KaeLi Deng
11/17/2025
Foundations of Programming: Python
Assignment 06
https://github.com/Kaedaimo/IntroToProg-Python-Mod06

Functions

Introduction

This module focuses on enhancing my understanding of fundamental programming concepts in Python, including functions, classes, and the separation of concerns (SoC). These concepts are essential for writing clean, organized, and maintainable code. functions provide a way to encapsulate repetitive code into reusable blocks, classes allow the organization of related functions and attributes into structured templates, and SoC helps divide a program into distinct layers to improve clarity, scalability, and reduce errors. Through this module, I learned not only how to implement these concepts, but also how they work together to create more effective and robust programs.

Function

In this module, I finally learned about functions. A function is like a reusable building block that you can use wherever it's needed. For example, previously, when writing code, I had to repeatedly write the same code for input and repeatedly provide the output. After learning this week's concepts, I can now define all this repetitive code into a single function. Besides defining functions, I also learned about parameters, which act as placeholders. They tell the function that it will receive some data in the future and that it needs to use this data to perform its work. The actual data passed to the function is called an argument. After running a function, I also need to bring the results back, which are done using return. Its purpose is to output the result of my computation. Here is a simple example from Figure 1: I defined a function called math_operation using a and b as parameters. Inside the function, I wrote formulas for reach variables. Since division cannot divide by zero, I specifically used an if-else statement to prevent errors. After the formulas, I had the function return the results of each variable. Finally, I used the result to collect the outputs and print to display them.

```
def math_operation(a,b):  1 usage
    addition = a + b
    substract = a - b
    multilication = a * b
    if b != 0:
        division = a / b
    else:
        division = "undefined"
    return addition, substract, multilication, division

result = math_operation( a: 6,  b: 2)
print(result)
```

```
Testtesttest ×

C:\Users\kaeli\Python\.venv\Scripts\python.exe C:\Users\kaeli\Python\PythonLabs\Testtesttest.py
(8, 4, 12, 3.0)

Process finished with exit code 0
```

Figure 1. Example of function.

Classes

This module mainly focused on learning the differences between classes and functions. To me, the biggest difference is like folders on a computer. Organizing each function into different classes is like sorting files into separate folders. In Python, a class is used to define a template for objects. This week, the focus was on static methods, which do not depend on an object's attributes or state and can be directly used for functionality related to the class logic. They can be called directly through the class name to perform operations. In figure 2, a class was defined as Math with two static methos, add and sub, which perform addition and subtraction. Because they are marked with @staticmethod, you told Python that these are static methods. As results, Math.add(1,2) returns 3, and Math.sub(2,3) returns -1. This stage helped me understand the powerful role of classes in Python; they not only improve code organization and reusability but also lay a solid foundation for learning object-oriented programming and developing more complex projects in the future.

```
class Math:  2 usages
    @staticmethod
    def add(a,b):
        return a + b
    @staticmethod  1 usage
    def sub(a,b):
        return a - b
print(Math.add( a: 1,  b: 2)) #Output 3
print(Math.sub( a: 2,  b: 3)) #Output -1
```

Figure 2. Example of class using class Math to calculate sum and difference between parameter a and b.

Separation of Concerns

From classes, the topic extended to the Separation of Concerns (SoC). This makes the code more maintainable, clear, scalable, and less prone to bugs. It divides the code into three main categories: presentation, logic, and data concerns, which correspond to the presentation layer, processing layer, and data layer, respectively. This lesson emphasized the order of code execution, and with SoC, programmers can know how to organize and integrate their code every time they write a program. The usual sequence is: the Data Layer, which stores and retrieves data; the Processing Layer, which applies rules, performs calculations, makes decisions, and handles errors; and finally, the Presentation Layer, which handles user interactions, collects input, and displays outputs.

Summary

Through this module, I gained a deeper understanding of how functions, classes, and the Separation of Concerns work together to create well-structured and maintainable code. Functions allow for reusable and efficient code blocks, classes provide templates to organize related functionality, and SoC ensures that code is divided into logical layers for clarity and scalability. Learning these concepts has strengthened my programming skills, improved my ability to write organized and reusable code, and prepared me for more advanced topics in object-oriented programming and complex project development.