

# Programação Orientada a Objetos: POO

Relacionamento entre Classes



Teoria?

Vocês já sabem o que é necessário!

Lutador
<ul style="list-style-type: none"><li>- nome</li><li>- nacionalidade</li><li>- idade</li><li>- altura</li><li>- peso</li><li>- categoria</li><li>- vitórias</li><li>- derrotas</li><li>- empates</li></ul>
<ul style="list-style-type: none"><li>apresentar()</li><li>status()</li><li>ganharLuta()</li><li>perderLuta()</li><li>empatarLuta()</li></ul>

```
// pseudo código
// Classe/Molde - LUTADOR
```

### Classe Lutador

```
privado nome: String
privado nacionalidade: String
privado idade: int
privado altura: real
privado peso: real
privado categoria: String
privado vitorias: int
privado derrotas: int
privado empates: int
```

```
publico metodo construtor(no: String,
                           na: String,
                           id: int,
                           al: real,
                           pe: real,
                           vi: int,
                           de: int,
                           em: int){
```

```
    nome = no
    nacionalidade = na
    idade = id
    altura = al
    setPeso(pe)
    vitorias = vi
    derrotas = de
    empates = em
```

```
}
```

```
publico metodo apresentar()
    (...)
fimMetodo
publico metodo status()
    (...)
fimMetodo
publico metodo ganharLuta()
    (...)
fimMetodo
publico metodo perderLuta()
    (...)
fimMetodo
publico metodo empatarLuta()
    (...)
fimMetodo
```

```
publico metodo getName()
    retorne nome
fimMetodo
publico metodo setName(no: String)
    nome = no
fimMetodo
publico metodo getPeso()
    retorne peso
fimMetodo
publico metodo setPeso(pe: real)
    peso = pe
    setCategoria()
fimMetodo
publico metodo setCategoria()
    se(peso<52.2)então
        categoria = "Inválida"
    senão se(peso<=70.3)então
        categoria = "Leve"
    senão se(peso<=83.9)então
        categoria = "Médio"
    senão se(peso<=120.2)então
        categoria = "Pesado"
    senão
        categoria = "Inválida"
    fimSe
fimMetodo
```

```
publico metodo ganharLuta()
    setVitorias(getVitorias() + 1)
fimMetodo
publico metodo perderLuta()
    setVitorias(getDerrotas() + 1)
fimMetodo
publico metodo empatarLuta()
    setVitorias(getEmpates() + 1)
fimMetodo
```

```
publico metodo apresentar()
    escreva("Lutador: " + getName())
    escreva("Origem: " + getNacionalidade())
    escreva(getIdade() + " anos")
    escreva(getAltura() + " m de altura")
    escreva("Pesando " + getPeso() + "Kgs")
    escreva("Ganhou " + getVitorias() + " vezes")
    escreva("Perdeu " + getDerrotas() + " vezes")
    escreva("Empatou " + getEmpates() + " vezes")
fimMetodo
publico metodo status()
    escreva(getNome() + " é um peso " + getCategoria())
    escreva(getVitorias()+"/"+getDerrotas()+"/"+getEmpates())
fimMetodo
```

FimClasse



```
// Programa Principal / Main / Instanciar o Objeto
// Objetos são como variáveis, mas são Instancias de uma Classe (reler até entender)
// Ao invés de instanciar 6 lutadores, é possível criar um vetor de 6 posições.

Lutador: vetor[0...2]
Lutador[0] = novo Lutador(
    "Fulano de Tal", "França",
    31, 1.75, 68.9,
    11, 3, 1)
Lutador[1] = novo Lutador(
    "Beltrano Silva", "Brasil",
    29, 1.68, 57.8,
    12, 2, 3)
Lutador[2] = novo Lutador(
    "Ruskamenegov Guthahs", "Russia",
    35, 1.88, 99.7,
    15, 2, 2)
// adicionar quantos + desejar...

Lutador[0].apresentar()
Lutador[1].status()
Lutador[2].getCategoria()
Lutador[0].ganharLuta()
Lutador[1].perderLuta()
Lutador[2].empatarLuta()
Lutador[0].apresentar()
Lutador[1].status()
Lutador[2].getCategoria()
```

# Atividade avaliativa

## n1.5 (parte 1/2)

a. Escolha um dentre os temas para trabalhar:

- **Herói** / BatalhaEpica
- **Monstro** / Combate
- **Mago** / DueloDeMagia
- **Rapper** / BatalhaDeRima
- **Atirador** / Tiroteio
- **TimeDeFutebol** / PartidaDeFutebol

b. Defina alguns Atributos, Métodos e Métodos Especiais (getter, setter e constructor) à Classe escolhida, de acordo com o tema;

