**What is OOP?**

*OOP* is a design philosophy. It stands for Object Oriented Programming. **O**bject-**O**riented **P**rogramming (*OOP*) uses a different set of programming languages than old procedural programming languages (*C, Pascal*, etc.). Everything in *OOP* is grouped as self sustainable "*objects*". Hence, you gain re-usability by means of four main object-oriented programming concepts.

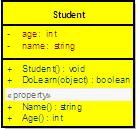
In order to clearly understand the object orientation, let’s take your “*hand*” as an example. The “*hand*” is a class. Your body has two objects of type hand, named left hand and right hand. Their main functions are controlled/ managed by a set of electrical signals sent through your shoulders (through an interface). So the shoulder is an interface which your body uses to interact with your hands. The hand is a well architected class. The hand is being re-used to create the left hand and the right hand by slightly changing the properties of it.

**What is an Object?**

An object can be considered a "*thing*" that can perform a set of **related** activities. The set of activities that the object performs defines the object's behavior. For example, the hand can grip something or a *Student* (*object*) can give the name or address.

In pure *OOP* terms an object is an instance of a class.

### What is a Class?



A class is simply a representation of a type of object. It is the blueprint/ plan/ template that describe the details of an object. A class is the blueprint from which the individual objects are created. Class is composed of three things: a name, attributes, and operations.

### What is Encapsulation (or Information Hiding)?

The encapsulation is the inclusion within a program object of all the resources need for the object to function - basically, the methods and the data. In OOP the encapsulation is mainly achieved by creating classes, the classes expose public methods and properties. The class is kind of a container or capsule or a cell, which encapsulate the set of methods, attribute and properties to provide its indented functionalities to other classes. In that sense, encapsulation also allows a class to change its internal implementation without hurting the overall functioning of the system. That idea of encapsulation is to hide how a class does it but to allow requesting what to do.

### What is Inheritance?

The ability of a new class to be created, from an existing class by extending it, is called inheritance.

There are various types of inheritance, depending on paradigm and specific language. A fundamental difference is whether one can inherit from only a single other object or class, which is known as *single inheritance,* or whether one can inherit from multiple other objects or classes, which is known as [*multiple inheritance*](http://en.wikipedia.org/wiki/Multiple_inheritance)*.*

### What is Polymorphism?

Polymorphism is a generic term that means 'many shapes'. More precisely Polymorphism means the ability to request that the same operations be performed by a wide range of different types of things.

At times, I used to think that understanding Object Oriented Programming concepts have made it difficult since they have grouped under four main concepts, while each concept is closely related with one another. Hence one has to be extremely careful to correctly understand each concept separately, while understanding the way each related with other concepts.

In OOP the polymorphisms is achieved by using many different techniques named method overloading, operator overloading, and method overriding,