<Kseniia Temnikova>

<09/04/2024>

<Programming Basics>

<Assignment_6>

<GitHub URL>

# Basic Python program

## Introduction

In this assignment, I developed a Python program to manage student enrollments for a course. The program allows users to input a student's first name, last name, and course name, then organizes this data using dictionaries and lists. Users can display the current registration data and save it to a CSV file for future reference.

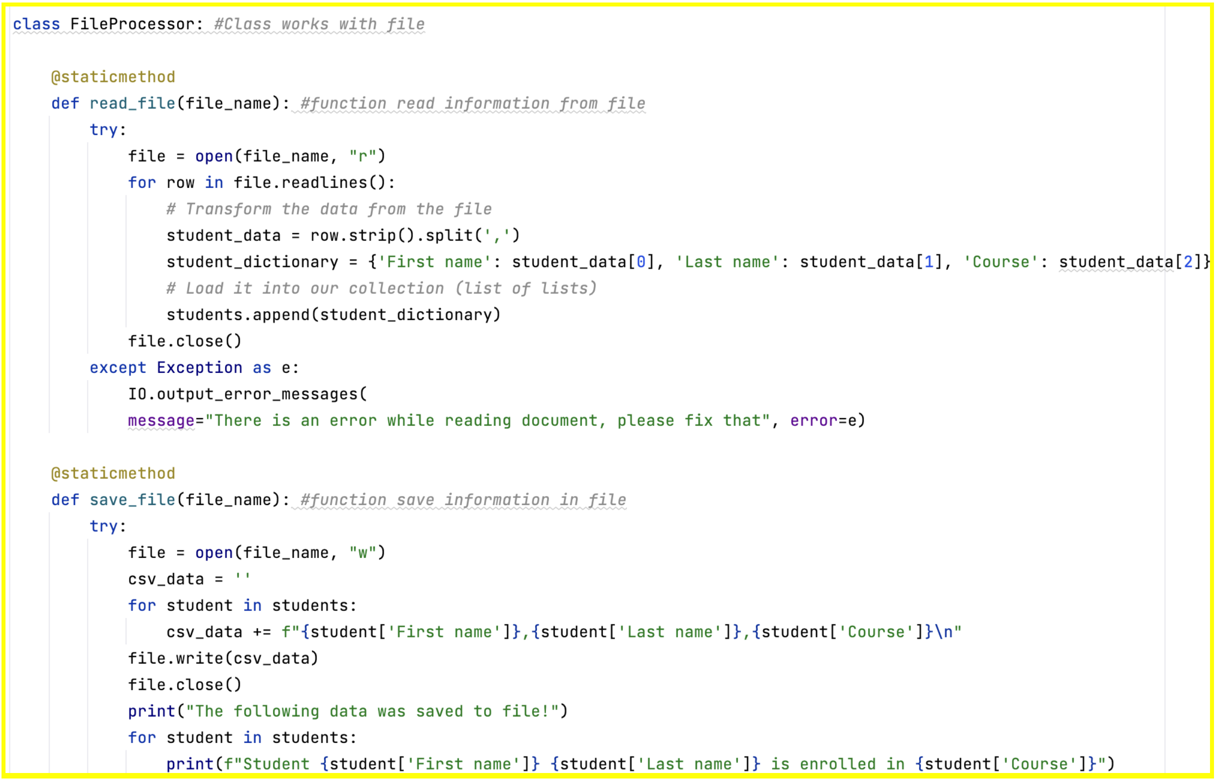
The program employs two classes, FileProcessor and IO, to handle file operations and user input/output, respectively. By using the @staticmethod decorator, I ensured that the methods within these classes can be called without needing to instantiate the class. This approach simplifies the code, as no object creation is required for processing tasks. Each static method is well-defined, serving specific purposes like reading from or writing to a file, handling user input, or displaying messages. This design maintains a clear separation of concerns, making the program more modular and easier to maintain.

Through this project, I gained valuable experience in structuring Python programs efficiently, utilizing classes and static methods to maintain clean, reusable, and functional code.

## *Classes*

In my program, I used two main classes: `FileProcessor` and `IO`. Each class handles different parts of the program to keep everything organized.

