

Name: David Michalove

Date: 17/05/23

Course Name: IT FDN 110 A Sp 23: Foundations Of Programming: Python

Assignment: Assignment05

To Do or Not To Do

Introduction

This paper is broken up into two main parts: *Structures* and *The Final Product*. There are two parts within *Structures*: *Thoughts*, *Pseudocode* and *The Code Itself*. *The Final Product* consists of two figures: figure 1, figure 2, and figure 3. Figure 1 shows the code as run through PyCharm CE, figure 2 shows *ToDoList.txt* file before the program has been run and Figure 3 shows *ToDoList.txt* file after the program has been run.

Structures

Thoughts

Dictionaries allowed for pairs of data to easily be called and utilized throughout the code. Using dictionaries within a list, as a table, showed both the flexibility of lists and gave a two fold structure that could be used throughout the program. The ability to read text files and save its contents to memory increases the range of our programming skills in a manner suited to the flexibility of lists. The two complement each other as was experienced when writing the first part of the program which captures and saves the data that is already present in the text file. The starting structure that was present gave another chance to experience flexibility when coding. That is to say, two ways of thinking, coding, had to come together by following the basic, backbone, structure throughout and backfilling said structure with suitable code. One theme, among many, of this project is *flexibility*.

Pseudocode

The Proposition:

"When the program starts, load each "row" of data in "ToDoList.txt" into a python Dictionary."

Main Steps:

1. Load data from *ToDoList.txt* file into a list of dictionary rows.
2. "Display a menu of choices to the user."
3. "Show the current items in the table."
4. "Add a new item to the list/Table."
5. "Remove a new item from the list/Table."
6. "Save tasks to the *ToDoList.txt* file."
7. Exit program

Name: David Michalove

Date: 17/05/23

Course Name: IT FDN 110 A Sp 23: Foundations Of Programming: Python

Assignment: Assignment05

Detailed Steps:

Step 1: Load data from ToDoList.txt file into a list of dictionary rows.

1. Open and read the file.
 - a. `strFile = open(objFile, "r")`
2. Run through the file and collect its information.
 - a. `for row in strFile:`
 - i. `lstRow = row.split(",")` # splits apart data based on a comma and returns a list.
 - b. Need to place items into a dictionary.
 - i. `dicRow = {"task":lstRow[0],"priority":lstRow[1].strip()}` # Needs to be within the for loop in Detailed Steps/Step 1/ 2a.
 - c. Need to save said data to the main list table.
 - i. `lstTable.append(dicRow)`

Step 2: "Display a menu of choices to the user."

This code has been provided by RRoot,1.1.2030, who created the started script.

Step 3: "Show the current items in the table."

1. Use a loop that goes through the table of dictionaries.
2. `for objRow in lstTable:`
 - a. `print(objRow)`

Step 4: "Add a new item to the list/Table."

1. Needs to get both the task and its priority from the user.
 - a. Saves user's input to two variables.
 - b. `strTask = input("Enter Task: ")`
 - c. `strPriority = input("Enter Priority: ")`
2. Needs to create a dictionary with the user's input.
 - a. `dicRow = {"task": strTask, "priority": strPriority}`
3. Needs to place this dictionary into the list table.
 - a. `lstTable.append(dicRow)`

Step 5: "Remove a new item from the list/Table."

1. Can use indexing to select which dictionary items will be deleted.
 - a. Need an indexing function.
 - i. `len()`
 - ii. `amountOptions = len(lstTable)`
 - b. User needs to be able to pick a certain index number for removing the desired dictionary.
 - i. Variable saved with an integer from the user
 - ii. `term = int(input("What task/priority do you wish to remove?\nPlease choose from 0 to "+ str(amountOptions - 1) + ": "))`

Name: David Michalove

Date: 17/05/23

Course Name: IT FDN 110 A Sp 23: Foundations Of Programming: Python

Assignment: Assignment05

2. The selected dictionary needs to be removed.
 - a. Can use .pop() to remove the dictionary and return it so that the user is notified again of what dictionary has been removed. Better to be redundant and transparent when removing data.
 - b. lstTable.pop(term)

Step 6: "Save tasks to the ToDoList.txt file."

1. Needs to save items but without repeating since what is already in the text file is in the table and just saving items in the table will cause those items to repeat.
 - a. Can clear out the text file or subtract lists
 - b. Subtracting lists leads to a problem when one dictionary is removed and this cannot be accounted for if the user deletes something after having saved it. If the user saves the table again after deleting a dictionary then the text file will have erroneous data that does not reflect the wish of the user.
 - c. Clearing out the text file before the table is saved solves the above problem.
 - i. open('ToDoList.txt', 'w').close()
2. Need to go through the list and save each dictionary key with its value side by side.
 - a. Can use a for loop that runs through the length of the list table while indexing each dictionary so that the key and value pair is saved each loop.
 - i. for i in range(len(lstTable)):
 1. dltem = lstTable[i]
 2. objFile.write(dltem.get("task") + ', ' + dltem.get("priority") + "\n")

QEF

The Code Itself

```
# ----- #
# Title: Assignment 05
# Description: Working with Dictionaries and Files
#             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
# <David Michalove>,<05/15/23>,Added code to complete assignment 5
# ----- #

# -- Data -- #
# declare variables and constants
objFile = "ToDoList.txt" # An object that represents a file
strData = "" # A row of text data from the file
dicRow = {} # A row of data separated into elements of a dictionary
{Task,Priority}
```

