Jae Barnett

03/12/2025

Foundations of Programming: Python

Assignment 07

https://github.com/jaeuw

Classes and Objects

Introduction

In this document I will be creating a program that builds upon the code written in the last assignment. The difference is this one will delve into core concepts of programming and version control. In the below you will see how this assignment elaborates on programming classes and elements in object-oriented programming.

Declarations in Python

Classes

In our previous assignment we demonstrated using functions within a class with the @staticmethod function decorator. This week we will be creating an object instance from a class to access data unique to each object with classes named “Person” and “student” These classes include properties for the student first name, last name and course name. Each object created from these types of classes has their own memory space and maintain their own set of data even if they’re part of the same class. We know this in object-oriented programming as data encapsulation.

```
class Person:  1 usage
    """
    A class that represents the first and last name of each student
    ChangeLog: (Who, When, What)
    JBarnett,03/13/2025,Created Class
    """
    def __init__(self, first_name: str = '', last_name: str = ''):
        self.first_name = first_name
        self.last_name = last_name

# TODO Add first_name and last_name properties to the constructor (Done)
# TODO Create a getter and setter for the first_name property (Done)

    @property  6 usages (2 dynamic)
    def first_name(self):
        return self.__first_name.title()
```

Figure 1: Person class

Constructors

Constructors are functions used to initialize objects of a class. They are automatically called with an oject is created and set up the initial data known as state for an object. The most common constructor in Python

is __init__ which stands for initializing the instance of the class. The constructor assigns initial values to an objects variables, they also set attribute values by using special functions to set each attribute value. Also, if a constructor isn't defined, then Python create an invisible default constructor without parameters. In this assignment we used the initializing constructor – the init method takes self as the first argument which refers to the instance of the class being created followed by any other arguments you want to pass for initialization as shown below.

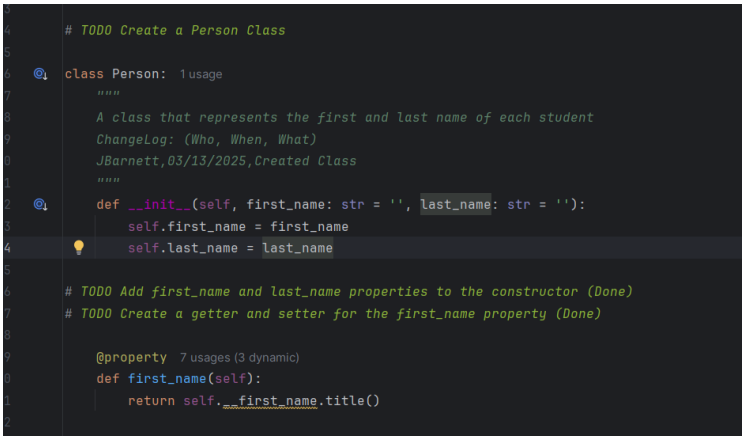


Figure 2: Initializing constructor

Attributes

In Python attributes are characteristics or properties associated with an object. They describe or store information about the object that they belong to and can be variables or methods. There are two main types of attributes:

- **Instance:** These are specific to each instance of a class and are defined within the __init__ method or directly assigned to an instance.
- **Class:** These are shared among all instances of a class and are defined outside any method within the class definition.

In the screenshot below the first name and last name are instance attributes for our assignment gathering student data.

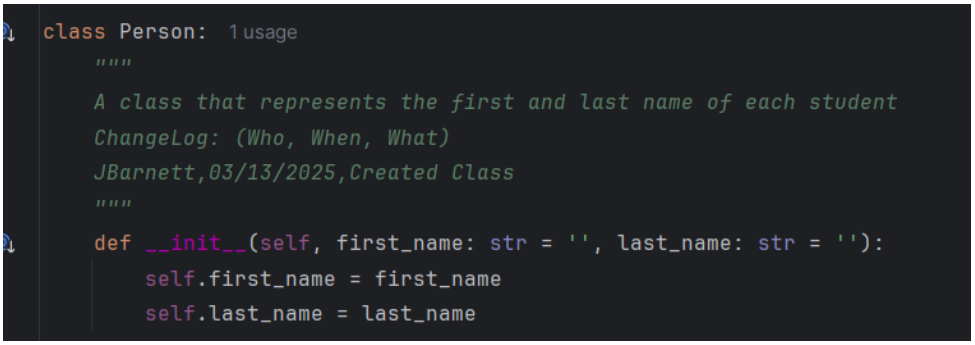
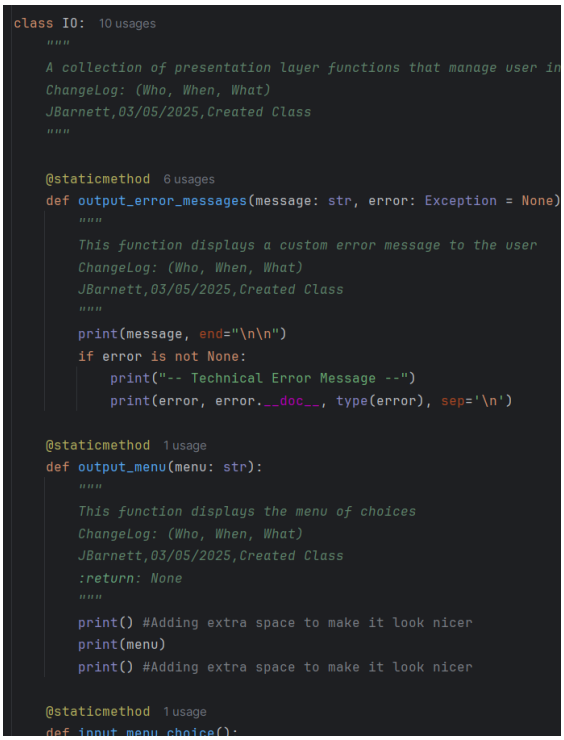


Figure 3: Instance Attributes

IO: This class can be found in the presentation layer. This class contains the functions that control input and output a user interacts with. For example, error messages, menu options, user input, etc.



```
class IO:  10 usages
    """
    A collection of presentation layer functions that manage user in
    ChangeLog: (Who, When, What)
    JBarnett,03/05/2025,Created Class
    """

    @staticmethod  6 usages
    def output_error_messages(message: str, error: Exception = None)
        """
        This function displays a custom error message to the user
        ChangeLog: (Who, When, What)
        JBarnett,03/05/2025,Created Class
        """
        print(message, end="\n\n")
        if error is not None:
            print("-- Technical Error Message --")
            print(error, error.__doc__, type(error), sep='\n')

    @staticmethod  1 usage
    def output_menu(menu: str):
        """
        This function displays the menu of choices
        ChangeLog: (Who, When, What)
        JBarnett,03/05/2025,Created Class
        :return: None
        """
        print() #Adding extra space to make it look nicer
        print(menu)
        print() #Adding extra space to make it look nicer

    @staticmethod  1 usage
    def input_menu_choice():
```

Figure 4: IO class

Summary

Object-oriented programming is essential because it contains elements like constructors, attributes, and properties which have proven to be quite helpful when creating a program that is easy to read and build. The person and student classes help to distinguish different sections of code and in doing so make it much easier to read and understand.