Demostracion ejercicio 3

Funciones usadas:
objetos_en :: [Either Personaje Objeto] $\Rightarrow$ [Objeto]
objetos_en[] = []
objetos_en universo = **map** objeto_de (**filter es_un_objeto** universo)

es_un_objeto :: Either Personaje Objeto $\rightarrow$ Bool
es_un_objeto (Left o) = False
es_un_objeto (Right p) = True

objeto_de :: Either Personaje Objeto $\rightarrow$ Objeto
objeto_de (Right o) = o

map :: $(a \rightarrow b) \rightarrow [a] \rightarrow [b]$
map _ [] = []
map f $(x : xs) = f\ x$: **map** $f\ xs$

elem :: Eq $a \Rightarrow a \rightarrow [a] \rightarrow$ Bool
elem _ [] = False
elem y $(x : xs) =$
| $y == x$ = True
| otherwise = **elem** $y\ xs$

filter :: $(a \rightarrow$ Bool$) \rightarrow [a] \rightarrow [a]$
filter _ [] = []
filter p $(x : xs) =$
| $p\ x = x$: **filter** $p\ xs$
| $otherwise$ = **filter** $p\ xs$

Queremos demostrar lo siguiente:

$\forall u$ :: Universo.$\forall o$ :: Objeto.elem o (objetos_en u) $\Rightarrow$ elem (Right o) u

Vamos a realizar induccion estructural sobre u siendo u::Universo

P(u) = $\forall o$ :: Objeto.elem o (objetos_en u) $\Rightarrow$ elem (Right o) u

Las listas tienen un unico caso base, cuando es vacia

Caso base $u = []$
P([]) = $\forall o$ :: Objeto.elem o (objetos_en []) $\Rightarrow$ elem (Right o) []
elem o ([]) $\rightarrow$ elem (Right o) [] $(def\ objetos\_en)$
$False$ $\rightarrow$ elem (Right o) [] $(def\ elem)$
False implica lo que sea, siempre dara true por lo tanto el caso base esta probado y vale P([])

Caso inductivo
Tratamos como hipotesis inductiva a P(u)
HI : P(u) =$\forall o$ :: Objeto.elem o (objetos_en u) $\Rightarrow$ elem (Right o) u
Quiero ver que cumple para P(u0:u)
P(u0:u) =$\forall o$ :: Objeto.elem o (objetos_en $u0 : u$) $\Rightarrow$ elem (Right o) $u0 : u$
u0 puede ser tanto Left Personaje como Right Objeto

Caso Right u0:
elem o (objetos_en $u0 : u$)
elem o (map objeto_de (filter es_un_objeto ($u0 : u$))) ($def\ objetos\_en$)
elem o (map objeto_de ($u0$: filter es_un_objeto (u))) ($def\ filter\ y\ def\ es\_un\_objeto$)
elem o (objeto_de u0 : (map objeto_de (filter es_un_objeto (u)))) $def\ map$
elem o (u0' : (map objeto_de (filter es_un_objeto (u)))) ($def\ objeto\_de$)
Notar que debido a la funcion *objeto_de* cambia el tipo de u0. pasa a ser de Right Objeto a Objeto.
Para simplificar la lectura vamos a decir que u0´ es u0 pero de tipo Objeto

Aplicando la definicion de elem sobre sus argumentos nos queda:
o == u0´ || elem o (map *objeto_de* (filter *es_un_objeto* u)

LLegado a este punto queremos que el resultado del "or" anterior sea True.
Para eso basta con ver que el lado izquierdo del "or" sea verdadero,
y, por otro lado que el lado, que el lado derecho sea verdadero y el izquierdo falso.
Si ambos son Falsos no importa ya que la implicación false $\rightarrow$ ...dara True siempre.

Caso o == u0´
o == u0´ || elem o (map *objeto_de* (filter *es_un_objeto* u)
True $\rightarrow$ elem (Right u0´) u0:u (como se dijo mas arriba u0 = Right u0´. Remplazando queda:
True $\rightarrow$ elem u0 u0:u
True $\rightarrow$ True ($def\ elem$)
$True\ Caso\ probado$

Caso elem o (map *objeto_de* (filter *es_un_objeto* u) =True
Este caso parte de la base de que o == u0´ es falso. Por lo tanto (Right o) $\neq u0$
elem o (map *objeto_de* (filter *es_un_objeto* u) $\rightarrow$ elem (Right o) uo:u
elem o ($objeto\_en\ u$) $\rightarrow$ elem (Right o) uo:u ($def\ objeto\_en$)
Recordando nuestra hipotesis inductiva sabemos que lo siguiente es verdadero:
HI : elem o ($objeto\_en\ u$) $\rightarrow$ elem (Right o) u
Si podemos probar que elem (Right o) u $\rightarrow$ elem (Right o) u0:u
Entonces por transitividad probamos: elem o ($objeto\_en\ u$) $\rightarrow$ elem (Right o) u0:u
$Empecemos$:
elem (Right o) u $\rightarrow$ elem (Right o) u0:u
elem (Right o) u $\rightarrow$ elem (Right o) u
Como sabemos que Right o $\neq u0$ podemos quitarlo de la lista por def de elem.
$A \rightarrow A$ es tautologia por lo tanto queda demostrada la implicacion por transitividad.

Habiendo terminado la implicacion del anterior subcaso queda probado
la demostracion para el caso Right u0.

Caso Left u0

Por ultimo hay que probar que la implicacion original vale con el constructor Left

elem o (map objeto_de (filter es_un_objeto $(u0 : u)$)) $(def\ objetos\_en)$

elem o (map objeto_de (filter es_un_objeto (u))) $(def\ filter\ y\ def\ es\_un\_objeto)$

Como es de constructor Left, u0 es filtrado por la funcion es_un_objeto
quedando la lista u de la misma forma que la hipótesis inductiva.

Habiendo probado todo lo anterior queda comprobada que la implicación:

P(u0:u) $=\forall o$ :: Objeto.elem o (objetos_en $u0 : u$) $\Rightarrow$ elem (Right o) $u0 : u$

es verdadera $\forall u :: Universo$

Entonces probamos que es valida la propiedad ☺

$\forall u$ :: Universo.$\forall o$ :: Objeto.elem o (objetos_en u) $\Rightarrow$ elem (Right o) u