Grupo	Grupo Haskell, Curry & Minions
Campanello, José Luis	1207/88
Delgado, Gonzalo Sebastián	38/17
Rankov González, Jorge Augusto Germán	714/23
Santos, Diego Hernán	874/03

Resolución Ejercicio 12 TP 2

Más abajo se puede observar el enunciado completo.

Resolución Ejercicio 12

El predicado replicar/3 se definió inicialmente como:

```
replicar(+E, +N, -L)
```

Y se implementó de la siguiente forma:

```
replicar(E, N, L) :- length(L, N), maplist(=(E), L).
```

Se pide evaluar si el predicado es reversible en el parámetro N, es decir:

```
replicar(+E, -N, -L)
```

Hay dos posibles comportamientos que se espera, en función de si el parámetro Lista también está definido. Los comportamientos son:

1. Si Lista está definido, el predicado debe fallar si los elementos de Lista no son todos Elem y si no falla, entonces N debe instanciarse a la longitud de Lista.
2. Si Lista no está definido, se espera obtener todas las posibles listas armadas con repetición de Elem (comenzando por 0 repeticiones - lista vacía).

Análisis de Comportamiento 1

En este caso asumimos que L y E tienen valor. El primer predicado que se utiliza es length/2, que es reversible en ambos parámetros, de forma que N tomará el como valor la longitud de la lista.

Luego de esto, se realiza un maplist/2 que verifica que todos los elementos de la lista sean iguales (unifiquen) con E. Entonces, si la lista es la repetición de un valor (o variable sin instanciar), el predicado tendrá éxito y N tomará el valor correcto. Si la lista no consiste de repeticiones del mismo valor, entonces el predicado fallará.

Consideramos que este primer posible comportamiento se verifica en la implementación.

Análisis de Comportamiento 2

En este caso, asumimos que sólo E tiene valor.

El primer predicado que se utiliza es length/2, que toma como parámetros N (longitud) y L (lista) y que por definición, si ninguno de los dos está instanciado (como es este caso), funciona como un generador de listas con elementos sin instanciar, comenzando por la lista de longitud 0, luego longitud 1, etc.

El predicado length/2 entonces genera todas las posibles listas.

El segundo predicado (maplist/2) se encarga de asegurar que todos los elementos de la lista tengan el mismo valor (E), de forma que las listas generadas sean las listas que cuyos elementos tienen el valor indicado por E.

Considerando que length/2, por la forma en que es usado, genera todas las posibles listas, el mismo no terminará nunca de generar casos, de forma que en este escenario, el predicado replicar/3 tampoco terminará nunca de generar casos.

Consideramos que este segundo posible comportamiento se verifica en la implementación.

Pruebas en consola de Prolog

Estas pruebas se pueden verificar por consola:

```
% swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 9.2.9)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free
```

```
software.
Please run ?- license. for legal details.
For online help and background, visit
https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).
?- [nonograma].
true.
?- replicar(c, N, [c, c, c]).
N = 3.
?- replicar(c, N, [c, c, a]).
false.
?- replicar(c, N, L).
N = 0,
L = [] ;
N = 1,
L = [c] ;
N = 2,
L = [c, c] ;
N = 3,
L = [c, c, c] ;
N = 4,
L = [c, c, c, c] ;
N = 5,
L = [c, c, c, c, c] .
?-
```

Información Adicional

Si bien no se solicitó, en rigor, por la forma en que está definido, el predicado replicar/3 es completamente reversible en los tres parámetros (ej: si se le pasa una lista que es la replicación del mismo valor, asigna valor para C y N).

Enunciado Ejercicio 12

Indicar si el predicado replicar/3 es reversible en el segundo argumento. En concreto se pide analizar si replicar(+Elem, -N, -Lista) funciona correctamente.