Name: David Michalove

Date: 17/05/23

Course Name: IT FDN 110 A Sp 23: Foundations Of Programming: Python

Assignment: Assignment05

```
lstTable = [] # A list that acts as a 'table' of rows
strMenu = "" # A menu of user options
strChoice = "" # A Capture the user option selection
strFile = None # File handle

# -- Processing -- #
# Step 1 - When the program starts, load any data you have
# in a text file called ToDoList.txt into a python list of dictionaries rows
# (like Lab 5-2)
print("\n Here are your current items if any:")
strFile = open(objFile, "r")
# Reads the data in the txt file
for row in strFile:
# Loops through the content within the txt file
    lstRow = row.split(",")
    # splits apart data based on a comma and returns a list
    dicRow = {"task":lstRow[0],"priority":lstRow[1].strip()}
    lstTable.append(dicRow)
    print(dicRow)
strFile.close()

# -- Input/Output -- #
# Step 2 - Display a menu of choices to the user
while (True):
    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 5] -
"))
    print() # Adding a new line for looks
    # Step 3 - Show the current items in the table
    if (strChoice.strip() == '1'):
        for objRow in lstTable:
            # Prints the contents of lstTable
            print(objRow)
        continue
    # Step 4 - Add a new item to the list/Table
    elif (strChoice.strip() == '2'):
        strTask = input("Enter Task: ")
        # Saves the user's input to the key: task.
        strPriority = input("Enter Priority: ")
        # Saves the user's input to the key: priority.
```

Name: David Michalove

Date: 17/05/23

Course Name: IT FDN 110 A Sp 23: Foundations Of Programming: Python

Assignment: Assignment05

```
dicRow = {"task": strTask, "priority": strPriority}
# Main dictionary that will make up the lstTable
lstTable.append(dicRow)
# Puts dictionary into a list making a table
continue

# Step 5 - Remove a new item from the list/Table
elif (strChoice.strip() == '3'):
    amountOptions = len(lstTable)
    # Gets the length of the table for indexing
    print("Here are your tasks so far:")
    for objRow in lstTable:
        # Prints the contents of lstTable
        print(objRow)

    term = int(input("What task/priority do you wish to remove?\nPlease
choose from 0 to " + str(amountOptions - 1) + ": "))
    # Gets the user's input for what term is wished to be removed
    while term >= amountOptions:
        # Uses the greater than or equal to cases because this part tests
for an out of range index selection
        print("Please only respond with a number from 0 to " +
str(amountOptions - 1) + ".")
        term = int(input("What task/priority do you wish to remove?\nPlease
choose from 0 to " + str(amountOptions - 1) + ": "))
    yesNo = input("You wish to remove: " + str(lstTable[term]) + "? [yes/no]
")

    # Asks user if they wish to proceed
    print(lstTable[term])
    if yesNo.lower() == "yes":
        lstTable.pop(term)
        # Removes and returns the selected dictionary so the user can see
what has been deleted
        continue
    elif (strChoice.strip() == '4'):
        # Step 6 - Save tasks to the ToDoToDoList.txt file
        open('ToDoList.txt', 'w').close()
        # Clears what is in the txt file so nothing will be duplicated by
writing lstTable into the txt file
        objFile = open("ToDoList.txt", "a")
        for i in range(len(lstTable)):
            # Saves each dictionary pair to a txt file
            dItem = lstTable[i]
            print(dItem.get("task") + ", " + dItem.get("priority"))
            objFile.write(dItem.get("task") + ', ' + dItem.get("priority") +
"\n")

        objFile.close()
        print("Saved!")
```

Name: David Michalove

Date: 17/05/23

Course Name: IT FDN 110 A Sp 23: Foundations Of Programming: Python

Assignment: Assignment05

```
        continue
# Step 7 - Exit program
elif (strChoice.strip() == '5'):
    finalYesNo = input("Do you wish to end the program? [yes/no] ")
    while finalYesNo.lower() != "yes" and finalYesNo.lower() != "no":
        # Ensures user will only respond within the asked for parameters
        print("Please only respond with: [yes/no] ")
        finalYesNo = input("Do you wish to end the program? [yes/no] ")
    if finalYesNo.lower() == "no":
        continue
    break # and Exit the program
```

