(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

$n \leq m \iff (\exists k : \mathbb{N}_0)(n + k = m)$

$even\ x \iff (\exists k)[x = 2 \cdot k]$

$odd\ x \iff (\exists k')[x = 2 \cdot k' + 1]$

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

product : ListNat → Nat

tidy : ListNat → Bool

takeEvens, *atEvens* : ListNat → ListNat

replaceAt, *addAt* : Nat → Nat → ListNat → ListNat

addNat, *removeAt* : Nat → ListNat → ListNat

pwAdd : ListNat → ListNat → ListNat

RESPOSTA. **Não** repita as tipagens na resposta!

```
product [] = 1
product (x:xs) = x * (product xs)
```

```
tidy [] = True
tidy (x:xs) = (even x) ∧ [¬(tidy xs)]
```

```
takeEvens [] = []
takeEvens (x:xs) =
    | even x = x : (takeEvens xs)
    | otherwise = takeEvens xs
```

```
pwAdd _ [] = []
pwAdd [] _ = []
pwAdd (x:xs) (y:ys) = (x+y) : (pwAdd xs ys)
```

```
removeAt 0 (x:xs) = xs
removeAt S(n) (x:xs) = x : (removeAt n xs)
```

```
addNat 0 xs = xs
addNat _ [] = []
addNat n (x:xs) = (n+x) : (addNat n xs)
```

```
replicate n 0 (x:xs) = n : xs
replicate n S(p) (x:xs) = x : (replicate n p xs)
replicate _ _ [] = error "lista vazia"
```

```
addAt n 0 (x:xs) = (n+x) : xs
addAt n S(p) (x:xs) = x : (addAt n p xs)
addAt _ _ [] = error "lista vazia"
```

funções auxiliares:

```
even : Nat → Bool
even 0 = True
even S0 = False
even SS(n) = even n
```

```
length : ListNat → Nat
length [] = 0
length (x:xs) = S(length xs)
```

```
atEvens [] = []
atEvens (x:xs) =
```

Q 1 2 3 4

[1, 11, 2, 15, 0]

(0, 1, 2)

True ∧ (False ∧ (True ∧ (False ∧ True)))

False ∧ False

True even 4 ∧ [¬False] ∧

xiss é p

[4] Tempo

4 : [ ]

(9) C

yuri $(a^b)^{s(0)}$

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[a^{b\cdot c} = (a^b)^c]$:

3

$*_1 : (\cdot)1$
$*_2 : (\wedge)1$

Sejam a, b, inteiros.
Faremos indução em b.
Para o caso base, calculamos:
$a^{b\cdot 0} \overset{*_1}{=} a^0 \overset{*_2}{=} s(0)$, e também $(a^b)^0 = (a^b)^{\wedge}(0) \overset{*_1}{=} s(0)$, então
— temos que $a^{b\cdot 0} = (a^b)^0$, por transitividade.

suponha que $a^{(b\cdot c)} = (a^b)^c$.
calculemos:
$a^{(b\cdot c)} \cdot a = a^{s(b\cdot c)} =$?

e
$(a^b)^c \cdot a =$ ...

?! mid-level!

alguém o tempo ☼

(18) T

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m$ ou $m \leq n]$:

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

product : ListNat → Nat

tidy : ListNat → Bool

takeEvens, *atEvens* : ListNat → ListNat

replaceAt, *addAt* : Nat → Nat → ListNat

addNat, *removeAt* : Nat → ListNat → ListNat

pwAdd : ListNat → ListNat → ListNat

Ex: [1,2,3] [10,12,13] = [11, 14, 16].

RESPOSTA. **Não** repita as tipagens na resposta!

```
product
product Nil = 1
product x:[] = x
product x:xs = x * (product x':xs')
```

(9) **C**

1

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[a^{b\cdot c} = (a^b)^c]$:

2

Sejam a, b : Nat.
Indução no c.
Base: $a^{b\cdot 0} = (a^b)^0$

~~Calculemos:~~

~~Sejam x : Nat tq $x = a^b$~~
~~Calculemos:~~
~~Passo indutivo: $a^{b\cdot k} = (a^b)^k \Rightarrow a^{b\cdot k+1} = (a^b)^{k+1}$~~

Sejam x : Nat tq $x = a^b$
Calculemos:
$a^{b\cdot 0} = a^0$
$= 1$
$= x^0$
$= (a^b)^0$

Passo indutivo: $a^{b\cdot k} = (a^b)^k \Rightarrow a^{b\cdot k+1} = (a^b)^{k+1}$
Seja k : Nat tq $k \geq 1$.
Suponha $a^{b\cdot k} = (a^b)^k$.

