Brynn Lampman

11/23/2022

IT FDN 110A

Assignment 06

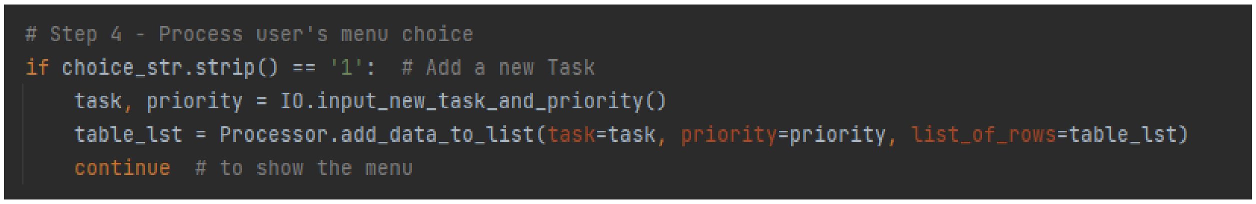
# Assignment 06 – Functions

## Introduction

This assignment was a lot like assignment 5, but assignment 6 will add functions to have the code more organized.  Setting up these functions will allow us to define the function with what we want the computer to do, then be able to call the functions later in the code.  The initial code was provided by the instructor, and we were to add in the functions where it said "TODO".

## Input

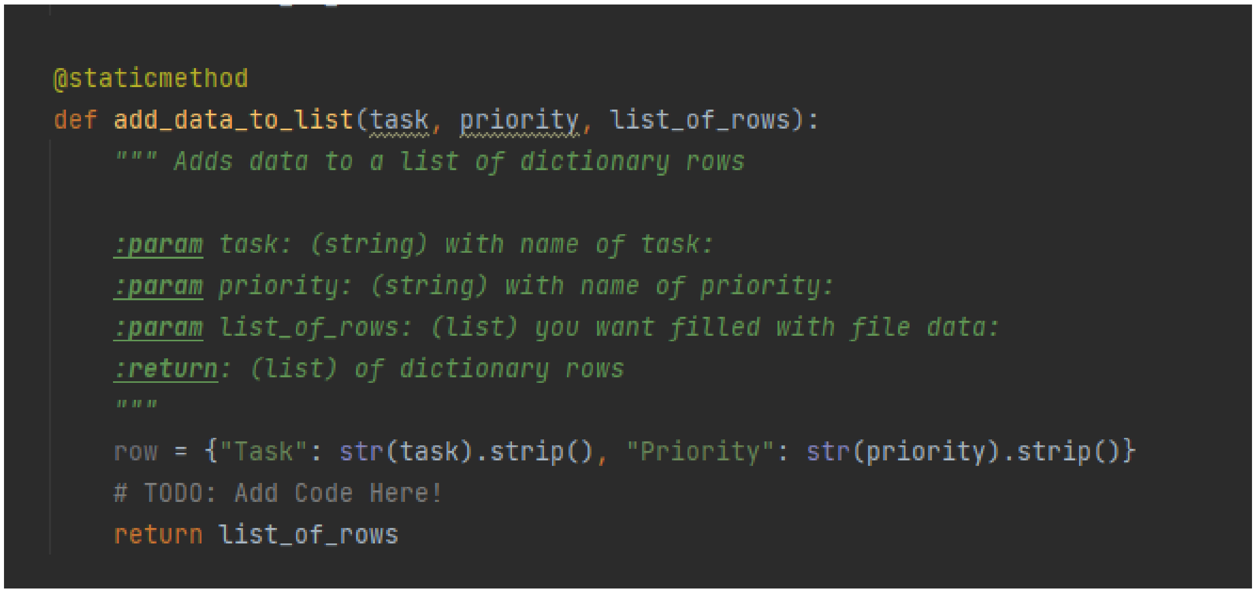
The first "TODO" was to Add code to add data to a list of dictionary rows.  Here you want to ask the user for input for task and priority.  This is done with defining (def) the function – 'add_data_to_list'. (See Figure 1)



```
# 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
```

Figure 1 – Processing the function add_data_to_list

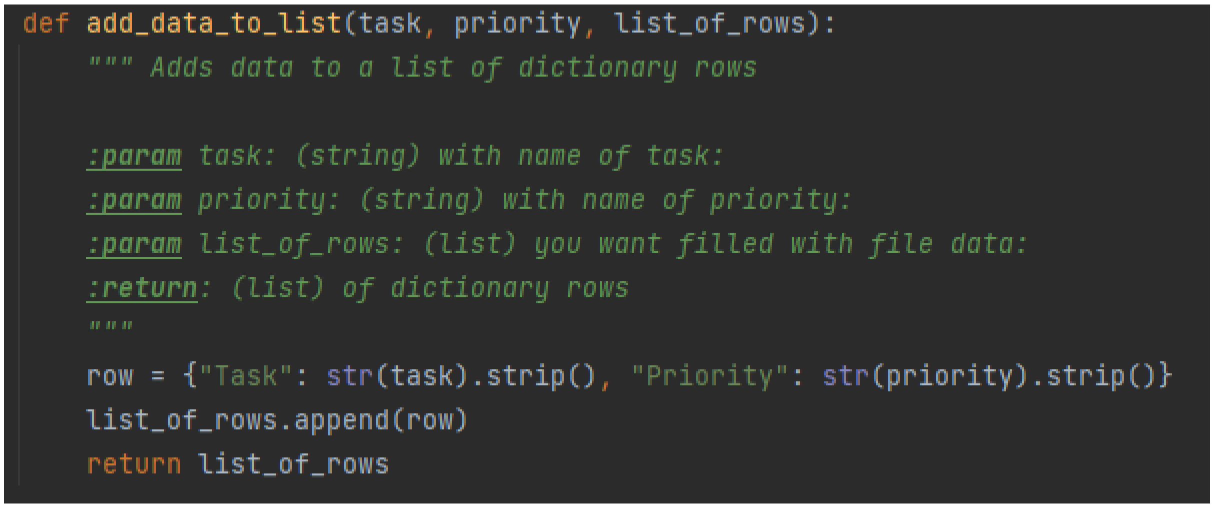
Defining the function 'add_data_to_list'.  (See Figure 2)



```
    @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()}
        # TODO: Add Code Here!
        return list_of_rows
```

