Kyle Chan

February 17, 2019

Intro to Python (IT FDN 100A)

Assignment 06

Github: <https://github.com/kylechan1987/IntroToProg-Python>

Assignment 05 Document

# Introduction

This assignment continues the concepts and material we learned from Assignment05. We continue our usage of lists and dictionaries but now combine the functionality of Python functions and classes. The goal of this assignment is to use functions and classes to help streamline our code to make it more robust as well as repeatable for similar types of tasks.

# Create Python script

The python script used for Assignment 06 is written in PyCharm which is an IDE which makes it easier to write and debug code. The python code is shown in (Figure 1). The first part of the code is the comment section which lists things like Name, Title, Date, and short description of the python script. Because of the length of the code, I decided to simply copy and paste the code below. I’ve created a section break so that the code itself can be seen on in a Landscape format for readability.

*# ---------------------------------------  
# Title: Assignment06 script (To Do List with Functions and Classes)  
# Name: Kyle Chan  
# Date: Feb 17, 2019  
# Course: Intro to Python  
# Comments: This code goes with Intro  
# to Python Class Assignment06  
# This assignment adds onto what we did with Assignment05  
# Create Python script file that manages a To Do list  
# Store data in a Python Dictionary  
# Create conditional loop with user options  
# Save file to* ***Todo.txt****# However we will use functions and classes to make our code more streamlined  
# We will start with using the Assignment05\_answer.py from class  
  
  
#-- Data --#  
# declare variables and constants*filename = **"C:\\_PythonClass\Assignment06\Todo.txt"** *# name of text file of todolist*table = [] *# empty table for list  
  
#-- Processing & Input/Output --#  
# Create class to put all functions in***class** todoList(object):  
  
 *# Create Functions needed for program  
 # This function reads the file* ***Todo.txt into a python dictionary and passes the file name as a parameter*** @staticmethod  
 **def** read\_file(filename, table):  
 objFile = open(filename, **"r"**)  
 **for** line **in** objFile:  
 strData = line.split(**","**) *# readline() reads a line of the data into 2 elements* dicRow = {**"Task"**: strData[0].strip(), **"Priority"**: strData[1].strip()}  
 table.append(dicRow)  
 objFile.close()  
 **return** table  
  
 *# This function prints the menu of options* @staticmethod  
 **def** print\_menu():  
 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 4] - "**))  
 print() *# adding a new line* **return** strChoice  
  
 *# This function shows the current items in the table and passes the lstTable as parameter* @staticmethod  
 **def** show\_data(table):  
 print(**"\*\*\*\*\*\*\* The current items ToDo are: \*\*\*\*\*\*\*"**)  
 **for** row **in** table:  
 print(row[**"Task"**] + **"("** + row[**"Priority"**] + **")"**)  
 print(**"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"**)  
  
 *# This function adds a new item to the list table and passes lstTable as parameter* @staticmethod  
 **def** add\_item(table):  
 strTask = str(input(**"What is the task? - "**)).strip()  
 strPriority = str(input(**"What is the priority? [high|low] - "**)).strip()  
 dicRow = {**"Task"**: strTask, **"Priority"**: strPriority}  
 table.append(dicRow)  
 **for** dicRow **in** table:  
 print(dicRow)  
 **return** table  
  
 *# This functions removes an item from the list table and passes lstTable as parameter* @staticmethod  
 **def** remove\_item(table):  
 *# 5a-Allow user to indicate which row to delete* strKeyToRemove = input(**"Which TASK would you like removed? - "**)  
 blnItemRemoved = **False** *# Creating a boolean Flag* intRowNumber = 0  
 **while** (intRowNumber < len(table)):  
 **if** (strKeyToRemove == str(  
 list(dict(table[intRowNumber]).values())[0])): *# the values function creates a list!* **del** table[intRowNumber]  
 blnItemRemoved = **True** *# end if* intRowNumber += 1  
 *# end for loop  
 # 5b-Update user on the status* **if** (blnItemRemoved == **True**):  
 print(**"The task was removed."**)  
 **else**:  
 print(**"I'm sorry, but I could not find that task."**)  
 **return** table  
  
 *# This functions save the data from list table into objfile  
 # 5b Ask if they want save that data* @staticmethod  
 **def** save\_file(table, filename):  
 **if** (**"y"** == str(input(**"Save this data to file? (y/n) - "**)).strip().lower()):  
 objFile = open(filename, **"w"**)  
 **for** dicRow **in** table:  
 objFile.write(dicRow[**"Task"**] + **","** + dicRow[**"Priority"**] + **"\n"**)  
 objFile.close()  
 input(**"Data saved to file! Press the [Enter] key to return to menu."**)  
 **else**:  
 input(**"New data was NOT Saved, but previous data still exists! Press the [Enter] key to return to menu."**)  
  
  
*#-- Processing --#  
# Step 1  
# When the program starts, load the any data you have  
# in a text file called* ***ToDo.txt into a python Dictionary.***lstTable = todoList.read\_file(filename,table)  
  
**while**(**True**):  
  
 *# Step 2  
 # Display a menu of choices to the user* strChoice\_selection = todoList.print\_menu()  
  
 *# Step 3  
 # Show the current items in the table* **if** (strChoice\_selection.strip() == **'1'**):  
 todoList.show\_data(lstTable)  
  