- **FileProcessor**:
  This class deals with tasks related to files. It reads student data from a CSV file and writes the data back to the file. By using this class, the program keeps file operations separate from other parts of the code, which helps make the program easier to understand and maintain. (Figure 1)



```python
class FileProcessor: #Class works with file

    @staticmethod
    def read_file(file_name): #function read information from file
        try:
            file = open(file_name, "r")
            for row in file.readlines():
                # Transform the data from the file
                student_data = row.strip().split(',')
                student_dictionary = {'First name': student_data[0], 'Last name': student_data[1], 'Course': student_data[2]}
                # Load it into our collection (list of lists)
                students.append(student_dictionary)
            file.close()
        except Exception as e:
            IO.output_error_messages(
            message="There is an error while reading document, please fix that", error=e)

    @staticmethod
    def save_file(file_name): #function save information in file
        try:
            file = open(file_name, "w")
            csv_data = ''
            for student in students:
                csv_data += f"{student['First name']},{student['Last name']},{student['Course']}\n"
            file.write(csv_data)
            file.close()
            print("The following data was saved to file!")
            for student in students:
                print(f"Student {student['First name']} {student['Last name']} is enrolled in {student['Course']}")
```

Figure 1: Class FileProcessor

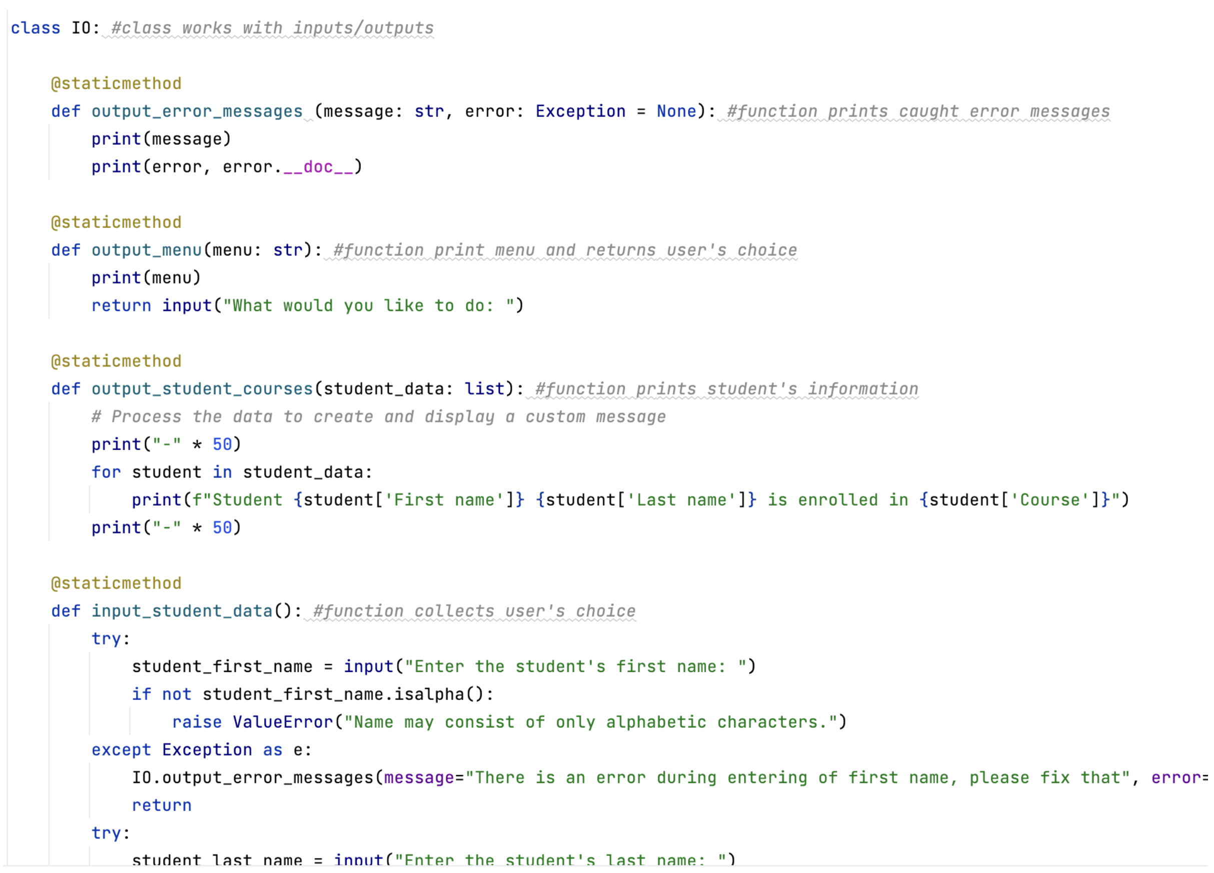
- **IO**:
  The IO class handles everything related to user interaction. It shows the menu to the user, takes user input, and displays messages about what's happening in the program. This class keeps all the code related to user communication in one place. (Figure 2)

Both classes use @staticmethod for their methods. This means you can use these methods without creating an instance of the class. This makes the code simpler and helps keep the program organized.

