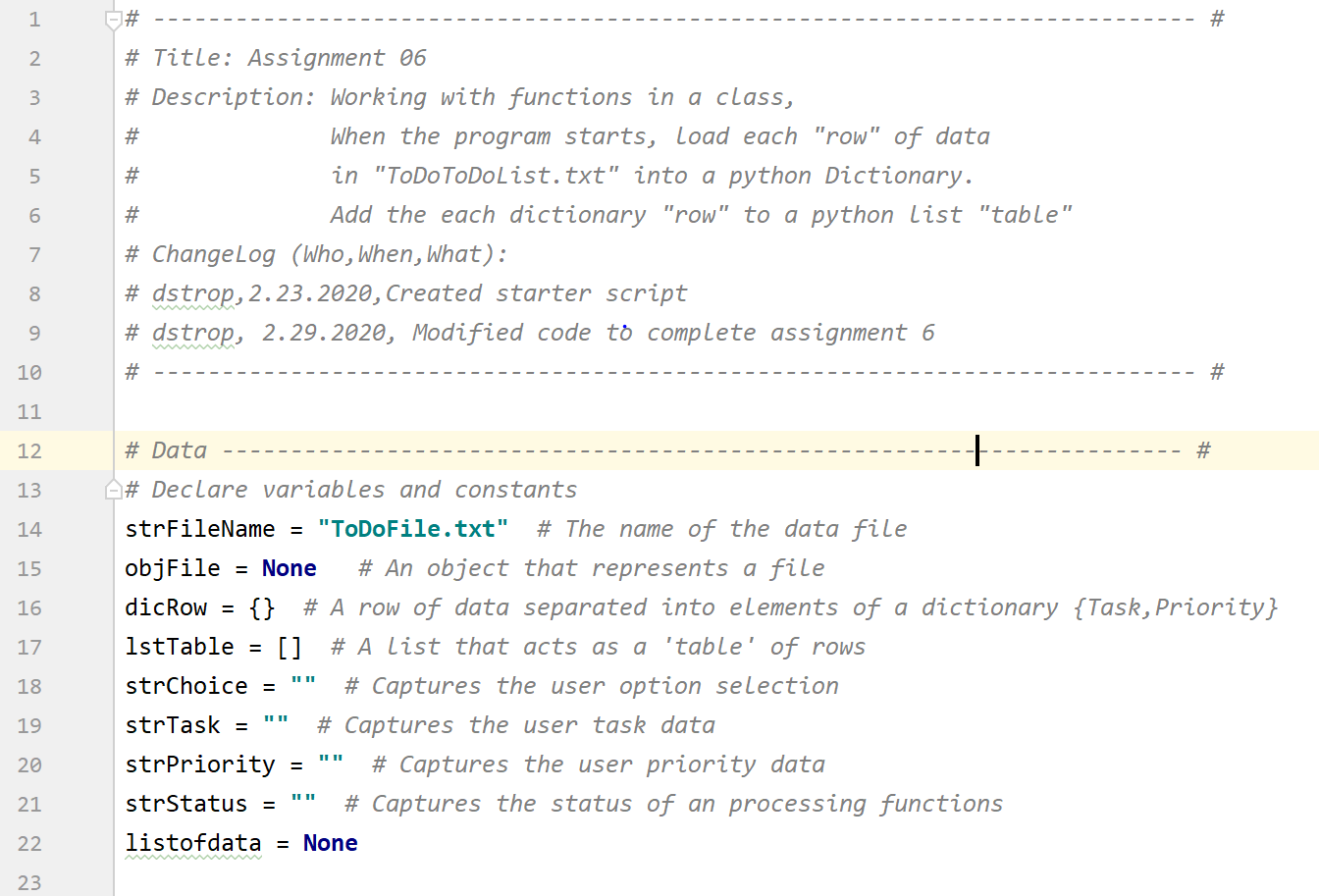
# Working with Functions and Classes

Introduction

This paper will cover how to use functions, how you can take functions and organize them into classes, and how to use variables, arguments and parameters to make your code function more effectively.

