Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
Laboratorio de Lenguajes de Programación (CI-3661)

Proyecto Haskell

Lambdamon, Gotta Apply'em All!

En un pequeño pueblo en la costa de la región de Funcionalia, llamado Pueblo Haskellia, un muchacho de 10 años parte en una gran aventura. Al inicio, se encuentra con el estimado Prof. Curry.

Prof. Curry: *¡Hola! ¡Bienvenido al mundo de Lambdamon! Mi nombre es Curry, pero muchos me llaman el Profesor Lambdamon.*

Este mundo está habitado por criaturas llamadas Lambdamon. Para algunas personas, los Lambdamon son mascotas. Otros los usan para pelear. En mi caso... ¡mi profesión es estudiarlos!

Tu propia leyenda Lambdamon está a punto de... ¿ah? ¿Qué no quieres ser un Maestro Lambdamon? ¿Qué quieres ser mi asistente? Ah bueno, en ese caso...

Después de una pequeña charla, el Prof. Curry presenta su línea de investigación: él quiere entender bien tanto las dinámicas de las batallas *Lambdamon* como construir un catálogo que presente a todos los *Lambdamon* existentes (llamémoslo *Lambdadex*). Para eso, le propone a ud., que sabe que está cursando esta asignatura, a construir tanto un *Lambdadex* como un simulador de batallas *Lambdamon*. Por preferencias obvias (Funcionalia, Pueblo Haskellia, Prof. Curry), el solicita que ud. implemente ambos programas en el lenguaje de programación Haskell (¡duh!).

*(**Advertencia:** Cualquier similitud con la vida real es pura coincidencia... nah, no lo es jajaja. A partir de ahora hablaremos en base a Pokémon. Pokémon es una marca registrada de Nintendo y de The Pokémon Company.)*

1 Conceptos principales

Para este proyecto se manejarán los siguientes conceptos:

- Un **tipo** es una propiedad que representa un valor elemental. Existen 17 tipos en Pokémon, planteados la siguiente definición en Haskell (usando los nombres en inglés de los tipos en cuestión) propuesta en la Figura 1.
- Una **especie** Pokémon (usualmente conocido como el Pokémon) es la familia a la que pertenece un Pokémon. Ejemplos de Pokémon son Pikachu (el más famoso), Bulbasaur, Charmander, Squirtle (los iniciales), et al. Tal especie está definida por un número (número de catálogo según el Dex), un nombre, uno o dos tipos elementales, y las estadísticas base que definen al pokémon: su vida (a partir de ahora llamda HP, del inglés

```
data Tipo
      = Bug
      | Dark
      | Dragon
      | Electric
      | Fighting
      | Fire
      | Flying
      | Ghost
      | Grass
      | Ground
      | Ice
      | Normal
      | Poison
      | Psychic
      | Rock
      | Steel
      | Water
      deriving (Bounded, Eq, Enum, Read, Show)
```

Figure 1: Definición del tipo `Tipo` en Haskell.

Hit Points), su ataque, su defensa, su ataque especial, su defensa especial y su velocidad (Las estadísticas base corresponden a enteros comprendidos entre 0 y 255).

Además, como una especie puede evolucionar a otra dado un criterio de evolución, una especie puede estar relacionada a otra por pre-evolución (que especie evoluciona a la actual, puede ser un valor nulo indicando que es un miembro inicial de la cadena) y puede estarlo a un conjunto de especies a las que éste pueda evolucionar (con su criterio correspondiente de evolución).

- Actualmente hay 649 especies pero trabajaremos con los primeros 251 monstruos *(hasta la generación Gold & Silver)*.

Vamos a tomar como ejemplo a Pidgeotto:

Pidgeotto es una especie de tipo `Normal`/`Flying`, número de catálogo 17 y con las siguientes estadísticas base:

- HP: 63
- Ataque: 60
- Defensa: 55
- Ataque especial: 50
- Defensa especial: 50
- Velocidad: 71

Pidgeotto evoluciona de Pidgey:

y evoluciona a Pidgeot al alcanzar el nivel 36:

- Un **ataque** es una acción que puede realizar un Pokémon para ocasionar un efecto a otro Pokémon. Para este proyecto trabajaremos solamente con ataques que ocasionen daño. Un ataque está comprendido por su nombre, el tipo de ataque (a diferencia de la especie, un ataque es de un sólo tipo), si el ataque es físico o no, los Puntos de Poder (o PPs) máximo del ataque (cuántas veces puedo usar el ataque) y el poder del ataque. Los ataques en el juego real pueden tener un efecto secundario y un valor de puntería, pero para efectos del proyecto, no trabajaremos con efectos secundarios y supondremos que todos los ataques conectan.

