```
Ejercicio 12 - Demostracion

A) P(e) : cantLit e = S (cantOp e)
 Queremos demostrar que vale ∀e :: Expr. P(e)
B)  El tipo es:
    data Expr = Const Float
                | Rango Float Float
                | Suma Expr Expr
                | Resta Expr Expr
                | Mult Expr Expr
                | Div Expr Expr

Esquema de inducción estructural:

    Casos Base:
            ∀n :: Float. P(Const n)
            ∀a, b :: Float. P(Rango a b)

    Casos inductivos:
            ∀e1, e2 :: Expr. P(e1) ∧ P(e2) ⇒ P(Suma e1 e2)
            ∀e1, e2 :: Expr. P(e1) ∧ P(e2) ⇒ P(Resta e1 e2)
            ∀e1, e2 :: Expr. P(e1) ∧ P(e2) ⇒ P(Mult e1 e2)
            ∀e1, e2 :: Expr. P(e1) ∧ P(e2) ⇒ P(Div e1 e2)

C)  data Nat = Z | S Nat

    suma :: Nat → Nat → Nat
    suma Z m = m -- {S1}
    suma (S n) m = S ( suma n m) -- {S2}

    cantLit :: Expr → Nat
    cantLit ( Const _) = S Z -- {L1}
    cantLit ( Rango _ _) = S Z -- {L2}
    cantLit ( Suma a b) = suma ( cantLit a) ( cantLit b) -- {L3}
    cantLit ( Resta a b) = suma ( cantLit a) ( cantLit b) -- {L4}
    cantLit ( Mult a b) = suma ( cantLit a) ( cantLit b) -- {L5}
    cantLit (Div a b) = suma ( cantLit a) ( cantLit b) -- {L6}

    cantOp :: Expr → Nat
    cantOp ( Const _) = Z -- {O1}
    cantOp ( Rango _ _) = Z -- {O2}
    cantOp ( Suma a b) = S ( suma ( cantOp a) ( cantOp b)) -- {O3}
    cantOp ( Resta a b) = S ( suma ( cantOp a) ( cantOp b)) -- {O4}
    cantOp ( Mult a b) = S ( suma ( cantOp a) ( cantOp b)) -- {O5}
    cantOp ( Div a b) = S ( suma ( cantOp a) ( cantOp b)) -- {O6}

    Asumimos el siguiente lema como valido.
    {CONMUT} ∀n, m :: Nat. suma n m = suma m n

    Casos Base:
        ∀n :: Float. P(Const n)
        ∀n :: Float. cantLit (Const n) = S (cantOp (Const n))
                                   S Z = S Z                    {L1} ∧ {O1}
```

```
        ∀a, b :: Float. P(Rango a b)
        ∀a, b :: Float. cantLit (Rango a b) = S (cantOp (Rango a b))
                                        S Z = S Z                {L2} ∧ {O2}


    Caso Inductivo: ∀e1, e2 :: Expr. (P(e1) ∧ P(e2)) ⇒ P(Suma e1 e2)
                    Suponemos que P(e1) ∧ P(e2) es verdadero

    HI: cantLit e1 = S (cantOp e1) ∧ cantLit e2 = S (cantOp e2)

    Queremos probar: cantLit (Suma e1 e2) = S (cantOp (Suma e1 e2))

    ⇒ cantLit (Suma e1 e2) = suma (cantLit e1) (cantLit e2)       {L3}
                            = suma (S cantOp e1) (S cantOp e2)     HI
                            = S (suma cantOp e1 (S cantOp e2))     {S2}
                            = S (suma (S cantOp e2) cantOp e1)     {CONMUT}
                            = S (S (suma cantOp e2 cantOp e1))     {S2}
                            = S (S (suma cantOp e1 cantOp e2))     {CONMUT}
                            = S (S (suma (cantOp e1) (cantOp e2))) Agregamos
Parentesis
                            = S (cantOp (Suma e1 e2))              {O3}

    Con esto queda probado por induccion estructural.

    Para Resta, Mult y Div es analogo. Cambiariamos {L3}/{O3} por {L4}/{O4},
{L5}/{O5}, {L6}/{O6} respectivamente.
```