Denise Albano
2/22/22
Foundations of Programming: Python
Assignment 06
https://github.com/dla425/IntoToProg-Python-Mod06

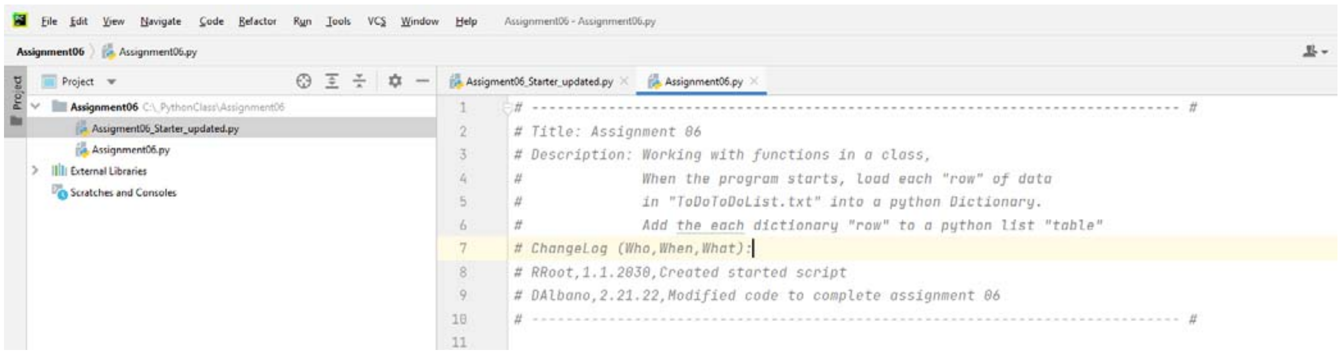
“ToDo File” Python Script

Introduction:

This paper will document the steps I took to modify a python script that manages a “ToDo list.” The script will load data from a file into a python list of dictionary objects. Several new functions are needed to organize the code in the script. In order to complete the assignment, I watched the course video in module 5 by Randall Root, read chapter 6 in text book, as well as reviewed and watched the additional web pages and videos.

Creating the Python Script:

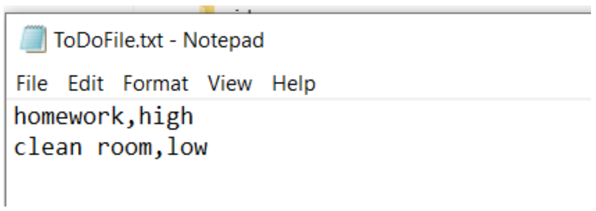
For this assignment I used PyCharm to modify the assignment06 starter python script. To start, I created a new project in PyCharm that uses the folder _PythonClass\Assignment06. Within this project, I created a new python script file named Assignment06.py (see figure 1) by copying the information from the assignment06 starter python script. In my assignment06.py script, I updated the header to include my name and date in the change log.


File Edit View Navigate Code Refactor Run Tools VCS Window Help
Assignment06 - Assignment06.py
Assignment06
Assignment06.py
Project
Assignment06 C:_PythonClass\Assignment06
Assigment06_Starter_updated.py
Assignment06.py
External Libraries
Scratches and Consoles
Assigment06_Starter_updated.py
Assignment06.py

```
# ---------------------------------------------------------------------------- #
# Title: Assignment 06
# Description: Working with functions in a class,
#              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
# DAlbano,2.21.22,Modified code to complete assignment 06
# ---------------------------------------------------------------------------- #
```

Figure 1: New project and python script in PyCharm

Since this assignment needed to read data from a text file, I created the ‘ToDoFile.txt’ file and added two rows of tasks and their associated priority (see Figure 2).



```
homework,high
clean room,low
```

Figure 2: Screenshot of ToDoFile.txt

Add data to list

In order to get the function add_data_to_list to work, I first needed to request the new task and priority to be added. I put the code to request this information from the user in the input/output class, under the input_new_task_and_priority function. To do this, I used two variables, task and priority, and the input()

function to request the new item from the user. By including the strip() method, I could make sure any leading or ending characters were removed. I then used the return statement to return the values captured to the add_data_to_list function. To add the new item as a row in the list, I used the keys 'task' and 'priority' with the associated values returned. I also used the strip() method to again remove any leading or ending characters from each value returned. I then used the append() method to add the new row to the list that makes up the table. I used the return statement again, to return the list of dictionary rows in the parameter, list_of_rows.

