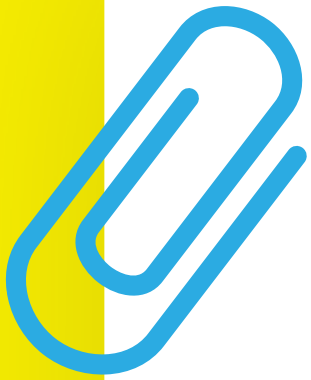
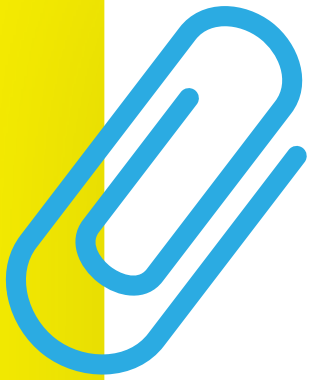


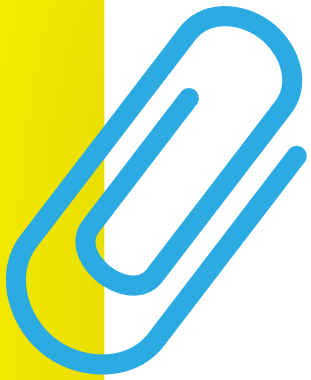
Principles of OOP

- Class
- Objects
- Polymorphism
- Encapsulation
- Inheritance
- Data Abstraction



Class

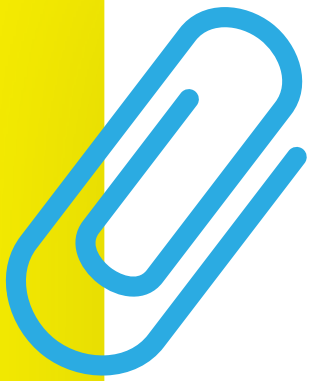
In Python, a class is a blueprint for creating objects. A class defines the attributes and methods of its objects. A class defines what an object of that class will look like and how it will behave.



Methods

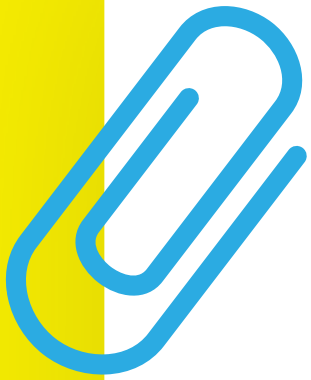
Methods are functions that are defined inside a class and are called on objects of that class.

They have access to the object's attributes and can modify them.



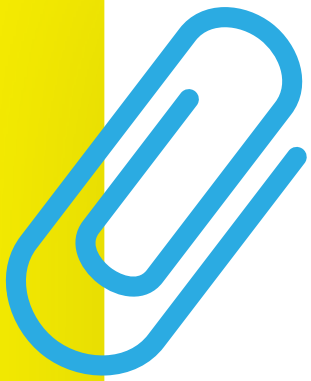
Polymorphism

Polymorphism allows objects of different types to be treated as objects of a common superclass. Subclass have access to all methods and attributes.



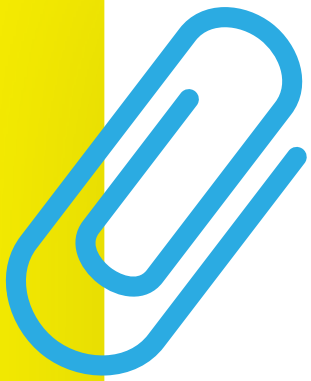
Encapsulation

Encapsulation helps protect the data from unauthorized access and ensures proper mainpulation.



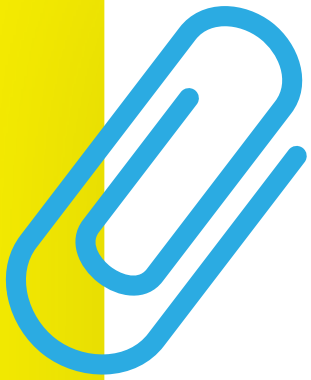
Encapsulation

- **Public: Public members (attributes and methods) are accessible from anywhere, both inside and outside the class.**
- **Private: They can only be accessed within the class itself and not from outside the class.**
- **Protected: They can be accessed within the class and its subclasses (derived classes), but not from outside the class hierarchy.**



Conclusion

Finally, Object-Oriented Programming (OOP) is a strong paradigm that enables developers to create and arrange code in a more ordered and modular fashion. Python, as an object-oriented language, offers extensive support for OOP principles.



More

See code examples and more on

https://medium.com/@sarperismetmakas/mastering-oop-in-python-e56c270ceee