Juego de Cacería

Se trata de un juego donde hay cazadores que recorren el planeta con el objetivo de atrapar los monstruos y así sumar puntos.

1) **Los mostruos**

Monstruos hay muchos y de cada uno se conoce su tamaño, peligrosidad y algunas características. Por ejemplo, tenemos a godzilla, con tamaño 100, peligrosidad 5 y sus caracteristicas son "terrestre", "colmillos" y "coraza". También esta sullivan, de tamaño 50, peligrosidad 1 y "pelos" y "simpatico" son sus características.
Se tiene registrada cuales son las características terribles, que son las mismas para todo el juego: Por ejemplo: "colmillos","garras","coraza","aguijón".

Se quiere saber cuantos puntos da un monstruo a quien lo caza. El puntaje que se obtiene al cazar un monstruo es equivalente al 20% de su tamaño por el doble de su peligrosidad, a lo que se le suman 10 puntos adicionales si tiene alguna característica considerada terrible. Hacer una solución en el paradigma funcional y otra en el paradigma lógico que resuelvan el mismo problema. Explicar similitudes y diferencias.

2) **Los cazadores funcionales**

De los cazadores se conoce su nombre, la cantidad de medallas recibidas y un cierto puntaje acumulado. También se conoce la especialidad de cada cazador, que si bien puede ser diferente entre unos y otros, suele repetirse para varios cazadores. La especialidad permite identificar a cuáles monstruos puede cazar y a cuáles no.
Las especialidades conocidas son:

- acuatico. Caza monstruos que tengan "acuatico" entre sus características
- peliplumifero. Caza monstruos que tengan "pelos" o "plumas" entre sus características
- todoTerreno. No se fija en las características, pero sólo caza mostruos de tamaño mayor a un valor indicado y peligrosidad entre 3 y 6.

Por ejemplo, el cazador llamado hunter tiene 1 medalla, 100 puntos y se especializa en acuáticos. Por otra parte, un cazadar llamado chasseur no tiene medallas, acumula 5000 puntos y su especialidad es todo terreno, con un valor tope de 50.

Implementar en Haskell:

a) Definir el tipo de dato y los datos de ejemplo mencionados, agregar otro cazador que también sea acuático, uno que sea peliploumifero y otro que sea todo terreno pero con otro valor como tope.
b) Puntaje de caceria: Se quiere saber cuántos puntos obtiene un cazador por cazar todos los monstruos que pueda (que su especialidad le permita) de un conjunto de monstruos dado.
c) Otorgar medalla: Hacer que un jugador reciba una medalla, que implica que aumenta en uno su cantidad de medallas y suma 100 puntos.

d) Cacería: Se quiere obtener cómo queda el cazador luego de ir de cacería, habiendo acumulado los puntos correspondientes a los monstruos que cazó. Si logró cazar más de 2 monstruos, recibe una medalla.
e) Inventar una nueva especialidad, hacer que alguno de los cazadores ahora tenga esa especialidad, y que todo siga funcionando correctamente.

Justificar la utilización (o no) de los conceptos de aplicación parcial, composición y orden superior.

3) Equipos lógicos de caza

Se quieren armar equipos de cacería, que estan compuestos por un cazador lider, un cazador asistente y una jauría de una cierta cantidad de perros. Como son muchas las posibles formas de combinar los equipos, se decidió implementar la solución con el paradigma logico.

Se cuenta con hechos sobre los cazadores, por ejemplo:
cazador(hunter, 1 , 100, acuatico).
cazador(chasseur, 0, 5000, todoTerreno(50)).
Completar la base de conocimiento con otros ejemplos (Usar los mismos que el ejercicio anterior).

Para formar un equipo, el lider debe tener más puntaje que el asistente y tener distinta especialidad (se considera que dos todo terreno son de la misma especialidad aunque tengan diferente valor tope). La cantidad de perros no debe ser menor a 3 ni mayor a 5 veces la cantidad de medallas de ambos cazadores juntos. Por ejemplo, sería válido un equipo que tenga por lider a chasserur, como asistente a hunter y 4 perros. Tambíen para los mismos cazadores si los perros fueran 3 o 5, pero no con otras cantidades.

Implementar en prolog los siguientes predicados, haciendo que sean totalmente inversibles, mostrar ejemplos de consulta y justificar:

a) hayEquipo/3. Relacione al cazador lider con el cazador asistente y la cantidad de perros de un equipo válido.

b) liderIndispensable/1. Permite encontrar, si lo hay, qué cazador resulta lider de todos los equipos posibles.
c) cazadorInutil/1. Debe permitir averiguar si algún cazador no puede formar parte de ningún equipo posible, ni como lider ni como asistente.

Nota: Existe un predicado llamado between/3