Encapsulation

Inheritance

# OOP

Polymorphism

Abstraction

BEHZAD DARA

SWIPE >>

```csharp
public class Person
{
    private int age;

    public int GetAge()
    {
        return age;
    }

    public void SetAge(int age)
    {
        if (age > 0)
        {
            this.age = age;
        }
    }
}
```

Encapsulation

Restrict direct access

BEHZAD DARA

**SWIPE >>**

Inheritance

Reuse parent

```
public class Animal
{
    public void Eat()
    {
        Console.WriteLine("This animal eats food.");
    }
}
```

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

BEHZAD DARA

SWIPE >>

```cs
public class Animal
{
    public virtual void Speak()
    {
        Console.WriteLine("The animal makes a
    }
}

public class Dog : Animal
{
    public override void Speak()
    {
        Console.WriteLine("The dog barks.");
    }
}

public class Cat : Animal
{
    public override void Speak()
    {
        Console.WriteLine("The cat meows.");
    }
}
```

Polymorphism

Child treated as parent

```cs
Animal myDog = new Dog();
```

BEHZAD DARA

**SWIPE >>**

Abstraction

hide complex implementation

```
public abstract class Shape
{
    public abstract double CalculateArea();
}
```

```
public class Rectangle(double width, double height) : Shape
{
    public override double CalculateArea()
    {
        return width * height;
    }
}
```

BEHZAD DARA