ISP - Interface Segregation Principle

O ISP afirma que uma interface não deve forçar classes a implementar métodos que elas não utilizam.

Benefícios:

- **Flexibilidade e Clareza:** Interfaces específica evitam que classes implementem métodos desnecessários.
- **Manutenção e Testes Fáceis:** Classes ficam mais fáceis de manter e testar, pois têm interfaces enxutas e focadas.

Exemplo de Má Prática: Interfaces grandes que obrigam classes a implementar métodos irrelevantes para elas.

ISP - Mau Exemplo

Problema: `Cachorro` é forçado a implementar `Voar()`, o que não faz sentido. Isso viola o ISP, pois a interface obriga a implementação de métodos que a classe não precisa.

```
public interface IAcoesAnimal
{
    void Comer();
    void Voar();
    void Nadar();
}

public class Cachoro : IAcoesAnimal
{
    public void Comer() => Console.WriteLine("Cachorro comendo.");
    // Problema
    public void Voar() => throw new NotImplementedException();
    public void Nadar() => Console.WriteLine("Cachorro nadando.");
}
```

ISP - Bom Exemplo

Solução: As interfaces `IComer`, `INadar` e `IVoar` são focadas e permitem que cada classe implemente apenas as interfaces que fazem sentido para ela.

```
public interface IComer
{
    void Comer();
}

public interface INadar
{
    void Nadar();
}

public interface IVoar
{
    void Voar();
}

public class Cachorro : IComer, INadar
{
    public void Comer() => Console.WriteLine("Cachorro comendo.");
    public void Nadar() => Console.WriteLine("Cachorro nadando.");
}

public class Passaro : IComer, IVoar
{
    public void Comer() => Console.WriteLine("Pássaro comendo.");
    public void Voar() => Console.WriteLine("Pássaro voando.");
}
```