## Functions

In my program, functions are used to perform different tasks and keep the code organized. Here's a summary of what each function does:

- **`output_error_messages(message: str, error: Exception = None)`**:
  This function shows error messages when something goes wrong. It helps the user understand what the problem is if there is an issue.
- **`output_menu(menu: str)`**:
  This function displays the main menu of the program and lets the user choose an option. It makes it easy for the user to see what they can do next.
- **`input_student_data()`**:
  This function asks the user for information about a student, like their first name, last name, and course. It checks if the names are valid (only letters) before adding them to the list of students.(Figure 3)



```python
class IO: #class works with inputs/outputs

    @staticmethod
    def output_error_messages (message: str, error: Exception = None): #function prints caught error messages
        print(message)
        print(error, error.__doc__)

    @staticmethod
    def output_menu(menu: str): #function print menu and returns user's choice
        print(menu)
        return input("What would you like to do: ")

    @staticmethod
    def output_student_courses(student_data: list): #function prints student's information
        # Process the data to create and display a custom message
        print("-" * 50)
        for student in student_data:
            print(f"Student {student['First name']} {student['Last name']} is enrolled in {student['Course']}")
        print("-" * 50)

    @staticmethod
    def input_student_data(): #function collects user's choice
        try:
            student_first_name = input("Enter the student's first name: ")
            if not student_first_name.isalpha():
                raise ValueError("Name may consist of only alphabetic characters.")
        except Exception as e:
            IO.output_error_messages(message="There is an error during entering of first name, please fix that", error=
            return
        try:
            student_last_name = input("Enter the student's last name: ")
```

Figure 2: Class IO

- **`output_student_courses(student_data: list)`**:
  This function shows the list of students and the courses they are registered for. It formats and prints the information so it is easy to read.
- **`read_data_from_file(file_name: str, student_data: list)`**:
  This function reads student data from a file and adds it to the list of students. It makes sure that data from the file is loaded correctly into the program.
- **`write_data_to_file(file_name: str, student_data: list)`**:
  This function saves the student data to a file. It converts the data into a CSV format and writes it to the file, so the information is saved for later use.

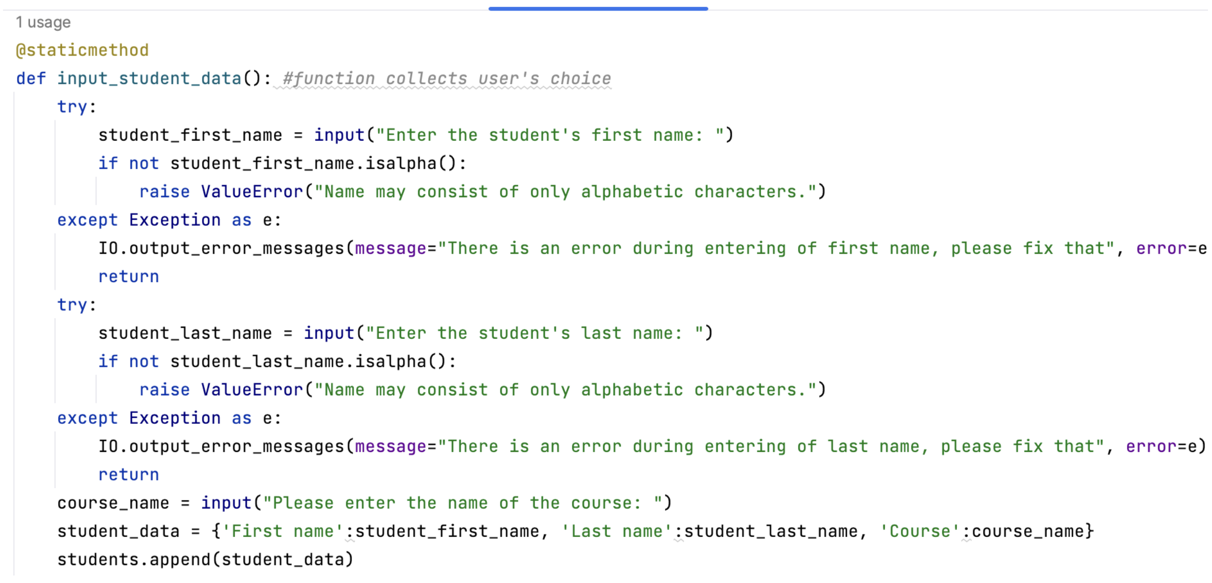
These functions help organize the program by separating different tasks into small, manageable pieces. This makes the code easier to read and work with.

