Hoang Nguyen

September 08, 2024

Foundations of Programming: Python

https://github.com/hoangkenuw/IntroToProg-Python/tree/IntroToProg-Python-Mod07

Python – Statements, Functions, and Classes

Introduction:

This week, I learned data class components and processing classes, I also learned about constructors, attributes, and properties. Additionally, I explored class inheritance and how it allows you to reuse code more efficiently, including how to override methods when needed.

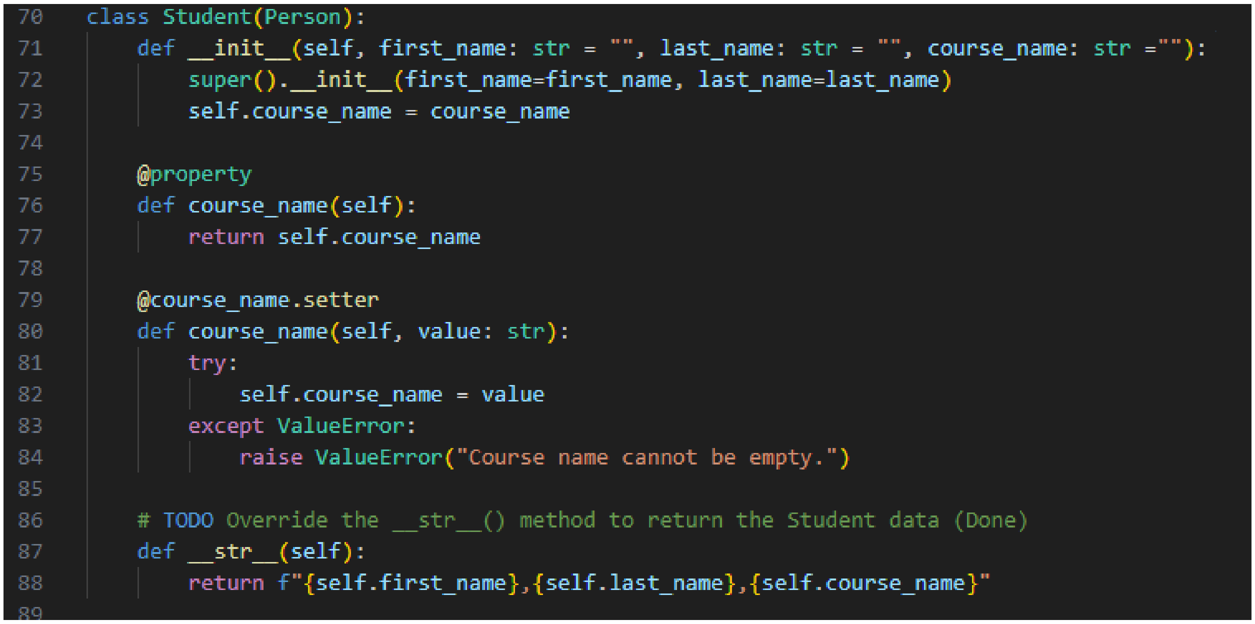
Statements:

In Python, “Statements” are organized into functions when a programmer wants to use them many times in a program. In complex scripts, the programmer is usually created using these three components, and one program can have many statements, many functions, and many classes. It all depends on how complex the program is. Using functions to organize statements and classes to organize functions makes complex programs easier to create and manage.

Classes, Functions and Constructors:

Last week, I learned that “Classes” are a way of grouping functions, variables, and constants by the name of the class. Grouping functions within classes creates a modular structure, making it easier to manage and maintain code. This week I learned a Constructor is a special method in a class, called __init__(), which automatically runs when an object is created. It’s used to initialize the object's attributes. Functions inside classes, often referred to as methods, define specific behaviors that an object created from the class can perform. By combining these, Python allows for organized and reusable code. In my assignment, I created a Student class that builds on a Person class by using inheritance. This means the Student class can take advantage of the existing features of Person—like the first_name and last_name—without needing to rewrite that part of the code. The constructor (__init__()) sets up the student's first name, last name, and course when a new student is created.

I added a property for course_name to ensure that the course is properly managed, allowing it to be set and retrieved easily, and I added a check to prevent empty course names.



```
class Student(Person):
    def __init__(self, first_name: str = "", last_name: str = "", course_name: str =""):
        super().__init__(first_name=first_name, last_name=last_name)
        self.course_name = course_name

    @property
    def course_name(self):
        return self.course_name

    @course_name.setter
    def course_name(self, value: str):
        try:
            self.course_name = value
        except ValueError:
            raise ValueError("Course name cannot be empty.")

    # TODO Override the __str__() method to return the Student data (Done)
    def __str__(self):
        return f"{self.first_name},{self.last_name},{self.course_name}"
```

Figure 1. Using Student class and constructor method __init__()

Summary:

In this week, I learned statements are individual lines of code that do something, like assigning values or printing output, while functions are reusable blocks of code that perform specific tasks. Classes are blueprints for creating objects that group data and behavior together. There are different types of classes, like data classes, which store information, presentation classes, which handle how data is displayed, and processing classes, which manage logic or actions. A constructor is a method that runs when an object is created, initializing its attributes, which hold data about the object. Inheritance allows a class to inherit features from another, making code more reusable and easier to manage.