Programação Orientada a Objetos

Herança

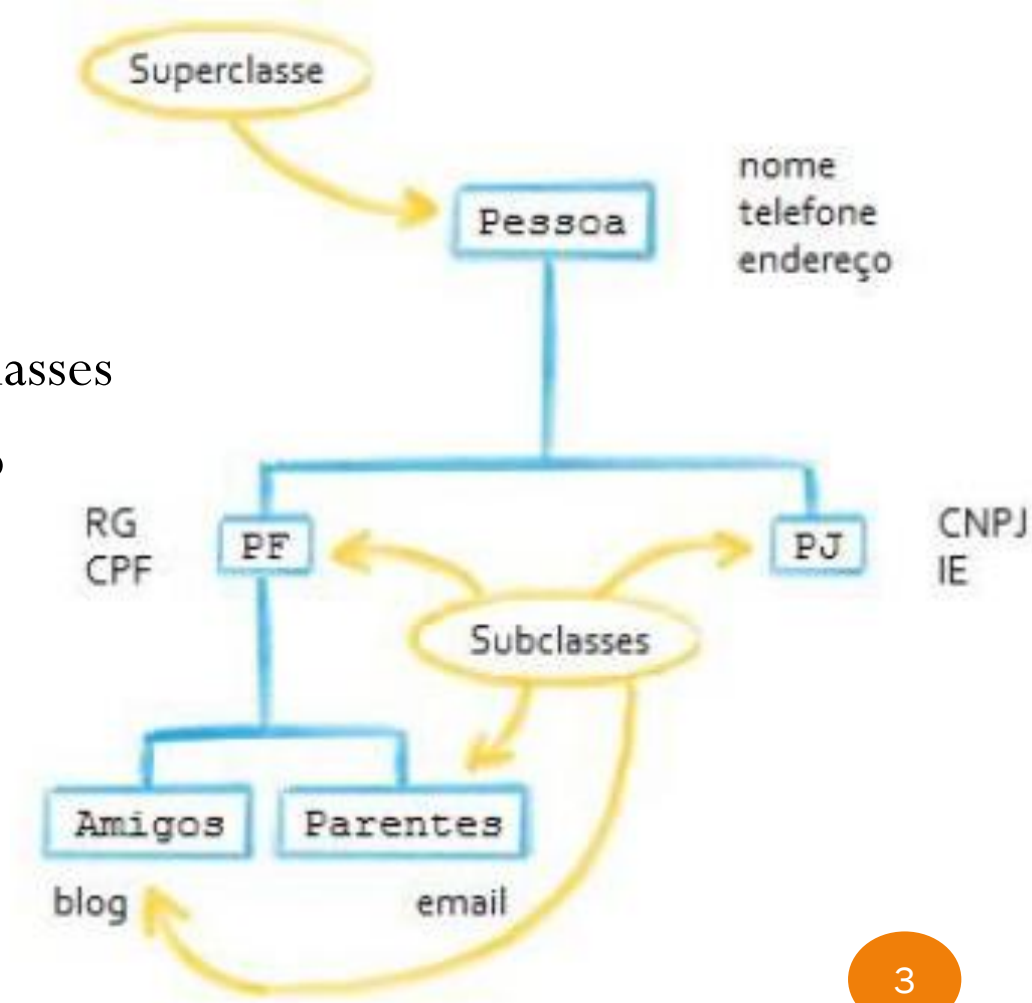
Professor Me.: Gustavo Siqueira Vinhal

Herança


Orientação a Objetos
Encapsulamento
Herança
Composição
Polimofismo
Princípio da abstração

Herança

- Cria uma nova classe a partir de uma classe existente:
  - absorvendo os dados e comportamentos da classe existente; e
  - aprimorando-a com novas capacidades.
- Adota um relacionamento hierárquico entre classes
- Permite melhor organização e reuso de código



Herança

- Subclasse ou classe derivada:
  - criada a partir de outra classe (classe mãe)
  - herda características da classe mãe
  - também possui características próprias
- Superclasse ou classe base:
  - concede características a classe derivada
- Relação: Subclasse estende a superclasse

