Saima Ahmed

November 26th, 2023

Foundation of Programming: Python

Assignment 07

**Creating a Python Script**

**Introduction:**

In this module, we delve into the core concepts of programming and version control. We begin by clarifying the distinctions between statements, functions, and classes. Next, we explore the world of classes in more detail, dissecting data classes, presentation classes, and processing classes, and understanding their unique purposes. We also learn about constructors, attributes, and properties. We also study the concept of class inheritance and the power it brings to code reuse and extensibility, including the concept of overridden methods.

**Function:**

In programming, a function is a reusable block of code that performs a specific task or a set of tasks. Functions are a fundamental concept in most programming languages and serve several important purposes.

**Parameters:**

Parameters allow you to make your functions more flexible and reusable because you can customize their behavior by providing different values when you call them. The parameters act as placeholders for the values that you want to pass to the function when you call it. The values you pass into parameters are called arguments.

**Classes and Functions:**

Another way to organize your code is by using classes. Classes are a way of grouping functions, variables, and constants by the name of the class. Using classes to group functions is a fundamental concept in programming. Grouping functions within classes creates a modular structure, making it easier to manage and maintain code. Classes provide a natural way to organize code by grouping functions and data needed by those functions, making code more structured and readable, especially in larger projects.

**The Separation of Concerns (SoC):**

The Separation of Concerns (SoC) is a fundamental software design principle that aims to enhance the maintainability, scalability, and readability of code by breaking it down into distinct, self-contained components, each responsible for a specific aspect of the application's functionality. This pattern encourages modularity, reduces code complexity, and makes it easier to manage and extend software systems.

**Attributes:**

In programming, an attribute is a piece of data, or a characteristic associated with an object. Attributes describe or store information about the object they belong to. Depending on the programming context, attributes are also referred by developers as fields.

**Constructor:**

A constructor is a special method (function) that is automatically called when an object of a class is created. Its primary purpose is to set an object's attributes values when it is created. Constructors are sometime called an initializer because it sets up any necessary initial data (known as state) for the object.

**Creating the Program:**

The objective of this week was to create a Python program that demonstrates using constants, variables, and print statements to display a message about a student's registration for a Python course with **the use of functions, classes, and using the separation of concerns pattern.**

* I started the program by first including the **Script Header** that shows Title and description about the program along with my name and current date as shown in Fig 1A screen shot of a computer

  Description automatically generated

**Fig 1: Creating Script Header**

* Using Separation of concerns (SoC) pattern, I organized my program into three main classes called ‘Person’, ‘FileProcessor’ and ‘IO’.
* The class ‘Person’ has a sub-class called ‘Student’. Both these classes belong to Data classes whose main objective is to organize data about the students. While processing and presentation classes just have methods, Data classes typically have Attributes, Constructor, Properties in addition to Methods.
* The class File Processor has mainly all the functions that handles the main operation of the code. In this code, ‘read\_data\_from\_file’ and ‘write\_data\_to\_file’ were included in the class FileProcessor.
* Class IO has mainly all the functions that handles taking inputs from the user and displaying outputs to the user. In this code I used the functions output\_error\_message, out\_menu, input\_menu\_choice, output\_student\_courses and input\_student\_data.
* The class ‘Student’ has an \_\_init\_\_ function that handles the attributes first\_name and last\_name of the student. I have also used getter and setter properties on these attributes and over ride the \_\_str()\_\_ method to return Person data as shown in the Fig 2

A screenshot of a computer program

Description automatically generated

**Fig 2: Main structure of class Person**

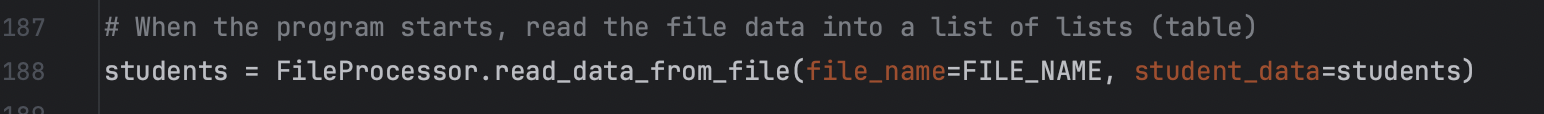
* Person class has a sub class called ‘Student’ which manages an extra attribute called ‘course\_name”. I have again used getter and setter properties on course\_name and override the \_\_str()\_\_ method to return student data as shown in Fig 3

A computer screen shot of a program

Description automatically generated

**Fig 3: Main structure of student class**

* The program starts by calling the function read\_data\_from\_file to read data form the json file Enrollments.json as shown in Fig3



**A computer screen shot of a program

Description automatically generated**

**Fig 3: Opening the json file in read mode**

* Then I used the While loop because I wanted to keep iterating the program for taking inputs, showing the output and saving data into a json file. I again used try-except exception technique to make sure user’s mistakes while entering the inputs could be caught in time as shown in Fig 4. The functions called were shown in the Fig4.

A screenshot of a computer program

Description automatically generated

**Fig 4: Processing data and exception handling.**

* For menu choice ‘1’ which was to register student for the course I used the function input\_student\_data as shown in Fig 5

**A screenshot of a computer program

Description automatically generated**

**Fig 5: Function input\_student\_data**

* For menu choice ‘2’ which was to display the current data, I used the function output\_student\_courses as shown in Fig 6

A computer screen shot of a program

Description automatically generated

**Fig 6: Function output\_student\_courses**

* For menu choice ‘ 3’ which is to save the data into the file I used the function write\_data\_to\_file**A computer screen shot of a program

  Description automatically generated**

**Fig 7 Function write\_data\_to\_file**

* For menu choice ‘4’ I used the ‘break’ command to break the while loop and end the program as shown in Fig 8

A screenshot of a computer program

Description automatically generated

**Fig 8: Ending the program**

**Running the program:**

First I made a json file called Enrollments.json and saved some data in it. Next I ran the program and took some more inputs from the user using input command. When I ran the program on PyCharm console, below is the output shown in Fig 9:

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**Fig 9: Output of the program on PyCharm console**

Following is my Enrollments.json file which has all the stored data in Fig 10

**A screenshot of a computer

Description automatically generatedFig 10: Json file**

When I ran the program on IDLE and Terminal, below are the outputs shown in Fig 11 and Fig 12 respectively.

A screenshot of a computer

Description automatically generated

**Fig11: Output of the program on IDLE**

**A screenshot of a computer program

Description automatically generated**

**Fig 12: Output of the program on Terminal**

**Summary:**

In this module, we looked at how to work with functions and classes. With the knowledge of While loop and conditional statements, we are able to make user interactive programs that can be run as many times as we want. We also got introduced to exception handling techniques using try-except.