Pillars of Oops their advantages and Limitations

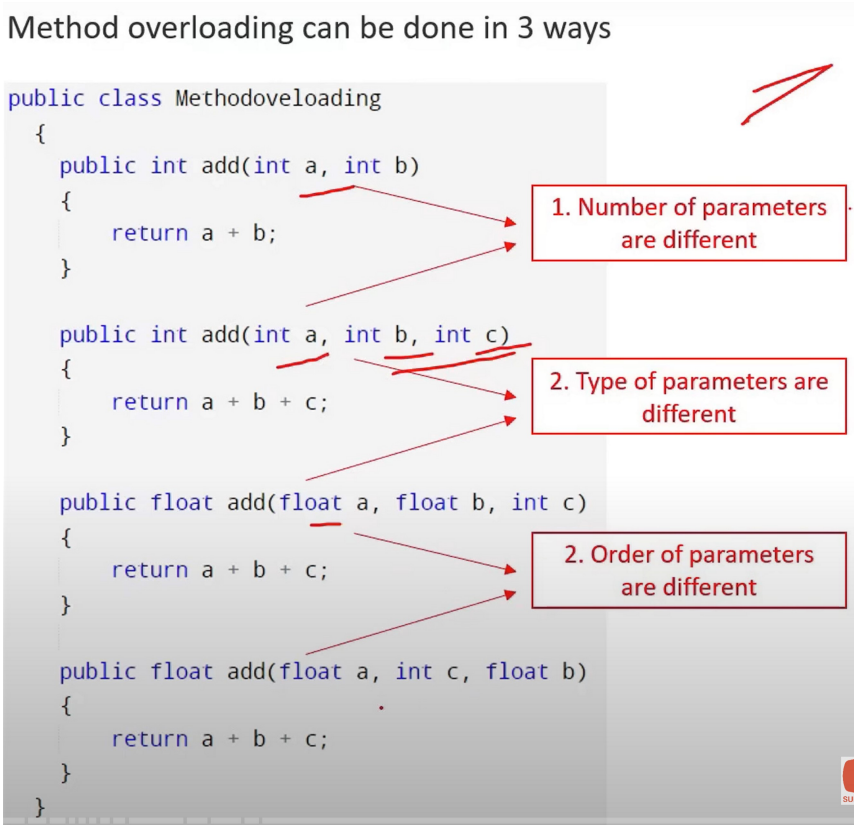
Friday, May 10, 2024 1:31 PM

What are four pillars of Oops and it's advantages

Polymorphism - Same name can do different ways.

Overloading Compile Time – Using Method overloading.

Allows a class to have multiple functions with same name but different signatures (Return type and input parameters)



Overriding Run Time – Using Method overriding.

Allows a subclass to override method from its parent class definition.

Method overloading is a kind of compile time polymorphism, in which we can create multiple methods of the **same name in the same class**, and all methods work in different ways.

Method overriding is creating a method in the **DERIVED** class with the **SAME NAME and SIGNATURE** as a method in the base class.

Overriding uses **VIRTUAL** keyword for base class method and **OVERRIDE** keyword for derived class method.

If you will remove Virtual and Override keyword then it will use baseClass Greetings() method and the output will be:

"baseClass Saying Hello!"

```
class baseClass
{
    public virtual void Greetings()
    {
        Console.WriteLine("baseClass Saying Hello!");
    }
}
class subClass : baseClass
{
    public override void Greetings()
    {
        Console.WriteLine("subClass Saying Hello!");
    }
}
class Program
{
    static void Main(string[] args)
    {
        baseClass obj1 = new subClass();

        obj1.Greetings();
    }
}
//Output: subClass Saying Hello!
```

Same method name but one is in base class and another is in derived class

This will call subclass Greetings() method because of overriding.

Abstraction - Reuse of code.

Process to hide the implementation and expose only the functionality to an end user.
That means, the end user will know, what it does rather how it does.
In C# you can implement it with both abstract class and interfaces.

Inheritance - Reuse of code.

Process of creation of new class by extending existing class.

Encapsulation - Helps achieve Principle of data hiding.

Is a process of wrapping variables and method in single unit.

Limitation of OOPS

So everything has some limitations, What I would say is "Oops is not good for Small scale apps", You could use procedural language there.
See, Oops need planning, so the question is to why to spend too much of your efforts in thinking oops all the time even for small one piece of code.