Aula 23

Prof: Henrique Augusto Maltauro

Desenvolvendo Algoritmos

Programação Orientada a Objetos (POO)

- Quatro pilares da POO

Principais fundamentos da POO.

- Abstração ✓
- Herança ✓
- Encapsulamento ✓
- Polimorfismo ✓

Programação Orientada a Objetos (POO)

- Interfaces

Conforme já havíamos comentado, quando se trata de herança, uma classe só pode herdar de uma única classe.

Contudo, surgem situações em que uma classe precisa herdar atributos e/ou métodos de mais de um lugar.

Como solução nesses casos, fazemos o uso de interfaces.

Programação Orientada a Objetos (POO)

- Interfaces

É importante entendermos que não existe a herança de interfaces, existe a implementação de interfaces.

Diferente das classes, a interface não define nenhuma lógica, ela simplesmente define a existência de atributos e métodos, e quem vai definir a lógica é a classe que vai implementar a interface.

Programação Orientada a Objetos (POO)

- Interfaces

A interface funciona como uma espécie de contrato a qual a classe está atrelada, pois ela define atributos que a classe vai ter, e métodos que a classe vai ser responsável por implementar.

Por convenção, nomeamos as interfaces com o prefixo I.

Programação Orientada a Objetos (POO)

- Interfaces (vamos imaginar um sistema escolar)

Ambos alunos e professores são pessoas.

Mas dentro do sistema escolar, o professor não é apenas uma pessoa, ele também é um funcionário da escola.

E a escola não vai ter apenas professores de funcionários, mas vai ter zeladores, bibliotecários, coordenadores, tesoureiros, diretores, etc.

Programação Orientada a Objetos (POO)

- Interfaces (vamos imaginar um sistema escolar)

Trazendo esse exemplo para um contexto de POO, nós teríamos as classes Aluno e Professor.

A classe Aluno implementa a interface IPessoa.

E a classe Professor implementa as interfaces IPessoa e IFuncionario.

Programação Orientada a Objetos (POO)

- C#: Interfaces

No caso do C#, os atributos das interfaces precisam receber os acessadores get e set, para indicar que aquela variável é de fato um atributo e não apenas uma variável qualquer.

Programação Orientada a Objetos (POO)

- C#: Interfaces

```
public interface IPessoa
{
    string Nome { get; set; }
    string CPF { get; set; }
}
```

Programação Orientada a Objetos (POO)

- C#: Interfaces

```
public interface IFuncionario
{
    decimal Salario { get; set; }
    void ReceberSalario();
}
```

Programação Orientada a Objetos (POO)

- C#: Interfaces

```
public class Professor : IPessoa, IFuncionario
{
    public string Nome { get; set; }
    public string CPF { get; set; }
    public decimal Salario { get; set; }

    public void ReceberSalario()
    {
        // Bloco de código do método
    }
}
```

Exercício

Exercício

1 - Crie a interface FiguraGeometrica com os atributos Altura e Largura e o método CalcularArea.

Crie as classes Quadrado, Retangulo e Triangulo, que devem implementar a interface FiguraGeometrica.

As classes devem ter o método construtor recebendo os atributos como parâmetros.

Exercício

2 - Crie a interface OperacaoMatecatica com o método Calculo, que recebe dois parâmetros do tipo double.

Crie as classes Soma, Subtracao, Multiplicacao e Divisao, que devem implementar a interface OperacaoMatecatica.

As classes não precisam ter o método construtor.