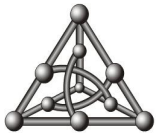
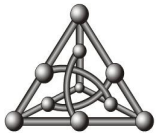


UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
Faculdade de Computação

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;
- 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.
- Crie uma pasta com o seu nome na sua Área de Trabalho e armazene todas as implementações da prova dentro dessa pasta. Você pode utilizar um editor de texto comum ou o Eclipse. Crie dentro dessa pasta um arquivo chamado *README.txt*, que contém um descritivo de como você organizou os exercícios (1 projeto no eclipse, vários projetos no eclipse, utilizou editor de texto comum, etc.).

BOA PROVA!



UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
Faculdade de Computação

Prova 2

Linguagem de Programação Orientada a Objetos — 2014

Turma P02

1. **[100 pontos]** O objetivo desta questão é resolver um problema utilizando interface gráfica juntamente com os conceitos de classes genéricas e tratamento de exceções.

   (a) **[50 pontos]** Implemente uma classe genérica *MyLinkedList<T>* capaz de oferecer uma série de métodos que gerenciam uma coleção de elementos de um tipo T de tamanho dinâmico, utilizando uma lista encadeada. 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 a exceção não verificada ArrayIndexOutOfBoundsException;
         E. clear, que limpa a lista;
         F. remove, que remove um elemento da lista em uma determinada posição $index$ recebida por parâmetro. Caso o índice solicitado não esteja no intervalo de elementos da lista, a implementação desse método deve lançar a exceção não verificada ArrayIndexOutOfBoundsException.

      ii. Classe genérica *Node<T>*, que modela um elemento do tipo T de uma lista encadeada. Essa classe deve armazenar um elemento do tipo T e a referência do próximo *Node<T>*;

      iii. Classe genérica *MyLinkedList<T>* que implementa a interface *MyList<T>* utilizando lista encadeada. Para isso, sua classe deve conter pelo menos o endereço do primeiro *Node<T>* e do último *Node<T>* da lista.

(b) **[50 pontos]** Implemente uma aplicação Java com uma interface gráfica capaz demonstrar que a classe genérica MyLinkedList<T> funciona para classes T quaisquer. Para isso, sua interface deve ter as seguintes características:

i. A janela principal deve ser implementada utilizando um JFrame um menu Integer, que deve conter os seguintes subitens: *Adicionar elemento no final*, *Alterar valor de uma posição*, *Consultar valor de uma posição*, *Remover elemento de uma posição*, *Limpar coleção* e *Mostrar tamanho*, que irão manipular os elementos de uma MyLinkedList de Integers;

ii. A janela principal deve ser organizada da seguinte maneira:
   A. Na região norte você deve exibir o conteúdo atual do MyLinkedList de Integers;
   B. Na região sul você deve exibir o conteúdo atual do MyLinkedList de Strings;
   C. As regiões central, leste e oeste estarão vazias.

iii. Quando o usuário selecionar o subitem Adicionar elemento no final do menu Integer, uma JOptionPane deve ser exibida no centro da aplicação, pedindo para que o usuário digite um valor inteiro. Adicione esse valor no final do MyLinkedList de Integer. 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, você deve atualizar a região norte da tela com os novos valores do vetor;

iv. Quando o usuário selecionar o subitem Consultar valor de uma posição do menu Integer, uma JOptionPane deve ser exibida no centro da aplicação, pedindo para que o usuário digite o índice da posição a ser consultada. Caso haja alguma exceção, você deve exibir um JOptionPane no centro da tela, informando ao usuário qual foi o problema. Caso contrário, você deve exibir um JOptionPane no centro da tela contendo o valor dessa posição no MyLinkedList de Integer;

v. Quando o usuário selecionar o subitem Remover elemento de uma posição do menu Integer, uma JOptionPane deve ser exibida no centro da aplicação, pedindo para que o usuário digite o índice da posição a ser removida. Remova o elemento que está nessa posição do MyLinkedList de Integer. Caso haja alguma exceção durante a remoção, você deve exibir um JOptionPane no centro da tela, informando qual foi o problema. Caso o elemento tenha sido removido com sucesso, você deve atualizar a região norte da tela com os novos valores do vetor;

vi. Quando o usuário selecionar o subitem Limpar coleção do menu Integer, remova todos os elementos da MyLinkedList de Integer. Caso haja alguma exceção durante a limpeza, você deve exibir um JOptionPane no centro da tela, informando qual foi o problema. Caso a limpeza tenha acontecido com sucesso, você deve atualizar a região norte da tela;

vii. As exceções devem ser tratadas separadamente.

Dicas:

- As classes necessárias para a implementação do menu são: JMenuBar, JMenu e JMenuItem;
- O evento de clique em um subitem de menu pode ser feito adicionando um *listener* que implementa a interface *ActionListener*;
- Para a exibição dos itens de cada uma das MyLinkedLists nas regiões norte e sul, você pode utilizar um componente de sua preferência (JLabel, JTextArea, etc.. Todos esses componentes possuem um método repaint, que pode ser útil durante a implementação;
- Para adicionar uma janela JOptionPane no frame, basta passar uma referência do frame atual como o primeiro parâmetro dos métodos estáticos da JOptionPane.