Figure 2 – Here is where the function 'add_data_to_list' is defined

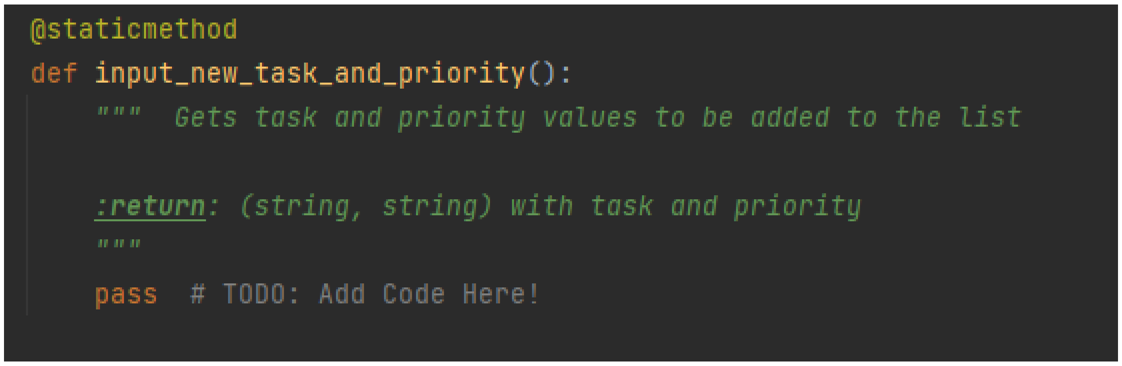
The code that will perform this is the function 'add_data_to_list'.  It appends the task and priority to the "ToDoFile.txt" and returns the list of rows. (See Figure 3)



```
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
```

Figure 3 – The code added for 'add_data_to_list'

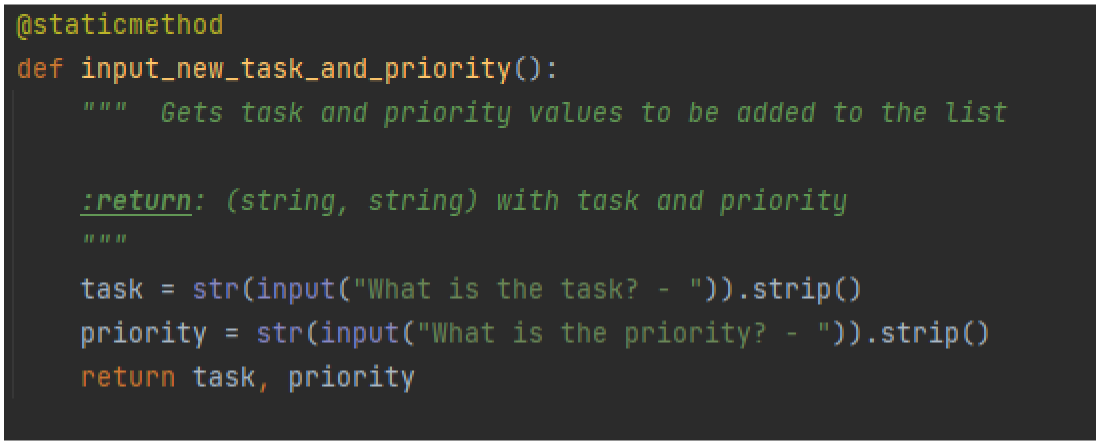
This information is retrieved from the user with option 1 – Add a new task with the function 'input_new_task_and_priority'. The "TODO"code here asks for the input from the user. (See Figure 4)



```
@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
    """
    pass  # TODO: Add Code Here!
```

Figure 4 – Defining the function 'input_new_task_and_priority'

The code asks "What is the Task?, and strips the spaces just in case they add any spaces, same with priority, then returns the task and priority. (Figure 5)

