Yuying Xie
May 28, 2025
Class: IT FDN 110 A
Assignment 06

Functions

Introduction

This assignment focuses on improving a course registration program by applying key programming concepts such as encapsulation, parameter passing, return values, and class-based design. I restructured the original procedural code into a modular format using functions and classes to promote code clarity, reusability, and maintainability. By separating concerns, adding parameters and return values, and documenting each part with Python docstrings, I enhanced the program's organization and made it easier to test, debug, and extend.

Key Concept

1. By declaring **parameters** in the function we can provide data to the function when you call it. This approach encapsulates the data required for the function within its parameters, making it clear what the function needs to operate correctly. It does not rely on global variable data which makes it less likely to cause errors and side-effects[1].
2. **Encapsulation** is a fundamental concept that involves bundling data and the processes that work on that data into a single "capsule" of code[2].
3. **Overloaded functions** refer to the ability to define multiple functions with the same name but different parameters (number or type). If you define multiple functions with the same name, the last definition will override the previous ones[3].
4. Passing argument: **Immutable objects** cannot be modified in place, so reassigning the parameter variable inside the function creates a new object. **Mutable objects** can be modified in place, so changes made inside the function affect the same object outside the function[4].
5. **Return values**: the return statement is used inside a function to send back a value (or multiple values) to the caller[5].
6. **Classes** are a way of grouping functions, variables, and constants by the name of the class[6].

---

[1] Mod06_Notes P15
[2] Mod06_Notes P15
[3] OpenAI. (2025). Overloaded functions in Python [Large language model response]. ChatGPT. https://chat.openai.com/
[4] Course video: https://www.youtube.com/watch?v=TAD_BczzOI0
[5] OpenAI. (2025). Return Values in Python [Large language model response]. ChatGPT. https://chat.openai.com/
[6] Mod 06_Notes P42

7. It is a common practice to include a header at the beginning of a class or function, which is known as docstring in python. It can be helpful when developers include additional notes in a docstring.
8. “In software development, a **"concern"** refers to a specific aspect or responsibility of a program's functionality. These concerns can include things like user interface (UI) logic, data storage, data processing, input validation, and more.”
9. “When concerns are mixed together, it becomes challenging to make changes or updates without affecting other parts of the code. By **separating concerns,** you can isolate changes, debug more effectively, and reduce the risk of introducing bugs.”

Edit Data Constants and Variables

I kept the Constants variable MENU and FILE_NAME unchanged since it’s okay to keep them in the global place. I commented out all other variables except for students and menu_choice since these two are needed when the first function is called (Figure 1).

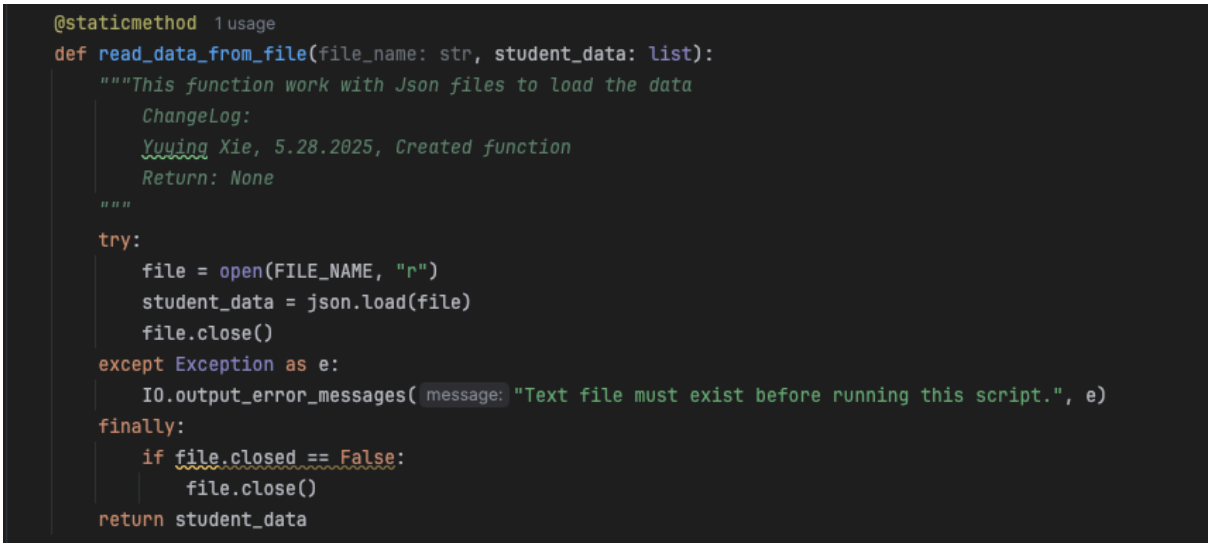
```
# Define the Data Constants
MENU: str = '''
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
----------------------------------------
'''
# Define the Data Constants
# FILE_NAME: str = "Enrollments.csv"
FILE_NAME: str = "Enrollments.json"

# Define the Data Variables and constants
# student_first_name: str = ''  # Holds the first name of a student entered by the user.
# student_last_name: str = ''  # Holds the last name of a student entered by the user.
# course_name: str = ''  # Holds the name of a course entered by the user.
# student_data: dict = {}  # one row of student data
students: list = []  # a table of student data
# file = None  # Holds a reference to an opened file.
menu_choice: str = ''  # Hold the choice made by the user.
```

