Eliza

Curso 2004/05, Fecha:27/01/2005

Enunciado 1 (busca) *Se representa un diccionario como una lista de tuplas, donde el primer elemento es la clave y el segundo, el valor*

```
> type Dicc a b = [(a,b)]
```

Construir un predicado busca::a →Dicc a b →b que devuelve el valor de una clave en un diccionario

```
?- busca 2 [(4,"hola"),(2,"juan"),(3,"pepe")]
juan
```

Enunciado 2 (leer) *Una frase puede representarse como una lista de cadenas de caracteres:*

```
> type Frase = [String]
```

Construir un programa que solicite una frase al usuario y devuelva dicha frase con las palabras en orden inverso de forma indefinida hasta que el usuario introduzca la palabra adios

Nota*: Puede utilizarse la función* ***words*** *::****String*** *→[****String****] que toma una frase y devuelve la lista de palabras de dicha frase*

Enunciado 3 (Patrones) *Los patrones estímulo/respuesta para el programa conversacional Eliza pueden definirse de la siguiente forma:*

```
> type Patron = [Elem]

> data Elem = N Int       -- Valor a encajar
>           | S String   -- Cadena de texto
>  deriving Show

> patron :: Patron → Patron
> patron [S "estoy", N 1]  = [S "estas", N 1, S "a menudo?"]
> patron [N 1,S "es", N 2] = [S "porque crees que", N 1, S "es", N 2, S "?"]
> patron [S "eres", N 1]   = [S "porque crees que soy", N 1, S "?"]
> patron [S "soy", N 1]    = [S "porque crees que eres", N 1, S "?"]
> patron [S "si"]          = [S "puedes ser mas expresivo?"]
> patron [S "no"]          = [S "no estas siendo muy negativo?"]
```

```
> patron [S "hola"]          = [S "cuentame algo"]
> patron [S "adios"]         = [S "hasta la proxima"]
> patron [N 1]               = [S "por favor, continua"]
```

Incorporar el código anterior al programa y añadir algún otro patrón

Enunciado 4 (encaja) *Añadir al programa anterior el siguiente predicado.*

```
> encaja :: Patron → Dicc Int Frase → Frase
> encaja (N n:ps) d = busca n d ++ encaja ps d
> encaja (S p:ps) d = p:encaja ps d
> encaja []       _ = []
```

Observar y justificar el funcionamiento del predicado ante las siguientes preguntas:

- *?−encaja([N 1,S "es",N 2] [(1,[S "juan"]),(2,[S "tonto"])]*
- *?−encaja ["hola",N 1] v =:= [hola,amigo,pepe]* ***where*** *v free*
- *?−encaja [N 1, S "es",N 2] v =:= [S "juan",S "es",S "tonto"]* ***where*** *v free*

Enunciado 5 (eliza) *Construir el programa conversacional Eliza que solicita frases del usuario y devuelve la respuesta que encaje según la lista de patrones que se haya definido.*

Enunciado 6 (estado(Opcional)) *Modificar el programa conversacional para que mantenga un estado en la conversación. Dicho estado puede contener la lista de patrones, de forma que durante la conversación sea posible añadir o quitar patrones*

Enunciado 7 (tema(Opcional)) *Modificar el programa Eliza con patrones que le permitan mantener una conversación sobre un tema concreto, por ejemplo: deportes, música, etc.*

Enunciado 8 (respuestaAlea(Opcional)) *Modificar el programa Eliza para que el formato de los patrones sea: patron(E,[R1,R2...Rn]). Es decir que a cada estímulo E le correspondan una serie de respuestas y el sistema seleccione dichas respuestas de forma aleatoria.*

La implementación actual de Curry no contiene la librería ***Random****, sin embargo, es relativamente fácil simular números pseudoaleatorios mediante listas infinitas y el algoritmo de congruencia lineal. Por ejemplo, el siguiente código devuelve una lista infinita de números pseudoaleatorios:*

```
> listaAlea:: Int → [Int]
> listaAlea n
>  = genera semilla
```

```
>     where
>         semilla  = 1234567
>         a        = 31415821
>         m        = 65536    -- 2 ^ 16
>         genera x = r: genera x'
>          where x' = (a * x + 1) `mod` m
>                r = ((x' * n) `div` m)
```

A su vez, los más intrépidos podéis utilizar el servicio Web que se ofrece en Random.org. Para ello debéis buscar la forma de utilizar servicios Web desde Curry, Animo!!

Enunciado 9 (chatBot(Opcional)) *Incorporar el programa* Eliza *a un* chatBot. *Un* chatBot *es un programa que* chatea *de forma automática en foros* IRC.

Puede obtenerse más información en

- *AliceBot*
- *FinOP*
- *Programming IRC Bots in Perl*

Enunciado 10 (Juegos(Opcional)) *Construir programas que resuelvan acertijos lógicos.*

Ejemplos:

- *Asignar dígitos a las letras de forma que se cumpla la suma:*

  $$\begin{array}{r} S\ E\ N\ D \\ +\ M\ O\ R\ E \\ \hline =\ M\ O\ N\ E\ Y \end{array}$$

- *Dada una lista de dígitos, buscar combinaciones de operaciones entre dichos dígitos que den como resultado el valor 24. Por ejemplo, dados los dígitos [2,3,7,8] , una solución podría ser ((8 + 7) − 3) ∗ 2*

- *Resolver el siguiente examen:*

```
1 : la primer pregunta cuya respuesta es A es:
      (A) 4    (B) 3    (C) 2    (D) 1    (E) ninguna de las anteriores
2 : las dos Ãºnicas preguntas consecutivas cuyas respuestas son idÃ©nticas
      (A) 3 y 4    (B) 4 y 5    (C) 5 y 6    (D) 6 y 7
(E) 7 y 8
3 : la siguiente pregunta cuya respuesta es A es:
      (A) 4    (B) 5    (C) 6    (D) 7    (E) 8
4 : la primer pregunta de nÃºmero par cuya respuesta es B es:
      (A) 2    (B) 4    (C) 6    (D) 8    (E) 10
5 : la Ãºnica pregunta de nÃºmero impar cuya respuesta es C es:
```

(A) 1 (B) 3 (C) 5 (D) 7 (E) 9

6 : Un pregunta cuya respuesta es D...

(A) aparece antes de Ã©sta pero no despues de Ã©sta

(B) aparece despuÃ©s de Ã©sta, pero no antes de Ã©sta

(C) aparece antes y despuÃ©s de Ã©sta

(D) no aparece

(E) ninguna de las anteriores

7 : la Ãºltima pregunta cuya respuesta es E es:

(A) 5 (B) 6 (C) 7 (D) 8 (E) 9

8 : el nÃºmero de preguntas cuyas respuestas son consonantes es:

(A) 7 (B) 6 (C) 5 (D) 4 (E) 3

9 : el nÃºmero de preguntas cuyas respuestas son vocales es:

(A) 0 (B) 1 (C) 2 (D) 3 (E) 4

10: la respuesta a esta pregunta es:

(A) A (B) B (C) C (D) D (E) E