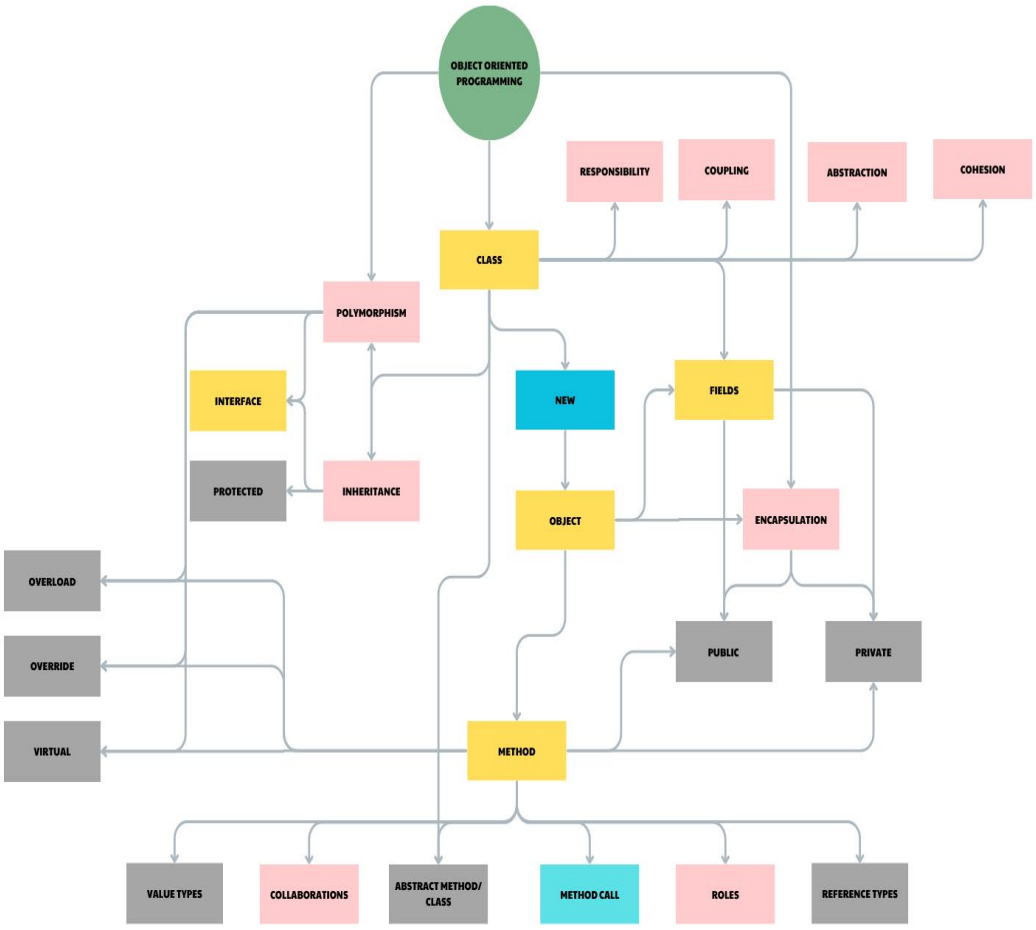

KEY CONCEPTS:

- **Abstraction:** This is the process of hiding the irrelevant details and only shows the essential features of the object. We usually use abstract classes or interfaces to achieve abstraction. For example, in the Drawing class, the internal workings of the methods, such as Load, Save, Draw…, are hidden from the users.
- **Encapsulation:** Encapsulation is when we make the fields in the class private, and we give the users access to those fields only via the public methods. We usually use encapsulation in class to enhance security and avoid inadvertent change. For example, the field “private readonly List<Shape> _shapes” in the Drawing program can only be accessed through the AddShape() and RemoveShape() methods. The encapsulation technique is associated with the terminology “private” and “public”.
- **Inheritance:** This is a mechanism where a new class is derived from an existing class. The derived class inherits all the properties and methods of the base class, and the derived class can also add its own features. For instance, the MyLine, MyRectangle, MyCircle are derived from the Shape class, so those derived classes have all the features of the Shape class.

- **Polymorphism:** This is a mechanism that we will create an interface, abstract class or using virtual for the methods. Then, any subclasses derived from the base class can use the implementations of the methods from the superclass by using the base keyword, or they can have their own implementations for their methods by using override keyword. For example, in the MyLine, MyCircle, MyRectangle, we use override keyword to implements the workings of the methods defined in the abstract Shape class.