README
PRÁCTICA 06

Liprandi Cortes Rodrigo: 317275605
Tinoco Miguel Laura Itzel: 316020189

19.Enero.2021

1 Observaciones:

Explicaciones de las soluciones.

1. **Relación traducir.**

   Para nuestra base de datos utilizamos un segmento de la canción "We will rock you - Queen":

   "Buddy, you're a boy, make a big noise Playing in the street, gonna be a big man someday You got mud on your face, you big disgrace..."

   Además ocupamos una relación auxiliar "append" la cual concatena dos listas.

   Para la función traducir, en el caso base, si nos pasan una lista vacía regresamos una lista vacía. En el caso en que nos pasen una lista no vacía, obtenemos la palabra de la cabeza, checamos su traducción en la base de datos y hacemos una lista con ella, esta la concatenamos con las listas que salgan al llamar la relación con la cola.

2. **Mundo de cubos**.

   Primero modelamos el mundo de los cubos descrito en la imagen del pdf de la practica con los hechos *cubo(X)* que nos dice que X es un cubo, *sobre(X, Y)* que nos dice que el cubo X esta sobre el cubo Y, el nombre de cada cubo es el color que tiene, después definimos las reglas:

   - *bloqueado(X):* un cubo X esta bloqueado si hay algún cubo sobre el.
   - *hastaArriba(X):* un cubo X esta hasta arriba si es que no esta bloqueado, es decir no hay ningún cubo sobre el.
   - *hastaAbajo(X):* un cubo X esta hasta abajo si no esta sobre algún otro cubo Y.
   - *mover(X, Y):* el cubo X se puede mover sobre Y solo si X y Y no están bloqueados, es decir que ambos estén hasta arriba.

3. **AFN**

   Primero modelamos el autómata con relaciones.
   Con la relación "ei()" lo que hicimos fue definir el estado inicial.
   Con la relación "ef()" definimos el estado final.
   Con las relaciones "delta()" definimos de qué estado podíamos ir a otro estado y la letra que aceptaban.
   Hicimos una relación "transicion()" la cual hace todo el trabajo. Nuestro caso base es cuando estamos en un estado pero ya no hay letras para checar, en ese caso lo que hacemos es checar si ya estamos en el estado final. En el caso recursivo lo que hacemos es checar el estado en el que estamos, ver la letra en la cabeza de la lista y ver si hay un camino a otro estado que acepte esa letra, si lo hay nos movemos a ese estado y continuamos con la cola de lista.

   La función aceptar llama a la función transicion en el estado inicial.

4. **Relación mezclar.**

   Primero definimos tres relaciones auxiliares:

   - *insertar:* Inserta un elemento en una lista ordenada.
   - *ordenar:* Toma los elementos de una lista y los inserta en otra lista de forma ordenada.
   - *mezclarAux:* Mezcla dos listas ya ordenadas en una nueva lista ordenada.

   Con estas relaciones ya podemos definir la relación mezclar(L1,L2,L), la cual primero ordena la lista "L1" en otra lista "X", la lista "L2" en otra lista "Y", y mezcla las listas ordenadas "X" y "Y" en la lista "L".

   Nota: Para resolver este ejercicio me guié con las fuentes citadas.

2 Fuentes consultadas:

- Prolog program to merge two ordered list generating an ordered list. (s.f.). Recuperado de: http://www.dailyfreecode.com/code/prolog-merge-two-ordered-list-3129.aspx
- Prolog.(s.f.). Recuperado de: https://www.uv.mx/personal/aguerra/files/2020/09/pia-03.pdf