Figure 1. Constants and Variables

Create Function for Each Part

In the original code, processes like reading and writing data, printing the menu, taking input, and displaying outputs were written inline, making the code difficult to manage. In the revised version, I moved each of these tasks into clearly defined functions such as

read_data_from_file(), write_data_to_file(), output_menu(), input_student_data(), and so on. I also added “@staticmethod” for each of them. See Figure 2 as an example.



```
@staticmethod 1 usage
def read_data_from_file(file_name: str, student_data: list):
    """This function work with Json files to load the data
        ChangeLog:
        Yuying Xie, 5.28.2025, Created function
        Return: None
    """
    try:
        file = open(FILE_NAME, "r")
        student_data = json.load(file)
        file.close()
    except Exception as e:
        IO.output_error_messages( message: "Text file must exist before running this script.", e)
    finally:
        if file.closed == False:
            file.close()
    return student_data
```

Figure 2. An example of function that I created

Add Parameters and Return Values

I also added input parameters and return values where appropriate, so each function can process data without relying on global variables. This makes the program logic easier to test and reuse while minimizing unintended side effects (also see figure 2 as an example).

Using Class – The Separation of Concerns Pattern

To implement better structure and modularity, I introduced two classes—FileProcessor and IO. The FileProcessor class contains static methods that handle reading and writing JSON data, effectively managing the program’s interaction with the file system. The IO class manages all user interaction, including input, output, menu navigation, and error messaging. This separation of concerns allows for cleaner organization and a clearer understanding of the program's layers: presentation (user interface), processing (business logic), and data access (file handling). With this structure, each class has a well-defined role, and changes in one part of the system will not break unrelated functionalities (See figure 3 FIlePRocessor class as an example).

```python
class FileProcessor:  2 usages
    """ A collection of processing layer functions that work with Json files
        ChangeLog:
        Yuying Xie, 5.27.2025,Created Class
        Yuying Xie, 5.28.2025,Added functions
        Return: student data:list
    """
    @staticmethod  1 usage
    def read_data_from_file(file_name: str, student_data: list):
        """This function work with Json files to load the data
            ChangeLog:
            Yuying Xie, 5.28.2025, Created function
            Return: None
        """
        try:
            file = open(FILE_NAME, "r")
            student_data = json.load(file)
            file.close()
        except Exception as e:
            IO.output_error_messages( message: "Text file must exist before running this script.", e)
        finally:
            if file.closed == False:
                file.close()
        return student_data
```

Figure 3 (Part1) FileProcessor Class

```python
    @staticmethod  1 usage
    def write_data_to_file(file_name: str, student_data: list):
        """This function write the data into file
            ChangeLog:
            Yuying Xie, 5.28.2025, Created function
            Return: None
        """
        try:
            file = open(FILE_NAME, "w")
            json.dump(students, file, indent=4)
            file.close()
            last_data_saved = True
            print("The following data was saved to file!")
            for student in students:
                print(f'Student {student["FirstName"]} '
                      f'{student["LastName"]} is enrolled in {student["CourseName"]}')
        except Exception as e:
            if file.closed == False:
                file.close()
            IO.output_error_messages( message: "Please check that the file is not open by another program.", e)
```

Figure 3 (Part1) FileProcessor Class

Add Document String

For all functions and classes I created or modified, I added Python docstrings that explain their purpose, usage, and change logs. These docstrings serve as in-code documentation that helps future readers (and myself) quickly understand what each function or class does without needing to examine its internal logic line-by-line. Including details about when the function was created and updated, as well as its parameters and return types, makes the code more maintainable and professional. This practice aligns with best standards in software development and promotes collaborative work.