  Un Pokémon puede tener o uno o dos tipos (a partir de ahora mencionado como el tipo `TipoPokémon`) y un ataque debe ser de un tipo.

  Por ejemplo, el ataque *Gust* es un ataque físico de tipo `Flying`, de poder 40 y 35 PPs.

- Un **monstruo** es un miembro de la especie. Un ejemplo de un monstruo famoso es el Pikachu de Ash (un miembro bien especial de la familia Pikachu). Un monstruo está definido por la especie a la que pertenece, un sobrenombre que pueda tener (se debe manejar, mas el sobrenombre como tal es opcional), el nivel en que está (un número entre 1 y 100), el HP actual (un número entre 0 y el HP máximo, introducido luego) y los ataques (que puede tener entre 1 y 4 ataques), cada ataque con su PP remanente (cuantas veces puedo usar el ataque antes que sea inútil). Adicionalmente, el monstruo puede tener, para cada estadística, dos valores adicionales:

  - Los Valores Individuales (que permiten tener variación entre distintos monstruos de la misma especie y el mismo nivel, representado en el juego como un valor entre 0 y 31, y que nunca varía en la vida del monstruo).
  - Los Valores de Esfuerzo (que determinan la experiencia que ha adquirido el monstruo en cierta estadística, un número entre 0 y 255).

  Para efectos del proyecto, puede suponer que los Pokémon a usar son perfectos (todos sus valores individuales valen 31 y sus valores de esfuerzo son 255).

  Como ejemplo de especie, considere un Pidgeotto que ud. entrenó. Su Pidgeotto tiene nivel 43, pero ha recibido suficientes golpes y le queda nada más 20 HP. Su Pidgeotto solamente conoce el ataque *Gust*, del cual le quedan 10 usos de los 35 máximos. Como se supone que los Pokémon a trabajar son perfectos a nivel de individualidad y esfuerzo, tiene 31 en todos sus valores individuales y 255 en sus valores de esfuerzo. Ah, y además se llama "Pajarraco".

Ud. debe crear, en Haskell, la representación en tipos de datos de los conceptos mencionados anteriormente, en particular los tipos `Especie`, `Monstruo` y `Ataque`. Los internos de la representación se deja a criterio de uds, siempre y cuando represente todos los conceptos y sus atributos.

En adición, ud. debe implementar una función que imprima de manera legible para un usuario los datos de una especie. Puede, y se recomienda, implementar esta función aparte de la derivación predefinida de Haskell. Esta función será usara para efectos del catálogo solicitado.

2 Implementando el simulador de batallas

2.1 Estadísticas actuales

Para poder implementar el simulador de batallas, se necesita primero poder calcular los valores estadísticos actuales correspondientes. Recordemos que hay seis estadísticas en un Pokémon:

- HP
- Ataque

- Defensa
- Ataque especial
- Defensa especial
- Velocidad

Una especie de Pokémon tiene un valor base, pero la estadística actual de un monstruo depende además de varios factores: en que nivel se encuentra el monstruo, su valor individual de la estadística en cuestión, y su valor de esfuerzo. Para poder realizar el cálculo del valor actual, se usarán dos fórmulas. La primera fórmula se usa para calcular el HP máximo de un monstruo, y se obtiene de la siguiente forma:

$$maxHp = \frac{(ivHp + (2 \cdot baseHp) + \frac{evHp}{4} + 100) \cdot nivel}{100} + 10$$

donde $evHp$ corresponde al valor de esfuerzo de HP que presenta el monstruo (255 según la suposición), $baseHp$ corresponde a la estadística base de HP de la especie, $ivHp$ corresponde al valor individual del monstruo (31 según la suposición) y $nivel$ corresponde al nivel del Pokémon actualmente.

Para las demás estadísticas, su valor actual se calcula mediante la siguiente fórmula:

$$estadistica = \frac{(iv + (2 \cdot base) + \frac{ev}{4}) \cdot nivel}{100} + 5$$

donde los valores iv, $base$ y ev corresponden al valor individual (31), valor base de especie y valor de esfuerzo (255) del monstruo según la estadística en cuestión (por ejemplo, si se calcula ataque, sería el valor individual, valor base y valor de esfuerzo de ataque).