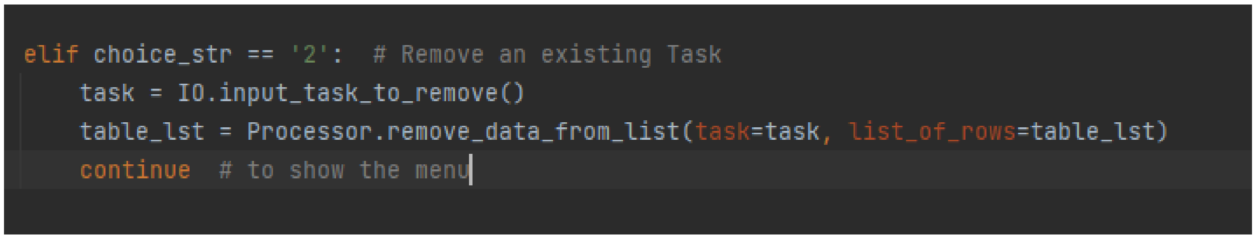


```
@staticmethod
def input_new_task_and_priority():
    """  Gets task and priority values to be added to the list

    :return: (string, string) with task and priority
    """
    task = str(input("What is the task? - ")).strip()
    priority = str(input("What is the priority? - ")).strip()
    return task, priority
```

Figure 5 – The code added to perform the function 'input_new_task_and_priority'

## Removing a task

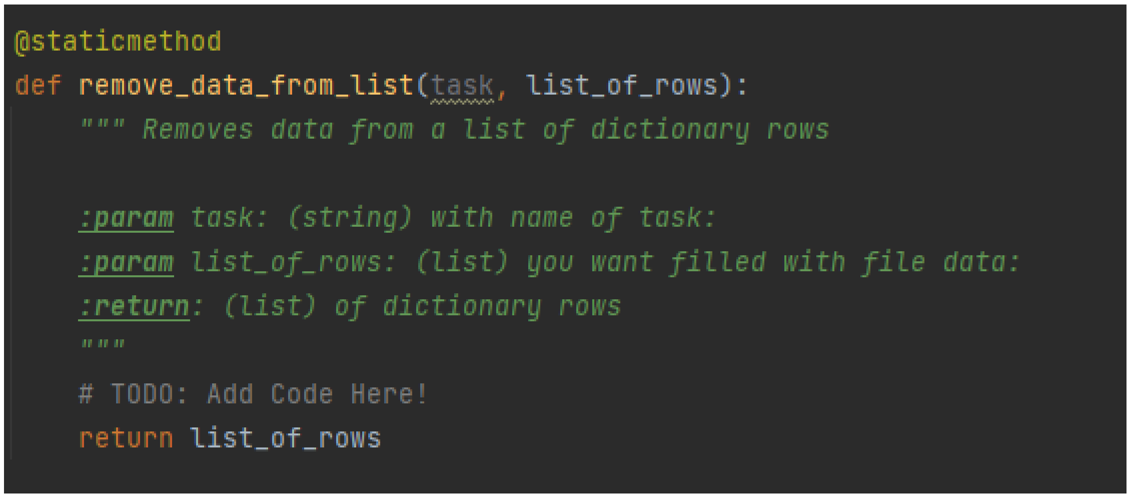
This program allows the user to remove a task from the ToDoFile.txt as well, with Option 2 on the menu. If they choose option 2, they are prompted to enter the task to remove. This is the function 'remove_data_from_list'. (See Figure 6)



```
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
```

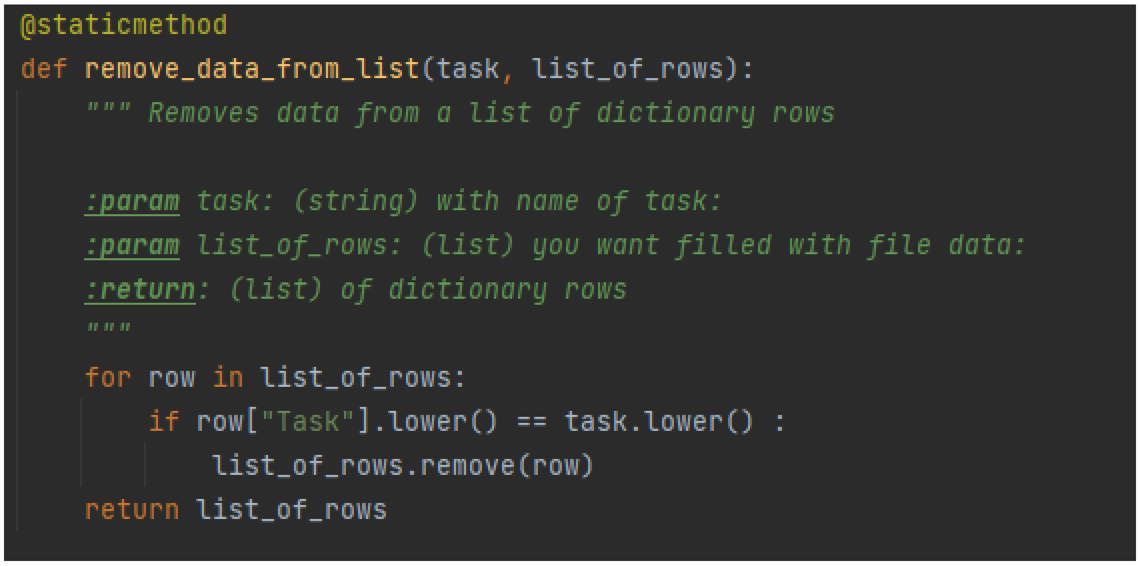
Figure 6 – Calling the function ‘remove_data_from_list’

The code “TODO” here prompts the user to enter the task to be removed, then removes it with the command ‘remove’ from the list_of_rows. (See Figure 7 and 8)



```
@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
    """
    # TODO: Add Code Here!
    return list_of_rows
```