Topic I: Header & Declaring the Variables

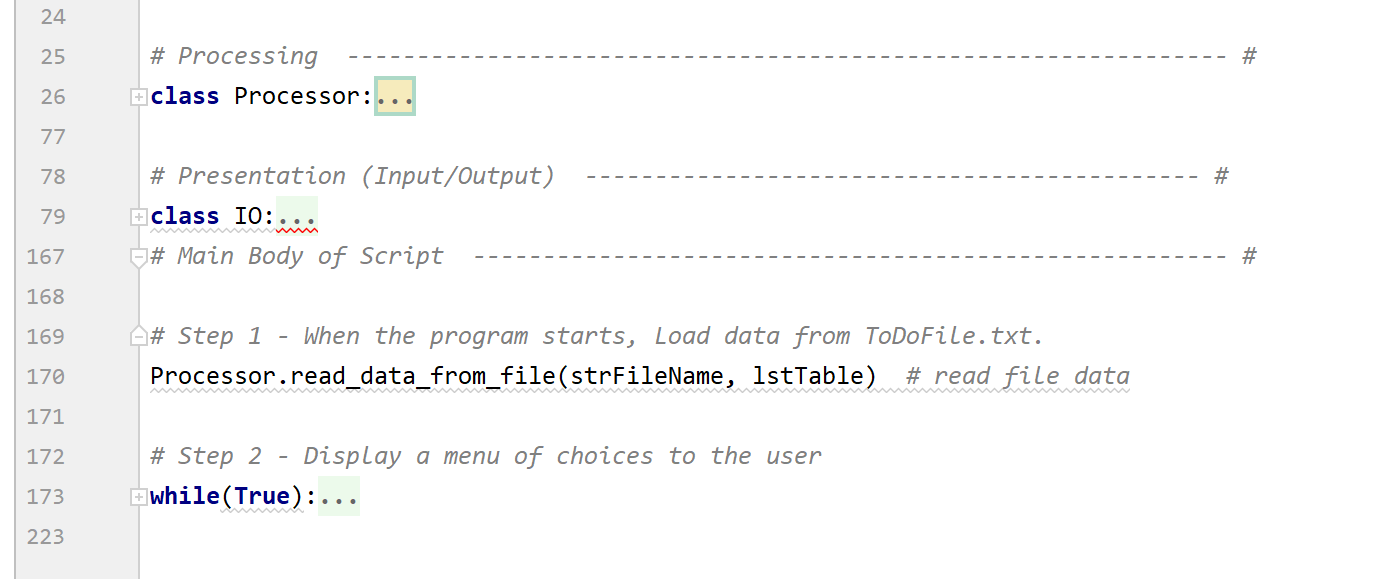
Begin by opening a new project in PyCharm and typing out the header, which is lined 1 through 10. At line 14 the variables are declared, objFile and strFileName are the variables that Python uses to call the ToDoFile.txt. The variables on lines 16 through 22 are used throughout the script to capture and organize data and user inputs.



***Figure 1: Example of pseudocode and declared variables.***

Topic II: Processing and Presentation Code

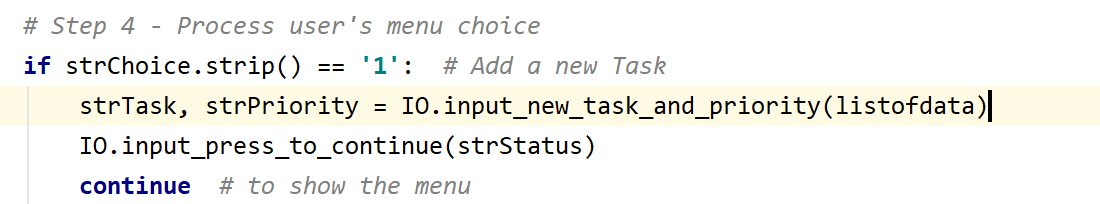
The script is broken out into processing and presentation code, as displayed in Figure 2. Through the creation of variables and arguments, the processing and presentation code can pass data between one another. The functionality of this code allows to choose an option (1-5) and depending on the choice, they are able to tell the code to perform an action. Those actions include, add data to a list, remove data from a list, save data to list, print data from a list, and exit the program. This paper will not walk through each of these functions and how class and functions are incorporated into the code.