Retomando nuestro ejemplo, vuestro Pidgeotto perfecto nivel 43 tendría las siguientes estadísticas actuales (usando las fórmulas introducidas anteriormente):

- HP: 147
- Ataque: 97
- Defensa: 92
- Ataque Especial: 88
- Defensa Especial: 88
- Velocidad: 106

2.2 Calculando el daño de un ataque

Ya conocidas las fórmulas de cálculo de estadísticas actuales, procedemos a calcular el daño que ocasiona un ataque de un monstruo (considerado el atacante) a otro (considerado el defensor). El daño de un monstruo viene dado por los siguientes factores:

- Nivel del monstruo que realiza el ataque.
- Si el ataque es físico o no.
- Si el ataque es físico, el ataque actual del monstruo atacante y la defensa actual del monstruo defensor. Si no es físico, entonces la versión especial de cada estadística será usada.
- El poder del ataque.
- Los tipos del ataque y de los dos monstruos.

La fórmula de daño de un ataque es la siguiente:

$$daño = \left(\frac{\left(\frac{(2 \cdot nivelAtacante)}{5} + 2 \right) \cdot poder \cdot \left(\frac{ataque}{defensa} \right)}{50} + 2 \right) \cdot modificador$$

donde *nivelAtacante* corresponde al nivel del monstruo atacante, *poder* al poder del ataque usado, *ataque* corresponde al valor actual de ataque del monstruo atacante en caso que el ataque sea físico, ataque especial en caso contrario; $defensa$ corresponde al valor actual de defensa del monstruo defensor en caso que el ataque sea físico, defensa especial en caso contrario y $modificador$ el factor obtenido de los siguientes modificadores:

- Si el tipo del ataque coincide con uno de los tipos del monstruo atacante, el ataque hace 1.5 veces más daño.
- Para cada tipo del monstruo defensor:
  - Si el tipo en cuestión es débil al tipo del ataque, el ataque hace el doble de daño.
  - Si el tipo en cuestión resiste el tipo del ataque, el ataque hace la mitad del daño.
  - Si el tipo en cuestión es inmune al tipo del ataque, el ataque no hace daño.
  - En el caso que el monstruo defensor tenga dos tipos, se aplica cada modificador por separado. Por ejemplo, si los dos tipos del monstruo defensor es débil al tipo del ataque, el ataque hace el cuadrúple de daño (doble por el primer tipo y doble del segundo).

Por ejemplo, considere a un Kabutops nivel 60 con las siguientes estadísticas (base / actual):

- Tipo: Rock/Water
- HP: 60 / 198
- Ataque: 115 / 199
- Defensa: 105 / 187
- Ataque Especial: 65 / 139
- Defensa Especial: 70 / 145
- Velocidad: 80 / 157

y tiene los siguientes ataques:

- *Body Slam*, tipo Normal fisico con 85 de poder y 15 PP.
- *Absorb,* tipo Grass especial con 20 de poder y 20 PP.
- *Water Gun*, tipo Water especial con 40 de poder y 25 PP. *(Mismo tipo que Kabutops.)*
- *Rock Throw*, tipo Rock fisico con 50 de poder y 15 PP. *(Mismo tipo que Kabutops.)*

y se enfrenta a un Charizard nivel 70:

- Tipo: Fire / Flying
- HP: 78 / 255
- Ataque: 84 / 188
- Defensa: 78 / 180
- Ataque Especial: 109 / 223
- Defensa Especial: 85 / 189
- Velocidad: 100 / 210

Cada uno de los ataques de Kabutops dan los siguientes resultados:

- Body Slam hace 48 puntos de daño.
- Absorb hace 2 puntos de daño. (*It's not very effective...)*
- Water Gun hace 50 puntos de daño. (*It's super effective!)*
- Rock Throw hace 172 puntos de daño *(It's super effective!)*

2.3 El sistema de batalla

Una batalla Pokémon (o Lambdamon) sigue un flujo bastante complicado. Para el proyecto, trabajaremos con la siguiente simplificación del sistema de batalla usado en el juego original.

2.3.1 Conformación de los equipos

Una batalla Pokémon se realiza entre dos entrenadores, que tienen cada uno, a su disposición, un equipo comprendido entre 1 y 6 monstruos.

2.3.2 Inicio de la batalla

Al comenzar la batalla, los entrenadores eligen a un monstruo de los distintos miembros de su equipo. El monstruo elegido se convertirá en el **monstruo actual**.