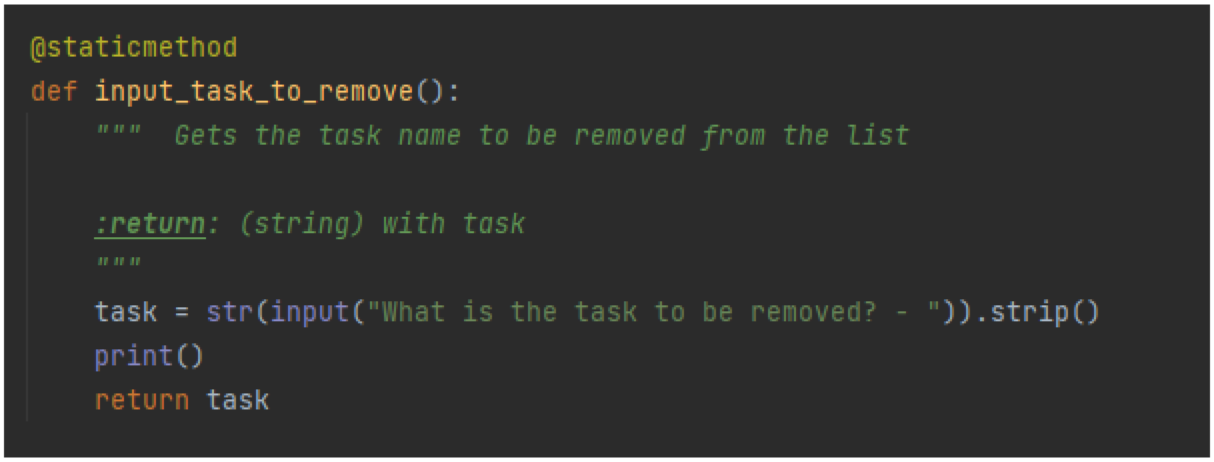
Figure 7 – Add code to help define function ‘remove_data_from_list’



```
@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
```

Figure 8 – Code entered to ‘remove_data_from_list’

When the user enters option 2, they are asked “Which task would you like to remove?” with the function ‘input_task_to_remove’. The code again removes any spaces with the command ‘strip’, then prints the new tasks. (See Figure 9)

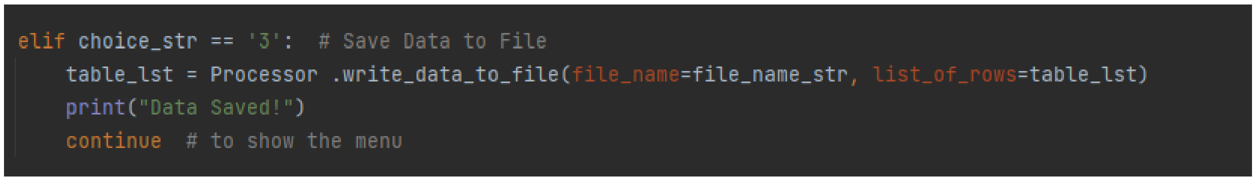


```
@staticmethod
def input_task_to_remove():
    """  Gets the task name to be removed from the list

    :return: (string) with task
    """
    task = str(input("What is the task to be removed? - ")).strip()
    print()
    return task
```

Figure 9 – Defining ‘input_task_to_remove’

## Save Data to File

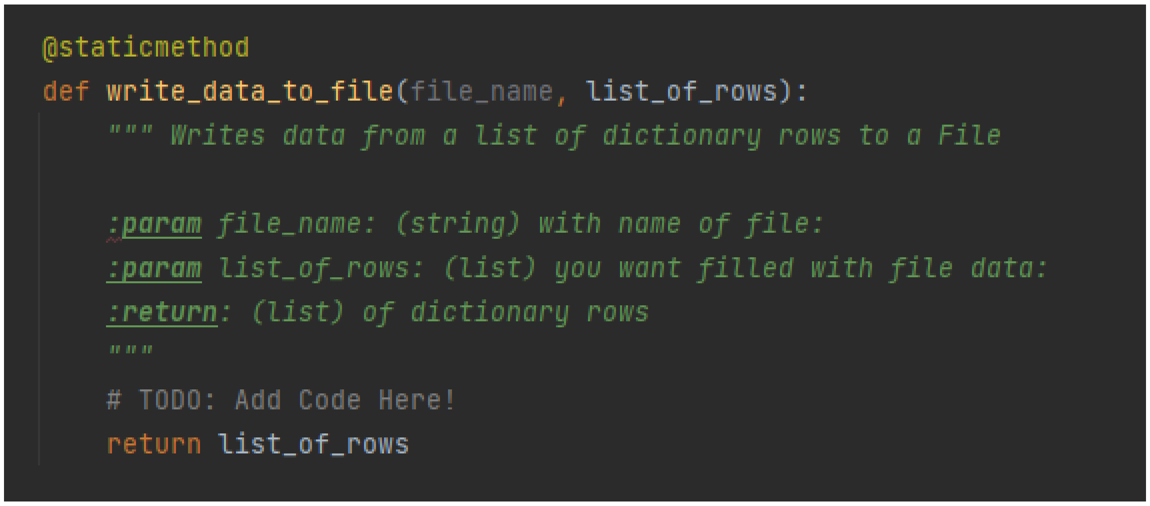
When the user chooses option 3 from the menu, this saves the data to the “ToDoFile.txt”. This is done with the function ‘write_data_to_file’. (See Figure 10)



```
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
```