Remove data from the list

In order to get the function remove_data_from_list to work, I first needed to request the task to be removed. I put the code to request this information from the user in the input/output class, under the input_task_to_remove function. To do this, I used the variables 'task' and the input() function to request the item to be removed from the user. By including the strip() method, I could make sure any leading or ending characters were removed. I then used the return statement to return the values captured to the remove_data_from_list function. To search the data, I used a 'for … in' statement to look row by row for the task to be removed. If the row contained the task to be removed, I used the remove() method to delete the row. I used the return statement again, to return the list of dictionary rows in the parameter, list_of_rows.

Write data to the file

To save the data to the "ToDoList.txt" file, I used the open() function with the 'w' access mode to open the "ToDoFile" text file for writing. I then used a 'for … in' statement to loop through the table by row. I used the write() method to save each row of data to the file. I then used the close() function to close the file and the return statement to return the list of dictionary row in the parameter, list_of_rows.

The final code for this assignment is shown below

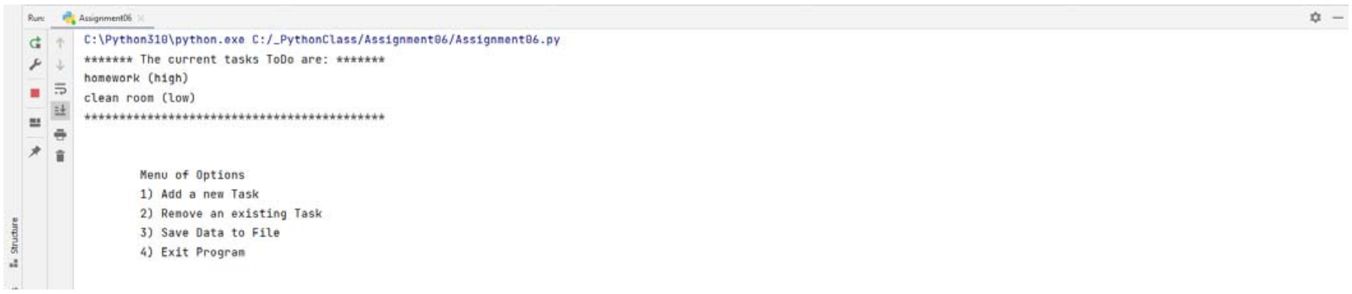
```
# ---------------------------------------------------------------------------- #
# Title: Assignment 06
# Description: Working with functions in a class,
#              When the program starts, load each "row" of data
#              in "ToDoToDoList.txt" into a python Dictionary.
#              Add the each dictionary "row" to a python list "table"
# ChangeLog (Who,When,What):
# RRoot,1.1.2030,Created started script
# DAlbano,2.21.22,Modified code to complete assignment 06
# ---------------------------------------------------------------------------- #

# Data ---------------------------------------------------------------------- #
# Declare variables and constants
file_name_str = "ToDoFile.txt"  # The name of the data file
file_obj = None  # An object that represents a file
row_dic = {}  # A row of data separated into elements of a dictionary
{Task,Priority}
table_lst = []  # A list that acts as a 'table' of rows
choice_str = ""  # Captures the user option selection


# Processing  --------------------------------------------------------------- #
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

    @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 filled with file data:
        :return: (list) of dictionary rows
        """
        row = {"Task": str(task).strip(), "Priority": str(priority).strip()}
        list_of_rows.append(row)
        return list_of_rows

    @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)
        return list_of_rows

    @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

# Presentation (Input/Output)  -------------------------------------------- #

class IO:
    """ Performs Input and Output tasks """

    @staticmethod
    def output_menu_tasks():
        """  Display a menu of choices to the user
        :return: nothing
        """
        print('''
        Menu of Options
        1) Add a new Task
        2) Remove an existing Task
        3) Save Data to File
        4) Exit Program
        ''')
        print()  # Add an extra line for looks

    @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

    @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 = str(input("Enter the task to be added: ")).strip()
        priority = str(input("Enter the priority of the task: ")).strip()
        return task, priority

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

        :return: (string) with task
        """
        task = str(input("Enter task to remove: ")).strip()
        return task

# Main Body of Script  ------------------------------------------------------ #

# 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
```