 *# Step 4  
 # Add a new item to the list/Table* **elif**(strChoice\_selection.strip() == **'2'**):  
 lstTable = todoList.add\_item(lstTable)  
  
 *#4a Show the current items in the table* todoList.show\_data(lstTable)  
  
  
 *# Step 5  
 # Remove a new item to the list/Table* **elif**(strChoice\_selection.strip() == **'3'**):  
 lstTable = todoList.remove\_item(lstTable)  
  
 *#5c Show the current items in the table* todoList.show\_data(lstTable)  
  
 *# Step 6  
 # Save tasks to the* ***ToDo.txt file* elif**(strChoice\_selection.strip() == **'4'**):  
 todoList.show\_data(lstTable)  
  
 *# Ask if they want to save data* todoList.save\_file(lstTable,filename)  
 **continue** *# to show the menu* **elif** (strChoice\_selection.strip() == **'5'**):  
 **break** *#and Exit the program*

## Python Classes and Functions

In Python, we can group code within Functions and Classes. This follows the concept of Abstraction and Encapsulation which is one of the 3 keys of Object Oriented Programming. According to our Module 06 notes, "Functions are a way of grouping one or more statements together. You can then execute these statements by calling the functions”. In this assignment, I created six different functions to perform the tasks we did from Assignment 05. One of the main advantages of grouping statements of codes into functions is repeatability. For example, if I wanted to make multiple Todo lists, I can give the separate files different names and still reuse the same functions to add/remove/save data. For this assignment, I also used the return statement within the function to return the desired data I want from that function. This is useful because I can specify different variable names if I wanted to make multiple Todo lists.

Alongside functions, we were also introduced to Python Classes. Similar to Functions, classes in Python group functions, variables, and constants based on our Module 06 notes. When doing this, Functions within its respective Class are now called methods. For this assignment, we put the six functions into a class I called todoList. In this fashion, to call out one of the functions I would have to call out with class\_name.function\_name() since functions are methods within classes.

# Launching from Command Window (Windows)

Once the Python script Assignmnet06\_script.py is created and saved in the desired location, the next step is to launch this python script on the Command Window. Figure 1 shows the steps in DOS to run the python file. To test my code, I entered multiple Tasks and priorities as well as deleted some entries to see if it works! Figure 2 shows the snapshot of the Console in PyCharm. Note: my computer uses windows 10 so launching python can be done with the command “py”. Lastly, Figure 3 shows the snapshot of the text file that gets saved when exiting the program.

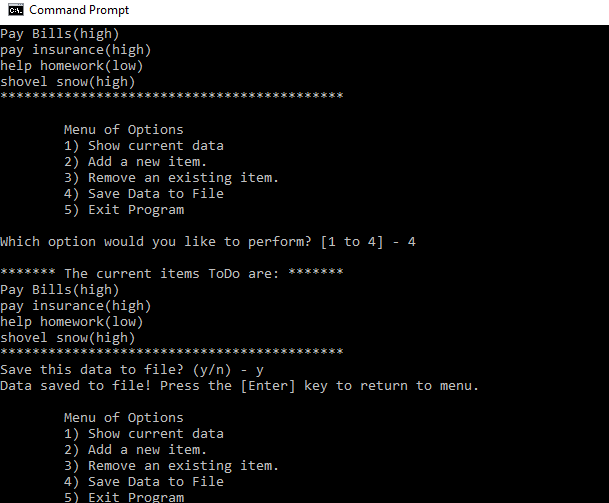


Figure 1. Launching Assignment05\_script.py in Command Prompt in Windows

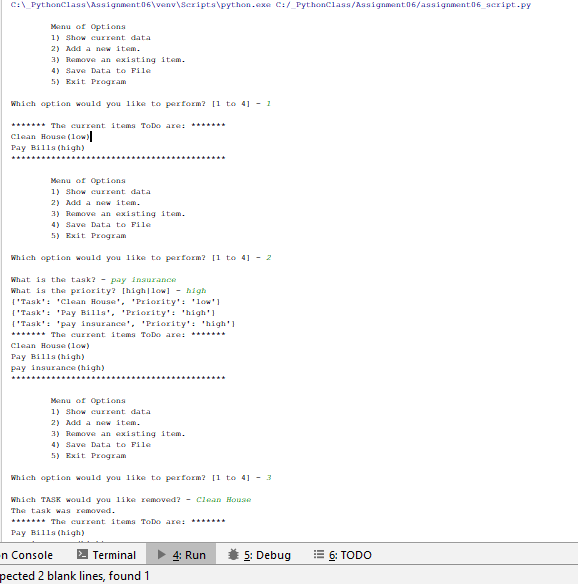


Figure 2. Snapshot of console in PyCharm

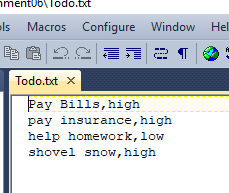


Figure 3. Snapshot of Todo.txt file with stored data of list

# Summary

Assignment 06 showed us how to use the power of functions and classes to simplify and organize our code to go along with what we learned about dictionaries and lists from the last assignment. Using functions allows us to group several statements into one “function” which helps with things that are constantly repeated. Likewise, Classes group functions and other variables which store these attributes as methods. Classes and functions help provide flexibility and robustness to our computer code by allowing us to easily repeat similar tasks along with organizing our code for better readability.