Universidade Federal Rural do Semi-Árido
Centro Multidisciplinar Pau dos Ferros
Departamento de Engenharias e Tecnologia
PEX1263 - Projeto Detalhado de Software
Profa. Huliane Medeiros da Silva

Lista de Exercícios

1. Elaborar um programa OO que:
    a. Implemente uma classe abstrata C1;
    b. Implemente duas classes concretas C2 e C3, ambas herdando de C1;
    c. Implemente duas classes concretas C4 e C5, ambas herdando de C2;
    d. Considere que todas as classes devem ter pelo menos um método e um atributo próprios;
    h. Demonstre no exercício:
        i. Sobrecarga de construtores;
        ii. Sobrecarga de métodos;
        iii. Sobreposição de métodos.
    i. Implemente a classe Principal, para testar todos os métodos das demais classes, contendo pelo menos um objeto de cada classe concreta.

2. Implemente o seguinte conjunto de classes em um programa Java.
    a. Classe: Porta
        i. Atributos: aberta, cor, dimensaoX, dimensaoY, dimensaoZ.
        ii. Métodos: void abre(), void fecha(), void pinta(String s), boolean estaAberta(). Para testar, crie uma porta, abra e feche a mesma, pinte-a de diversas cores, altere suas dimensões e use o método estaAberta para verificar se ela está aberta.
    b. Classe: Casa
        i. Atributos: cor, porta1, porta2, porta3;
        ii. Método: void pinta(String s), int quantasPortasEstaoAbertas(), int totalDePortas(). Para testar, crie uma casa e pinte-a. Crie três portas e coloque-as na casa; abra e feche as mesmas como desejar. Utilize o método quantasPortasEstaoAbertas() para imprimir o número de portas abertas.
    c. Classe: Edificio
        i. Atributos: cor, totalDePortas, totalDeAndares, portas[];
        ii. Métodos: void pinta(String s), int quantasPortasEstaoAbertas(), void adicionaPorta(Porta p), int totalDePortas(), void adicionarAndar(), int totalDeAndares().
        iii. Para testar, crie um edifício, pinte-o. Crie seis portas e coloque-as no edifício através do método adicionaPorta(), abra e feche-as como desejar. Utilize o método quantasPortasEstaoAbertas para imprimir o número de

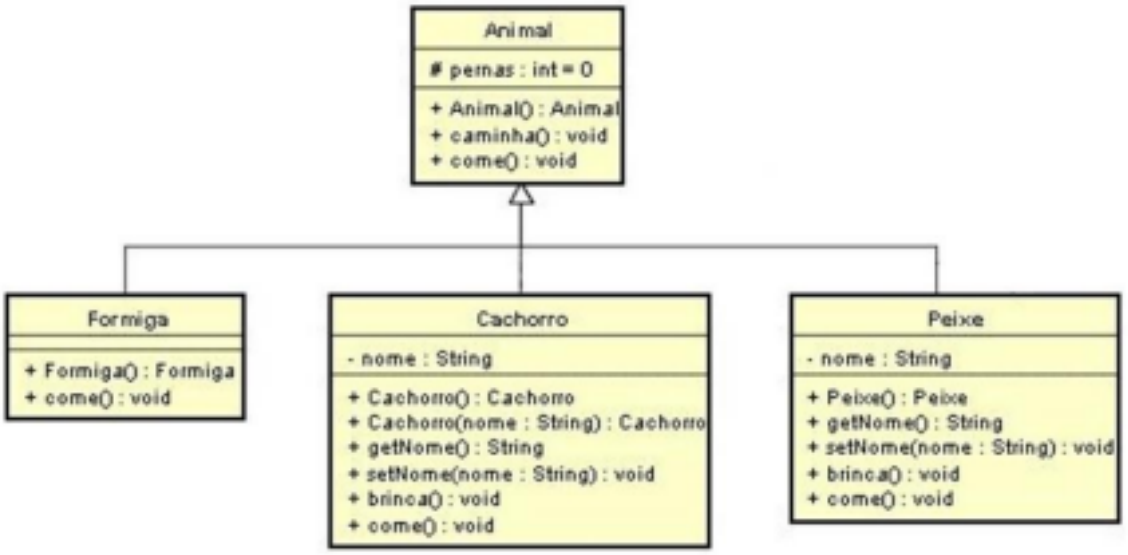
portas abertas e o método totalDePortas para imprimir o total de portas em seu edifício. Crie alguns andares utilizado o método adicionarAndar e retorne o número total de andares utilizando o método totalDeAndares.

d. As classes Casa e Edifício ficaram muito parecidas. Crie a classe Imovel e coloque nela tudo o Casa e Edificio tem em comum. Faça Imovel superclasse de Casa e Edificio. Note que alguns métodos em comum não poderão ser implementados por Imovel. Logo, esses deverão ser abstratos.

