Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
AD2 de Programação III
1° semestre de 2016

Exercício (ENTREGAR OS ARQUIVOS EM MÍDIA, PARA FINS DE TESTE, JUNTAMENTE COM A AD IMPRESSA):
Considere que sua empresa seja contratada pela Superliga de Vôlei Nacional para resolver o problema da divulgação dos resultados das competições que ela mantém.

Sua empresa deve desenvolver um algoritmo que, dados um arquivo de entrada (**LIDO SOMENTE UMA VEZ PELO SEU PROGRAMA**), com todos os jogos de algum turno da Superliga, e o número de times que devem ser classificados para a próxima fase (**N**), retorne os **N** times da próxima fase em um arquivo de saída bem especificado. Sabe-se, analisando as regulamentações da CBV que, em caso de:

- VITÓRIA ou por 3x0, ou por 3x1 – o vencedor ganha três pontos;
- DERROTA ou por 0x3, ou por 1x3 – o perdedor não ganha nenhum ponto;
- VITÓRIA por 3x2 – o vencedor ganha dois pontos; e
- DERROTA por 2x3 – o perdedor ganha um ponto.

Se times estiverem empatados em pontuação, o critério de desempate é o número de vitórias. Se eles também estiverem empatados em número de vitórias, o critério de desempate será o saldo de pontos em sets, isto é, a diferença entre os todos os pontos feitos por uma quipe e todos os pontos que os outros times fizeram nela.

O seu programa deve retornar, no arquivo de saída **"out-"** acrescido do nome de arquivo de entrada, os **N** times da próxima fase.

Por exemplo, dados os resultados de um turno deste ano (Liga 2015-2016), cujo o nome é **result-fase2-2016.txt**:

```
VOLEI NESTLE vs REXONA-ADES/3-2/25-22/14-25/26-24/19-25/15-10
DENTIL-PRAIA CLUBE vs CAMPONESA/MINAS/3-2/22-25/25-15/21-25/25-22/15-12
REXONA-ADES vs VOLEI NESTLE/3-1/21-25/25-22/25-23/25-16
CAMPONESA-MINAS vs DENTIL-PRAIA CLUBE/3-1/21-25/25-17 25-22 25-22
REXONA-ADES vs VOLEI NESTLE/3-0/25-20/25-23/25-16
DENTIL-PRAIA CLUBE vs CAMPONESA-MINAS/3-0/25-21/25-18/25-22
```

E seu programa for executado com a seguinte linha de comando **java AD2-2016 result-fase2-2016.txt 2**, seu programa deve retornar, no arquivo **out-result-fase2-2016.txt**, os seguintes times:

```
REXONA-ADES
DENTIL-PRAIA CLUBE
```

LEMBRE-SE: SEU PROGRAMA DEVE EXECUTAR COM QUAISQUER DADOS INFORMADOS COMO PARÂMETROS DE ENTRADA. SE O SEU PROGRAMA RESOLVER SOMENTE O PROBLEMA DO EXERCÍCIO SUPRACITADO, SUA QUESTÃO SERÁ TOTALMENTE DESCONTADA.