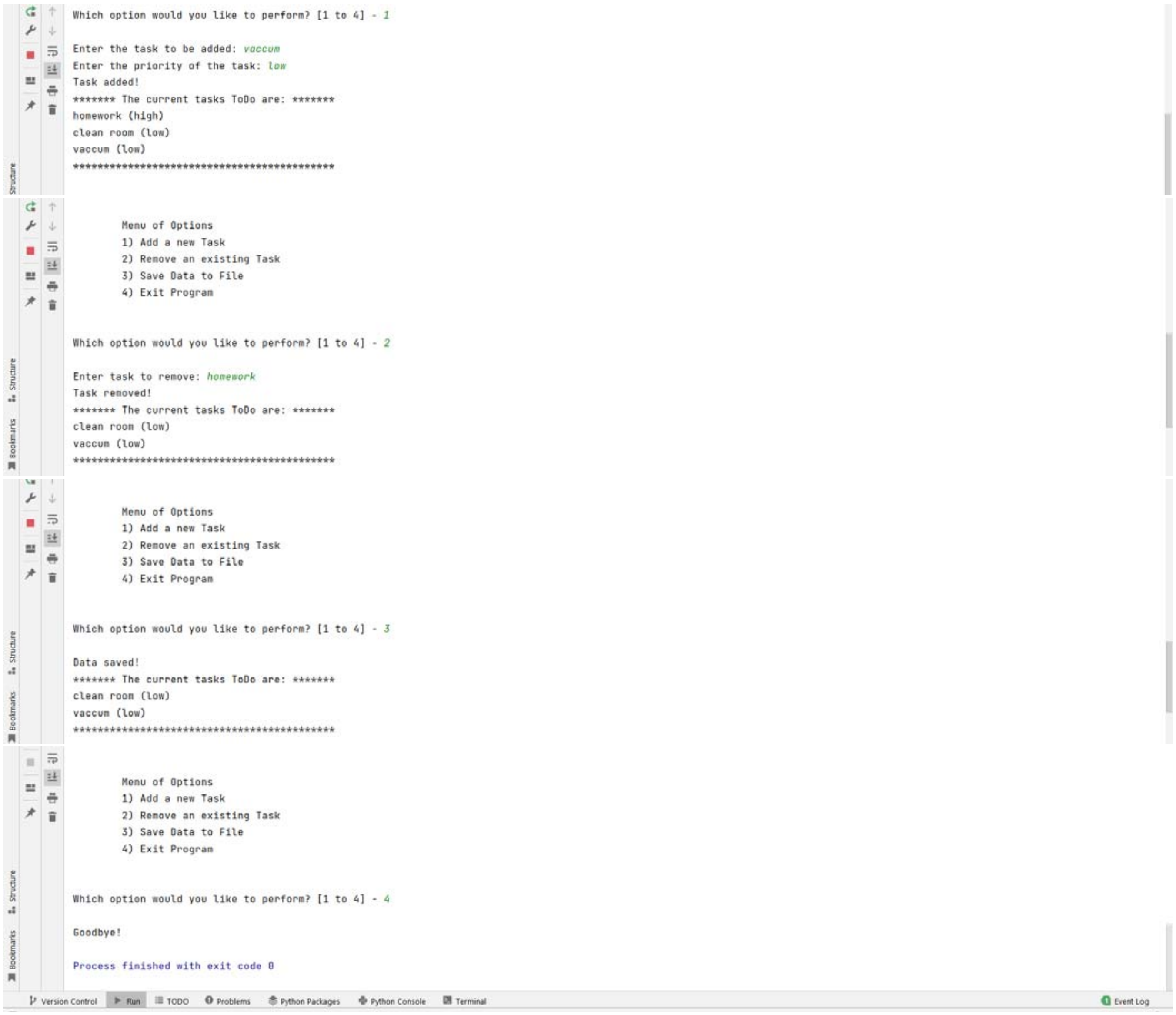
```python
# Step 2 - Display a menu of choices to the user
while (True):
    # Step 3 Show current data
    IO.output_current_tasks_in_list(list_of_rows=table_lst)  # Show current data in
the list/table
    IO.output_menu_tasks()  # Shows menu
    choice_str = IO.input_menu_choice()  # Get menu option

    # 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)
        print("Task added!")
        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)
        print("Task removed!")
        continue  # to show the menu

    elif choice_str == '3':  # Save Data to File
        table_lst = 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 program:

