List of OOP’s Concepts:

1.Abstraction

2.Encapsulation

3.Inheritance

4.Polymorphism

**What is Polymorphism:**

“Poly” means “many” and “morphism” means “forms”

Polymorphism means same object having different behaviours

There are two types of polymorphisms:

1. Compile time/Static polymorphism(Method Overloading-same method name with different parameters):

This type of polymorphism is achieved by method overloading.

This is also called as early binding.

Eg.

Void add(int x, int y)

{

}

Void add(float x, float y)

{

}

1. Run time/Dynamic polymorphism(Method Overriding-same method name with same parameters):

This type of polymorphism is achieved by method overriding.

This is also called as late binding.

Eg.

Void add(int x, int y)

{

Return x+y;

}

Void add(int x, int y)

{

Return x\*y;

}

**Abstraction**

Abstraction allows us to represent complex real world in simplest manner. It is process of identifying the relevant qualities and behaviors an object should possess, in other word represent the necessary feature without representing the back ground details. Abstraction is a process of hiding work style of an object and showing only those information which are required to understand the object. Abstraction means putting all the variables and methods in a class which are necessary.

**Encapsulation**

It is a process of hiding all the internal details of an object from the outside real world. The word Encapsulation, like Enclosing into the capsule. It restrict client from seeing its internal view where behavior of the abstraction is implemented. In Encapsulation, generally to hide data making it private and expose public property to access those data from outer world. Encapsulation is a method for protecting data from unwanted access or alteration. Encapsulation is the mechanism by which Abstraction is implemented.

**What is the difference between Abstraction and Encapsulation?**

Public class Customer

{

public string CustomerCode = “”;

public string CustomerName = “”;

public void Add ()

{

Validate();

CreateDBObjects();

// Code for Add

}

Private bool Validate()

{

//Validation Code

}

Private bool CreateDBObjects()

{

// CreateDBObjects code

}

}

In the above example we need only the add function and we really not interested in Validate and CreateDBObject functions so we are declaring both these functions as private under the customer class and they are not accessible from outside members this is called as Encapsulation.

And for the user we are only showing add function and hiding all other information from customer class this is called as Abstraction.