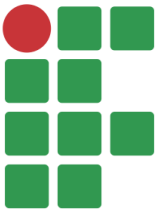


INSTITUTO FEDERAL DA PARAÍBA
CAMPUS CAMPINA GRANDE
BACHARELADO EM ENGENHARIA DA COMPUTAÇÃO
DISCIPLINA DE POO e LAB. POO
PROF. VICTOR ANDRÉ PINHO DE OLIVEIRA

Atividade Unidade II - 4 - Herança

Lista Única

Instruções

Responda às questões abaixo. Pode usar este próprio documento. Questões práticas devem ser anexadas separadamente.

As primeiras 5 questões valem 1. As questões 6 a 11 valem 2.

Questões

1. O que é herança e quais os benefícios que ela traz para o desenvolvimento de software?

   R: A herança é uma forma de reaproveitamento de código, em que você pode criar uma classe com características iguais a uma outra, e a partir disso, adicionar novas.

2. Qual a diferença entre classe básica e classe derivada?

   R: A classe básica é a primeira classe de uma herança, de onde todas as outras serão derivadas. Ela, geralmente, vai possuir os métodos mais primordiais das classes envolvidas. Por sua vez, classes derivadas são aquelas que herdam características das classes básicas.

3. Qual a diferença entre uma classe básica direta e indireta?

   R: Uma classe básica direta é quando uma classe herda diretamente a classe básica, indireta é quando herda uma classe que já herdou uma classe básica, isso em um, ou mais níveis.

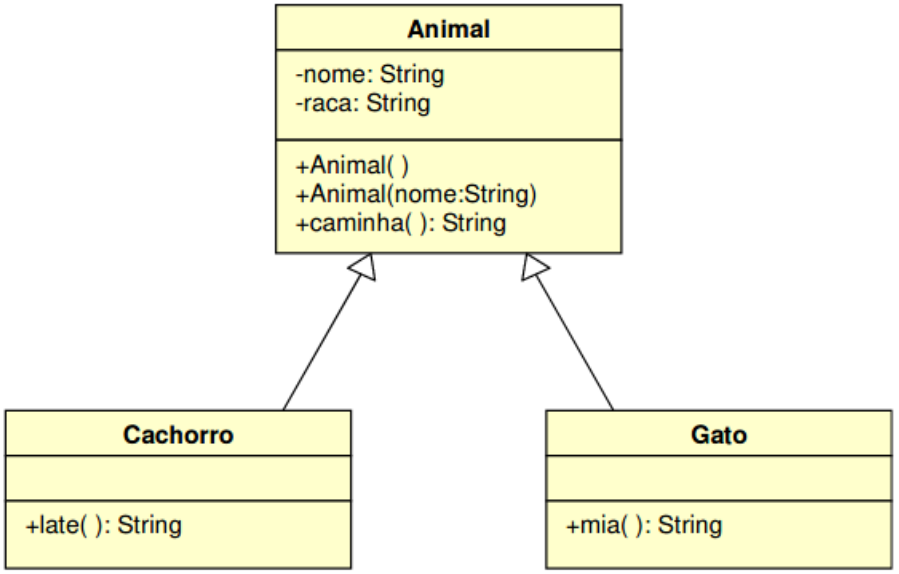
4. Em termos de relacionamento entre classes, qual a diferença entre a composição e a herança?

   R: Uma composição é um relacionamento "tem um" e herança é um relacionamento "é um", uma herança cria classes filhas, já a composição faz com que classes usem recursos de outras classes, sem que elas tenham, obrigatoriamente, algo em comum.

5. Como o modificador protected se diferencia dos modificadores public e private? Em que contexto o protected deve ser usado?

   R: O modificador public permite acesso a qualquer código externo a classe e o private proíbe qualquer acesso externo à própria classe, inclusive das classes filhas. Já o protected fornece um acesso intermediário, onde ele permite acesso às classes filhas. Deve ser usado protected quando você deseja que somente as classes filhas tenham acesso a classe.

6. Implemente o diagrama de classes abaixo:



7. Crie uma classe Imovel que possui como atributos um endereço e um preço. Forneça métodos get e set para esses atributos. Em seguida, crie uma classe ImovelNovo que herda Imovel e possui um adicional no preço. Forneça um método get e um método set para esse atributo adicional. Crie também uma classe ImovelVelho que herda Imovel e possui um desconto no preço. Forneça um método get e um método set para esse atributo desconto. As classes ImovelNovo e ImovelVelho devem fornecer também um método getPreco que sobrescreve o método getPreco da classe básica Imovel e considera, respectivamente, o adicional ou desconto. Mostre que suas classes estão funcionando corretamente por meio de exemplos na função main.

8. Crie uma classe chamada Pessoa que tenha como atributo protected o nome da pessoa. Em seguida, crie duas outras classes chamadas PessoaFisica e PessoaJuridica que herdam da classe Pessoa. A classe PessoaFisica terá como atributos privados o CPF e o nome da pessoa, enquanto a classe PessoaJuridica terá como atributos privados o CNPJ, a razão social e o nome fantasia. Note que o atributo nome da PessoaFisica e nome fantasia da PessoaJuridica são herdados de Pessoa, isto é, é o atributo nome de Pessoa. Crie métodos get e set para todos os atributos das três classes. A classe básica Pessoa deve ter uma função amiga que

sobrecarrega o operador << para imprimir o nome da pessoa na tela. Mostre que suas classes estão funcionando corretamente por meio de exemplos na função main.

9. Crie uma classe chamada Funcionario que herde da classe PessoaFisica da questão 8. Essa classe deverá ter como atributos privados a matrícula, o salário base do funcionário, a carga horária mensal (quantidade de horas mensais) e a quantidade de horas trabalhadas no mês. Além disso, a classe terá um método público chamado calculaSalarioBruto que não terá nenhum parâmetro e deverá ser capaz de calcular e retornar o salário bruto através da seguinte equação: salarioBase * quantidadeHorasTrabalhadas/cargaHorariaMensal. Por fim, crie métodos get e set para os atributos. Note que a quantidade de horas trabalhadas não poderá superar a carga horária mensal e nem ser inferior a 0. Garanta isso dentro da classe. A classe Funcionario deve ter uma função amiga que sobrecarrega o operador << para imprimir os dados do funcionário na tela. Mostre que suas classes estão funcionando corretamente por meio de exemplos na função main.

10. Crie uma classe chamada Cliente que herde da classe PessoaFisica da questão 8. Essa classe deverá ter atributos privados que armazenem um telefone e um endereço. Crie métodos get e set para esses atributos. A classe Cliente deve ter uma função amiga que sobrecarrega o operador << para imprimir os dados do cliente na tela. Mostre que suas classes estão funcionando corretamente por meio de exemplos na função main.

11. Crie uma classe chamada Empresa que herde da classe PessoaJuridica da questão 8. Essa classe deverá ter uma lista de funcionários e uma outra lista de clientes (pode ser array de tamanho fixo). Crie métodos para adicionar funcionários e clientes (não precisa se preocupar em excluir). Crie um método para imprimir os funcionários e outro para imprimir os clientes. Crie também um método chamado calcularFolhaDePagamento que deverá calcular o salário bruto de todos os funcionários e retornar o total a ser gasto com os funcionários. Mostre que suas classes estão funcionando corretamente por meio de exemplos na função main.