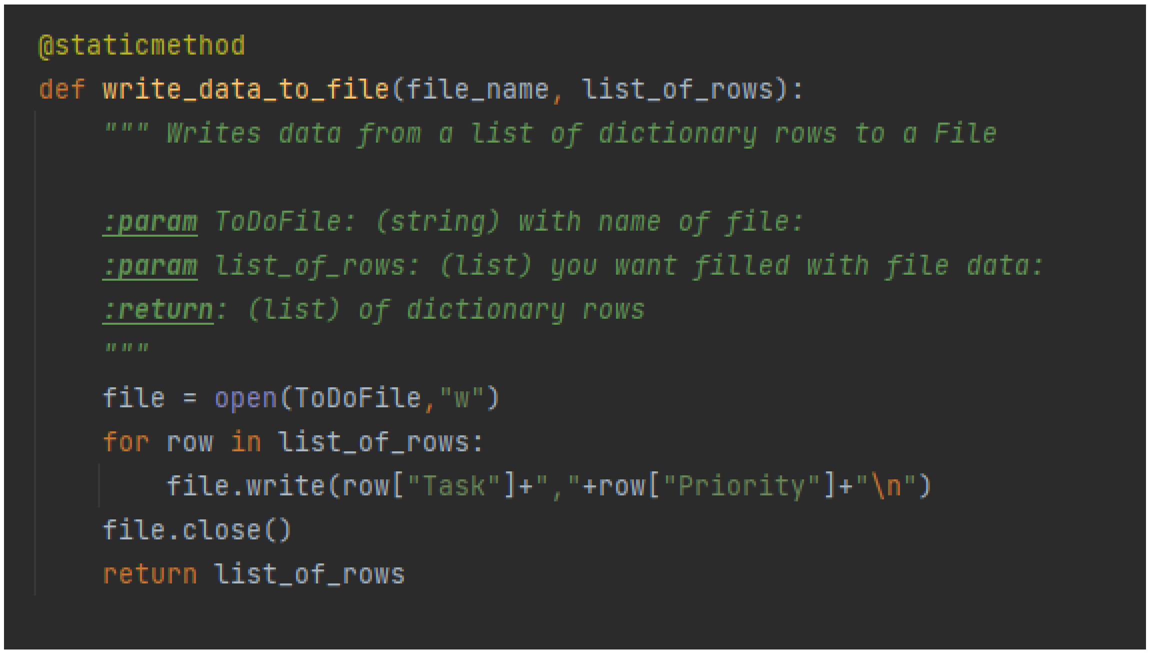
Figure 10 – Calling the function ‘write_data_to_file’

The code "TODO" here tells the computer to write (w) to the "ToDoFile" the "Task" and "Priority". (See figures 11 and 12)



```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
    """
    # TODO: Add Code Here!
    return list_of_rows
```

Figure 11

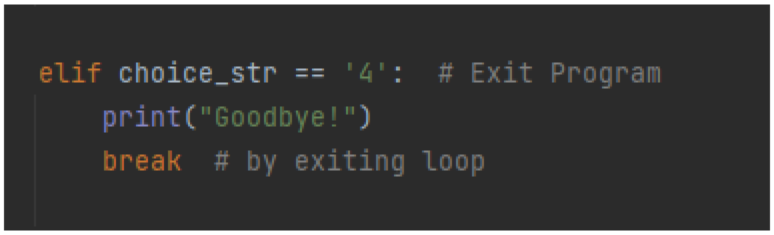


```python
@staticmethod
def write_data_to_file(file_name, list_of_rows):
    """ Writes data from a list of dictionary rows to a File

    :param ToDoFile: (string) with name of file:
    :param list_of_rows: (list) you want filled with file data:
    :return: (list) of dictionary rows
    """
    file = open(ToDoFile,"w")
    for row in list_of_rows:
        file.write(row["Task"]+","+row["Priority"]+"\n")
    file.close()
    return list_of_rows
```

Figure 12 – Writing the rows to the "ToDoFile"

## Exiting the Program

When users choose option 4, the program prints "Goodbye", then exits the loop with the command 'break'. (See Figure 13)

