Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
Disciplina: Programação III
AP3 1° semestre de 2009.

Nome –

Assinatura –

Observações:

1. Prova sem consulta e sem uso de máquina de calcular.
2. Use caneta para preencher o seu nome e assinar nas folhas de questões e nas folhas de respostas.
3. Você pode usar lápis para responder as questões.
4. Ao final da prova devolva as folhas de questões e as de respostas.
5. Todas as respostas devem ser transcritas nas folhas de respostas. As respostas nas folhas de questões não serão corrigidas.

Questão 1) (2.5 pontos)
Suponha o trecho de código abaixo que apresenta uma classe simples que modela uma conta bancária.

```
1  class ContaCorrente {
2     int conta;
3     float saldo;
4     public ContaCorrente(int pConta, float pSaldo) {
5           this.conta = pConta; this.saldo = pSaldo;
6     }
7     public float obtemSaldo() {
8           return saldo;
9     }
10    public void realizaDeposito(float valor) {
11          this.saldo = this.saldo + valor;
12    }
13    public void realizaSaque(float valor) {
14          this.saldo = this.saldo - saldo;
15    }
16 }
17
18 public class AP3_2009_1_Q1 {
19    public static void main(String[] args) {
20          ContaCorrente c = new ContaCorrente(1, 500);
21          c.realizaSaque(1000);
22    }
23 }
```

Na linha 21 é executada uma suposta operação de saque de 1000 dinheiros. Entretanto, na linha 20, a conta é criada com saldo inicial de 500 dinheiros. Ou seja, essa é uma situação excepcional que deve ser tratada. Para tal, criamos a classe abaixo a qual pode ser utilizada em 2 situações opcionais na classe ContaCorrente: 1) saque acima do saldo existente; 2) depósito de valor negativo.

```
class OperacaoIlegal extends Exception {
      enum TipoOperacao {saque, deposito};
      ContaCorrente cc;
      float valor;
      TipoOperacao operacao;
      public OperacaoIlegal(ContaCorrente contaCorrente, float valor,
TipoOperacao op) {
            this.cc = contaCorrente;
            this.valor = valor;
            this.operacao = op;
      }
}
```

RESPOSTA:

a) Altere a classe ContaCorrente de forma que exceções do tipo OperacaoIlegal sejam lançadas.

```
class OperacaoIlegal extends Exception {
      enum TipoOperacao {saque, deposito};
      ContaCorrente cc;
      float valor;
      TipoOperacao operacao;
      public OperacaoIlegal(ContaCorrente contaCorrente, float valor,
TipoOperacao op) {
            this.cc = contaCorrente;
            this.valor = valor;
            this.operacao = op;
      }
}

class ContaCorrente {
      int conta;
      float saldo;
      public ContaCorrente(int pConta, float pSaldo) {
            this.conta = pConta; this.saldo = pSaldo;
      }
      public float obtemSaldo() {
            return saldo;
      }
      public void realizaDeposito(float valor) throws OperacaoIlegal {
            if (valor < 0)
                  throw new OperacaoIlegal(this, valor,
OperacaoIlegal.TipoOperacao.deposito);
            else
                  this.saldo = this.saldo + valor;
      }
```

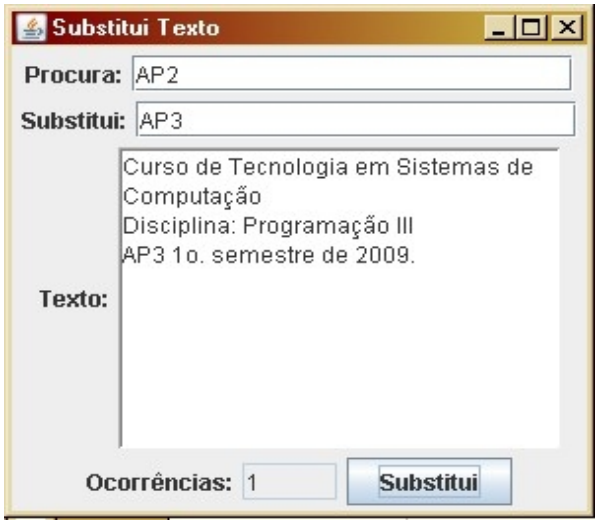
```
	public void realizaSaque(float valor) throws OperacaoIlegal {
		if (valor >= this.saldo)
			throw new OperacaoIlegal(this, valor,
OperacaoIlegal.TipoOperacao.saque);
		else
			this.saldo = this.saldo - saldo;
	}
}

public class AP3_2009_1_Q1 {
	public static void main(String[] args) {
		ContaCorrente c = new ContaCorrente(1, 500);
		try {
			c.realizaSaque(1000);
		} catch (OperacaoIlegal e) {
			System.out.println("Operação de " + e.operacao + " com
valor " + e.valor + " inválida para a conta " + e.cc.conta + "!");
		}
	}
}
```

b) É necessário modificar algo no método main() entre as linhas 19 e 22? Justifique sua resposta.

Como ilustrado no código acima, sim. O método realizaSaque() agora pode levantar uma exceção, o que obriga que o programador trate a possível exceção (bloco try-catch) em "tempo de compilação", ou seja, antes da execução do programa.

