



Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação

Disciplina: Programação III

AP3 2º semestre de 2010.

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) (4 pontos)

Supondo que você tenha uma empresa de software, e que a Associação Mundial de Tênis contrate sua empresa para escrever um programa que informe, automaticamente, o ranking dos jogadores de tênis. Este ranking segue o sistema de pontos corridos. Seu software deve contabilizar, somente, os quatro grand slams (Roland Garros – RG, Austrália Open – AO, Wimbledon – WI e US Open – US). A pontuação destes torneios é a seguinte:

Colocação	Sigla	Pontos
Vencedor	WIN	2000
Finalista	FIN	1200
Semifinalista	SF	720
Quartas-de-final	QF	360
Oitavas-de-final	OF	180
Terceira rodada	TR	90
Segunda rodada	SR	45
Primeira rodada	FR	10

O dado de entrada é um arquivo texto cujo o conteúdo é formado pelo número de jogadores e suas posições nestes torneios nos dois últimos anos (neste caso os anos de 2009 e 2010, respectivamente). Para o seguinte exemplo de arquivo:

4

Roger Federer/RG WIN SF/AO FIN WIN/WI WIN SF/US FIN SF
Rafael Nadal/RG SF WIN/AO WIN SF/WI QF WIN/US SF WIN
Novak Djokovic/RG SF SF/AO QF QF/WI QF FIN/US SF FIN
Andy Murray/RG SF OF/AO SF FIN/WI TR TR/US SR SR

O seu software deve informar o seguinte ranking:

- 1 Rafael Nadal (de 3800 para 6720): subiu 1 posicao
- 2 Roger Federer (de 6400 para 4160): desceu 1 posicao
- 3 Novak Djokovic (de 2160 para 3480): inalterada posicao
- 4 Andy Murray (de 1575 para 1515): inalterada posicao

Resposta:

```
import java.io.*;
class Rank{
    int posicao;
    int npontos;
    Rank(int np){
        npontos = np;
        posicao = 0;
    }
    void modificaPontos(int np){ npontos += np; }
    public String toString(){
        String resp = " " + npontos;
        return resp;
    }
}

class Jogador{
    String nome;
    Rank anterior, atual;
    Jogador(String n){
        nome = n;
        anterior = new Rank(0);
        atual = new Rank(0);
    }
    void incluiPontos(int np, boolean rank){
        if(rank) atual.modificaPontos(np);
        else anterior.modificaPontos(np);
    }
    public String toString(){
        String resp = nome + "\t (de" + anterior.toString() + " para" +
atual.toString() + "): ";
        return resp;
    }
}

class WTA{
    Jogador[] tabela;
    WTA(int num){ tabela = new Jogador[num]; }
    void inclui(int ind, Jogador j){
        if(ind < tabela.length) tabela[ind] = j;
    }
}
```

```

void ordena(){
    int i, j;
    Jogador temp;
    for(i = 0; i < tabela.length; i++)
        for(j = (i + 1); j < tabela.length; j++)
            if(tabela[i].anterior.npontos < tabela[j].anterior.npontos){
                temp = tabela[i];
                tabela[i] = tabela[j];
                tabela[j] = temp;
            }
    for(i = 0; i < tabela.length; i++) tabela[i].anterior.posicao = (i
+ 1);
    for(i = 0; i < tabela.length; i++)
        for(j = (i + 1); j < tabela.length; j++)
            if(tabela[i].atual.npontos < tabela[j].atual.npontos){
                temp = tabela[i];
                tabela[i] = tabela[j];
                tabela[j] = temp;
            }
    for(i = 0; i < tabela.length; i++) tabela[i].atual.posicao = (i +
1);
}
}
public String toString(){
    String resp = "";
    for(int i = 0; i < tabela.length; i++){
        resp += (i + 1) + "\t" + tabela[i].toString();
        int aux = tabela[i].atual.posicao - tabela[i].anterior.posicao;
        if(aux == 0) resp += "inalterada posicao\n";
        else if(aux > 0) resp += "desceu " + aux + " posicao\n";
        else resp += "subiu " + Math.abs(aux) + " posicao\n";
    }
    return resp;
}
}

public class Teste_Q1{
    public static void main(String[] args) throws IOException {
        int n, i, j, atual, ant;
        BufferedReader in = new BufferedReader(new FileReader(args[0]));
        try{
            String linha;
            linha = in.readLine();
            n = Integer.parseInt(linha);
            WTA tenis = new WTA(n);
            String[] partes;
            for(i = 0; i < n; i++){
                linha = in.readLine();
                partes = linha.split("/");
                Jogador jog = new Jogador(partes[0]);
                for(j = 1; j <= 4; j++){
                    String pos[] = partes[j].split(" ");
                    int k, pontos[] = new int [2];
                    for(k = 0; k < 2; k++){
                        if(pos[k + 1].equals("WIN")) pontos[k] = 2000;
                        else if(pos[k + 1].equals("FIN")) pontos[k] = 1200;
                        else if(pos[k + 1].equals("SF")) pontos[k] = 720;
                    }
                }
            }
        }
    }
}

```

```

        else if(pos[k + 1].equals("QF")) pontos[k] = 360;
        else if(pos[k + 1].equals("OF")) pontos[k] = 180;
        else if(pos[k + 1].equals("TR")) pontos[k] = 90;
        else if(pos[k + 1].equals("SR")) pontos[k] = 45;
        else pontos[k] = 10;
    }
    jog.incluiPontos(pontos[0], false);
    jog.incluiPontos(pontos[1], true);
}
tenis.inclui(i, jog);
}
in.close();
tenis.ordena();
System.out.println(tenis);
}catch(Exception e){ System.out.println(e); }
}
}

```

Questão 2) (3 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:

```

/*
Classe principal que inicia a janela da aplicação
*/
public class AP3_2009_1_Q2 {
    public static void main(String[] args) {
        new JText2();
    }
}

```

```

/*
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) {
                e1.printStackTrace();
            }
            // 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());
            }
        }
    }
}

```

```
    }  
}  ocorrencias = null;  
  
}
```

Questão 3) (3 pontos)

Defina um Dicionário através de uma interface Java. Um dicionário, também chamado de tabela associativa, é um TAD (Tipo Abstrato de Dados) que permite a armazenagem de valores associados a chaves (você pode encarar um dicionário como um array que é indexado por chaves ao invés de números). Projete seu dicionário para usar objetos quaisquer como chaves e valores.

Note que a forma de se obter um valor do dicionário é fazer uma consulta através da chave que foi usada para armazenar esse valor. Além dessa consulta normal, por chave, um dicionário deve também permitir que todos os pares (chave/valor) existentes sejam obtidos, um a um. Dessa forma é possível descobrir todas as informações contidas em um dicionário, mesmo sem conhecer as chaves. Definam classes e interfaces para que o programa abaixo funcione.

```
public class Teste {
    public static void main(String[] args) {
        InterfaceDicionario d = new Dicionario();
        d.insere("João", "512-1313");
        d.insere("Maria", "512-2299");
        System.out.println("O telefone de João é:
"+d.consulta("João"));
        System.out.println("O telefone de Maria é:
"+d.consulta("Maria"));
    }
}
```

Resposta:

```
interface InterfaceDicionario {
    void insere(String string, String string2);
    String consulta(String string);
}

class Dicionario implements InterfaceDicionario {
    Map<String, String> dicionario = new HashMap<String, String>();
    public String consulta(String string) {
        return (String)dicionario.get(string);
    }
    public void insere(String string, String string2) {
        dicionario.put(string, string2);
    }
}
```