RESPOSTA:

```
import java.io.*;

class Time{
  String nome;
  int num_vit, pontos, ptos_favor, ptos_contra;

  Time(String n, int nv, int ptos, int ptosf, int ptosc){
    nome = n;
    num_vit = nv;
    pontos = ptos;
    ptos_favor = ptosf;
    ptos_contra = ptosc;
  }

  public String toString(){
    return nome + " " + pontos + " " + num_vit + " " + ptos_favor + " "
+ ptos_contra + "\n";
  }
}


class NoLista{
  Time time;
  NoLista prox;

  NoLista(Time t){
    time = t;
    prox = null;
  }
}


class ListaTimes{
  NoLista prim;

  ListaTimes (){ prim = null; }

  NoLista busca (String nome_time){
    NoLista p = prim;
    while((p != null) && (!p.time.nome.equals(nome_time))) p = p.prox;
    return p;
  }

  void insereTime(Time t){
    NoLista p = busca(t.nome);
    if(p == null){
      p = new NoLista(t);
      p.prox = prim;
      prim = p;
    }
```

```
    else{
      p.time.num_vit += t.num_vit;
      p.time.pontos += t.pontos;
      p.time.ptos_favor += t.ptos_favor;
      p.time.ptos_contra += t.ptos_contra;
    }
  }

   public String toString(){
     NoLista p = prim;
     String resp = "";
     while(p != null){
       resp += p.time.toString();
       p = p.prox;
     }
     return resp;
   }
 }


public class AD2_2016_1{
  public static void main(String[] args) throws IOException{
    FileReader in = new FileReader(args[0]);
    BufferedReader br = new BufferedReader(in);
    FileWriter out = new FileWriter("out-" + args[0]);
    BufferedWriter bw = new BufferedWriter(out);
    String s;
    ListaTimes l = new ListaTimes();

    try{
       s = br.readLine();
       while (s != null){
         String partes[] = s.split("/");
         String nomes[] = partes[0].split(" vs ");

         String sets[] = partes[1].split("-");
         int set_time1 = Integer.parseInt(sets[0]);
         int set_time2 = Integer.parseInt(sets[1]);
         int total = set_time1 + set_time2;
         int ptos_favor = 0, ptos_contra = 0;
         for (int i = 0; i < total; i++){
          String pontos[] = partes[i + 2].split("-");
          ptos_favor += Integer.parseInt(pontos[0]);
          ptos_contra += Integer.parseInt(pontos[1]);
         }

         if(set_time1 == 3){
           if(set_time2 <= 1){
                    l.insereTime(new  Time(nomes[0],  1,  3,  ptos_favor,
ptos_contra));
                   l.insereTime(new  Time(nomes[1],  0,  0,  ptos_contra,
ptos_favor));
            }
```

```
          else{
                    l.insereTime(new Time(nomes[0], 1, 2, ptos_favor,
ptos_contra));
                   l.insereTime(new Time(nomes[1], 0, 1, ptos_contra,
ptos_favor));
          }
        }
        else{
          if(set_time1 <= 1){
                    l.insereTime(new Time(nomes[0], 0, 0, ptos_favor,
ptos_contra));
                   l.insereTime(new Time(nomes[1], 1, 3, ptos_contra,
ptos_favor));
          }
          else{
                    l.insereTime(new Time(nomes[0], 0, 1, ptos_favor,
ptos_contra));
                   l.insereTime(new Time(nomes[1], 1, 2, ptos_contra,
ptos_favor));
          }
        }
        s = br.readLine();
      }

      System.out.println("Antes de ordena...\n" + l);
      ordena(l);
      System.out.println("Depois de ordena...\n" + l);
      out.write(imprime(l, Integer.parseInt(args[1])));
    }
    catch (Exception e) { System.out.println("ERRO"); }
    finally{ out.close(); }
  }


  static String imprime(ListaTimes l, int n){
    String resp = "";
    NoLista p = l.prim;
    for(int i = 0; i < n; i++){
      resp = resp + p.time.nome + "\n";
      p = p.prox;
    }
    return resp;
  }


  static void ordena (ListaTimes l){
    NoLista p;
    for(p = l.prim; p != null; p = p.prox){
      NoLista q, maior = p;
      for(q = p.prox; q != null; q = q.prox){
        if(maior.time.pontos < q.time.pontos) maior = q;
        else
          if((maior.time.pontos == q.time.pontos) && (maior.time.num_vit
< q.time.num_vit)) maior = q;
```

```
          else
                          if((maior.time.pontos  ==  q.time.pontos)  &&
(maior.time.num_vit  ==  q.time.num_vit)  &&  ((maior.time.ptos_favor  -
maior.time.ptos_contra)  <  (q.time.ptos_favor  -  q.time.ptos_contra)))
maior = q;
      }

      if(maior != p){
        Time temp = p.time;
        p.time = maior.time;
        maior.time = temp;
      }
    }
  }
}
```