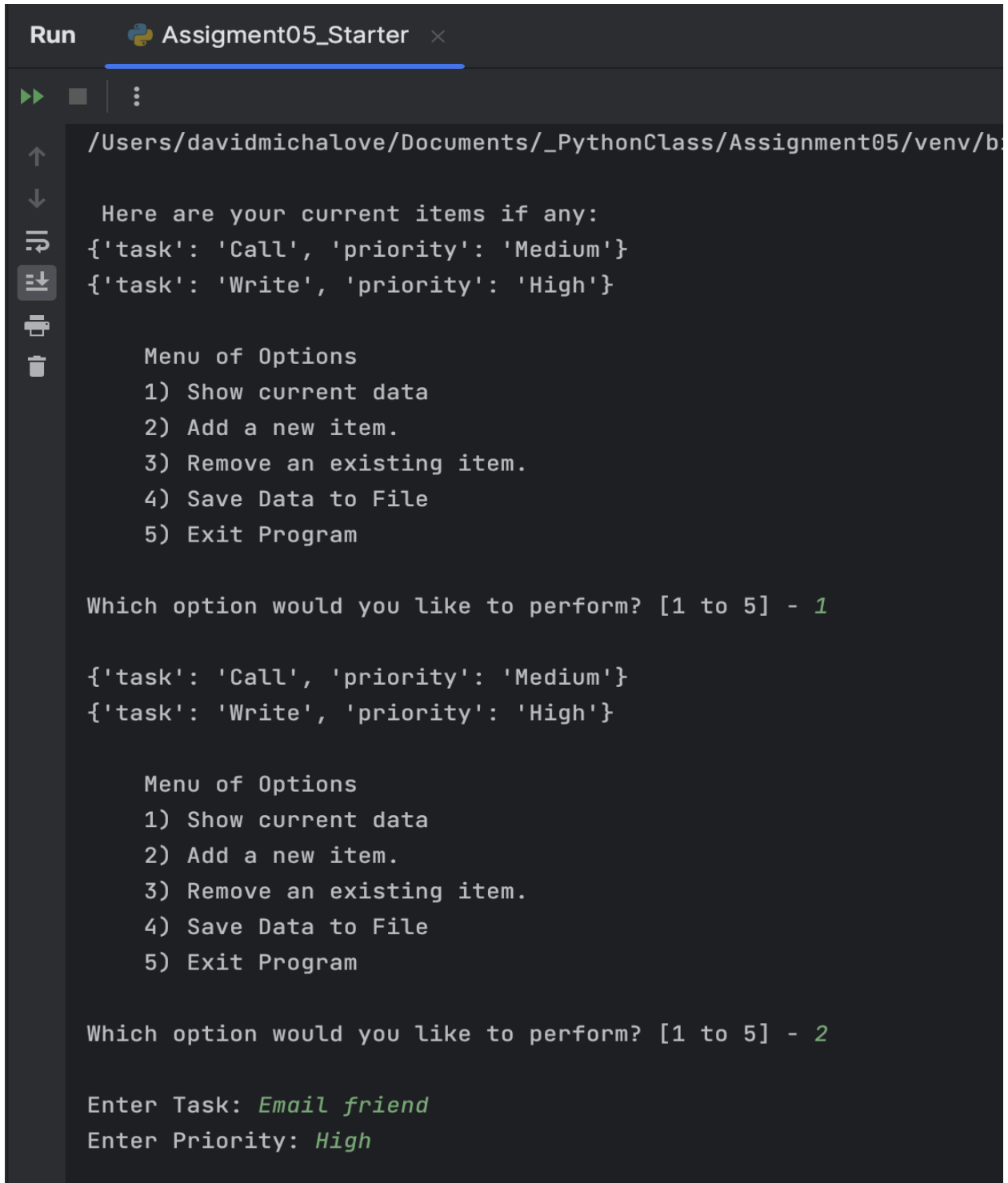
Name: David Michalove

Date: 17/05/23

Course Name: IT FDN 110 A Sp 23: Foundations Of Programming: Python

Assignment: Assignment05

The Final Product



```
Run Assignment05_Starter x
/Users/davidmichalove/Documents/_PythonClass/Assignment05/venv/bin/python3
Here are your current items if any:
{'task': 'Call', 'priority': 'Medium'}
{'task': 'Write', 'priority': 'High'}

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

Which option would you like to perform? [1 to 5] - 1

{'task': 'Call', 'priority': 'Medium'}
{'task': 'Write', 'priority': 'High'}

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

Which option would you like to perform? [1 to 5] - 2

Enter Task: Email friend
Enter Priority: High
```

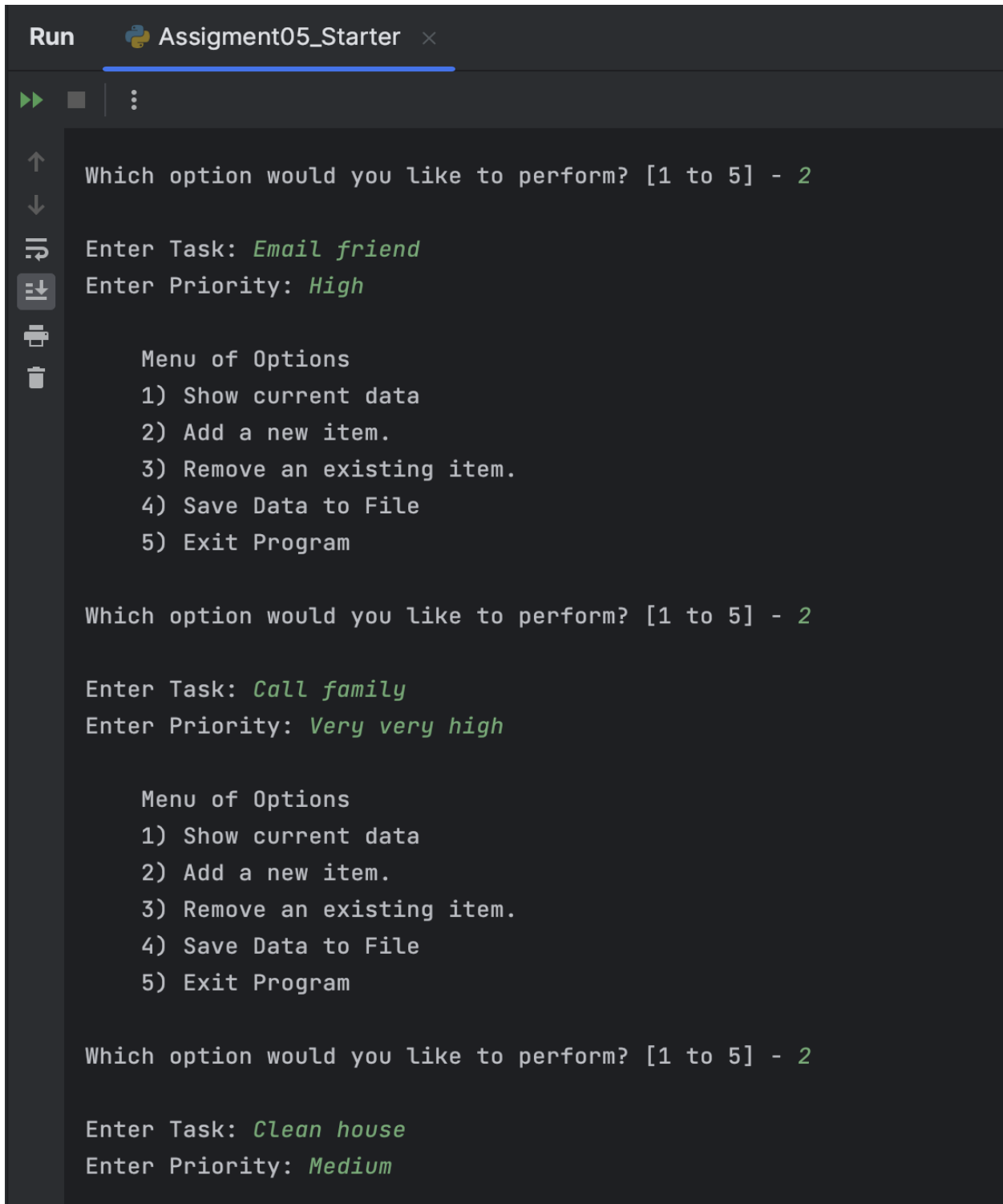
Figure 1.0: The code as run through PyCharm CE