Test results

See figure 4 I tested result in Pycharm; Figure 5 the json file and appendix 1 the result in Mac Terminal.

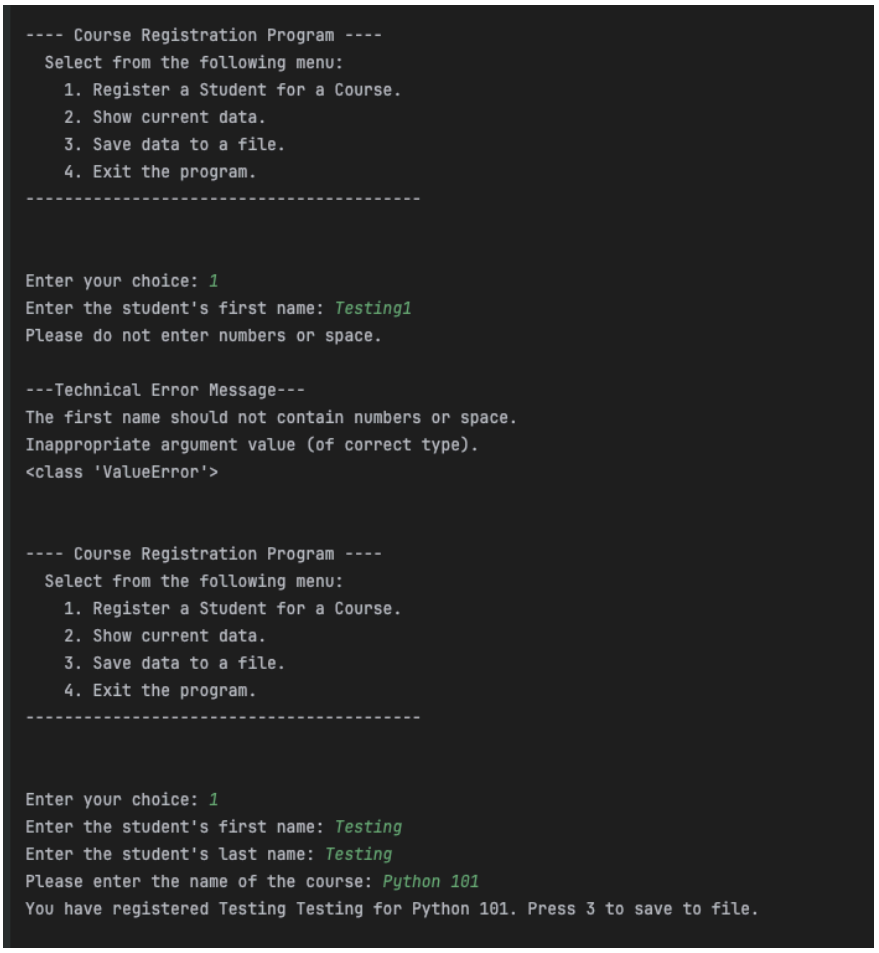


Figure 4. Test result in Pycharm

```
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
----------------------------------------


Enter your choice: 2
------------------------------------------------

The current data is:
Student Bob Smith is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student Testing Testing is enrolled in Python101
Student Testing Testing is enrolled in Python 101
------------------------------------------------


---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
----------------------------------------


Enter your choice: 3
The following data was saved to file!
Student Bob Smith is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student Testing Testing is enrolled in Python101
```

Figure 4. Test result in Pycharm

```
Student Testing Testing is enrolled in Python 101


---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
----------------------------------------


Enter your choice: 1
Enter the student's first name: Yuying
Enter the student's last name: Testing
Please do not enter numbers or space.

---Technical Error Message---
The last name should not contain numbers or space.
Inappropriate argument value (of correct type).
<class 'ValueError'>


---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
----------------------------------------
```

Figure 4. Test result in Pycharm

```
Enter your choice: Yuying
Invalid choice

Please only choose option 1, 2, or 3


---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
----------------------------------------


Enter your choice: 1
Enter the student's first name: Yuying
Enter the student's last name: Testing
Please enter the name of the course: Python 1-1
You have registered Yuying Testing for Python 1-1. Press 3 to save to file.


---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
----------------------------------------
```

Figure 4. Test result in Pycharm

```
---- Course Registration Program ----
  Select from the following menu:
    1. Register a Student for a Course.
    2. Show current data.
    3. Save data to a file.
    4. Exit the program.
----------------------------------------


Enter your choice: 4
Program Ended

Process finished with exit code 0
```

Figure 4. Test result in Pycharm

Figure 5. Results in Json file

Summary

In summary, I improved the original script by modularizing the code into clearly defined functions, encapsulating related functionalities into two distinct classes following the separation of concerns pattern, and documenting all key parts of the code with meaningful docstrings. These changes enhance the program's clarity, maintainability, and robustness. By replacing a procedural structure with a more structured and object-oriented approach, the revised script not only performs the same functions as the original but does so in a cleaner, more professional way that supports future growth and debugging.

