6. O diagrama seguinte representa o combinador *catamorfismo* (de naturais) que se começou a estudar na última aula teórica, onde a notação $(\!|\, g \,|\!)$ abrevia $cata\ g$ então usada:

$$
\begin{tikzcd}
\mathbb{N}_0 \arrow[d, "(\!| g |\!)"'] & 1 + \mathbb{N}_0 \arrow[l, "\mathsf{in}"'] \arrow[d, "id + (\!| g |\!)"] \\
B & 1 + B \arrow[l, "g"]
\end{tikzcd}
\qquad \text{onde} \qquad
\left\{
\begin{array}{l}
\mathsf{in} = [\mathsf{zero}\ , \mathsf{succ}] \\
\mathsf{zero}\ \_ = 0 \\
\mathsf{succ}\ n = n + 1
\end{array}
\right.
$$

Assumindo a seguinte propriedade universal desse combinador,

$$k = (\!|\, g \,|\!) \ \equiv\ k \cdot \mathsf{in} = g \cdot (id + k) \qquad \text{(F4)}$$

mostre que o combinador “ciclo-for”definido por

$$\mathsf{for}\ b\ i = (\!|\, [\underline{i}\ , b] \,|\!)$$

se converte na seguinte versão “pointwise”:

$$\mathsf{for}\ b\ i\ 0 = i$$
$$\mathsf{for}\ b\ i\ (n+1) = b\ (\mathsf{for}\ b\ i\ n)$$

Resolução

$$for\ b\ i = (\!|[\underline{i}, b]|\!)$$

{ propriedade universal, fazendo $k = for\ b\ i$ }

$$for\ b\ i = (\!|[\underline{i}, b]|\!) \equiv (for\ b\ i)\ .\ in = [\underline{i}, b]\ .\ (id + (for\ b\ i))$$

{ def. in = [zero, succ] }

$$\equiv (for\ b\ i)\ .\ [zero, succ] = [\underline{i}, b]\ .\ (id + (for\ b\ i))$$

{ fusão-$+$, **lei (20)** ; absorção-$+$, **lei (22)** }

$$\equiv [(for\ b\ i)\ .\ zero, (for\ b\ i)\ .\ succ] = [\underline{i}\ .\ id, b\ .\ (for\ b\ i)]$$

{ natural-id, **lei (1)** ; eq-$+$, **lei (27)** }

$$(for\ b\ i)\ .\ zero = \underline{i}$$

$$(for\ b\ i)\ .\ succ = b\ .\ (for\ b\ i)$$

{ igualdade extensional, **lei (71)** }

$$((for\ b\ i)\ .\ zero)\ n = \underline{i}\ n$$

$$((for\ b\ i)\ .\ succ)\ n = (b\ .\ (for\ b\ i))\ n$$

{ def-comp, **lei (72)** }

$$for\ b\ i\ (zero\ n) = \underline{i}\ n$$

$$for\ b\ i\ (succ\ n) = b\ (for\ b\ i\ n)$$

{ def. zero, def. succ, def-const, **lei (74)** }

$$for\ b\ i\ 0 = i$$

$$for\ b\ i\ (n+1) = b\ (for\ b\ i\ n)$$

Haskell

In [1]:

```
for b i 0 = i
for b i n = b (for b i (n-1))

-- type checking

:t for

-- testing for succ, (*2), n!
body (x,y)=(x+1,x*y)

for succ 1 4
for (*2) 1 4
for body (1,1) 5
fac = snd . for body (1,1)
fac 5
```

```
for :: forall t1 t2. (Eq t1, Num t1) => (t2 -> t2) -> t2 -> t1 -> t2
```

```
5
16
(6,120)
120
```