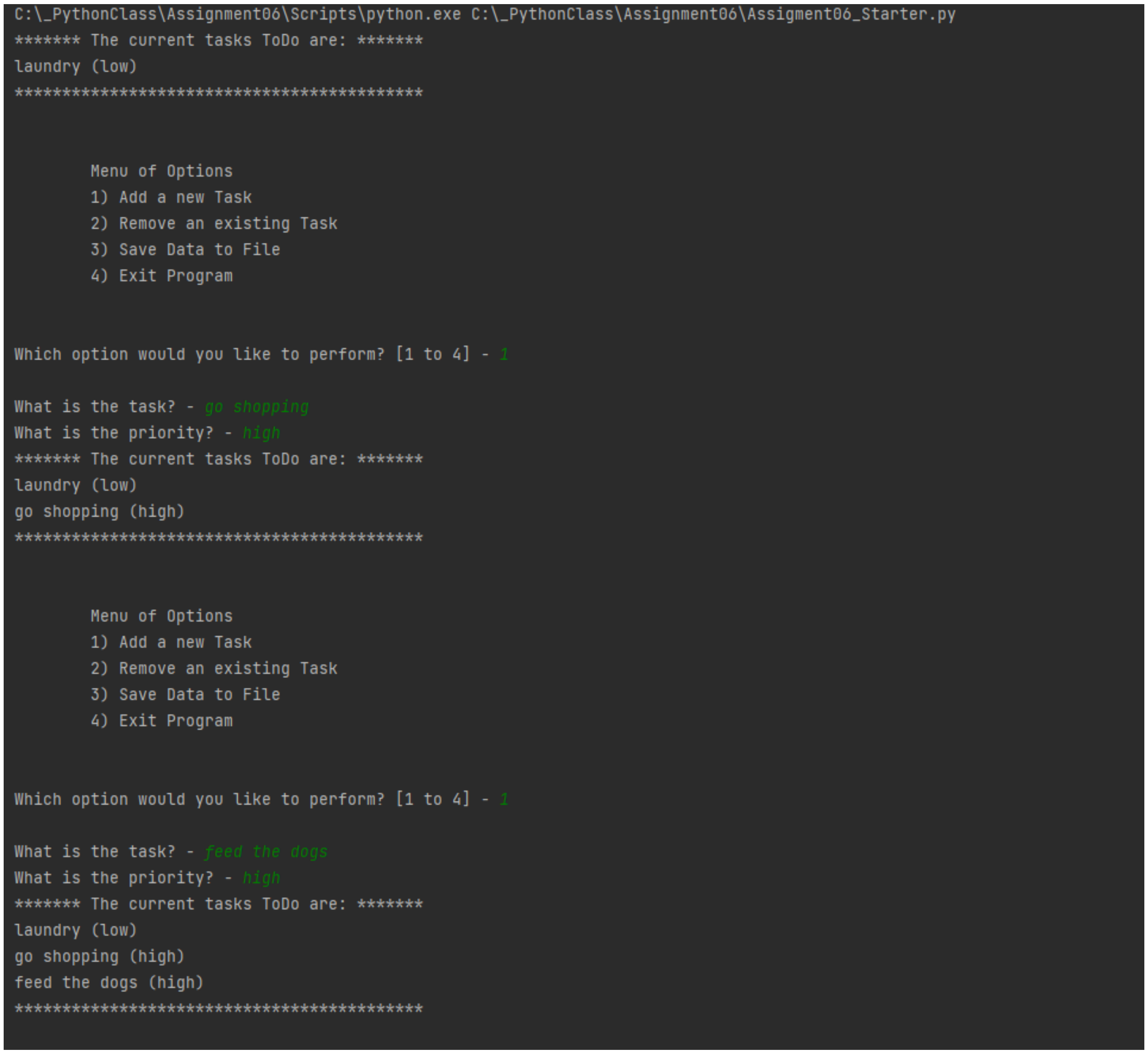


```
elif choice_str == '4':  # Exit Program
    print("Goodbye!")
    break  # by exiting loop
```