~~Calculemos:~~
Logo $a^{b\cdot k} \cdot a = (a^b)^k \cdot a$ [multiplicando ambos os lados]
Logo $a^{b\cdot k+1} = (a^b)^{k+1}$?!?!

assim, mesmo fechando tua demons. tu acabe tendo demonstrado só para o 0.

como isso ajuda?!?!

(18) **T**

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

~~Indução no n:~~
~~Seja m : Nat.~~
~~Base: $0 \leq m$ ou $m \leq 0$.~~
~~Caso $0 \leq m$:~~
~~Seja n : Nat.~~

~~Indução no n:~~
~~Seja m : Nat~~

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

0

≤ : Nat × Nat → ~~BoolProp~~
: 0 ≤ n → ~~Todos~~ 0 é menor que n;
: n ≤ 0 → ~~Todos~~ n não é menor que 0

Ev : Nat → Prop
: 0 → 0 é par
: S0 → S0 não é par
: SSn → Ev n

Od : Nat → Prop
: 0 → 0 ~~Não~~ é ímpar
: S0 → S0 é ímpar
: SSn → Od n

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

$$\begin{aligned} \mathit{product} &: \mathsf{ListNat} \to \mathsf{Nat} & \mathit{replaceAt}, \mathit{addAt} &: \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{ListNat} \\ \mathit{tidy} &: \mathsf{ListNat} \to \mathsf{Bool} & \mathit{addNat}, \mathit{removeAt} &: \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat} \\ \mathit{takeEvens}, \mathit{atEvens} &: \mathsf{ListNat} \to \mathsf{ListNat} & \mathit{pwAdd} &: \mathsf{ListNat} \to \mathsf{ListNat} \to \mathsf{ListNat} \end{aligned}$$

RESPOSTA. **Não** repita as tipagens na resposta!

(9) **C**

mid-level!

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[a^{b\cdot c} = (a^b)^c]$:

Toda casa ①
Indução
Sejam a, b ① $a^{b\cdot 0} \stackrel{?}{=} (a^b)^0$
Base

Calculamos
$a^{b\cdot 0} = a^0$ [· -1]
$= 1$ [$\wedge$ -1]
$= (a^b)^0$ [<-]

← Consegues escrever além do ASCII com a mão!

4

Passo Indutivo
$(\forall k)[(\forall a)(\forall b)[a^{b\cdot k} = (a^b)^k] \Rightarrow [(\forall a)(\forall b)[a^{b\cdot sk} = (a^b)^{sk}]]$
Seja k t.l.q. $(\forall a)(\forall b)[a^{b\cdot k} = (a^b)^k]$ (H.I) ① Sejam a, b ①

aqui não!

Calculamos:
$a^{b\cdot sk} = a^{b\cdot k + b}$ [· - 2]
$= (a^b)^k + b$ [H.I.]
$= a^{b\cdot sk}$ [· <- 2]

...mesmo prob.

(18) **T**

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

Só isso mesmo.

a+b> a,b=0

(8) **I**

0 INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

SEJAM a e b naturais
ENTÃO a e b = 0 ou
a = 0, b > 0 ou
b = 0, a > 0. LOGO a ≤ b ou b ≤ a.