Name: David Michalove

Date: 17/05/23

Course Name: IT FDN 110 A Sp 23: Foundations Of Programming: Python

Assignment: Assignment05



```
Run Assignment05_Starter x
>>
↑
↓
↶
↷
Enter Task: Email friend
Enter Priority: High

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

Which option would you like to perform? [1 to 5] - 2

Enter Task: Call family
Enter Priority: Very very high

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

Which option would you like to perform? [1 to 5] - 2

Enter Task: Clean house
Enter Priority: Medium
```

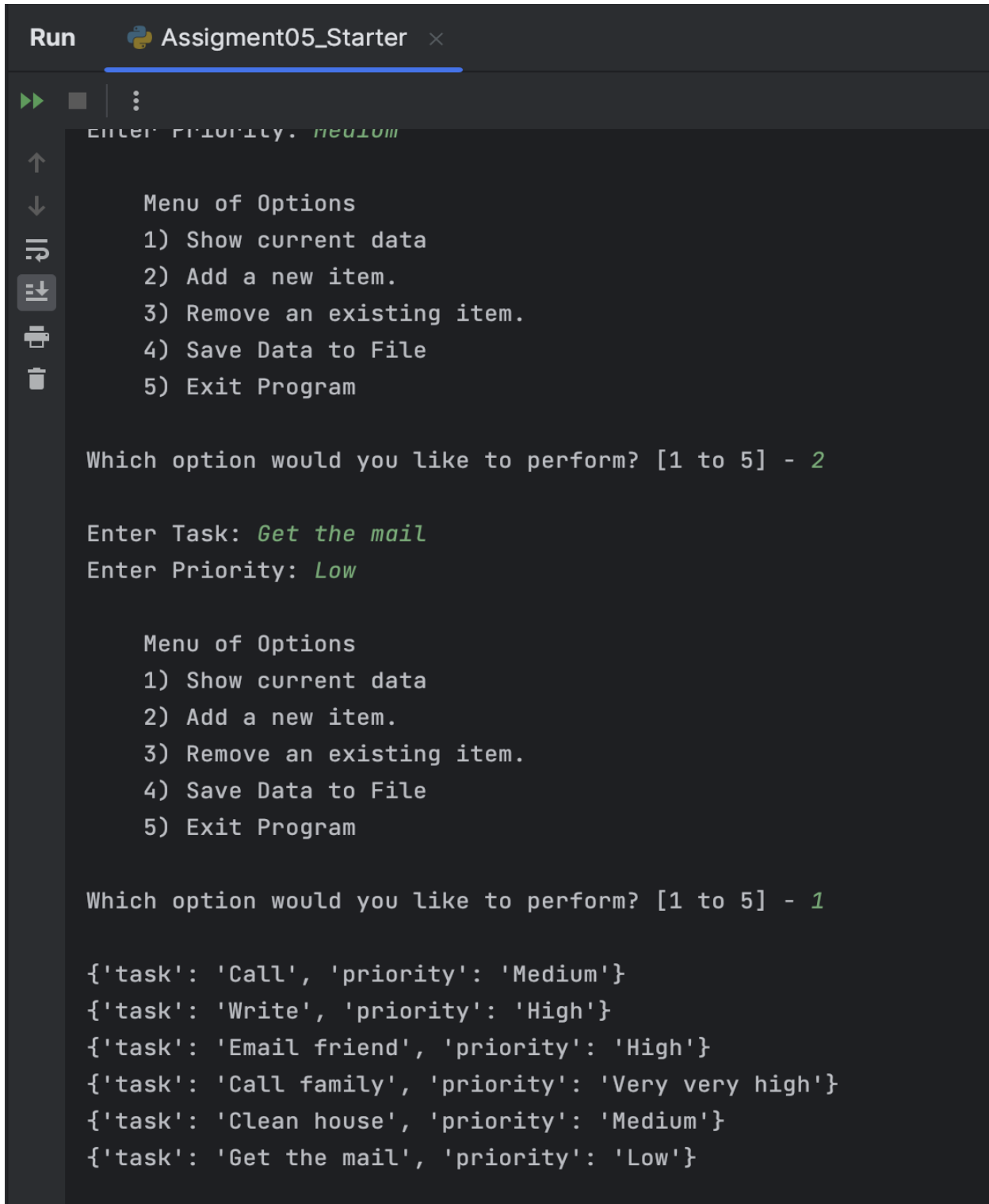
Figure 1.1: The code as run through PyCharm CE

Name: David Michalove

Date: 17/05/23

Course Name: IT FDN 110 A Sp 23: Foundations Of Programming: Python

Assignment: Assignment05



```
Run Assignment05_Starter x
Enter Priority: medium

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

Which option would you like to perform? [1 to 5] - 2

Enter Task: Get the mail
Enter Priority: Low

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

Which option would you like to perform? [1 to 5] - 1

{'task': 'Call', 'priority': 'Medium'}
{'task': 'Write', 'priority': 'High'}
{'task': 'Email friend', 'priority': 'High'}
{'task': 'Call family', 'priority': 'Very very high'}
{'task': 'Clean house', 'priority': 'Medium'}
{'task': 'Get the mail', 'priority': 'Low'}
```

