Introducción a la Programación

Primer Parcial - Turno Noche

- El parcial se aprueba con 6 puntos
- Podrás utilizar las siguientes funciones del prelude
  - Listas: head, tail, last, init, length, elem, ++
  - Tuplas: fst, snd
  - Operaciones lógicas: &&, ||, not
  - Constructores de listas: (x:xs), []
  - Constructores de tuplas: (x,y)
- Si querés utilizar Hunit para testear tu código acá tenés un script de ejemplo.

¡Vamos campeón!

En Exactas se está jugando un torneo de fútbol y la facultad le pidió a los alumnos de IP programar algunas funcionalidades en Haskell. Los datos con los que contamos para esto son los nombres de los equipos que participan del torneo, los nombres de los goleadores de cada uno de dichos equipos, y la cantidad de goles convertidos por esos jugadores. Los nombres de los equipos y sus respectivos goleadores serán modelados mediante tuplas de tipo `(String, String)`, donde la primera componente representa el nombre del equipo, y la segunda representa el nombre del goleador de dicho equipo.

En los problemas en los cuales se reciban como parámetros secuencias *goleadoresPorEquipo* y *goles*, cada posición de la lista *goles* representará la cantidad de goles obtenidos por el goleador del equipo que se encuentra en en esa misma posición de *goleadoresPorEquipo*. Por ejemplo, si la lista *goleadoresPorEquipo* es [("Sacachispas","Robertino Giacomini"), ("Fénix","Matías Domínguez")] y la lista *goles* es [3, 5], eso indica que Robertino Giacomini metió 3 goles, y Matías Domínguez 5.

---

1) Goles de no goleadores [1 punto]

problema golesDeNoGoleadores (*goleadoresPorEquipo*: seq‹String x String›,*goles*:seq< Z >, *totalGolesTorneo*: Z) : Z {
  requiere: {equiposValidos(*goleadoresPorEquipo*)}
  requiere: {|*goleadoresPorEquipo*| = |*goles*|}
  requiere: {Todos los elementos de *goles* son mayores o iguales a 0}
  requiere: {La suma de todos los elementos de *goles* es menor o igual a *totalGolesTorneo*}
  asegura: {*res* es la cantidad de goles convertidos en el torneo por jugadores que no son los goleadores de sus equipos}
}

---

2) Equipos Válidos [3 puntos]

problema equiposValidos (*goleadoresPorEquipo*: seq‹String x String›) : Bool {
  requiere: {True}
  asegura: {(*res* = true) <=> *goleadoresPorEquipo* no contiene nombres de clubes repetidos, ni goleadores repetidos, ni jugadores con nombre de club}
}

---

3) Porcentaje de Goles [3 puntos]

problema porcentajeDeGoles (*goleador*: String, *goleadoresPorEquipo*: seq‹String x String›,*goles*:seq< Z >) : R {
  requiere: {La segunda componente de algún elemento de *goleadoresPorEquipo* es *goleador*}
  requiere: {equiposValidos(*goleadoresPorEquipo*)}
  requiere: {|*goleadoresPorEquipo*| = |*goles*|}
  requiere: {Todos los elementos de *goles* son mayores o iguales a 0}
  requiere: {Hay al menos un elemento de *goles* mayores estricto a 0}
  asegura: {*res* es el porcentaje de goles que marcó *goleador* sobre el total de goles convertidos por goleadores}
}

Para resolver este ejercicio pueden utilizar la siguiente función que devuelve como Float la división entre dos números de tipo Int:

```
division :: Int -> Int -> Float
division a b = (fromIntegral a) / (fromIntegral b)
```

---

4) Botín de Oro [3 puntos]

problema botinDeOro (*goleadoresPorEquipo*: seq‹String x String›, *goles*:seq< Z >) : String {
  requiere: {equiposValidos(*goleadoresPorEquipo*)}
  requiere: {|*goleadoresPorEquipo*| = |*goles*|}
  requiere: {Todos los elementos de *goles* son mayores o iguales a 0}
  requiere: {|*goles*| > 0}
  asegura: {*res* es alguno de los goleadores de *goleadoresPorEquipo* que más tantos convirtió de acuerdo a *goles*}
}

Completa aca el codigo Haskell:

A continuación te dejamos una estructura básica para resolver los ejercicios. Este código no pretende resolver ningun caso de los ejercicios planteados, es sólo una plantilla.

```
module Solucion where

-- Ejercicio 1
golesDeNoGoleadores :: [(String, String)] -> [Int] -> Int  -> Int
golesDeNoGoleadores _ _ _ = 0

-- Ejercicio 2
equiposValidos :: [(String, String)] -> Bool
equiposValidos _ = True

-- Ejercicio 3
porcentajeDeGoles :: String -> [(String, String)] -> [Int] -> Float
porcentajeDeGoles _ _ _ = 0.0

-- Ejercicio 4
botinDeOro :: [(String, String)] -> [Int] -> String
botinDeOro _ _ = ""
```

Enviar