(18) **R**

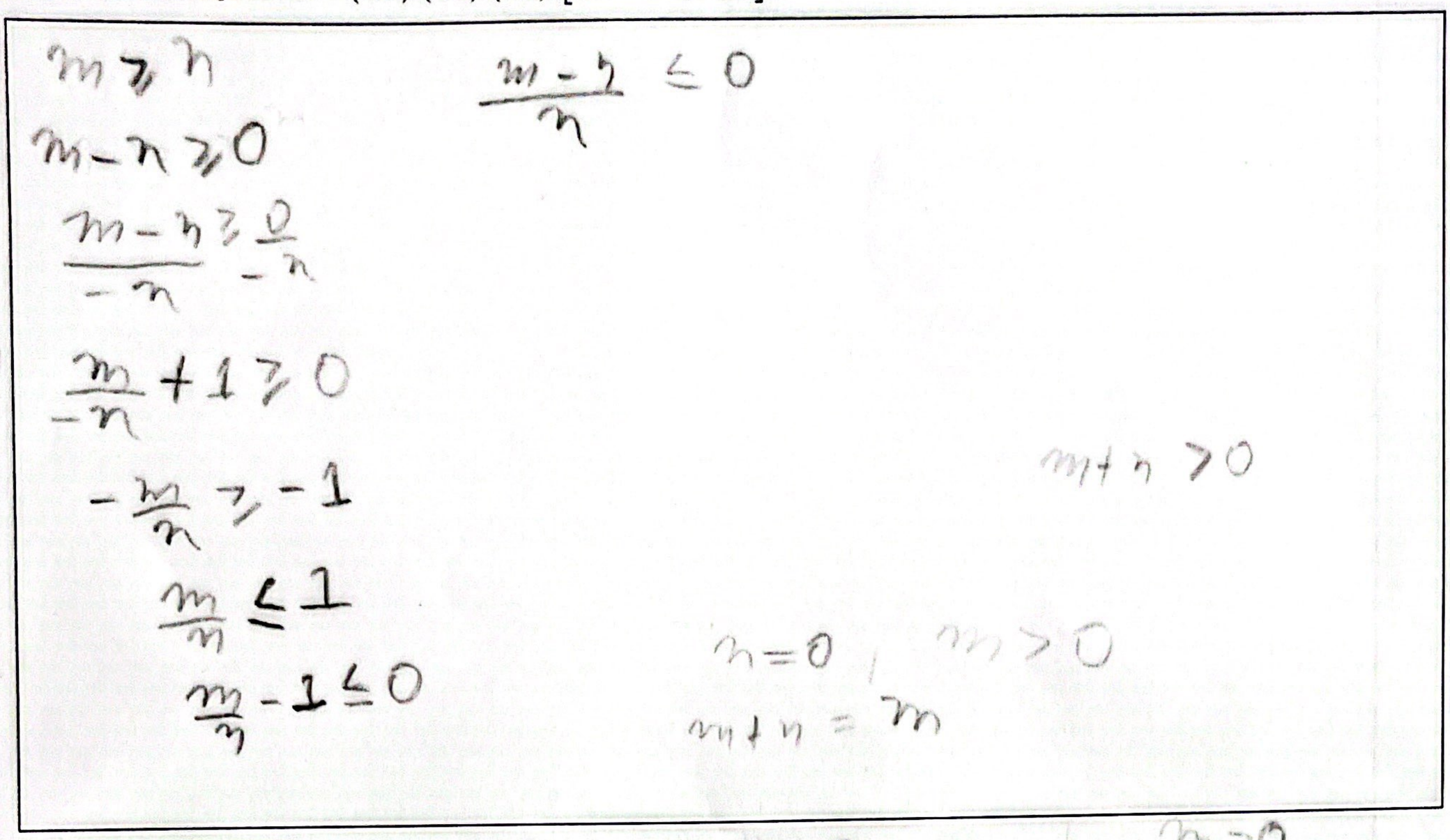
Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

$$\begin{aligned} \mathit{product} &: \mathsf{ListNat} \to \mathsf{Nat} & \mathit{replaceAt}, \mathit{addAt} &: \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{ListNat} \\ \mathit{tidy} &: \mathsf{ListNat} \to \mathsf{Bool} & \mathit{addNat}, \mathit{removeAt} &: \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat} \\ \mathit{takeEvens}, \mathit{atEvens} &: \mathsf{ListNat} \to \mathsf{ListNat} & \mathit{pwAdd} &: \mathsf{ListNat} \to \mathsf{ListNat} \to \mathsf{ListNat} \end{aligned}$$

RESPOSTA. **Não** repita as tipagens na resposta!

(9) **C**

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,\left[a^{b\cdot c} = (a^b)^c\right]$:



(18) **T**

0 DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

SEJAM n, m naturais
SEJA A UM CONJUNTO dos números naturais t.q
$n \in A$ e $m \in A$
ENTÃO $n \geq 0$ e $m \geq 0$

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

product : ListNat → Nat
tidy : ListNat → Bool
takeEvens, *atEvens* : ListNat → ListNat
replaceAt, *addAt* : Nat → Nat → ListNat
addNat, *removeAt* : Nat → ListNat → ListNat
pwAdd : ListNat → ListNat → ListNat

RESPOSTA. **Não** repita as tipagens na resposta!

```
product : ListNat → Nat
product [] = S O
product (x:xs) = (n . x) : product n xs

takeEvens : ListNat → ListNat
takeEvens [] = []
takeEvens (x:xs)
  | isEven x = x : takeEvens xs
  | otherwise = takeEvens xs

pwAdd : ListNat → ListNat → ListNat
pwAdd [] _ = []
pwAdd _ [] = []
pwAdd (x:xs) (y:ys) = (x+y) : pwAdd xs ys

addNat : Nat → ListNat → ListNat
addNat n [] = []
addNat n (x:xs) = (n+x) : addNat n xs

removeAt : Nat → ListNat → ListNat
removeAt n (x:xs)

Auxiliar
isEven : Nat → Bool
isEven 0 = True
isEven (S 0) = False
isEven (S (S n)) = isEven n
```

(9) **C**

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[a^{b \cdot c} = (a^b)^c]$:

(10) **T**

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

9

Seja n : Nat

Indução

Base:
Seja m : Nat
Escolho - L
- Testemunho m
Calculamos:

$0 + m = m + 0 \quad [(+).\text{Com}]$

$= m \quad [(+).1]$

Passo Indutivo

Seja $k \in \mathbb{N}$ $(\forall m)\,[k \leq m \text{ ou } m \leq k]$

Seja m : Nat

Caso $m \leq k$:

Temos $k \leq Sk$ [Lemma 1]

Como $m \leq k$ e $k \leq Sk$, logo $m \leq Sk$

[(≤).succ]

Caso $k \leq m$:

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

isso não é um padrão, pois ev não é um construtor.

