\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 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 esMenor : N N -> Bool .

op esMenorIgual : N N -> Bool .

op esMayor : N N -> Bool .

op esMayorIgual : N N -> Bool .

op maximo : N N -> N .

op minimo : N 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)) .

eq esMenor(cero, cero) = false .

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

eq esMenor(sucesor(sucesor(n)), n) = false .

eq esMenor(n, sucesor(sucesor(n))) = true .

eq esMenor(sucesor(cero), sucesor(sucesor(n))) = true .

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

eq esMenor(cero, sucesor(n)) = true .

eq esMenor(sucesor(cero), cero) = false .

eq esMenor(cero, sucesor(cero)) = true .

eq esMenor(sucesor(cero), sucesor(cero)) = false .

eq esMenor(n, n) = false .

eq esMenor(n, sucesor(n)) = true .

eq esMenor(sucesor(n), n) = false .

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

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

eq esMenorIgual(cero, cero) = true .

eq esMenorIgual(sucesor(sucesor(n)), n) = false .

eq esMenorIgual(n, sucesor(sucesor(n))) = true .

eq esMenorIgual(sucesor(cero), sucesor(sucesor(n))) = true .

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

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

eq esMenorIgual(cero, sucesor(n)) = true .

eq esMenorIgual(sucesor(cero), cero) = esMenor(sucesor(cero), cero) .

eq esMenorIgual(cero, sucesor(cero)) = esMenor(cero, sucesor(cero)) .

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

eq esMenorIgual(n, n) = esMenorIgual(cero, cero) .

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

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

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

eq esMenorIgual(n, sucesor(m)) = esMenorIgual(n, suma(sucesor(cero), m))

eq esMayor(cero, cero) = false .

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

eq esMayor(n, sucesor(sucesor(n))) = false .

eq esMayor(sucesor(sucesor(n)), n) = true .

eq esMayor(sucesor(sucesor(n)), sucesor(cero)) = true .

eq esMayor(sucesor(n), cero) = true .

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

eq esMayor(sucesor(cero), cero) = true .

eq esMayor(cero, sucesor(cero)) = false .

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

eq esMayor(n, n) = esMayor(cero, cero) .

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

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

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

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

eq esMayorIgual(cero, cero) = true .

eq esMayorIgual(sucesor(sucesor(n)), n) = true .

eq esMayorIgual(n, sucesor(sucesor(n))) = false .

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

eq esMayorIgual(sucesor(sucesor(n)), sucesor(cero)) = true .

eq esMayorIgual(n, sucesor(sucesor(n))) = false .

eq esMayorIgual(sucesor(n), cero) = true .

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

eq esMayorIgual(sucesor(cero), cero) = true .

eq esMayorIgual(cero, sucesor(cero)) = false .

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

eq esMayorIgual(n, n) = esMayorIgual(cero, cero) .

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

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

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

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

eq maximo(cero, cero) = cero .

eq maximo(n, n) = n .

eq maximo(n, sucesor(sucesor(n))) = sucesor(sucesor(n)) .

eq maximo(sucesor(sucesor(n)), n) = sucesor(sucesor(n)) .

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

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

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

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

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

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

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

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

eq maximo(n, sucesor(n)) = sucesor(n) .

eq maximo(sucesor(n), n) = sucesor(n) .

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

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

eq minimo(cero, cero) = cero .

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

eq minimo(sucesor(n), sucesor(n)) = sucesor(n) .

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

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

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

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

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

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

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

eq minimo(n, sucesor(n)) = n .

eq minimo(sucesor(n), n) = n .

eq minimo(sucesor(sucesor(n)), n) = n .

eq minimo(n, sucesor(sucesor(n))) = n .

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

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

eq minimo(n, n) = n .

endfm

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

fmod VOCAL is

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

protecting BOOL .

sort V .

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

ops A E I O U : -> V .

op esIgual : V V -> Bool .

op esDistinta : V V -> Bool .

op esMenor : V V -> Bool .

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

var v w : V .

eq esIgual(v, v) = true .

eq esIgual(v, w) = false .

eq esDistinta(v, w) = not esIgual(v, w) .

eq esMenor(v, v) = false .

eq esMenor(A, E) = true .

eq esMenor(A, I) = true .

eq esMenor(A, O) = true .

eq esMenor(A, U) = true .

eq esMenor(E, I) = true .

eq esMenor(E, O) = true .

eq esMenor(E, U) = true .

eq esMenor(I, O) = true .

eq esMenor(I, U) = true .

eq esMenor(O, U) = true .

eq esMenor(U, O) = false .

eq esMenor(U, I) = false .

eq esMenor(U, E) = false .

eq esMenor(U, A) = false .

eq esMenor(O, I) = false .

eq esMenor(O, E) = false .

eq esMenor(O, A) = false .

eq esMenor(I, E) = false .

eq esMenor(I, A) = false .

eq esMenor(E, A) = false .

endfm

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

fmod ARBOLBINARIODEVOCALES is

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

protecting BOOL .

protecting NATURAL .

protecting VOCAL .

sort AB .

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

op arbolVacio : -> AB .

op esVacio : AB -> Bool .

op construir : V AB AB -> AB .

op construirRaiz : V -> AB .

op altura : AB -> N .

op numNodos : AB -> N .

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

var v w t : V .

var a1 a2 : AB .

var n : N .

eq esVacio(arbolVacio) = true .

eq esVacio(construir(v, a1, a2)) = false .

eq construirRaiz(v) = construir(v, arbolVacio, arbolVacio) .

eq altura(arbolVacio) = cero .

eq altura(construir(v, a1, a2)) = sucesor(maximo(altura(a1), altura(a2))) .

eq numNodos(arbolVacio) = cero .

eq numNodos(construir(v, a1, a2)) = sucesor(suma(numNodos(a1), numNodos(a2))) .

endfm