***Figure 2: an example of code organized by processing and presentation pseudocode.***

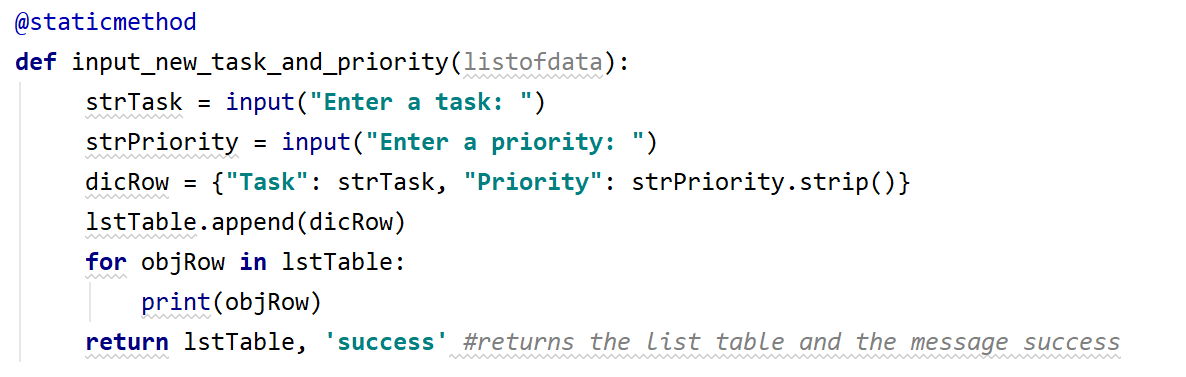
Topic III: Adding User Data to a List

This is best explained by how the program operates. When the user enters 1, using input statements and assigning them values, the user can input a task and priority. Once this data is entered the program delivers these values to the function, within a class, to perform the processing of adding the task to the list. Figure 3 shows one way to write the code. This code essentially reads: when the user selects 1, run the function IO.input\_new\_task\_and\_priority, and add strTask, strPriority to the listofdata. Once this is complete, the program can continue.



**Figure 3: IO.input\_new\_task\_and\_priority is a user friendly function name.**

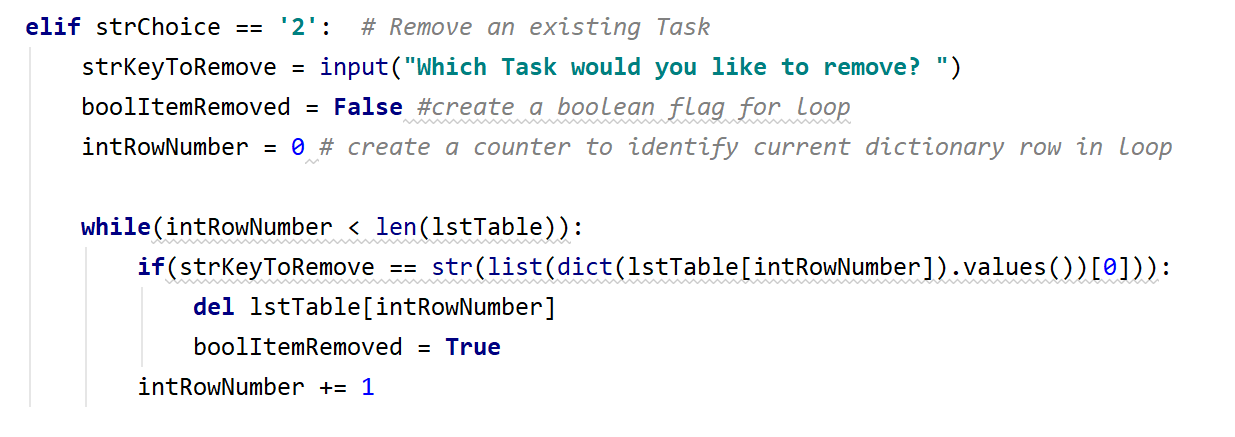
The program calls the function and runs the script to capture the users input. Using the **def** statements we define the function and its variable which is listofdata. The script reads: capture the users input, add to the dictionary row, and print the rows as a table. Figure 4 outlines this function.



***Figure 4: An example of a function that adds data to a list.***

Topic III: Removing a Row from the To Do List

Next, we write the script to remove an item from a list. In this code the program will capture the item the user wants to remove. This is named strKeyToRemove. The ‘0’ tells the program where to start on the list table. The code uses a while loop to run through the list and remove the entry that matches the strKeyToRemove.



***Figure 5: An example of code to remove an item from a list***

Topic IV: Summary

In this paper we have reviewed how functions and classes can be used to organize your code. The remaining operations are documented in the python script sent in with this document.