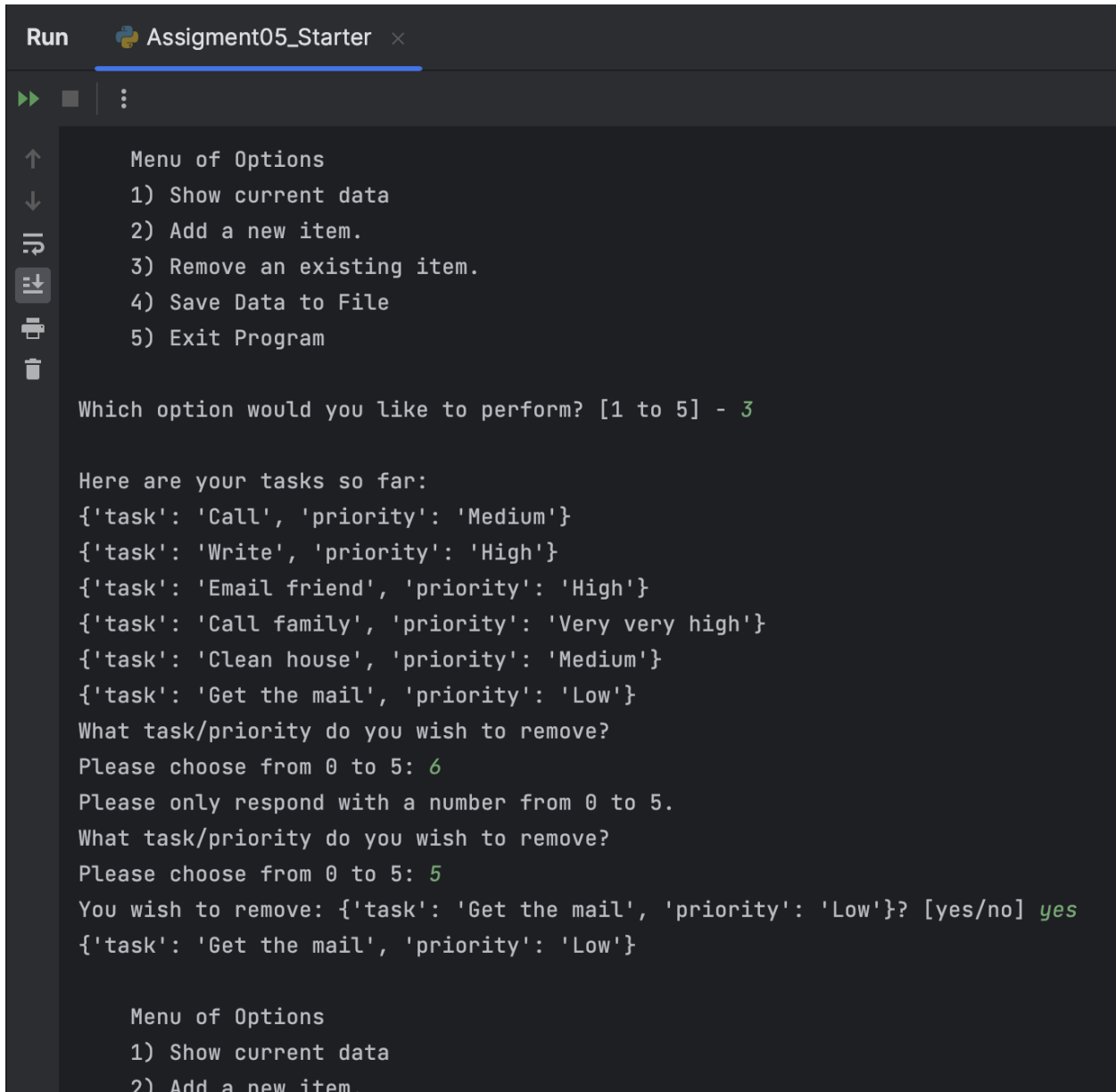
Figure 1.2: The code as run through PyCharm CE

Name: David Michalove

Date: 17/05/23

Course Name: IT FDN 110 A Sp 23: Foundations Of Programming: Python

Assignment: Assignment05



```
Run  Assigment05_Starter x
>>
↑      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
⇅
Which option would you like to perform? [1 to 5] - 3

Here are your tasks so far:
{'task': 'Call', 'priority': 'Medium'}
{'task': 'Write', 'priority': 'High'}
{'task': 'Email friend', 'priority': 'High'}
{'task': 'Call family', 'priority': 'Very very high'}
{'task': 'Clean house', 'priority': 'Medium'}
{'task': 'Get the mail', 'priority': 'Low'}
What task/priority do you wish to remove?
Please choose from 0 to 5: 6
Please only respond with a number from 0 to 5.
What task/priority do you wish to remove?
Please choose from 0 to 5: 5
You wish to remove: {'task': 'Get the mail', 'priority': 'Low'}? [yes/no] yes
{'task': 'Get the mail', 'priority': 'Low'}

Menu of Options
1) Show current data
2) Add a new item.
```