Appendix 1

(base) xy@MacBook-Pro-683 workspace % cd "/Users/xy/Documents/workspace/Python Fundation101"
(base) xy@MacBook-Pro-683 Python Fundation101 % python3 Assignment06.py

---- Course Registration Program ----
  Select from the following menu:

1. Register a Student for a Course.
2. Show current data.
3. Save data to a file.
4. Exit the program.

Enter your choice: 1
Enter the student's first name: NewTesting
Enter the student's last name: NewTesting
Please enter the name of the course: Python 101
You have registered NewTesting NewTesting for Python 101. Press 3 to save to file.

---- Course Registration Program ----
Select from the following menu:
1. Register a Student for a Course.
2. Show current data.
3. Save data to a file.
4. Exit the program.

Enter your choice: 2

The current data is:
Student Bob Smith is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student Testing Testing is enrolled in Python101
Student Testing Testing is enrolled in Python 101
Student Yuying Testing is enrolled in Python 1-1
Student NewTesting NewTesting is enrolled in Python 101

---- Course Registration Program ----
Select from the following menu:
1. Register a Student for a Course.
2. Show current data.
3. Save data to a file.
4. Exit the program.

Enter your choice: 3
The following data was saved to file!
Student Bob Smith is enrolled in Python 100

Student Sue Jones is enrolled in Python 100
Student Testing Testing is enrolled in Python101
Student Testing Testing is enrolled in Python 101
Student Yuying Testing is enrolled in Python 1-1
Student NewTesting NewTesting is enrolled in Python 101

---- Course Registration Program ----
  Select from the following menu:
        1. Register a Student for a Course.
        2. Show current data.
        3. Save data to a file.
        4. Exit the program.

Enter your choice: 1
Enter the student's first name: ReallyNEW
Enter the student's last name: Test Error
Please do not enter numbers or space.

---Technical Error Message---
The last name should not contain numbers or space.
Inappropriate argument value (of correct type).
<class 'ValueError'>

---- Course Registration Program ----
  Select from the following menu:
        1. Register a Student for a Course.
        2. Show current data.
        3. Save data to a file.
        4. Exit the program.

Enter your choice: 1
Enter the student's first name: TestError1
Please do not enter numbers or space.

---Technical Error Message---
The first name should not contain numbers or space.
Inappropriate argument value (of correct type).
<class 'ValueError'>

---- Course Registration Program ----
  Select from the following menu:

1. Register a Student for a Course.
2. Show current data.
3. Save data to a file.
4. Exit the program.

Enter your choice: 1
Enter the student's first name: FinalTesting
Enter the student's last name: FinalTesting
Please enter the name of the course: Python 1-1
You have registered FinalTesting FinalTesting for Python 1-1. Press 3 to save to file.

---- Course Registration Program ----
Select from the following menu:
1. Register a Student for a Course.
2. Show current data.
3. Save data to a file.
4. Exit the program.

Enter your choice: 2

The current data is:
Student Bob Smith is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student Testing Testing is enrolled in Python101
Student Testing Testing is enrolled in Python 101
Student Yuying Testing is enrolled in Python 1-1
Student NewTesting NewTesting is enrolled in Python 101
Student FinalTesting FinalTesting is enrolled in Python 1-1

---- Course Registration Program ----
Select from the following menu:
1. Register a Student for a Course.
2. Show current data.
3. Save data to a file.
4. Exit the program.

Enter your choice: 3
The following data was saved to file!

Student Bob Smith is enrolled in Python 100
Student Sue Jones is enrolled in Python 100
Student Testing Testing is enrolled in Python101
Student Testing Testing is enrolled in Python 101
Student Yuying Testing is enrolled in Python 1-1
Student NewTesting NewTesting is enrolled in Python 101
Student FinalTesting FinalTesting is enrolled in Python 1-1

---- Course Registration Program ----
  Select from the following menu:
        1. Register a Student for a Course.
        2. Show current data.
        3. Save data to a file.
        4. Exit the program.

Enter your choice: 4
Program Ended
(base) xy@MacBook-Pro-683 Python Fundation101 %