Especificación y Verificación de Software

Máster en Investigación en Informática (UCM)

Hoja 3 Curso 2009/2010

En los ejercicios que sea posible, ejecuta los módulos en Maude y comprueba que no se producen errores.

Ejercicio 1 Considera los siguientes módulos:

```
fmod NAT1 is
  sort Nat .
  op 0 : -> [Nat] .
  op s : [Nat] -> [Nat] .

  var N : [Nat] .
  mb 0 : Nat .
  cmb s(N) : Nat if N : Nat .
endfm
```

```
fmod NAT2 is
  sort Nat .
  op 0 : -> [Nat] .
  op s : [Nat] -> [Nat] .

  var N : [Nat] .
  mb 0 : Nat .
  mb s(N) : Nat .
endfm
```

¿Son equivalentes? ¿Y si se añaden las declaraciones

```
  sort NatInf .
  subsort Nat < NatInf .
  op infinity : -> NatInf .
```

a cada uno de ellos? ¿Cuáles son las álgebras iniciales de los módulos así extendidos? Señala, en particular, las interpretaciones de las familias y de los tipos.

Ejercicio 2 Define una teoría que importe los números naturales y distinga uno entre ellos. Utilízala para escribir un módulo parametrizado que define un predicado `mult` que devuelva `true` para los múltiplos de dicho número distinguido y `false` en los demás casos. Prueba que funciona utilizando distintas vistas.

Ejercicio 3 Define un módulo parametrizado por la teoría del ejercicio anterior en el que se defina el tipo de las clases de equivalencia módulo el número distinguido. Declara una operación que asocie a cada número su clase, así como dos operaciones para la suma y el producto de clases de equivalencia, con sus correspondientes ecuaciones.

Ejercicio 4 Construye un módulo parametrizado `PAIR{X :: TRIV, Y :: TRIV}` que tome dos argumentos y defina el tipo de los pares de elementos pertenecientes a los tipos pasados como argumento. Utiliza este módulo para definir `PAIR-LEX{X :: STOSET, Y :: STOSET}`, que defina el orden lexicográfico sobre dichos pares de elementos.

Ejercicio 5 Construye un módulo que importe `BIN-TREE{X :: TRIV}` y que contenga una operación `op niveles : BinTree{X} -> List{X}` que devuelva el recorrido de un árbol *por niveles*. Indicación: utiliza una cola de árboles.

Ejercicio 6 Define un módulo parametrizado para describir los *árboles generales* cuyos nodos contienen elementos de un tipo dado y tienen un número variable de hijos. Especifica al menos las siguientes operaciones: calcular la altura de un árbol, el número de hojas, el grado de un árbol, el recorrido en preorden y el recorrido por niveles.

Ejercicio 7 Construye una teoría `DATABASE{X :: DATA}` que construya una base de datos (representada como un conjunto, o lista, o multiconjunto, ...) sobre elementos del tipo que se le pasa como argumento. La teoría `DATA` debe definir, además del tipo principal, dos tipos adicionales que servirán de identificadores de los datos; uno de ellos deberá ser unívoco, lo que vendrá expresado con las correspondientes ecuaciones. La base de datos debe contener operaciones que permitan realizar búsquedas sobre los dos tipos de identificadores.