Departamento de Computação

Disciplina: Técnicas de Programação I (CMP 1048)

Prof. MSc. Anibal Santos Jukemura

Exercícios: HERANÇA ÚNICA (SIMPLES)

Para cada exercício crie um novo projeto ou separe as classes por pacotes (packages). Teste todos os programas na classe principal (classe que contém o método main).

Exercício 1: Crie a **classe Imovel**, que possui um **endereço (String)** e um **preço (double)**.

a) Crie uma **classe Novo**, que **herda Imovel** e possui um **adicional (double)** no preço. Crie métodos de acesso e impressão deste valor adicional.
b) Crie uma **classe Velho**, que **herda Imovel** e possui um **desconto (double)** no preço. Crie métodos de acesso e impressão para este desconto.
c) Teste as classes novo e velho, criando objetos para os mesmos.
d) Imprima o endereço, preço e adicional para o imóvel novo.
e) Imprima o endereço, preço e desconto para o imóvel velho.

Exercício 2: Implemente a classe **Funcionario** com um atributo private String nome e outro atributo private String cpf (faça os métodos GET/SET do tipo private).

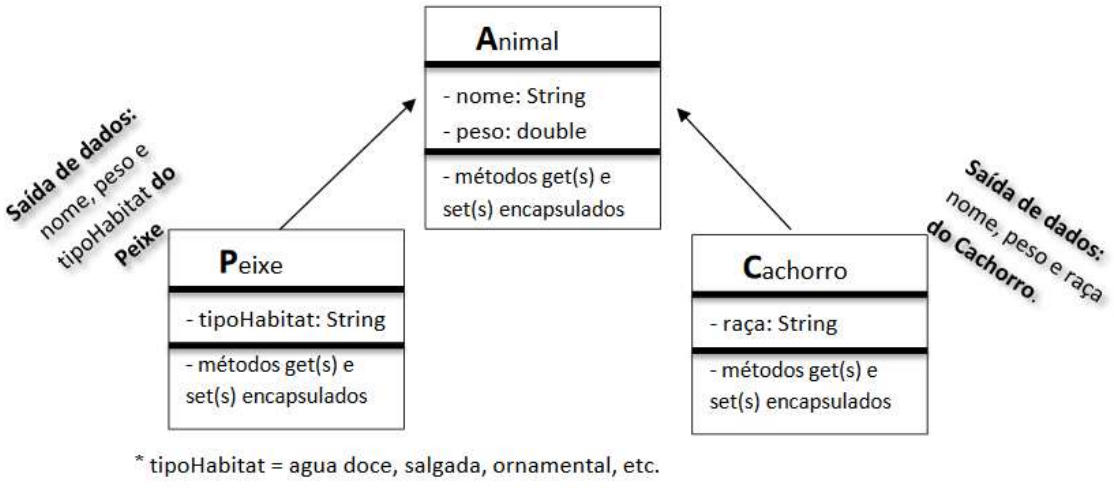
a) crie a classe **Assistente**, que também é um funcionário, e que possui um número de matrícula (faça os métodos GET/SET do tipo private).
b) Sabendo que os Assistentes Técnicos possuem um bônus salarial (private double bonus) e que os Assistentes Administrativos possuem um turno (atributo private String) qu pode ser dia ou noite, crie as classes **Tecnico** e **Administrativo**. (atributo private String). faça os métodos GET/SET do tipo protected. Crie um método public chamado imprimir para ambas as classes Tecnico e Administrativo. Use esse método para imprimir todos os dados das respectivas classes: nome, cpf, matricula e bonus para a classe Tecnico; e nome, cpf, matricula e turno para a classe Administrativo.
   Obs.: Note que um Administrativo é um Assistente que também é um funcionário. Da mesma forma, um Técnico é um Assistente que também é um Funcionário.
c) O que há de errado? Como corrigir o problema, sem usar o modificador public?
d) Após corrigir o código, crie um objeto Técnico e outro Administrativo. Utilize os métodos SET para preencher todos os atributos de ambas as classes. O que há de errado? Como corrigir o problema através do modificador protected, criando um construtor para Tecnico e outro para Administrativo?
e) Após corrigir o problema, teste o programa executando o método imprimir de cada uma das classes.
f) É possível criar um objeto do tipo Assistente ou Funcionário? Implemente e discuta essa possibilidade.

Exercício 3: Crie uma classe chamada **Ingresso** que possui um valor em reais (double) e um método imprimeValor(). Crie um construtor para essa classe, passando como parâmetro, um valor em reais para iniciar o atributo da classe.

a. Crie uma classe VIP, que **herda Ingresso** e possui um valor adicional (double). Crie um método que retorne o valor do ingresso VIP (com o adicional incluído). Observe a necessidade de um construtor para VIP.
b. Crie uma classe Normal, que **herda Ingresso** e possui um método que imprime: "Ingresso Normal". Observe a necessidade de um construtor para classe Normal.
c. Crie uma classe **CamaroteInferior** que possui a localização do ingresso (String) e métodos **public** para acessar e imprimir esta localização (GET/SET). O camaroteInferior tem vagas somente para ingressos normais. Ou seja, **normal** é um atributo da classe (Associação).
d. Crie uma classe **CamaroteSuperior** que possui a localização do ingresso (String) e métodos **public** para acessar e imprimir esta localização (GET/SET). O camaroteSuperior tem vagas somente para ingressos VIP. Ou seja, **normal** é um atributo da classe (Associação).
e. Crie um objeto para uma vaga no camarote inferior e outro objeto para o camarote superior. Imprima o tipo do ingresso, a localização da vaga no camarote e o valor do ingresso. Se for VIP, imprima os valores normal e o adicional.
   Obs.: para TODOS os atributos, utilize o modificador de acesso **private**.

