Ing. de Sistemas e Informática
Matemáticas Discretas
Jorge M. Londoño P.

Práctica 9
Procedimientos recursivos y listas

Introducción

De forma análoga a como en la inducción matemática se define un caso base y se realiza el paso inductivo (para n+1) partiendo de un caso previamente conocido (la hipótesis para n), en programación se pueden definir procedimientos recursivos: Se definen reglas para establecer uno o varios casos base (*e.g.* para n=0) y reglas recursivas que calculen los demás casos haciendo uso de casos anteriores.

Ejemplo 1: Función factorial

La función factorial se define recursivamente así:

$$factorial(n)=\begin{cases} 1 & n=0 \\ n*factorial(n-1) & n>0 \end{cases}$$

Esta función se implementa mediante el siguiente procedimiento recursivo:

```
% factorial/2 : Calcula factorial de n recursivamente
% arg1 = valor n
% arg2 = Resultado n!
factorial(0,1) :- !.
factorial(N,X) :- M is N-1, factorial(M,Y), X is N*Y.
```

El símbolo ! se utiliza en Prolog para indicar un corte. Cuando la evaluación de una regla llega a un corte, el Prolog no evaluará otras reglas y no hará retrocesos a metas anteriores al corte.

Ejemplo 2: Agregando elementos a una lista

El proceso de construcción de una lista es un proceso que se describe fácilmente de forma recursiva. En primer lugar se define un caso base que inicialice la lista, y

una regla recursiva que agregue elementos a esta lista.

El siguiente programa obtiene la lista de factoriales desde un valor inicial M:

```
% listaFactoriales/2 : Lista de factoriales desde M hasta 0
% arg1 : Valor actual
% arg2 : La lista de los factoriales
listaFactoriales(0,L) :- factorial(0,X), L=[X], !.
listaFactoriales(M,L) :- factorial(M,F), N is M-1,
                         listaFactoriales(N,K), L=[F|K].
```

Observamos que para el 0, el factorial se define como 0!=1 y de esta forma se inicializa la lista con el factorial de 0 en el caso base. Para los demás valores de M, se le calcula el factorial a M, se obtiene la lista de todos los factoriales desde N=M-1 y a este resultado se le agrega el factorial de M.

Es muy común la necesidad de utilizar limites arbitrarios de comienzo y fin cuando se hacen procesos recursivos (o iterativos). Esto lo podemos lograr fácilmente utilizando un parámetro adicional para indicar el valor final:

```
% listaFactoriales/3 : Lista de factoriales desde M hasta Z
% arg1 : Valor actual
% arg2 : La lista de los factoriales
% arg3 : Limite inferior del rango a considerar
listaFactoriales(M,L,Z) :- M=:=Z, factorial(M,X), L=[X], !.
listaFactoriales(M,L,Z) :- factorial(M,F), N is M-1,
                           listaFactoriales(N,K,Z), L=[F|K].
```

Ejercicio 1

Leonardo de Pisa (más conocido como Fibonacci) descubrió la sucesión de Fibonacci:

0, 1, 1, 2, 3, 5, 8, 13, ...

la cual tiene innumerables aplicaciones para describir todo tipo de fenómenos naturales.

La secuencia se puede definir recursivamente observando que siempre empieza en 0,1 y que de ahí en adelante todo término es la suma de los dos que le preceden. En forma de recurrencia, la sucesión se define así:

$$fib(n)=\begin{cases} n & n=0,1 \\ fib(n-1)+fib(n-2) & n\geqslant 2 \end{cases}$$

a) Definir un procedimiento recursivo fib/2 que dado un valor n calcule el correspondiente número de fibonacci. El primer argumento es N y el segundo es el resultado.

b) Ilustrar el correcto funcionamiento calculando fib(n) para n=0,2,9,14,27. Comprobar que la respuesta obtenida sea correcta.

c) Se puede calcular fib(40)? fib(50)? fib(100)? Qué ocurre en estos casos?

d) La secuencia no esta definida para n<0. Qué hace el programa si se le pide calcular fib(-1) ? Agregar la condición necesaria al programa para que no se presenten errores en casos como este. Hacer la consulta nuevamente y verificar su correcto funcionamiento.

Ejercicio 2

a) Hacer un procedimiento para obtener la lista de todos los números de Fibonacci desde un valor inicial M hasta 0.

b) Ilustrar el correcto funcionamiento para algunos valores positivos de M (mayores a 10).

c) Ilustrar el correcto funcionamiento para cuando M=0, -1.

Ejercicio 3

a) Hacer un nuevo procedimiento que obtenga la lista de números de Fibonacci descentiente desde un valor inicial M hasta un valor final Z.

b) Ilustrar el correcto funcionamiento del programa para algunas combinaciones de M y Z, tales que M>Z.

c) Qué ocurre cuando M=Z? Y cuando M<Z? Son correctos estos casos? Corregir el programa en caso de ser necesario y hacer la consulta nuevamente para verificar su correcto funcionamiento.

Una nota práctica:

Cuando una lista es larga, el Prolog la imprime en forma abreviada. Para imprimir la lista completa hay dos opciones[1]:

- Presionar **w** cuando se muestra el resultado de una consulta y el Prolog espera el Enter o el ';' (reintento).
- Cambiar la configuración por defecto para imprimir la lista completa. Esto se logra con la consulta:

```
set_prolog_flag(answer_write_options,[max_depth(0)]).
```

Informe:

Enviar la base de datos (`<ApellidoNombre>-<ID>-bd9.pl`) y el informe incluyendo todas las consultas solicitadas (`<ApellidoNombre>-<ID>-consultas9.txt`).

No olvidar documentar apropiadamente los hechos y reglas en la base de datos.

1 https://stackoverflow.com/questions/8231762/swi-prolog-show-long-list