Questão 2) (2.5 pontos)
Escreva um programa que crie a janela abaixo:



Ao clicar no botão Substitui, **todas** as ocorrências da primeira palavra (no exemplo da figura, "AP2") devem ser substituídas pela segunda palavra (no exemplo, "AP3") na caixa de texto. O campo "Ocorrências" deve ser atualizado com a quantidade de substituições necessárias.

Dicas: 1) A caixa de texto pode ser uma instância da classe JTextArea da interface Swing. 2) O número de substituições é o número de ocorrências da primeira palavra.

RESPOSTA:

```
import java.awt.*;
import java.awt.event.*;

import javax.swing.*;
import javax.swing.text.BadLocationException;

/*
 Classe que modela a janela principal e seus componentes
 Esta classe implementa a interface ActionListener, a qual
 permite que esta classe trate as ações disparadas pelo
 usuário na janela criada.
 */
class JTexto2 implements ActionListener {
      JFrame frame = new JFrame("Substitui Texto");
      JLabel procura = new JLabel("Procura:");
      JTextField tf = new JTextField(20);
      JLabel substitui = new JLabel("Substitui:");
      JTextField tf2 = new JTextField(20);
      JLabel texto = new JLabel("Texto:");
      JTextArea caixaTexto = new JTextArea(10, 20);
      JButton bt = new JButton("Substitui");
      JLabel qtd = new JLabel("Ocorrências:");
      JTextField tf3 = new JTextField(4);

      /*
       Construtor da classe da janela principal, a qual cria
       os componentes visuais e os inicializa.
       */
      public JTexto2() {
            tf.setEditable(true);
            tf.addActionListener(this);
            bt.addActionListener(this);
            Container c = frame.getContentPane();
            c.setLayout(new FlowLayout(FlowLayout.CENTER,5,5));

      caixaTexto.setBorder(BorderFactory.createLoweredBevelBorder());
            c.add(procura); c.add(tf);
            c.add(substitui); c.add(tf2);
            c.add(texto); c.add(caixaTexto);
            c.add(qtd); c.add(tf3); tf3.setEditable(false);
            c.add(bt);
            frame.setSize(300, 280);
            frame.setVisible(true);
      }

      /*
       Método que trata as ações disparadas pelo usuário. Neste
       caso, apenas o clique no botão para substituição das
       palavras no texto.
       */
```

```
	public void actionPerformed(ActionEvent e) {
		Object o = e.getSource();
		if (o == bt) {
			String texto = null;
			try {
				texto = caixaTexto.getDocument().getText(0,
caixaTexto.getDocument().getLength());
			} catch (BadLocationException e1) {
				System.out.println("Erro na manipulação da caixa
de texto!");
			}
			// Se as caixas de texto não estão vazias
			if (tf.getText().length() > 0 &&
tf2.getText().length() > 0)
			{
				// Cálculo da quantidade de ocorrências
				String ocorrencias[] =
texto.split(tf.getText());
				tf3.setText(String.valueOf(ocorrencias.length -
1));
				// Substituição do texto
				String textoNovo =
texto.replaceAll(tf.getText(), tf2.getText());
				caixaTexto.replaceRange(textoNovo, 0,
caixaTexto.getDocument().getLength());
			}
		}
	}
}

/*
 Classe principal que inicia a janela da aplicação
 */
public class AP3_2009_1_Q2 {
	public static void main(String[] args) {
		new JTexto2();
	}
}
```

Questão 3) (2.5 pontos)
Implemente o tipo abstrato de dados lista genérica em JAVA. Só é necessário apresentar a definição do tipo e o cabeçalho dos métodos de construção, verificação de lista vazia, inclusão e exclusão de elemento (não é preciso codificar os métodos da lista). Justifique a sua implementação, enfocando o modo como se obtém a generalidade da lista.

Questão 4) (2.5 pontos)
Considere uma aplicação que tenha por objetivo gerar um relatório de disciplinas cursadas pelos alunos. Neste relatório, para cada disciplina existente, deve-se informar a maior média entre os alunos que cursaram a disciplina e o nome deste aluno. As disciplinas serão apresentadas no relatório de saída em ordem decrescente de média. O dado de entrada é um arquivo texto que registra cada disciplina cursada por aluno, com o seu nome e sua respectiva média obtida. Um exemplo de um arquivo de entrada é mostrado a seguir:

```
INF1001/Fulano das Couves/7.3
INF1620/Sicrano da Silva/6.7
INF1620/Beltrano Raimundo/8.4
INF1001/Sicrano da Silva/8.7
INF1620/ Fulano das Couves/7.2
```

Escreva um programa que leia o arquivo de entrada, gere um arquivo de saída, cujo nome é "saida-maior-" acrescido do nome do arquivo de entrada, com as informações agrupadas por disciplina. No exemplo acima, o arquivo de saída seria:

```
INF1001     Maior nota: 8.7   Nome: Sicrano da Silva
INF1620     Maior nota: 8.4   Nome: Beltrano Raimundo
```

Um exemplo de uso desse programa seria java calcMediaDisc notas.txt, onde notas.txt é o nome do arquivo de entrada.