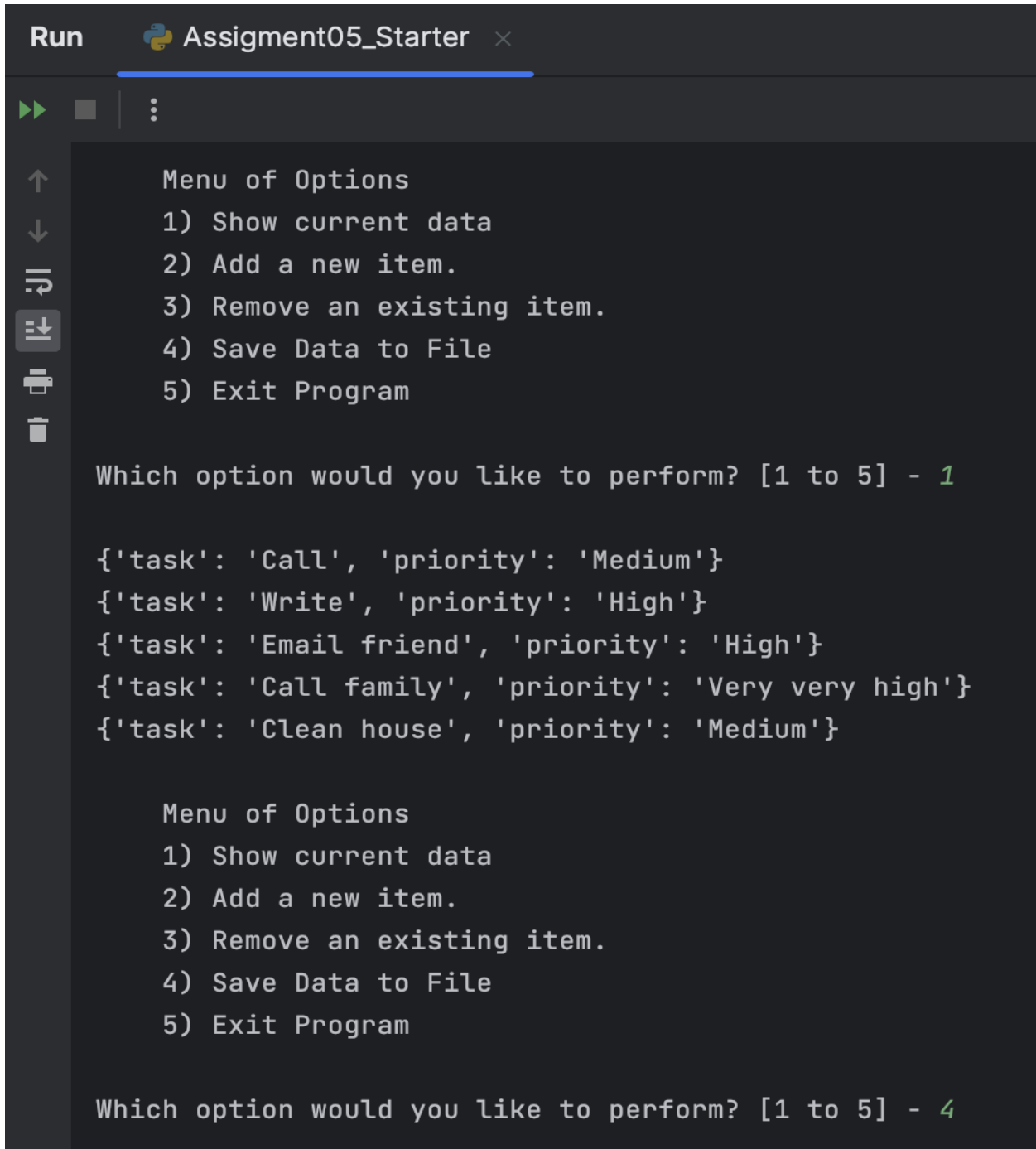
Figure 1.3: The code as run through PyCharm CE

Name: David Michalove

Date: 17/05/23

Course Name: IT FDN 110 A Sp 23: Foundations Of Programming: Python

Assignment: Assignment05



```
Run Assignment05_Starter x

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

Which option would you like to perform? [1 to 5] - 1

{'task': 'Call', 'priority': 'Medium'}
{'task': 'Write', 'priority': 'High'}
{'task': 'Email friend', 'priority': 'High'}
{'task': 'Call family', 'priority': 'Very very high'}
{'task': 'Clean house', 'priority': 'Medium'}

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

Which option would you like to perform? [1 to 5] - 4
```