- b1. Opcional: se desejar, crie também uma Interface para a Classe, informando quais os métodos obrigatórios;

c. Crie um Diagrama UML para a Classe escolhida (Herói, Monstro, Mago, Rapper, Atirador, TimeDeFutebol), para deixar claro, de forma simples, o que foi definido anteriormente;

d. A partir do Diagrama UML, desenvolva sua Classe em código;

e. Instancie alguns objetos, já passando os valores dos atributos por meio do método Construtor;

f. Utilize alguns métodos da Classe e realize algumas mudanças nos valores dos atributos.

```
// pseudo código
// Classe/Molde - LUTA
Classe Luta
    // agregação = tem 1 (uma Classe, tem outra Classe)
    // o tipo do parâmetro é a Classe
    privado desafiado: Lutador
    privado desafiante: Lutador
    privado rounds: int
    privado aprovada: boolean

    public metodo marcarLuta()
        (...)
    fimMetodo
    public metodo lutar()
        (...)
    fimMetodo

    // posso utilizar um objeto também como parâmetro
    // já que funciona como um 'tipo' (int, double...)
    public metodo setDesafiado(dd: Lutador)
        desafiado = dd
    fimMetodo
    public metodo getDesafiado()
        retorne desafiado
    fimMetodo
```

```
//adicionar todos os demais getters e setters restantes...

// regras (criatividade...)
// luta somente entre lutadores da mesma categoria
// luta somente entre lutadores diferentes (o obvio precisa ser verificado
// código != bola de cristal)
// luta somente pode acontecer se estiver aprovada
// unicos resultados possíveis: vitoria de um dos lutadores OU empate
```

```
//o parametro do método é o objeto do tipo Lutador
publico metodo marcarLuta(lutador1, lutador2: Lutador)
    // .getCategoria() é um método da Classe Lutador
    // como estamos usando um objeto dessa Classe, podemos utilizar
    // seus métodos e atributos
    se(lutador1.getCategoria()==lutador2.getCategoria() e lutador1!=lutador2) entao
        aprovada = verdadeiro
        desafiante = lutador1
        desafiado = lutador2
    senao
        aprovada = falso
        desafiante = null
        desafiado = null
    fimSe
fimMetodo

publico metodo lutar()
    se(aprovada) então
        desafiante.apresentar()
        desafiado.apresentar()
        //aleatorio: 0=empate, 1=vitoria do desafiante, 2=vitoria do desafiado
        vencedor = aleatorio(0..2)
        escolha(vencedor)
        caso 0:
            Escreva("Empatou!")
            desafiante.empatarLuta()
            desafiado.empatarLuta()
        caso 1:
            Escreva("Vencedor: " + desafiante.getNome())
            desafiante.ganharLuta()
            desafiado.perderLuta()
        caso 2:
            Escreva("Vencedor: " + desafiado.getNome())
            desafiante.perderLuta()
            desafiado.ganharLuta()
    fimEscolha
    senao
        Escreva("A luta não pode acontecer.")
    fimSe
fimMetodo
fimClasse
```



```
// Programa Principal / Main / Instanciar o Objeto (CONTINUAÇÃO)

Lutador: vetor[0...2]
Lutador[0] = novo Lutador(
    "Fulano de Tal", "França",
    31, 1.75, 68.9,
    11, 3, 1)
Lutador[1] = novo Lutador(
    "Beltrano Silva", "Brasil",
    29, 1.68, 57.8,
    12, 2, 3)
Lutador[2] = novo Lutador(
    "Ruskamenegov Guthahs", "Russia",
    35, 1.88, 99.7,
    15, 2, 2)

// adicionar quantos + desejar...

// agora, além de instanciar objetos da Classe/tipo/molde Lutador...
// posso também instanciar e utilizar objetos da Classe/tipo/molde Luta!
Luta UFC001 = novo Luta()

// chamo um método da classe luta, passando como parâmetro objetos do tipo/Classe Lutador
UFC001.marcarLuta(Lutador[0], Lutador[1])
UFC001.lutar()
```



# Atividade avaliativa

## n1.5 (parte 2/2)

a. Crie a segunda Classe de acordo com o tema escolhido (BatalhaEpica, Combate, DueloDeMagia, BatalhaDeRima, Tiroteio, PartidaDeFutebol);

b. Nessa Classe, crie pelo menos um método que espere receber como parâmetro, objetos da outra classe;

Ex.: marcarPartida(time01, time02: TimesDeFutebol);

c. Na Main, instancie pelo menos um objeto desta nova Classe (dica: para ser possível utilizar o que foi criado pelo item b);

d) Através dos objetos instanciados anteriormente, realize chamadas de métodos que utilizem objetos da outra Classe como parâmetro (dica: utilize o que foi criado no item b);

e) Exiba valores em tela para deixar claro o que está acontecendo;

f) Deixe comentários pelo código, para explicar os atributos, métodos, parâmetros, instâncias e chamadas de métodos.