3. Implemente o seguinte conjunto de classes em um programa Java: Veículo, carro, motocicleta, caminhão, brinquedo. Estabeleça as relações entre elas corretamente. Utilize os conceitos de abstração, herança, polimorfismo e encapsulamento.

4. Elabore um programa em Java que:
   a. Declare uma classe abstrata A com um atributo encapsulado, dois construtores sobrecarregados (um que não recebe parâmetros e outro que inicializa o valor do atributo), e seus métodos gets e sets;
   b. Declare em A dois métodos abstratos;
   c. Declare três subclasses, B, C e D, que herdam de A;
   d. Em cada subclasse declare os respectivos atributos próprios b1, c1 e d1, seus métodos próprios e construtores que inicializem todos os atributos;
   e. Declare uma classe E que tenha um atributo ArrayList que seja capaz de guardar objetos de B, C e D e dois métodos estáticos: um para retornar o ArrayList e outro para retornar um elemento específico do ArrayList a partir de uma busca;
   f. Declare a classe Principal contendo objetos das classes B, C e D;
   g. Insira estes objetos no ArrayList da classe E;
   h. Teste todos os métodos, inclusive os da classe E.

5. Escreva as classes em Java de acordo com o diagrama abaixo e uma classe Principal que teste seus métodos (não é necessário fazer um menu na classe principal; basta instanciar um objeto de cada classe e chamar todos os seus métodos).



6. A empresa XPTO necessita desenvolver um sistema para catalogar itens colecionáveis (livros, CDs, DVDs e revistas). O objetivo deste sistema é manter os itens colecionáveis,

organizados por tipo. O sistema deve permitir cadastrar os dados comuns e os específicos de cada tipo de item. Os dados comuns são: identificação única, título e data de aquisição. Para os livros é importante manter também, o nome da editora e o ano de publicação. Já para os CDs, é interessante manter o gênero. Para os DVDs é importante armazenar o tipo (musical, filme ou dados). Por fim, das revistas é interessante manter o ano de publicação, o volume e a editora. Desenvolva um modelo de classes (diagrama de classe) que melhor represente as informações acima, utilizando os conceitos da programação orientada a objetos. Implemente as classes e testes-as, criando, ao menos, um objeto de cada tipo.

7. Implemente a classe Voo em que cada objeto representa um vôo que acontece em determinada data e em determinado horário. Cada vôo possui no máximo 100 passageiros, e a classe permite controlar a ocupação das vagas. A classe deve ter os seguintes métodos:

construtor	configura os dados do vôo (número do vôo e data – objeto da classe data)
proximoLivre	retorna o número da próxima cadeira livre
verifica	verifica se a cadeira, cujo número foi recebido como parâmetro, está ocupada
ocupa	ocupa determinada cadeira do vôo, cujo número é recebido como parâmetro, e retorna verdadeiro se a operação foi bem sucedida e falso caso contrário
vagas	retorna o número de cadeiras vagas disponíveis (não ocupadas) no vôo

getVoo	retorna o número do vôo
getData	retorna a data do vôo (na forma de objeto)
clone	o objeto clona a si próprio (idem método clone da classe Data)

Implemente a classe Principal para testar todos os métodos da classe Voo.

8. Escreva uma classe VooMarcado herdeira da classe Voo (questão anterior), que permita definir quantas cadeiras existem no máximo no voo e permita também dividir o avião em ala de fumantes e não fumantes. Para isto, esta classe deve acrescentar os atributos necessários e adicionar os seguintes métodos:

construtor	além dos parâmetros recebidos pelo construtor da superclasse, receberá também como parâmetros o número de vagas do voo e quantas cadeiras serão destinadas para fumantes
maxVagas	determina o número máximo de cadeiras no voo
cadeirasFumantes	determina quantas cadeiras estão destinadas aos fumantes (as demais serão automaticamente destinadas aos não fumantes); as cadeiras dos fumantes serão sempre as últimas do avião

tipo	recebe como parâmetro o número da cadeira e retorna ‘F’ se for uma cadeira para fumantes e ‘N’ se for para não fumantes

Os métodos *proximoLivre*, *verifica* e *ocupa* da superclasse devem ser adaptados para tratar o número máximo de vagas informado, ao invés do número fixo de 100. Implemente a classe Principal para testar todos os métodos das classes Voo e VooMarcado.