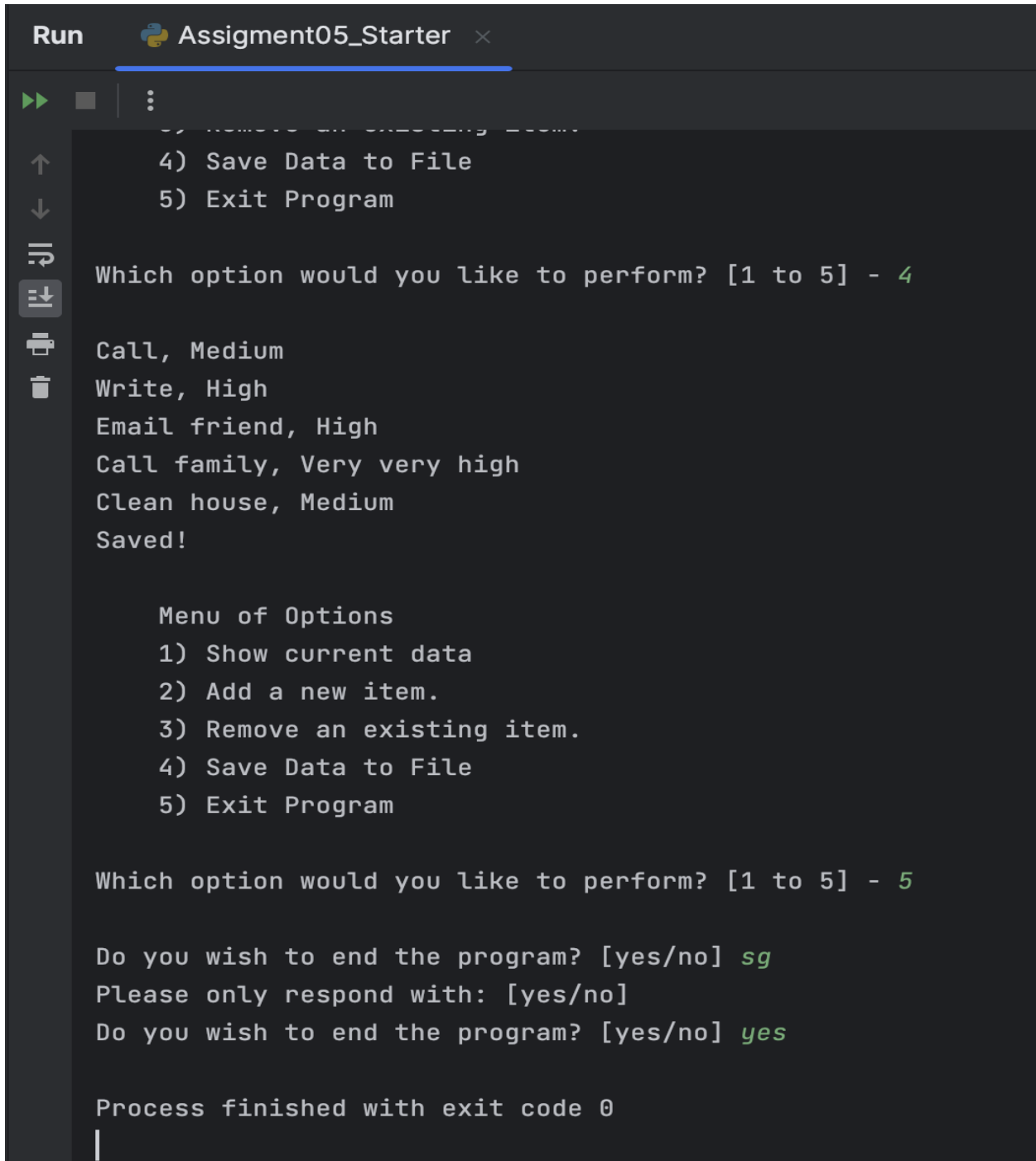
Figure 1.4: The code as run through PyCharm CE

Name: David Michalove

Date: 17/05/23

Course Name: IT FDN 110 A Sp 23: Foundations Of Programming: Python

Assignment: Assignment05



```
Run Assignment05_Starter x
>>
4) Save Data to File
5) Exit Program

Which option would you like to perform? [1 to 5] - 4

Call, Medium
Write, High
Email friend, High
Call family, Very very high
Clean house, Medium
Saved!

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

Which option would you like to perform? [1 to 5] - 5

Do you wish to end the program? [yes/no] sg
Please only respond with: [yes/no]
Do you wish to end the program? [yes/no] yes

Process finished with exit code 0
|
```

