Classificação no Brasileirão

Marco A L Barbosa malbarbo.pro.br

No Campeonato Brasileiro de Futebol, também chamado de Brasileirão, diversos times disputam o título de melhor time do Brasil.

Os times jogam todos contra todos em dois turnos, ida e volta, e vão acumulando pontos. Cada vitória gera três pontos, cada empate um ponto e a derrota não gera pontos. Os times são classificados de acordo com o seu desempenho, de forma que o time que tem melhor desempenho fica na frente do time que tem menos pontos. O desempenho é determinado primeiramente pelo número de pontos, se dois ou mais times tem o mesmo número de pontos, então, os seguintes critérios são utilizados para o desempate: número de vitórias, saldo de gols (número de gols feito menos o número de gols sofridos) e ordem alfabética (Sim, é injusto! Mas essa é uma simplificação...).

Projete uma função que receba como entrada uma lista de strings que descreve os resultados dos jogos e produza um lista de strings com a classificação dos times.

Cada string da lista de entrada tem o resultado de um jogo escrito da forma "Anfitrião Gols Visitante Gols" (os nomes dos times anfitrião e visitante não têm espaço). Por exemplo, a linha "Sao-Paulo 3 Flamengo 1", descreve o jogo em que o Sao-Paulo era anfitrião e marcou 3 gols e o Flamengo era visitante e marcou 1 gol.

Supondo que a lista de entrada seja

```
(list "Sao-Paulo 1 Atletico-MG 2"
      "Flamengo 2 Palmeiras 1"
      "Palmeiras 0 Sao-Paulo 0"
      "Atletico-MG 1 Flamengo 2")
```

O seu programa deve produzir a seguinte lista de strings (uma string por linha) de saída

```
(list "Flamengo 6 2 2"
      "Atletico-MG 3 1 0"
      "Palmeiras 1 0 -1"
      "Sao-Paulo 1 0 -1")
```

Onde depois do nome do time aparece o número de pontos, o número de vitórias e o saldo de gols. Se você quiser ir além, pode formatar a saída para ficar mais legível

```
(list "Flamengo     6  2   2"
      "Atletico-MG  3  1   0"
      "Palmeiras    1  0  -1"
      "Sao-Paulo    1  0  -1")
```

Requisitos

O programa deve:

1) Funcionar de forma genérica para qualquer quantidade de resultados e para quaisquer times que apareçam nos resultados;
2) Ter pelo menos uma função recursiva;
3) Utilizar as funções `map`, `filter` e `foldr` pelo menos uma vez cada;
4) Utilizar apenas as funções pré-definidas em Racket que estão no documento "Resumo da linguagem Racket".

Critérios de avaliação

O trabalho será avaliado de acordo com os seguintes critérios:

1) Utilização do processo de projeto de programas
   (a) Definição e documentação adequadas dos tipos de dados
   (b) Consistência e completude da especificação e implementação
   (c) Diversidade e completude dos exemplos
2) Uso adequado do paradigma funcional e das construções da linguagem

Desenvolvimento

O uso de lista de strings na entrada e saída da função principal permite que o programa funcione com dados na entrada e saída padrão com o seguinte trecho de código (supondo que a sua função chame `classifica-times`):

```
(display-lines (classifica-times (port->lines)))
```

Com isso você pode executar o seu programa na linha de comando para ele ler os resultados do arquivo `jogos.txt` e mostrar a classificação na tela:

```
racket classificao.rkt < jogos.txt
```

Note que internamente o seu programa deve trabalhar com estruturas, ele não deve trabalhar com strings nem para o resultado e nem para o desempenho. Dessa forma, a primeira coisa que seu programa deve fazer é transformar a lista de strings de entrada em uma lista de resultados e a última coisa deve ser transformar a classificação em uma lista de strings.

Resolver este problema "de uma" vez é complicado, então você deve decompô-lo em subproblemas menores e desenvolver um plano (sequência de etapas) de resolução dos subproblemas.

A seguir eu sugiro um plano de resolução, você pode usar esse plano ou criar o seu próprio.

- Descobrir os nomes dos times
- Calcular os pontos, número de vitórias e saldo de gols de um time por vez
- Classificar os times de acordo com o desempenho de cada um

Para classificar os times você pode adaptar a função que faz a ordenação por inserção, que está no material, ou implementar outro método de ordenação.

Você pode usar o seguinte trecho de código como partida:

```
;; ListaString -> ListaString
(define (classifica-times sresultados)
    ;; Transforma a lista de strings da entrada em uma lista de resultados
    (define resultados (map string->resultado sresultados))
    ;; Encontra o nome dos times
    ;; ListaResultado -> ListaString
    (define times (encontra-times resultados))
    ;; Calcula o desempenho de cada time
    ;; ListaString ListaResultado -> ListaDesempenho
    (define desempenhos (calcula-desempenhos times resultados))
    ;; Faz a classificao dos times pelo desempenho
    ;; ListaDesempenho -> ListaDesempenho
    (define classificacao (classifica desempenhos))
    ;; Transforma classificação (lista de desempenhos) em uma lista de strings
    (map desempenho->string classificacao))
```