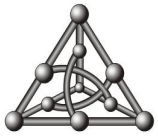
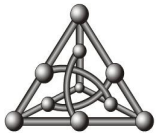


UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
Faculdade de Computação

Orientações iniciais:

- A prova é individual e sem consulta;
- A prova contém uma questão prática valendo **100 pontos**, que equivalem à nota **10**;
- A prova tem duração de mínima de **40 minutos** e duração máxima de **1 hora e 40 minutos**;
- Utilize os conceitos de encapsulamento em todos os exercícios;
- Utilize a IDE Eclipse para implementar a prova;
- Crie uma pasta com o seu nome na sua Área de Trabalho e armazene o projeto Eclipse dentro dela.
- O acesso à documentação do Java está disponível em *http://docs.oracle.com*.
- Caso você tenha alguma dúvida referente ao enunciado das questões durante a prova, pergunte em voz alta. O professor não irá atendê-lo individualmente;

BOA PROVA!



UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
Faculdade de Computação

Prova Optativa

Linguagem de Programação Orientada a Objetos — 2014

Turma P02

1. **[30 pontos]** O objetivo dessa questão é analisar a sua capacidade de abstrair uma situação do mundo real com os conceitos de orientação a objetos. Para isso, considere o contexto descrito a seguir:

   Os impostos dos Brasil são arrecadados com base no tipo de contribuinte: pessoa física ou pessoa jurídica. Independente do tipo de pessoa (física ou jurídica), todos são contribuintes do imposto de renda e devem ter as seguintes informações armazenadas: nome e renda bruta. Além disso, todo contribuinte deve ter a capacidade de calcular imposto, mas a maneira como esse imposto é calculado depende do tipo de contribuinte. Para pessoa jurídica, o imposto deve corresponder a 10% da renda bruta da empresa. Já para pessoa física, o imposto deve ser calculado de acordo com a seguinte tabela:

Renda Bruta	Alíquota	Parcela a Deduzir
R$0,00 a R$ 1.400,00	0%	R$0,00
R$1.400,01 a R$ 2.100,00	10%	R$100,00
R$2.100,01 a R$ 2.800,00	15%	R$270,00
R$2.800,01 a R$ 3.600,00	25%	R$500,00
R$3.600,01 ou mais	30%	R$500,00

   A coluna Parcela a Deduzir representa o valor que deve ser subtraído do imposto após o cálculo realizado com a alíquota.

   Deve ser possível registrar, de maneira , o valor total de impostos que já foram calculados de todos os contribuintes.

   Modele o problema acima utilizando o máximo de conceitos de orientação a objetos vistos no decorrer da disciplina. Logo após, crie uma classe chamada TesteImposto, que testa todas as classes criadas utilizando polimorfismo.

2. **[70 pontos]** O objetivo desta questão é resolver um problema utilizando interface gráfica juntamente com os conceitos de classes genéricas e tratamento de exceções.

   (a) **[35 pontos]** Implemente uma classe genérica *MyArrayList<T>* capaz de oferecer uma série de métodos que gerenciam uma coleção de elementos de um tipo T de tamanho dinâmico, utilizando um vetor de elementos do tipo T. Para isso, crie as seguintes classes:

      i. Interface genérica *MyList<T>*, que fornece uma assinatura para os seguintes métodos:
         A. size, que retorna o tamanho da lista;
         B. isEmpty, que responde se a lista está vazia;
         C. add, que adiciona no final da lista um elemento do tipo T recebido por parâmetro. Em todas as implementações dessa interface, uma lista deve conter no máximo 10 elementos. Quando esse método for invocado e a lista já contiver 10 elementos, a exceção verificada ListFullException deve ser lançada;
         D. get, que retorna um elemento do tipo T que está em uma determinada posição $index$ recebida por parâmetro. Caso o índice solicitado não esteja no intervalo de elementos válidos da lista, a implementação desse método deve lançar e informar explicitamente o lançamento da exceção não verificada ArrayIndexOutOfBoundsException;
         E. clear, que limpa a lista;
         F. remove, que remove da lista um elemento x recebido por parâmetro. Caso o elemento não esteja na lista, a implementação desse método deve lançar a exceção verificada ElementNotFoundException.

      ii. Classe genérica *MyArrayList<T>* que implementa a interface *MyList<T>*. A principal restrição dessa classe é que ela deve aceitar apenas elementos que sejam comparáveis.

   (b) **[35 pontos]** Implemente uma aplicação Java com uma interface gráfica capaz demonstrar que a classe genérica MyArrayList<T> funciona corretamente. Para isso, sua interface deve ter as seguintes características:

      i. A janela principal deve ser implementada utilizando um JFrame e um menu Float, que deve conter os seguintes subitens: *Adicionar*, *Remover*, e *Listar*, que irão manipular os elementos de um MyArrayList de Floats;

      ii. Quando o usuário selecionar o subitem Adicionar do menu Float, uma JOptionPane deve ser exibida no centro da aplicação, pedindo para que o usuário digite um valor Float. Adicione esse valor no final da instância de MyArrayList de Float. Caso haja alguma exceção durante a inserção, você deve exibir um JOptionPane no centro da tela, informando ao usuário qual foi o problema. Caso contrário, apenas insira o elemento no final da instância de MyArrayList;

      iii. Quando o usuário selecionar o subitem Remover do menu Float, uma JOptionPane deve ser exibida no centro da aplicação, pedindo para que o usuário digite o valor Float que deve ser removido. Se esse elemento existir, remova-o. Caso haja

alguma exceção durante a remoção, você deve exibir um JOptionPane no centro da tela, informando qual foi o problema;

iv. Quando o usuário selecionar o subitem Listar coleção do menu Float, liste os elementos presentes na instância de MyArrayList de Float, em qualquer posição da tela.

v. As exceções devem ser tratadas separadamente;

vi. Deve existir apenas um objeto handler que manipula os eventos dos três itens de menu existentes.