Herança

- A superclasse representa um conjunto maior de objetos do que as subclasses.
  - Superclasse **Veículo:** representa carros, caminhões, barcos, bicicletas…
  - Subclasse **Carro:** representa um subconjunto específico de veículos

- Relação de hierarquia: "é um"
  - Carro "é um"Veículo

Herança

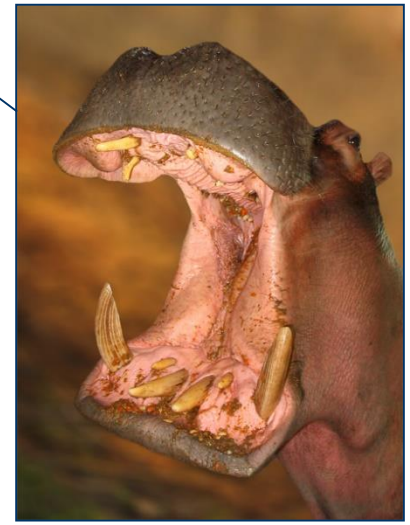
- A superclasse é mais geral do que suas subclasses.
- Uma subclasse é uma especialização de uma superclasse;
- A superclasse é uma generalização de subclasses;

Herança

Superclasse
Mamífero

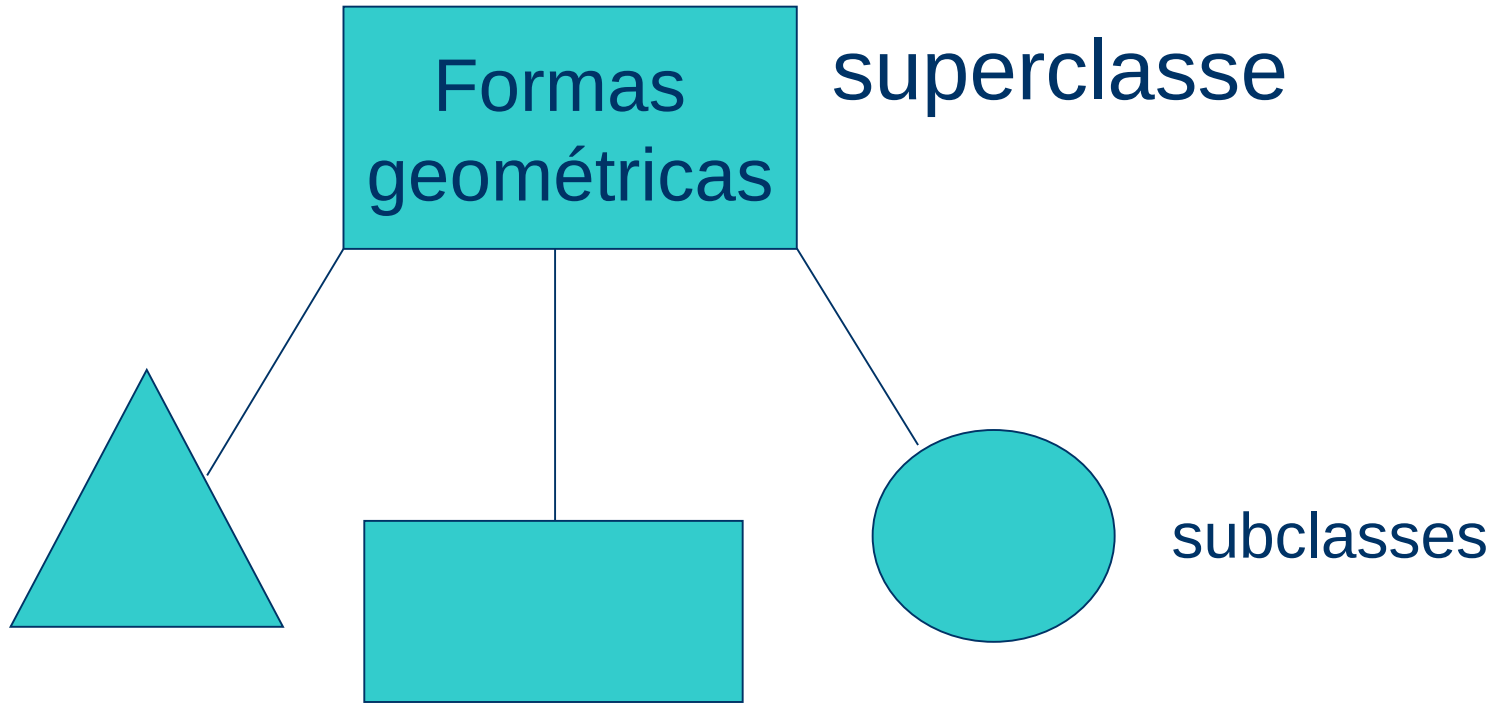
Guaxinim

Cachorro

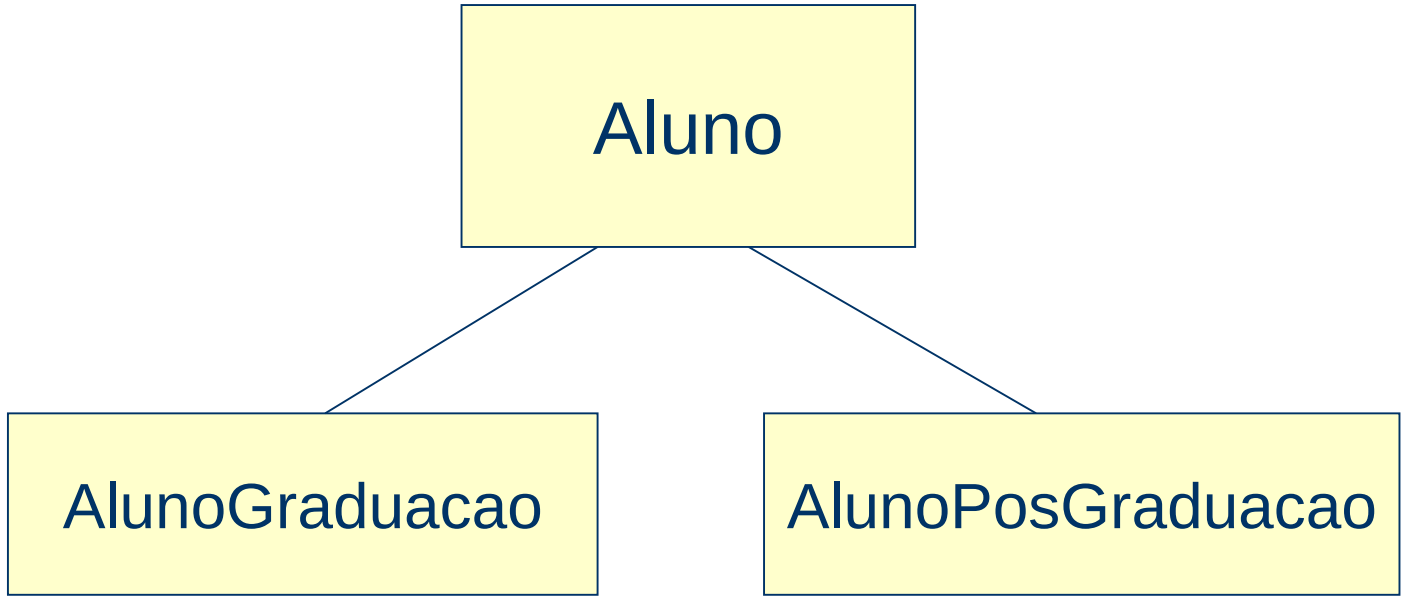


Hipopótamo

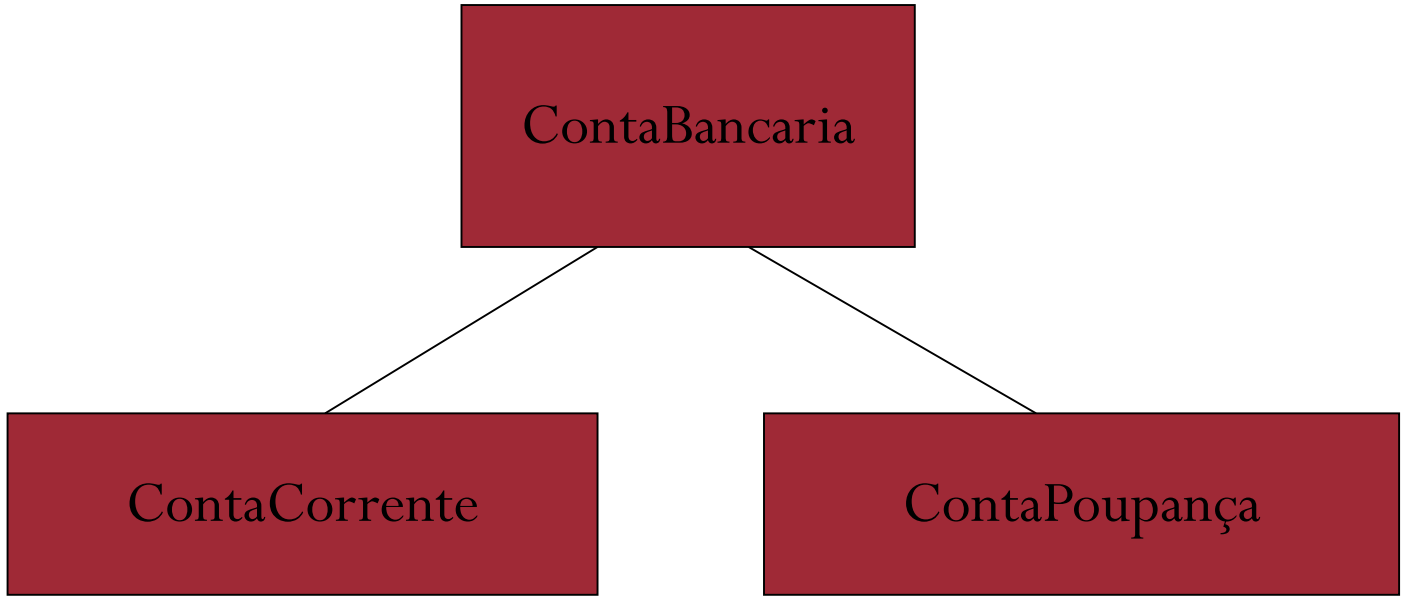
Herança



Herança

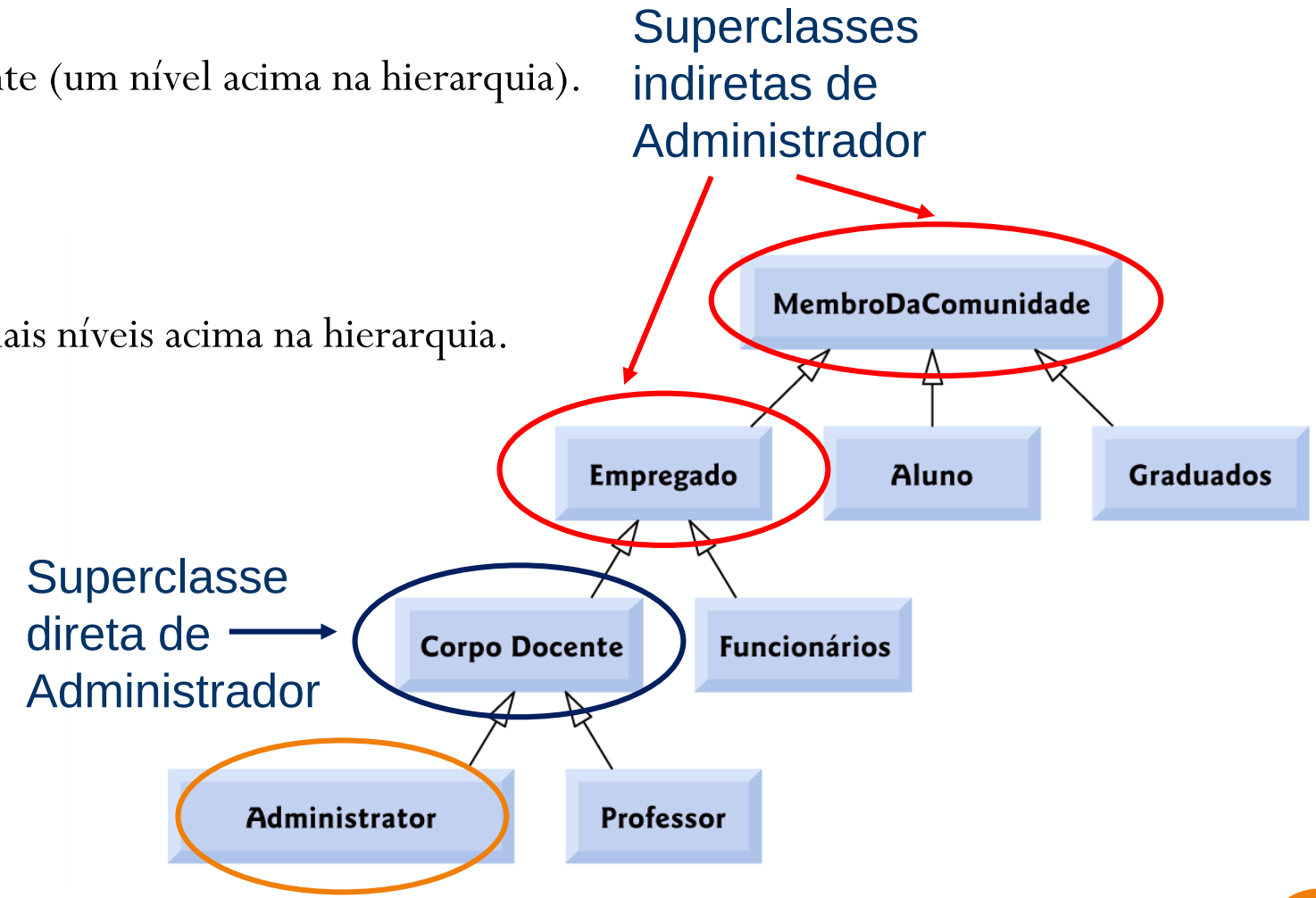


Herança



Herança

- Superclasse direta:
  - Herdada explicitamente (um nível acima na hierarquia).
- Superclasse indireta:
  - Herdada de dois ou mais níveis acima na hierarquia.



