**Object Oriented Analysis and Design**

**Benefits of Object Oriented**

* Ease of programming , maintainence , enhancements
* Map to real world entities

**Object Oriented Concepts**

**Major Elements of Object Oriented**

* **Abstraction**

Focus only on relevant details (data members and methods) and ignore/delete the non-essential details from the class design

* **Encapsulation**

Binding of data members and methods into a single unit called class

* **Data Hiding**

Use of appropriate access specifiers (private or protected) to avoid the visibility and accessibility of class members outside the class design

* **Inheritance**

Design for extendibility and in the process of extending, we also wish to reuse the members of the base class with appropriate access specifiers

Inheritance establishes an Is-A relationship between the derived class and the base class

* **Polymorphism**

Many forms of behaviour of a given object is called polymorphism.It is classified into 2 types

* Static Polymorphism – demonstrated using method overloading concept
* Dynamic Polymorphism – demonstrated using method overriding concept

**Minor Elements of Object Oriented**

* **Class**

It is conceptual representation of an object and no memory is allocated for the data members during the class design

* **Object**

It is a real-world entity having a well-defined identity, state and behaviour. Size of an object is the sum of the memory of the non-static data members in the class design. Static members are allocated memory during the class loading phase.

Non static members are accessible only via an object and hence they have an object scope. Static members are accessible using the class name itself and hence they have a class scope

Identity -> name of the class for which the object is created

State -> values what we give for the data members when an object is created

Behaviour -> the methods in the class are the behaviours of an object

**Method overloading rules**

* All the overloaded method should be defined within the same class
* The return type of the overloaded methods can be same or different
* The parameter signatures should be different either by the number of parameters, types of parameters or order of parameters
* The binding of the method call with the method definition should happen at compile time itself

**Method overriding rules**

* The methods should be across classes related via inheritance
* The return type and the parameter signature of the overridden and overriding methods should be exactly the same
* The binding of the method call with the method definition should have an runtime