## Test

For testing the program, I focused on ensuring it handles user inputs and performs operations correctly. I tested the program's ability to accept and store a student's first name, last name, and course name, verifying that this data is saved properly in a list. I also checked that the program displays the collected information back to the user in a clear, comma-separated format.

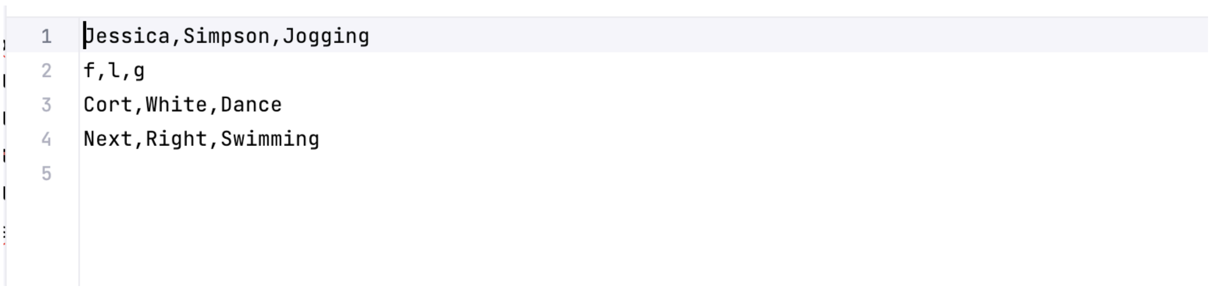
I confirmed that the program can save the user's input to a CSV file, ensuring that the data is written correctly and can be viewed in a text editor. I tested the program's capability to handle multiple registrations, verifying that each set of data is stored in a list of dictionaries and displayed accurately when requested.

Furthermore, I checked that the program correctly saves all registrations to the CSV file, preserving the data for future use. (Figure 4) Finally, I ran tests in both PyCharm and the terminal to ensure the program works consistently and as expected in different environments.(Figure 5)



```
1 usage
@staticmethod
def input_student_data(): #function collects user's choice
    try:
        student_first_name = input("Enter the student's first name: ")
        if not student_first_name.isalpha():
            raise ValueError("Name may consist of only alphabetic characters.")
    except Exception as e:
        IO.output_error_messages(message="There is an error during entering of first name, please fix that", error=e
        return
    try:
        student_last_name = input("Enter the student's last name: ")
        if not student_last_name.isalpha():
            raise ValueError("Name may consist of only alphabetic characters.")
    except Exception as e:
        IO.output_error_messages(message="There is an error during entering of last name, please fix that", error=e)
        return
    course_name = input("Please enter the name of the course: ")
    student_data = {'First name':student_first_name, 'Last name':student_last_name, 'Course':course_name}
    students.append(student_data)
```