Herança

- Herança única:
  - Herda de uma superclasse.

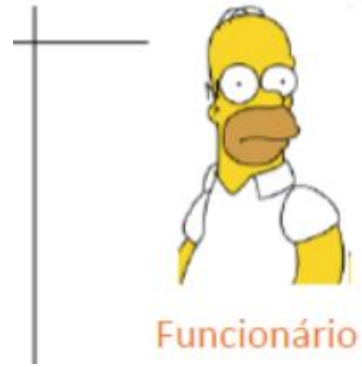
- Herança múltipla:
  - Herda de múltiplas superclasses.
  - O Java não suporta herança múltipla.

Exemplo

- Criar uma classe para aluno, professor e funcionário.

- Primeiro passo: quais os atributos e métodos de cada classe?

Exemplo




```
class Professor{
    String nome;
    String email;
    String telefone;
    int idade;
    char sexo;
    double salario;
    String disciplina;
    }
```

```
class Aluno{
    String nome;
    String email;
    String telefone;
    int idade;
    char sexo;
    String matricula;
    float nota;
    }
```

```
class Funcionario{
    String nome;
    String email;
    String telefone;
    int idade;
    char sexo;
    double salario;
    }
```

Exemplo

```
class Pessoa{
    String nome;
    String email;
    String telefone;
    int idade;
    char sexo;
    }
```