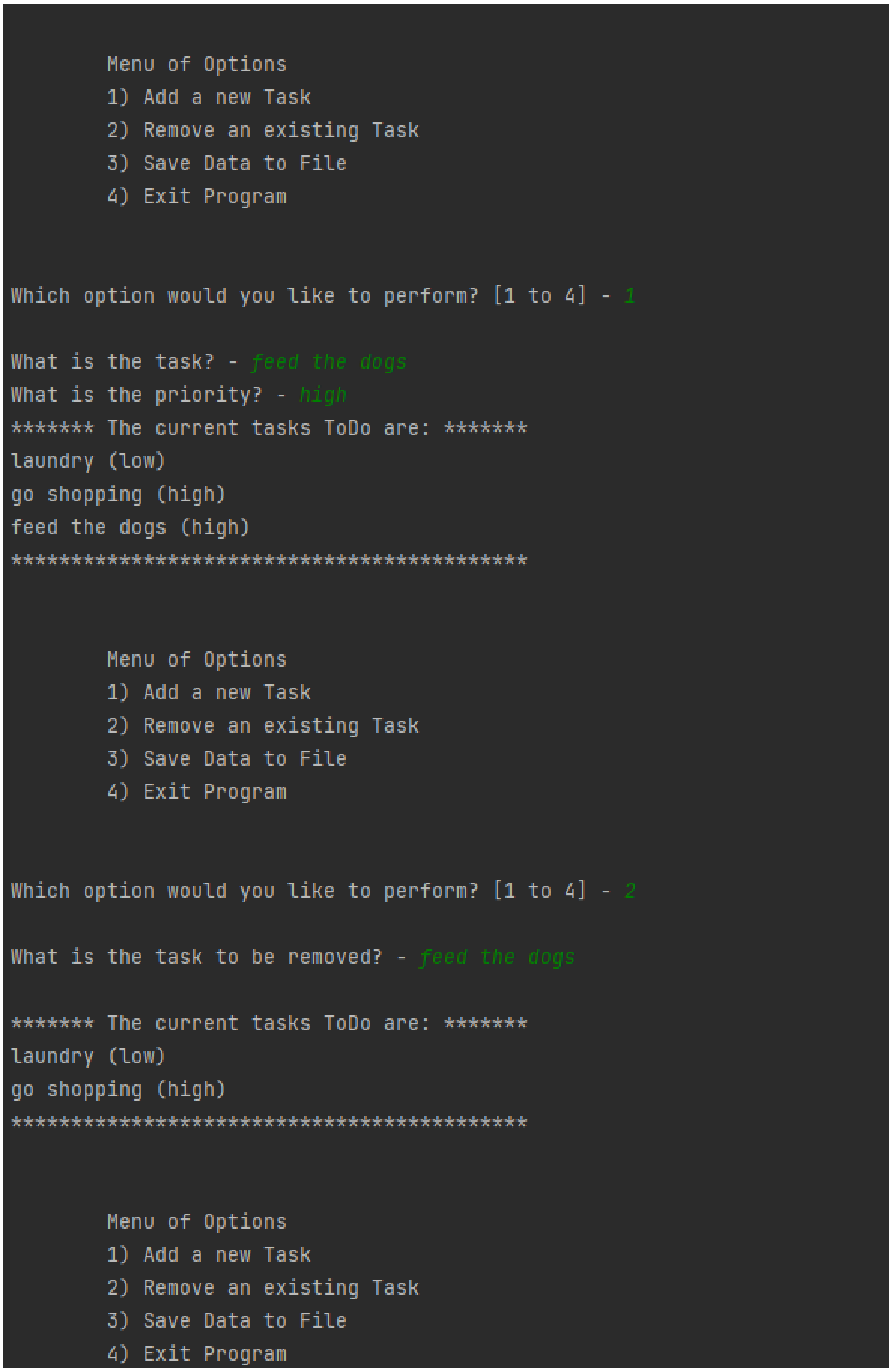
Figure 13 – Exiting the program

## Testing the Code

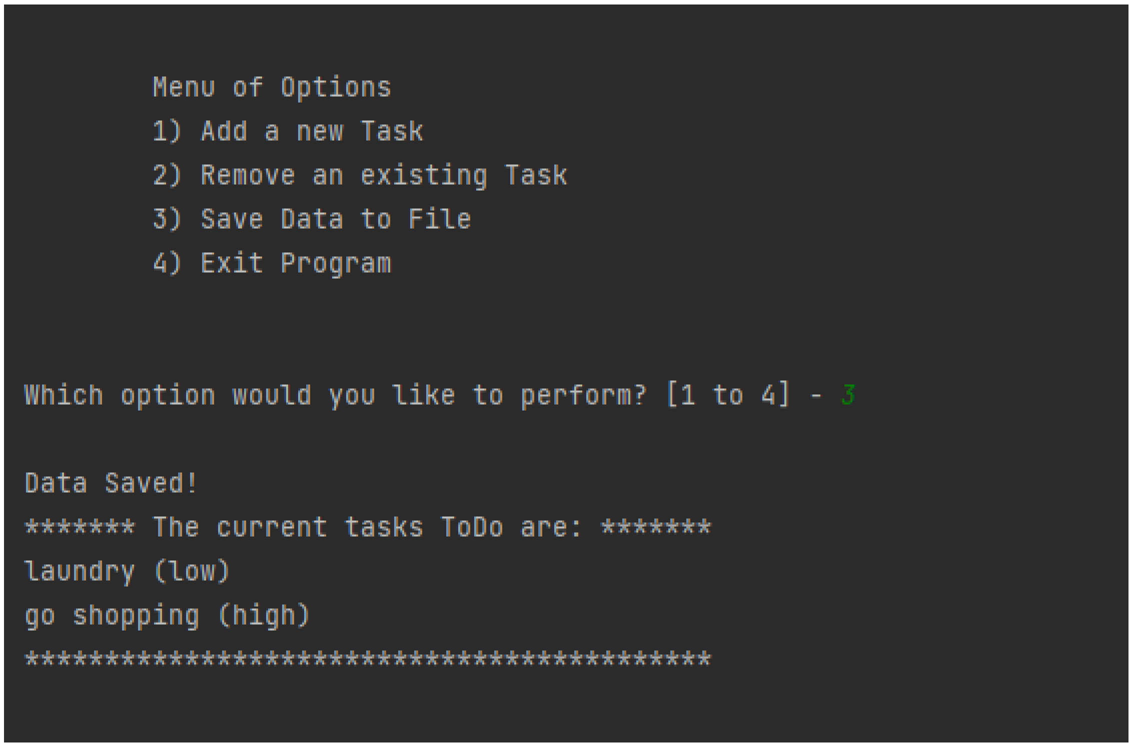
Here is the program running in PyCharm



```
C:\_PythonClass\Assignment06\Scripts\python.exe C:\_PythonClass\Assignment06\Assigment06_Starter.py
******* The current tasks ToDo are: *******
laundry (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

What is the task? - go shopping
What is the priority? - high
******* The current tasks ToDo are: *******
laundry (low)
go shopping (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] - 1

What is the task? - feed the dogs
What is the priority? - high
******* The current tasks ToDo are: *******
laundry (low)
go shopping (high)
feed the dogs (high)
*******************************************
```