I did find that since the starter script for Assignment06 used a slightly different way of naming variables, I needed to use more caution when using my code from Assignment05. I also need to be careful to use the parameter names instead of the variable names I used in Assignment05. Once I sorted through all of that, I was able to make the new code I added to the script run successfully. Screenshots of the progam running can be seen in Figure 3.



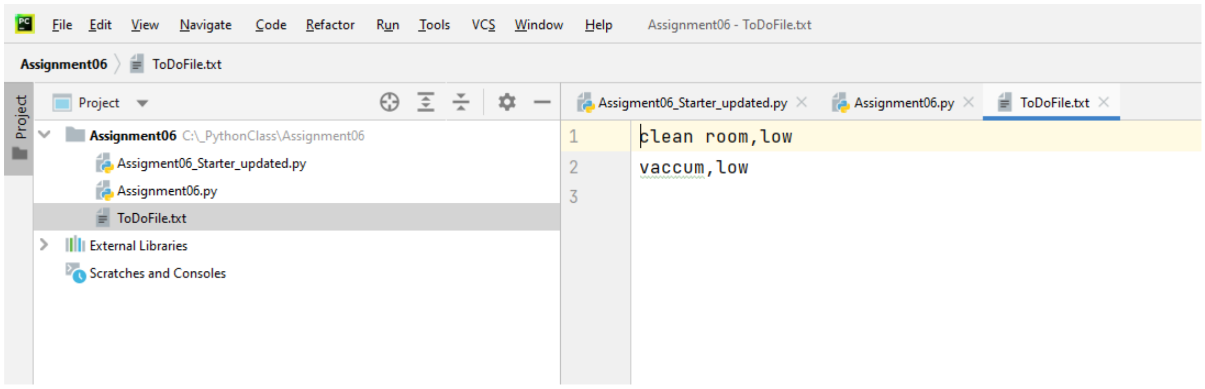
```
C:\Python310\python.exe C:/_PythonClass/Assignment06/Assignment06.py
******* The current tasks ToDo are: *******
homework (high)
clean room (low)
*******************************************



