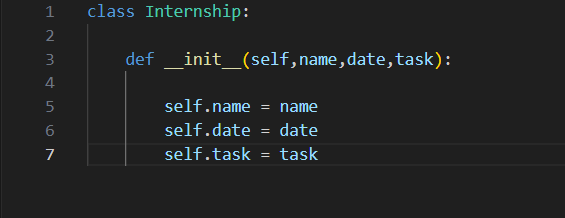
OOPS Concepts in Python Documentation

Object-Oriented Programming (OOP) is a programming paradigm that uses objects and classes to organize and structure code. Python is a versatile programming language that fully supports OOP principles. This documentation provides an overview of essential OOP concepts in Python.

1. Class & Objects:

In Python, a class is a blueprint for creating objects. It defines a set of attributes and methods that the objects created from the class will have.



Objects are the instances of the class. They encapsulate the data and behavior.

A black background with colorful text

Description automatically generated

1. Methods & Class Attributes:

Classes have variables which can be defined as Attributes

A black rectangular object with a black border

Description automatically generated

,and they also have functions which can be defined as Methods.

A computer screen with text

Description automatically generated

1. Class Methods & Static Methods:

We can use two kinds of methods other than regular methods(functions). We use class decorators to declare the class method which uses class as an argument so that we can change the class variables dynamically from a method.

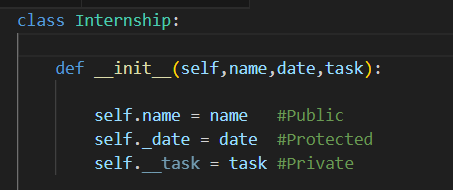
A computer screen with text

Description automatically generated

1. Encapsulation:

Access Modifiers: Python uses access modifiers like public, protected, and private to control the visibility of attributes and methods.

Getter and Setter Methods: Encapsulation involves using getter and setter methods to access and modify private attributes, ensuring controlled access.



A computer screen shot of a program

Description automatically generated

1. Inheritance:

Creating a Subclass: Inheritance allows a new class (subclass) to inherit attributes and methods from an existing class (parent class).

Overriding Methods: Subclasses can provide a specific implementation for a method defined in the parent class.

Types:

Single level Inheritance

Multiple Inheritance

Multilevel Inheritance

Hierarchical Inheritance

Hybrid Inheritance

A screen shot of a computer program

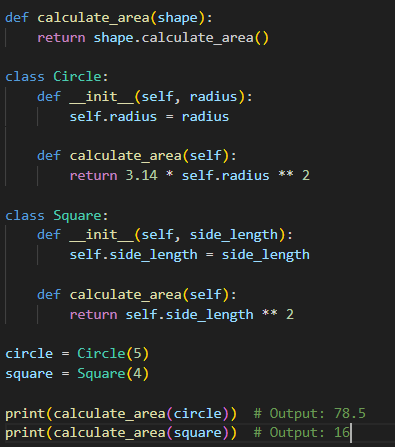
Description automatically generated

1. Polymorphism:

Polymorphism allows objects to be treated as instances of their parent class, promoting code flexibility. It can be achieved through method overloading and method overriding.

Method Overloading: Polymorphism allows the definition of multiple methods with the same name but different parameters.

Method Overriding: Different classes can have methods with the same name, providing specific implementations.



1. Abstraction:

Abstraction involves hiding the complex implementation details of an object and exposing only the necessary features or functionalities. It is Achieved through abstract classes and abstract methods.

A screen shot of a computer program

Description automatically generated