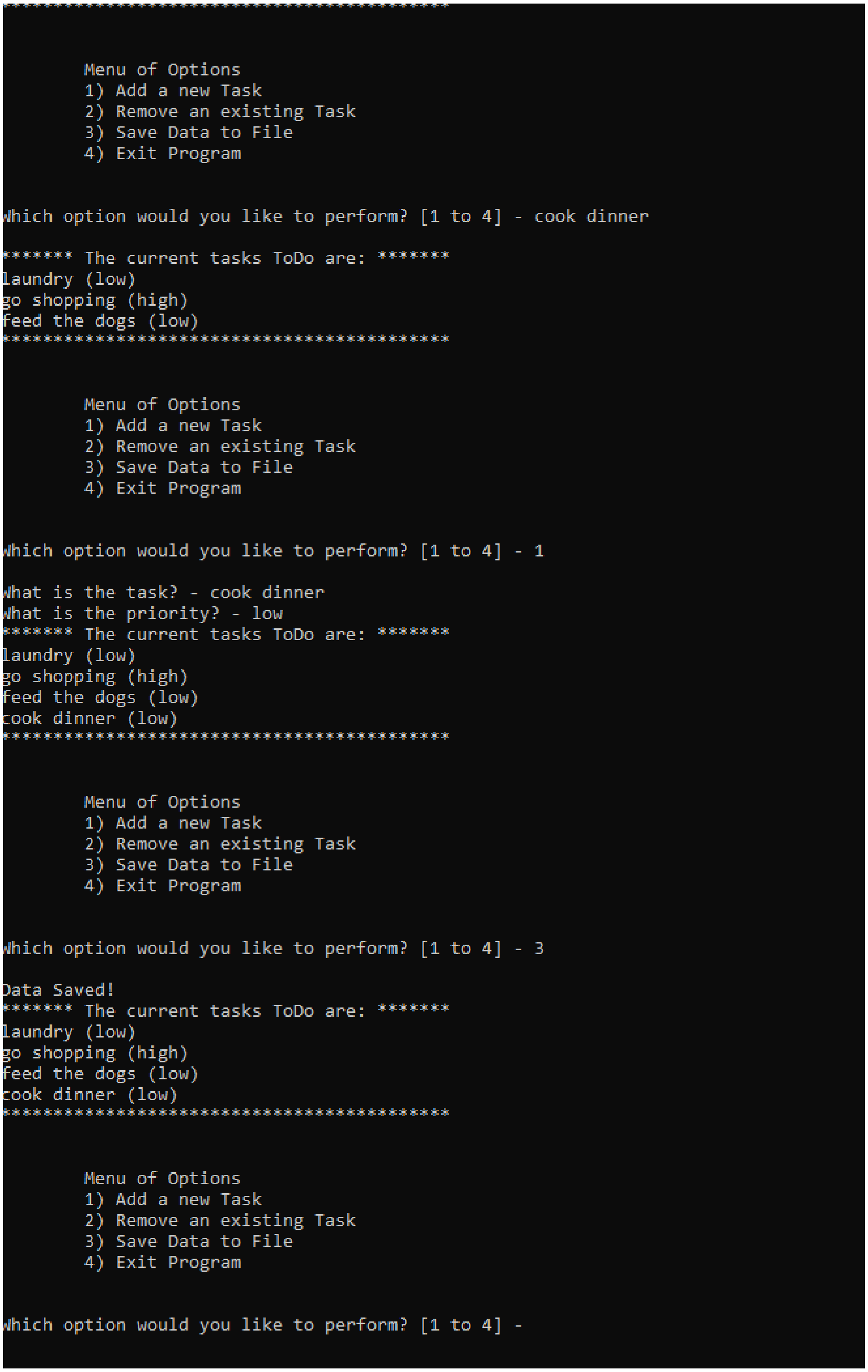


```

        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] - 1

What is the task? - feed the dogs
What is the priority? - high
******* The current tasks ToDo are: *******
laundry (low)
go shopping (high)
feed the dogs (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

What is the task to be removed? - feed the dogs

******* The current tasks ToDo are: *******
laundry (low)
go shopping (high)
*******************************************


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

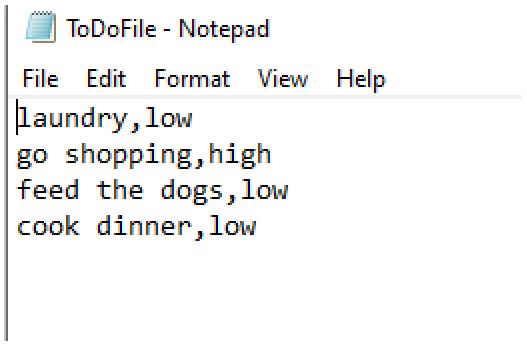


```
        Menu of Options
        1) Add a new Task
        2) Remove an existing Task
        3) Save Data to File
        4) Exit Program


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

Data Saved!
******* The current tasks ToDo are: *******
laundry (low)
go shopping (high)
*******************************************
```

Testing in command prompts



```
*******************************************


        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] - cook dinner

******* The current tasks ToDo are: *******
laundry (low)
go shopping (high)
feed the dogs (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

What is the task? - cook dinner
What is the priority? - low
******* The current tasks ToDo are: *******
laundry (low)
go shopping (high)
feed the dogs (low)
cook dinner (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: *******
laundry (low)
go shopping (high)
feed the dogs (low)
cook dinner (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] -
```



```
laundry,low
go shopping,high
feed the dogs,low
cook dinner,low
```

## Summary

By calling functions, it saves the programmer time and energy by not having to write the same thing repeatedly. If the programmer writes the function once, then just calls the function every time they want the same task done, it makes the code cleaner and faster.