Haskell — Peano P92085_ca

Definim els nombres de naturals de la forma següent:

```
data Nat = Z | S Nat deriving Show
```

A més, definim la funció genèrica d'ordre superior següent:

```
rec :: a → (Nat → a → a) → Nat → a
rec base step Z = base
rec base step (S n) = step n (rec base step n)
```

Definiu les les funcions següents, tenint en compte que només podeu substituir els `undefined` per expressions no recursives i que no podeu usar operacions dels enters.

```
isEven :: Nat → Bool      -- indica si un natural és parell o no
isEven = rec base step
  where
    base = undefined
    step = undefined

add :: Nat → Nat → Nat    -- retorna la suma de dos naturals
add = rec base step
  where
    base = undefined
    step = undefined

mul :: Nat → Nat → Nat    -- retorna el producte de dos naturals
mul = rec base step
  where
    base = undefined
    step = undefined

fact :: Nat → Nat         -- retorna el factorial d'un natural
fact = rec base step
  where
    base = undefined
    step = undefined
```

Observacions

Descarregueu-vos el fitxer `code.hs` i completeu-lo.

mul pot usar *add* i *fact* pot usar *mul*.

El Jutge dóna puntuacions parcials per a cada funció (20 punts) i per l'exemple (20 punts).

A l'hora de corregir es tindrà en compte la correcció, la consició, la senzillesa, l'elegància de la solució proposada, però no l'eficiència.

Exemple d'entrada	Exemple de sortida
`map isEven [Z, S Z, S (S Z), S (S (S Z))]`	`[True,False,True,False]`
`add (S (S (S Z))) (S (S Z))`	`S (S (S (S (S Z))))`
`mul (S (S (S Z))) (S (S Z))`	`S (S (S (S (S (S Z)))))`
`fact (S (S (S Z)))`	`S (S (S (S (S (S Z)))))`

Informació del problema

Autor : Jordi Petit
Generació : 2023-04-21 15:39:08