OOP Concepts

1. Class and Object

- **Class**: A blueprint for creating objects.
- **Object**: An instance of a class.

Example:

```
public class Car
{
   public string Color;
   public void Drive()
   {
      Console.WriteLine("Car is driving");
   }
  }
class Program
{
   static void Main()
   {
      Car myCar = new Car();
      myCar.Color = "Red";
      myCar.Drive(); // Output: Car is driving
   }
}
```

---

2. Encapsulation

- Wrapping data and methods into a single unit (class).
- Use of **access modifiers** like `private`, `public`, `protected`.

Example:

```
public class BankAccount
{
    private double balance;

    public void Deposit(double amount)
    {
        if (amount > 0) balance += amount;
    }

    public double GetBalance()
    {
        return balance;
    }
}
```

---

3. Inheritance

- Allows a class to inherit members (fields, methods) from another class.
- `:` symbol is used to inherit.

Example:

```
public class Animal
{
    public void Eat()
    {
        Console.WriteLine("Eating...");
    }
}

public class Dog : Animal
{
    public void Bark()
    {
        Console.WriteLine("Barking...");
    }
}
```

---

4. Polymorphism

- **Compile-time (Method Overloading)**: Same method name with different parameters.
- **Run-time (Method Overriding)**: Derived class provides specific implementation of a method.

Method Overloading Example:

```
public class MathOperations
{
    public int Add(int a, int b) => a + b;
    public double Add(double a, double b) => a + b;
}
```

Method Overriding Example:

```
public class Animal
{
    public virtual void MakeSound()
    {
        Console.WriteLine("Animal sound");
    }
}

public class Cat : Animal
{
    public override void MakeSound()
    {
        Console.WriteLine("Meow");
    }
}
```

OOP Concepts

5. Abstraction

- Hides internal implementation and shows only the necessary details.
- Achieved using **abstract classes** or **interfaces**.

Abstract Class Example:

```
csharp
CopyEdit
public abstract class Shape
{
    public abstract void Draw();
}

public class Circle : Shape
{
    public override void Draw()
    {
        Console.WriteLine("Drawing Circle");
    }
}
```