Exercício 4: Crie as classes conforme descritas no o diagrama UML. Faça o relacionamento (herança) entre as classes. Crie objetos da superclasse e das duas sub-classes para testar o programa. Use os seguintes exemplos de animais para saídas de dados:

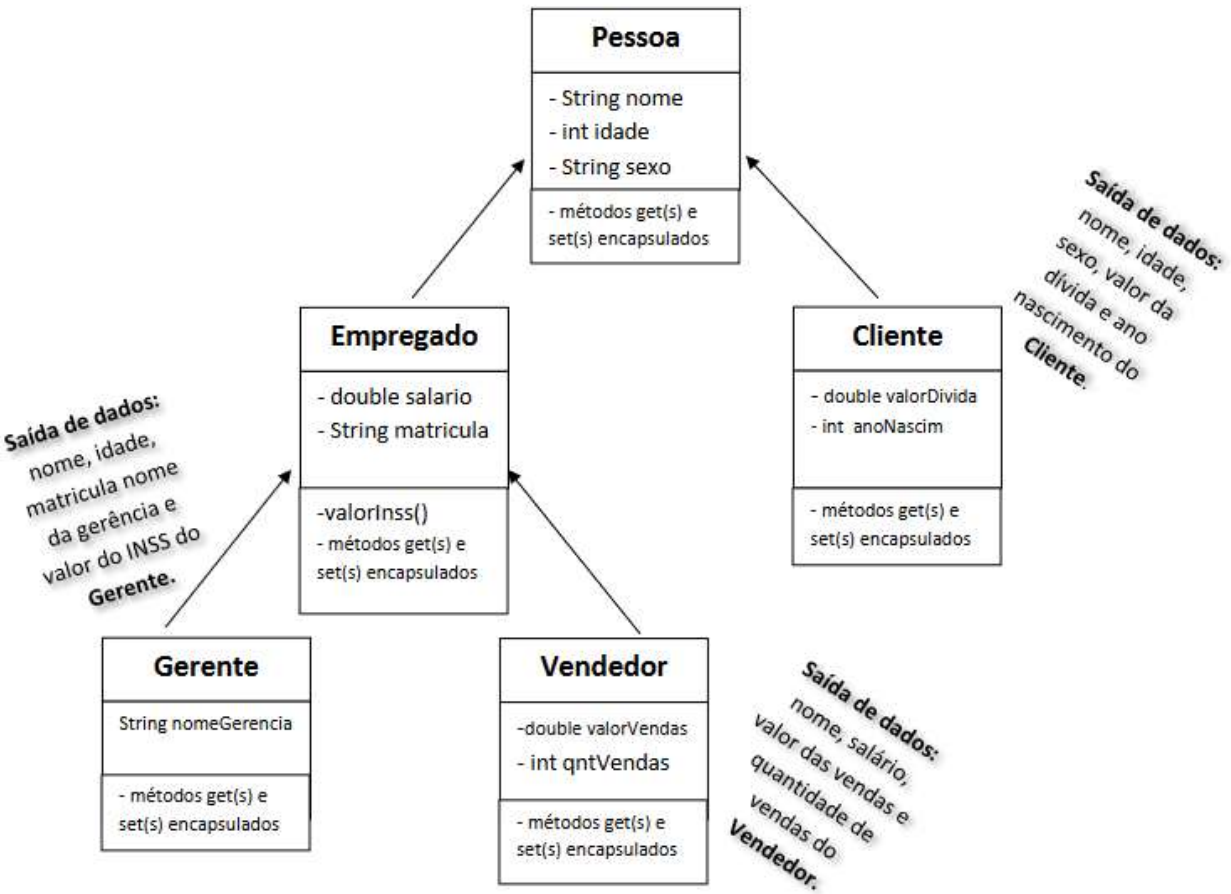
Cachorro: Nome: Rex Peso : 30 kg Raça : Boxer	Peixe: Nome : Neon Peso : 0.2 Kg Habitat: ornamental	Animal: Nome: Gato Peso : 10 Kg



* tipoHabitat = agua doce, salgada, ornamental, etc.

Exercício 5: Crie as classes conforme descritas no o diagrama UML. Faça o relacionamento (herança) entre as classes. Crie objetos da superclasse e das duas sub-classes para testar o programa. Use os seguintes exemplos para saídas de dados:

	Pessoa	Empregado	Cliente	Gerente	Vendedor
Nome	Joao	Ana	Almir	Tania	Igor
Idade	33	21	40	30	
Sexo	Masculino	Feminino	Masculino		
Salario		5000.00		15000.00	7000.00
matricula		1400		1300	
valorDivida			2000.00		
anoNascim			1976		
nomeGerencia				Atendimento	
valorVendas					14000.00
atdeVendas					120
valorInss		salario – salario*0.05		salario – salario*0.05	



valorInss retorna (salario – salario*0.05)