Documentación Técnica del Archivo `prelude.lam`

Este archivo implementa una serie de funciones en cálculo lambda utilizando codificaciones de Church y otras construcciones funcionales típicas del formalismo. A continuación, se detalla cada función.

BOOLEANOS (Codificación de Church)

`true`

Definición: `\t f. t`
Descripción: Representa el valor booleano verdadero.
Parámetros: Dos argumentos (t, f).
Retorno: Retorna el primer argumento.
Ejemplo: `true x y` devuelve `x`.

`false`

Definición: `\t f. f`
Descripción: Representa el valor booleano falso.
Parámetros: Dos argumentos (t, f).
Retorno: Retorna el segundo argumento.
Ejemplo: `false x y` devuelve `y`.

`if`

Definición: `\b x y. b x y`
Descripción: Estructura condicional.
Parámetros: Un booleano b y dos alternativas.
Retorno: `x` si `b = true`, `y` si `b = false`.
Ejemplo: `if true 1 2` devuelve `1`.

`not`

Definición: `\x. if x false true`
Descripción: Negación lógica.
Parámetros: Un booleano.
Retorno: Su valor negado.
Ejemplo: `not true` devuelve `false`.

`and`

Definición: `\x y. if x y false`
Descripción: Conjunción lógica.
Parámetros: Dos booleanos.
Retorno: `true` si ambos son `true`.
Ejemplo: `and true false` devuelve `false`.

`or`

Definición: `\x y. if x true y`
Descripción: Disyunción lógica.
Parámetros: Dos booleanos.
Retorno: `true` si alguno es `true`.
Ejemplo: `or true false` devuelve `true`.

`xor`

Definición: `\x y. if x (not y) y`
Descripción: Disyunción exclusiva.
Parámetros: Dos booleanos.
Retorno: `true` si exactamente uno es `true`.
Ejemplo: `xor true false` devuelve `true`.

NÚMEROS NATURALES (Numerales de Church)

`zero`

Definición: `\f x. x`
Descripción: Representa el número 0.
Parámetros: Una función f y un valor x.
Retorno: x sin aplicar f.
Ejemplo: `zero f x` devuelve `x`.

`succ`

Definición: `\n f x. f (n f x)`
Descripción: Sucesor de un número.
Parámetros: Un numeral n.
Retorno: El número n+1.
Ejemplo: `succ zero` representa `one`.

---

pred

Definición: `\n f x. n (\g h. h (g f)) (\u. x) (\u. u)`
Descripción: Predecesor de un número.
Parámetros: Un numeral.
Retorno: n - 1 (o zero si n = 0).
Ejemplo: `pred one` devuelve `zero`.

---

plus

Definición: `\m n f x. m f (n f x)`
Descripción: Suma de dos numerales.
Parámetros: Dos numerales.
Retorno: m + n.
Ejemplo: `plus two three` es `five`.

---

sub

Definición: `\m n. (n pred) m`
Descripción: Resta de numerales.
Parámetros: m y n.
Retorno: m - n.
Ejemplo: `sub five two` devuelve `three`.

---

mul

Definición: `\m n f. m (n f)`
Descripción: Multiplicación.
Parámetros: Dos numerales.
Retorno: m * n.
Ejemplo: `mul two three` devuelve `six`.

---

even

Definición: `\n. n not true`
Descripción: Predicado de paridad.
Parámetros: Un numeral.
Retorno: `true` si n es par.
Ejemplo: `even two` devuelve `true`.

---

iszero

Definición: `\m. m (\x. false) true`
Descripción: Predicado de cero.
Parámetros: Un numeral.

Retorno: `true` si m = zero.
Ejemplo: `iszero zero` devuelve `true`.

`lte`, `gte`, `eq`, `lt`, `gt`

Comparaciones entre numerales basadas en `sub`, `iszero` y combinaciones lógicas:

- `lte`: `\m n. iszero (sub m n)` — menor o igual.
- `gte`: `\m n. iszero (sub n m)` — mayor o igual.
- `eq`: `\m n. and (lte m n) (gte m n)` — igualdad.
- `lt`: `\m n. and (lte m n) (not (gte m n))` — menor que.
- `gt`: `\m n. and (gte m n) (not (lte m n))` — mayor que.

Ejemplos: - `eq two two` → `true` - `lt one two` → `true` - `gt three one` → `true`

Alias de Numerales

Numerales definidos como sucesores:

- `one = succ zero`
- `two = succ one`
- `three = succ two`
- `four = succ three`
- ...
- `ten = succ nine`

Cada uno representa el número correspondiente en codificación de Church.