Todos los monstruos de ambos entrenadores comienzan con los PPs de sus ataques al máximo (si *Gust* tiene 35 de PP máximo, el ataque comenzará con 35 PPs).

2.3.3 Flujo de batalla

Las batallas se realizan por medio de turnos. En cada turno, se solicita a cada uno de los entrenadores alguna de las siguientes acciones:

- Elegir uno de los posibles movimientos que conoce el monstruo actual. El entrenador puede elegir únicamente movimientos que tengan al menos 1 PP disponible.
- Cambiar el monstruo actual a otro monstruo del equipo. Solamente puede elegir monstruos que estén conscientes (siendo un monstruo consciente aquél que tenga al menos 1 HP al momento del cambio).
- Rendirse. Esta acción acaba la pelea en victoria para el contrincante, al menos que ambos entrenadores se rindan. En el caso que ambos se rindan, la pelea termina en empate.

Después de que ambos entrenadores elijan su acción, si ninguno se rindió, el turno se desempeña de la siguiente forma:

1. Si algún entrenador solicitó realizar un cambio:
   (a) Si ambos entrenadores solicitaron cambio, se realizan ambos cambios y se pasa al siguiente turno.
   (b) Si el otro entrenador no realizó cambio:
      i. Se realiza el cambio del entrenador en cuestión (llamémoslo entrenador cambiante).

ii. El monstruo del entrenador que no cambió realiza su acción y hace daño al nuevo monstruo activo del entrenador cambiante.

iii. Si el monstruo del entrenador cambiante pierde todo su HP (su HP actual es menor o igual a 0), se declara inconsciente.

2. Si ningún entrenador realizó cambio:

   (a) Se comparan las velocidades actuales de los monstruos activos. El que tenga más velocidad ataca primero.

      i. En el caso que tengan la misma velocidad, dar prioridad a uno de los dos entrenadores **siempre**. Tal prioridad será definida en la sección 3.

   (b) El monstruo más rápido realiza su ataque y le ocasiona daño al monstruo rival.

      i. Si el monstruo que recibió daño pierde todo su HP (su HP actual es menor o igual a 0), se declara inconsciente.

   (c) Si el monstruo que ataca de último no está inconsciente, él realiza su ataque y ocasiona daño al monstruo rápido.

      i. Si el monstruo rápido pierde todo su HP (su HP actual es menor o igual a 0), se declara inconsciente.

3. Si hay algún monstruo inconsciente, el entrenador en cuestión debe cambiar a otro monstruo. Si el entrenador no puede cambiar de monstruo, se termina la partida dando como ganador al rival.

4. Si la partida no se ha terminado, proceda al siguiente turno.

3 La interfaz con el usuario

Su aplicación debe interactuar con el usuario mediante una interfaz vía consola. Su programa se debe llamar `pokesim`, y recibe cuatro argumentos:

- Una dirección donde se ubique el archivo de especies. El archivo de especies es un archivo de texto separado por comas (`CSV`, de *Comma Separated Values*), donde cada línea está definida de esta forma:

  ```
  numero,nombre,tipo1,tipo2,hp,ataque,defensa,ataque_especial,defensa_especial,
  velocidad,padre,evolucion
  ```

  donde cada dato corresponde al dato relacionado para la especie, `padre` corresponde al número de catálogo de la pre-evolución y `evolucion` corresponde a la razón de evolución, una cadena de caracteres indicando como el Pokémon evoluciona.

  - En el caso que la especie no tenga una pre-evolución, no habrá ningún contenido entre las comas.
  - En el caso que un monstruo tenga un solo tipo, `tipo2` no tendrá nada entre las dos comas.

  Por ejemplo, el Pidgeotto que definimos anteriormente tendría la siguiente línea:

  ```
  17,Pidgeotto,Normal,Flying,63,60,55,50,50,71,16,Nivel 18
  ```

- Una dirección donde se ubique el archivo de ataques. El archivo de ataques es un archivo `CSV` donde cada línea está definida de esta forma:

  ```
  nombre,tipo,fisico,pp,poder
  ```

  donde cada dato corresponderá al dato pertinente definido en los ataques. (`fisico` puede valer `True` o `False`).