***Figure 3: input_student_data***



```
Jessica,Simpson,Jogging
f,l,g
Cort,White,Dance
Next,Right,Swimming

```

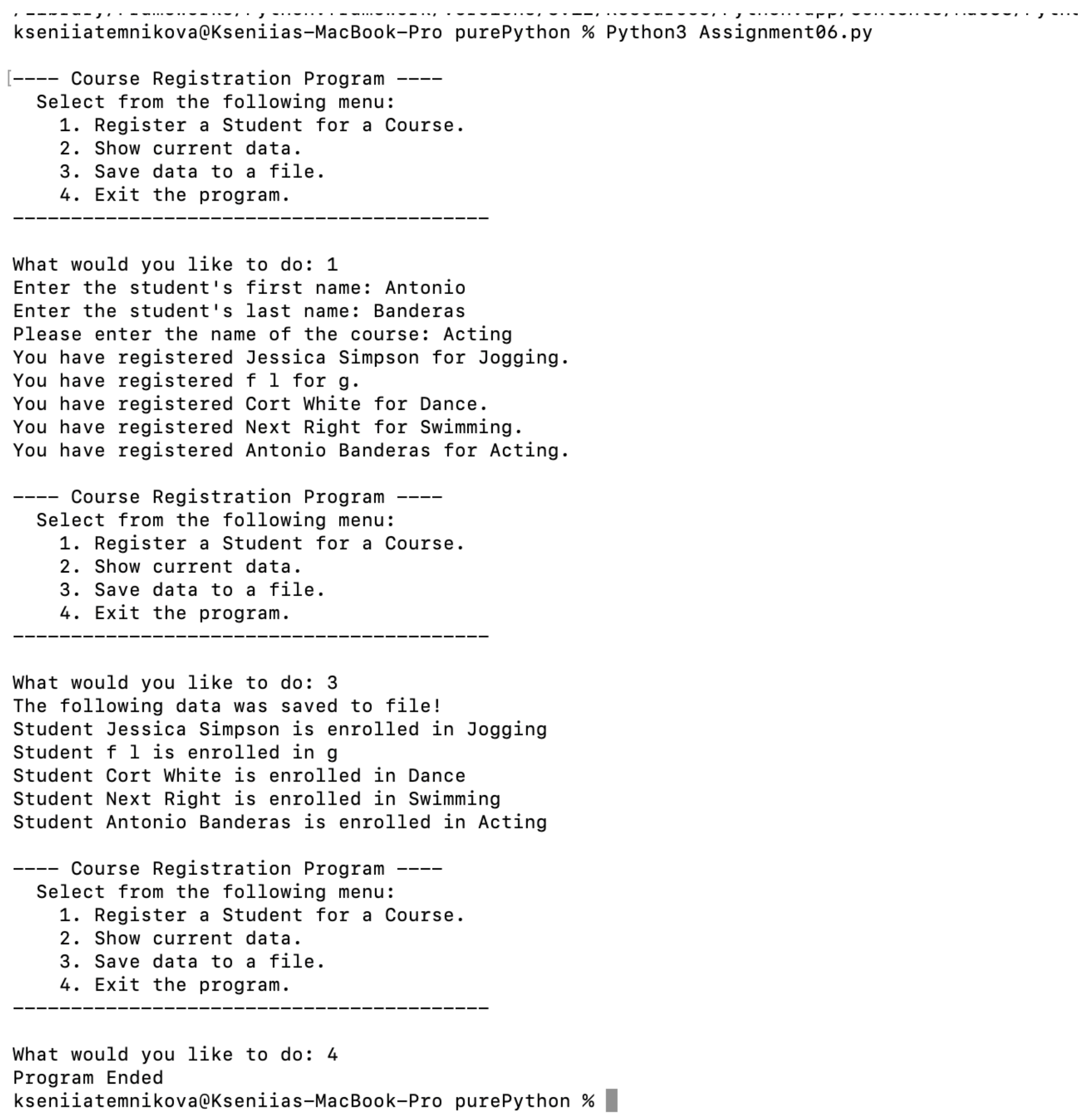
***Figure 4: csv file***

**Summary**

In this assignment, I developed a Python program that manages student enrollments for a course. The program allows users to input student details, view current registrations, and save the data to a file.

To achieve this, I used constants to store fixed values, such as the menu text and file name, ensuring these values remain consistent throughout the program. I also organized the program into two classes: `FileProcessor`, which handles file operations, and `IO`, which manages user interactions. Both classes use `@staticmethod` to simplify the code and avoid unnecessary object creation.

By following these practices, the program maintains a clear and modular structure, making it easy to understand, use, and maintain.



```
kseniiatemnikova@Kseniias-MacBook-Pro purePython % 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.
-----------------------------------------

What would you like to do: 1
Enter the student's first name: Antonio
Enter the student's last name: Banderas
Please enter the name of the course: Acting
You have registered Jessica Simpson for Jogging.
You have registered f l for g.
You have registered Cort White for Dance.
You have registered Next Right for Swimming.
You have registered Antonio Banderas for Acting.

---- 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.
-----------------------------------------

What would you like to do: 3
The following data was saved to file!
Student Jessica Simpson is enrolled in Jogging
Student f l is enrolled in g
Student Cort White is enrolled in Dance
Student Next Right is enrolled in Swimming
Student Antonio Banderas is enrolled in Acting

---- 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.
-----------------------------------------

What would you like to do: 4
Program Ended
kseniiatemnikova@Kseniias-MacBook-Pro purePython %
```

***Figure 5: Test in Terminal***