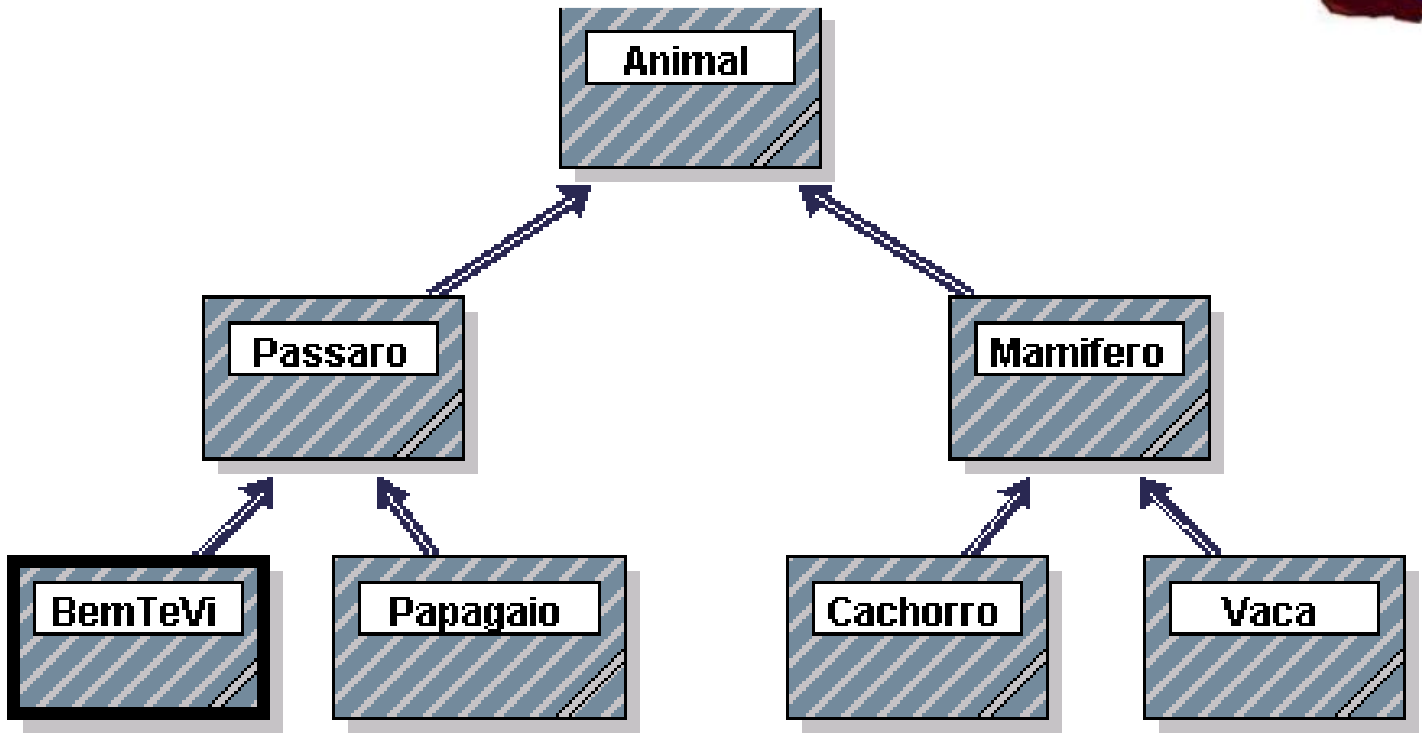
```
class Professor extends Pessoa{
    double salario;
    String disciplina;
    }
```

```
class Aluno extends Pessoa{
    String matricula;
    float nota;
    }
```

```
class Funcionario extends Pessoa{
    double salario;
    }
```

Prática:

1 – Escreva o código para implementar a seguinte hierarquia de classes:



Prática:

Atributos da classe “Animal” :

• nome

Métodos da classe “Animal”:

• imp(): Imprime o nome e a classe do animal

• talk(): Imprime “Me not falar”

Métodos da classe “Passaro”:

• talk(): Imprime “Piu piu”

Métodos da classe “BemTeVi”:

• talk(): Imprime “bem-te-vi!”

Atributos da classe “Papagaio”:

• vocabulario: frases que o papagaio consegue dizer (Considere que são no máximo 10)

Prática:

Métodos da classe “Papagaio”:

- talk(): Imprime o vocabulario
- addFrase(String f): Adiciona uma frase ao vocabulário

A classe mamífero não tem atributos ou métodos.

Atributos da classe “Cachorro”:

- lateMuito: um valor lógico que indica se o cachorro late muito

Métodos da classe “Cachorro”:

- setLateMuito(): configura o atributo como verdadeiro.
- setLatePouco(): configura o atributo como falso.
- talk(): Imprime “Au au”

Métodos da classe “Vaca”;

- talk(): Imprime “Muuuuuu”

Prática:

1 – Crie um programa que implemente contas (corrente e poupança).


Obrigado!