  Por ejemplo, el ataque *Gust* se definiría en el CSV de la siguiente forma:

  ```
  Gust,Flying,True,35,40
  ```

- Dos direcciones, correspondiente a los equipos del Entrenador 1 y Entrenador 2. Cada archivo corresponderá a un `CSV` de máximo 6 líneas, donde cada línea define a un monstruo del equipo. Tales líneas contendrán la siguiente información:

  ```
  numero,sobrenombre,nivel,ataque1,ataque2,ataque3,ataque4
  ```

  donde:

  - `numero` corresponde al número de especie a la que pertenece tal monstruo.
  - `sobrenombre` corresponde al sobrenombre del monstruo.
  - `nivel` corresponde al nivel actual del monstruo.
  - `ataque1`, ... ,`ataque4` corresponde al nombre de ataque en cuestión. Recuerde que debe haber al menos un ataque. Si el monstruo no tiene algún ataque, a partir de `ataque4`, no coloque información.
    * Si el monstruo tiene 3 ataques, `ataque4` debe estar en blanco.
    * Si tiene 2 ataques, `ataque3` y `ataque4` debe estar en blanco.
    * Si tiene 1 ataque, `ataque2`, `ataque3` y `ataque4` debe estar en blanco.

  Por ejemplo, su Pidgeotto (llamado "Pajarraco") tendría la siguiente entrada en el archivo `CSV`: *(Observe las comas sin contenido. Corresponde a cada ataque que "Pajarraco" no tiene.)*

  ```
  17,Pajarraco,43,Gust,,,
  ```

Después de que los archivos se hayan leído, se procede a la batalla. Las convenciones para la batalla serán las siguientes:

- El primer monstruo definido en los archivos de los equipos fungirá como el monstruo elegido por cada entrenador en la batalla.
- El Entrenador 1 (el primer equipo suministrado) es el que tendrá prioridad en el caso que haya empate de velocidad.

Durante la batalla, cada entrenador dispone de los siguientes comandos:

- Elegir un ataque: El comando tendrá la siguiente sintaxis:

  ```
  atacar n
  ```

  donde n corresponde al número de ataque usado (entre 1 y 4).

- Cambiar de monstruo. El comando tendrá la siguiente sintaxis:

  ```
  cambiar n
  ```

  donde n corresponde al número de monstruo a cambiar.

- Rendirse. Su sintaxis será:

  ```
  rendirse
  ```

- Obtener información de catálogo, cuya sintaxis es:

  ```
  info [yo|rival]
  ```

  y la ejecución de este comando mostrará la información de catálogo del monstruo actual o rival.

- Obtener información de ayuda para el entrenador, cuya sintaxis es:

```
ayuda
```

y mostrará la lista de ataques (con el número de comando pertinente), cuantos PPs quedan en cada ataque y la lista de monstruos que quedan en el equipo (con su número).

Después de leer los CSVs y cargar la información, el flujo de su programa es el siguiente:

1. Solicita al Entrenador 1 su comando. Si solicita información de catálogo o ayuda, debería volver a solicitar comando al Entrenador 1.
2. Solicita al Entrenador 2 su comando. Si solicita información de catálogo o ayuda, debería volver a solicitar comando al Entrenador 2.
3. Se ejecuta el flujo de batalla indicado en la sección 2.3.3.
4. En el caso que la pelea no haya terminado:
   (a) Si hay algún monstruo inconsciente, se le pregunta al entrenador afectado a que Pokémon desea cambiar.
   (b) Volver al paso 1.

Los formatos a mostrar y los mensajes a usar se deja a su discreción. Sin embargo, si desea tomar inspiración en algo ya hecho, puede revisar la infinidad de videos de pelea de Pokémon que están publicados en páginas como YouTube. Si desea un ejemplo en particular, puede usar este: `http://www.youtube.com/watch?v=azG7U6Id-6E`.

4 Detalles de entrega

- Ud. debe entregar los archivos solicitados, en adición a un `README` indicando el estado de su proyecto y un `Makefile` para poder realizar la compilación de su proyecto, en un archivo `.tar.gz` llamado `proy1-gXX.tar.gz`, donde `XX` corresponde a su número de grupo. Por ejemplo, el grupo 00 debe entregar el archivo `proy1-g00.tar.gz`.
  - El contenido de su proyecto debe residir en un directorio llamado `gXX`, donde `XX` es el número de su grupo. Ud. debe comprimir el directorio. Si su proyecto **no** cumple con esta norma, su proyecto no será evaluado y tendrá **cero (0)** puntos.
- Su proyecto debe ser entregado a más tardar el día domingo 17 de febrero, a las 11.59 pm, tanto al Prof. Pérez como al Prep. Gómez. Sus direcciones de correo están publicadas en la página web. Cualquier entrega después de esa hora ocasionará que su proyecto no sea evaluado y tendrá **cero (0)** puntos.