Figure 1.5: The code as run through PyCharm CE

Name: David Michalove

Date: 17/05/23

Course Name: IT FDN 110 A Sp 23: Foundations Of Programming: Python

Assignment: Assignment05

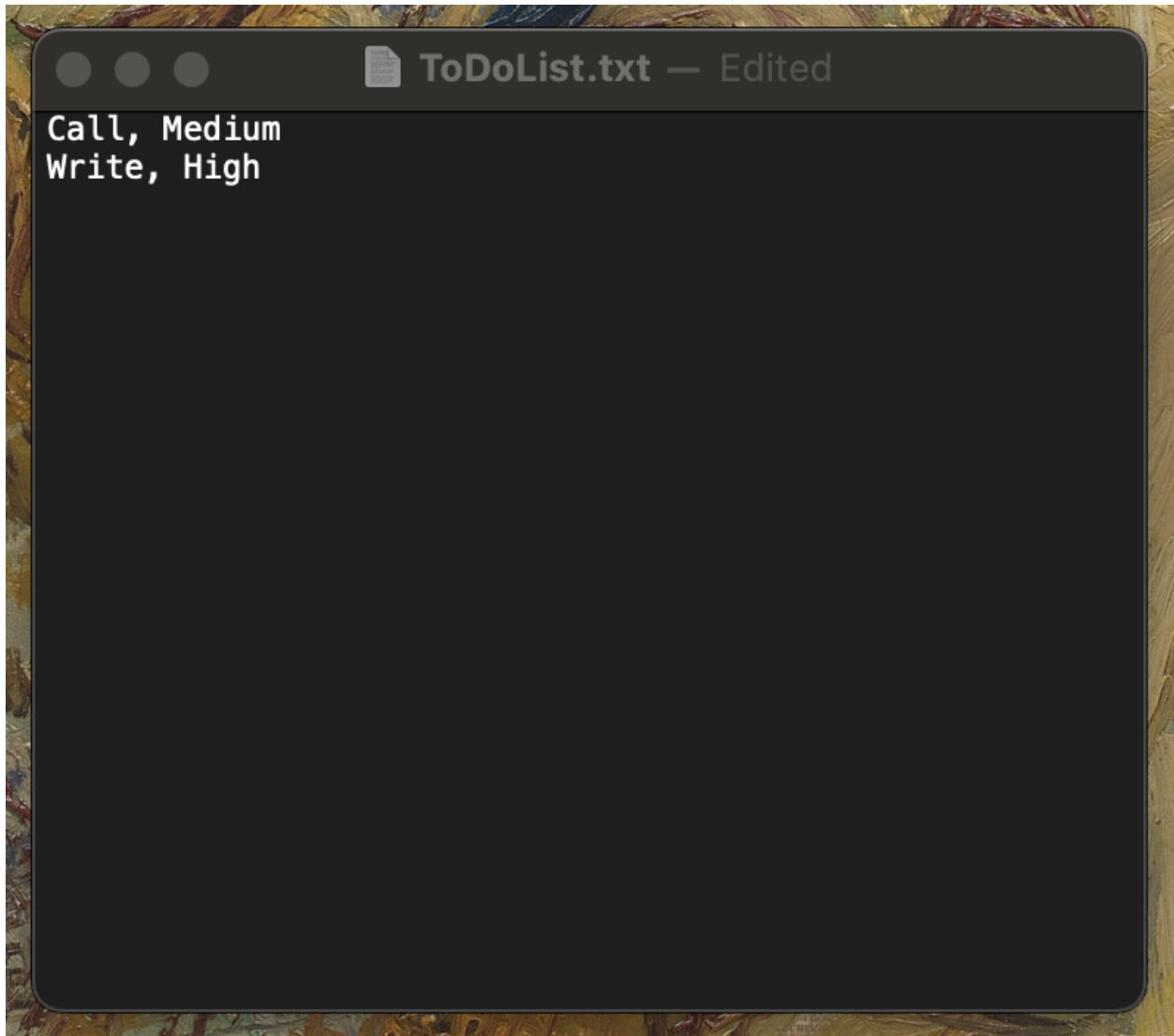


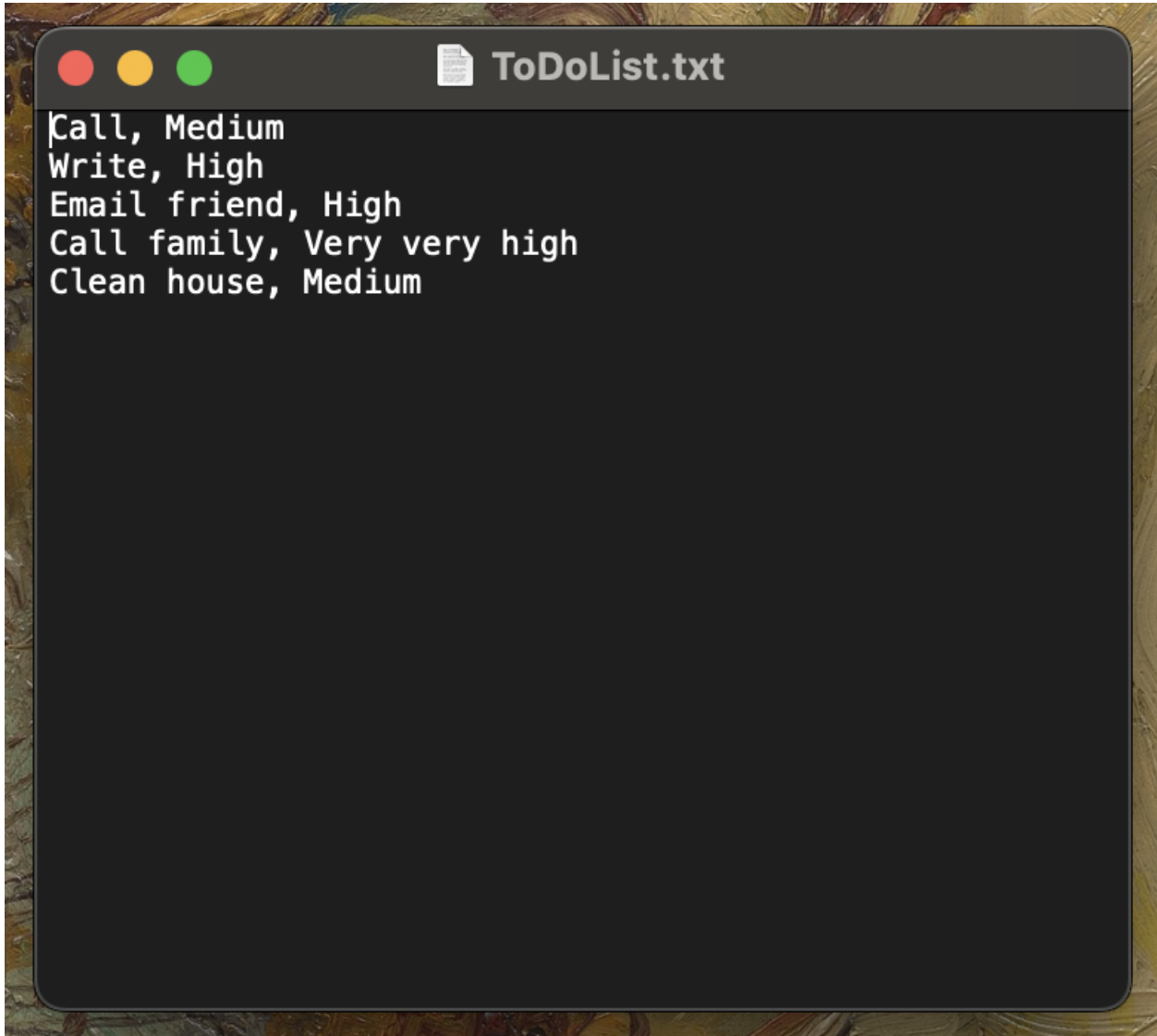
Figure 2: *ToDoList.txt* before the code has been run.

Name: David Michalove

Date: 17/05/23

Course Name: IT FDN 110 A Sp 23: Foundations Of Programming: Python

Assignment: Assignment05



```
Call, Medium
Write, High
Email friend, High
Call family, Very very high
Clean house, Medium
```

Figure 3: *ToDoList.txt* after the code has been run.

Summary

This paper went through two main parts: *Structures* and *The Final Product*. In *Structures* there are three parts therein: *Thoughts*, *Pseudocode*, and *The Code Itself*. The *Thoughts* section takes up the theme of flexibility in programming. The *Pseudocode* shows the primary steps taken before writing the program in PyCharm. *The Code Itself* shows the code as written in PyCharm. *The Final Product* showed the code's outcome as run in PyCharm and the before and after results for the text file.