\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* NOMBRE \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

fmod NATURAL is

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* CONJUNTOS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

protecting BOOL .

sort N .

sort NoN .

subsort NoN < N .

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* SINTAXIS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

op cero : -> N .

op sucesor : N -> N .

op suma : N N -> N .

op esCero : N -> Bool .

op esIgual : N N -> Bool .

op esDistinto : N N -> Bool .

op producto : N N -> N .

op potencia : N N -> N .

op cuadrado : N -> N .

op factorial : N -> N .

op NODEFINIDO : -> NoN .

op INFINITO : -> NoN .

op NEGATIVO : -> NoN .

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* SEMANTICA \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

var n m : N .

eq suma(cero, n) = n .

eq suma(sucesor(m), n) = sucesor(suma(m, n)) .

eq esCero(cero) = true .

eq esCero(sucesor(n)) = false .

eq esIgual(cero, n) = esCero(n) .

eq esIgual(sucesor(n), cero) = false .

eq esIgual(sucesor(n), sucesor(m)) = esIgual(n, m) .

eq esDistinto(n, m) = not esIgual(n, m) .

eq producto(cero, cero) = cero .

eq producto(cero, n) = cero .

eq producto(cero, sucesor(n)) = cero .

eq producto(sucesor(cero), sucesor(cero)) = sucesor(cero) .

eq producto(sucesor(cero), n) = n .

eq producto(sucesor(cero), sucesor(n)) = sucesor(n) .

eq producto(sucesor(m), n) = suma(producto(m, n), n) .

eq potencia(cero, cero) = NODEFINIDO .

eq potencia(n, cero) = sucesor(cero) .

eq potencia(cero, n) = cero .

eq potencia(sucesor(cero), n) = sucesor(cero) .

eq potencia(n, sucesor(cero)) = n .

eq potencia(n, sucesor(n)) = producto(potencia(n, n), n) .

eq potencia(n, sucesor(m)) = producto(potencia(n, m), n) .

eq cuadrado(cero) = cero .

eq cuadrado(sucesor(cero)) = sucesor(cero) .

eq cuadrado(n) = potencia(n, sucesor(sucesor(cero))) .

eq factorial(cero) = sucesor(cero) .

eq factorial(sucesor(cero)) = sucesor(cero) .

eq factorial(sucesor(sucesor(cero))) = sucesor(sucesor(cero)) .

eq factorial(sucesor(n)) = producto(factorial(n), sucesor(n)) .

endfm