Linguagem de Programação II

Análise e Desenvolvimento de Sistemas

2º semestre de 2019

Prof. Me. Renato Carioca Duarte

Herança

- A herança permite que você crie novas classes que podem ser reutilizadas e/ou estendidas.
- Classes cujos membros são herdados são chamadas de classes base, e classes que herdam esses membros são chamadas de classes derivadas.

```
class Base
{

}

class Derivada : Base
{

}
```

Herança

- Uma classe derivada pode ter apenas uma classe base direta.
- Se ClassC é derivada da ClassB e ClassB é derivada da ClassA, então a ClassC herda os membros declarados em ClassB e ClassA, pois a herança é transitiva.
- Conceitualmente, uma classe derivada é uma especialização da classe base.

```
class ClassA
{
}
class ClassB : ClassA
{
}
class ClassC : ClassB
{
}
```

Herança

Com esse recurso, praticamente todos os membros da classe base passarão a fazer parte da classe derivada. Existem algumas ressalvas no processo de herança que devem ser consideradas:

- membros **private** não são herdados;
- somente membros **protected** são herdados;
- somente o construtor sem parâmetros (se houver) é herdado;
- todas as classes derivam de object, portanto, a classe herdada e a derivada conterão os membros de object;
- não é permitido eliminar uma funcionalidade já existente, mas é possível redefini-la;
- para redefinir métodos de classes bases, estes devem ser virtuais (**virtual**);

Exemplo 1: Herança Simples

```
public class Conta        //classe base
{
    public int Numero { get; set; }
    public double Saldo { get; private set; }
}

public class ContaPoupanca : Conta            //classe derivada
{
    public int JurosMensais { get; set; }
}
```

Exemplo 2 – Duas classes derivadas

```
class A
{
    public void Exibir()
    {
        Console.WriteLine("Método da classe A");
    }
}
class B : A   // A classe B deriva da classe A
{
    public void Mostrar()
    {
        Console.WriteLine("Método da classe B");
    }
}
class C : A   // A classe C é derivada da classe A
{
    public void Apresentar()
    {
        Console.WriteLine("Método da Classe C");
    }
}
```

Exemplo 2 – Duas classes derivadas

```
static void Main(string[] args)
{
    B obj1 = new B();
    C obj2 = new C();
    obj1.Exibir();
    obj1.Mostrar();
    obj2.Apresentar();
}
```

Exemplo 3 – Herança com 3 níveis

```
class A
{
    public void Exibir()
    {
        Console.WriteLine("Método da classe A");
    }
}
class B : A   // A classe B deriva da classe A
{
    public void Mostrar()
    {
        Console.WriteLine("Método da classe B");
    }
}
class C : A   // A classe C é derivada da classe A
{
    public void Apresentar()
    {
        Console.WriteLine("Método da Classe C");
    }
}
```

Exemplo 3 – Herança com 3 níveis

```
static void Main(string[] args)
{
    C obj = new C();
    obj.Exibir();
    obj.Mostrar();
    obj.Apresentar();
    Console.ReadKey();
}
```

Exercícios

1. Fazer no Visual Studio o exemplo 2 com as 3 classes A, B c C e o Main.
2. Fazer no Visual Studio o exemplo 3 com as 3 classes A, B c C e o Main.