        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] - 1

Enter the task to be added: vaccum
Enter the priority of the task: low
Task added!
******* The current tasks ToDo are: *******
homework (high)
clean room (low)
vaccum (low)
*******************************************



        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 task to remove: homework
Task removed!
******* The current tasks ToDo are: *******
clean room (low)
vaccum (low)
*******************************************



        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: *******
clean room (low)
vaccum (low)
*******************************************



        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] - 4

Goodbye!

Process finished with exit code 0
```

Figure 3: Screenshot of the script running PyCharm

After running the program, I check the ToDoFile.txt file to confirm what was saved was what my program said was saved.



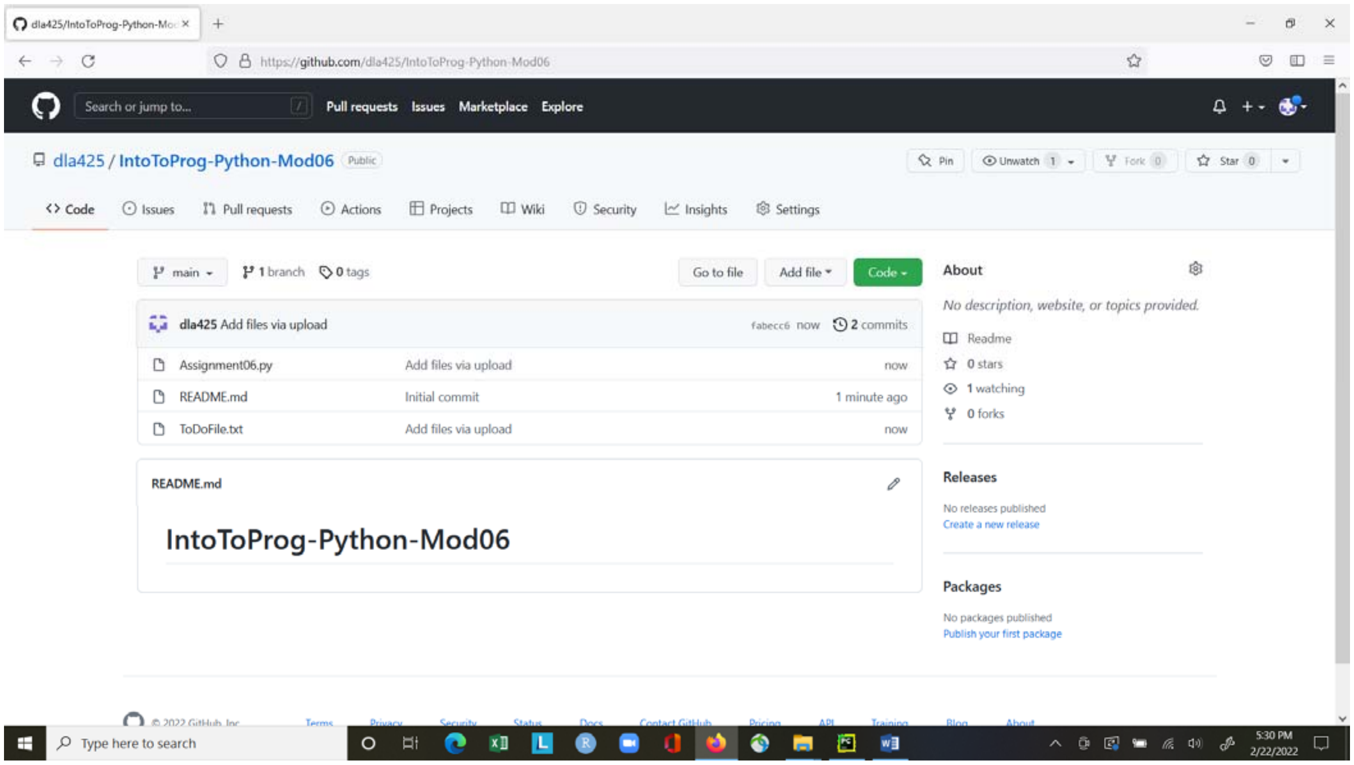
File Edit View Navigate Code Refactor Run Tools VCS Window Help Assignment06 - ToDoFile.txt
Assignment06 › ToDoFile.txt
Project
Assignment06 C:_PythonClass\Assignment06
Assigment06_Starter_updated.py
Assignment06.py
ToDoFile.txt
External Libraries
Scratches and Consoles
Assigment06_Starter_updated.py
Assignment06.py
ToDoFile.txt

```
clean room,low
vaccum,low
```

Figure 4: Screenshot of text file after running the script

Post Files to GitHub

After creating a GitHub repository named “IntoToProg-Python-Mod06” in my account, I uploaded the Assignement06 files and committed the changes. As a last step, I shared the link to my GitHub repository on the canvas discussion board for peer review.



https://github.com/dla425/IntoToProg-Python-Mod06

dla425 / IntoToProg-Python-Mod06 Public

main · 1 branch · 0 tags

dla425 Add files via upload · fabecc6 now · 2 commits

Assignment06.py	Add files via upload	now
README.md	Initial commit	1 minute ago
ToDoFile.txt	Add files via upload	now

README.md

IntoToProg-Python-Mod06

About

No description, website, or topics provided.

Readme · 0 stars · 1 watching · 0 forks

Releases: No releases published

Packages: No packages published

5:30 PM 2/22/2022

Figure 5: Screenshot of files loaded into my GitHub account

Summary

To complete this assignment, I needed to understand the difference between a function and class, as well as between an argument and parameter. It was important to also understand how to read and write data from and to a file. I found the examples in the course video, textbook, and additional materials to be extremely helpful in understanding the concepts needed to create a working program.