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

product : ListNat → Nat

tidy : ListNat → Bool

takeEvens, *atEvens* : ListNat → ListNat

replaceAt, *addAt* : Nat → Nat → ListNat

addNat, *removeAt* : Nat → ListNat → ListNat

pwAdd : ListNat → ListNat → ListNat

RESPOSTA. **Não** repita as tipagens na resposta!

```
Product
| Product []     = 1
| Product (x:xs) = x * Product xs

Tidy
Tidy [ev(x)] = True
Tidy [odd(x)] = False
Tidy x:xs = ¬(tidy xs)

TakeEvens
 takeEvens [] = []
 takeEvens ev(x):xs = [x] ++ takeEvens xs
 takeEvens odd(x):xs = takeEvens xs

atEvens
 atEvens [] = []
 atEvens [x] = [x]
 atEvens [x,y] = [x]
 atEvens

replaceAt
 replaceAt a b [] = []
 replaceAt a b xs =

addNat
 addNat x [] = [x]
 addNat x xs = x:xs

removeAt
 removeAt x [] = []
 removeAt x ms = ms - [x]
```

cuidado: a negação de tidy não é bagunça-total!

(9) **C**

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,\left[a^{b\cdot c} = (a^b)^c\right]$:

(18) **T**

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

Sejam n,m.
Separo em casos. ← Parece que tu tá usando tricotomia de (<). Qual é correção?
Caso n<m:
Escolho-L.
Uso m-n como testemunha.
Caso m<n:
Escolho-R.
Uso n-m como testemunha.
Caso n=m:
Escolho-L.
Uso 0 como testemunha.

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

$\mathit{product} : \mathsf{ListNat} \to \mathsf{Nat}$ ✓ $\quad \mathit{replaceAt}, \mathit{addAt} : \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat}$ ✓

$\mathit{tidy} : \mathsf{ListNat} \to \mathsf{Bool}$ ✓ $\quad \mathit{addNat}, \mathit{removeAt} : \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat}$

$\mathit{takeEvens}, \mathit{atEvens} : \mathsf{ListNat} \to \mathsf{ListNat}$ ✓ $\quad \mathit{pwAdd} : \mathsf{ListNat} \to \mathsf{ListNat} \to \mathsf{ListNat}$ ✓

RESPOSTA. **Não** repita as tipagens na resposta!

```
product : ListNat → Nat
product []     = 1
product (x:xs) = x * product xs
```

```
pwAdd : ListNat → ListNat → ListNat
pwAdd _ [] = []
pwAdd [] _ = []
pwAdd (x:xs) (y:ys) = x+y : pwAdd xs ys
```

```
takeEvens : ListNat → ListNat
takeEvens []     = []
takeEvens (x:xs) = if (even x) then x : takeEvens xs
                               else takeEvens xs
```

```
even : Nat → Bool
even 0     = True
even S0    = False
even S(Sn) = even n
```

```
odd : Nat → Bool
odd 0     = False
odd S0    = True
odd S(Sn) = odd n
```

```
addNat : Nat → ListNat → ListNat
addNat _ []     = []
addNat n (x:xs) = x+n : addNat n xs
```

```
atEvens : ListNat → ListNat
atEvens []       = []
atEvens [x]      = [x]
atEvens (x:y:ys) = x : atEvens ys
```

```
tidy : ListNat → Bool
tidy []       = True
tidy (x:y:ys) = even x && odd y && tidy ys
```

continua na Lemmata

(9) **C**

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[a^{b\cdot c} = (a^b)^c]$:

(18) **T**

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

Seja n: Nat
Indução.
Parte [n ≤ 0 ou 0 ≤ n]
escolho R.
Seja K: Nat,
Calculamos
0 + K = n
n = K + 0 [(+).am] ✓
K = n [(+).1 n := K]

Parte $(\forall_k)$ [n ≤ K ou K ≤ n ⇒ n ≤ SK ou SK ≤ n]
Seja K: Nat tq. n ≤ K ou K ≤ n.
Caso ?, tq n + l = K

Escolho L
Seja m: Nat tq n + m = SK
Calculamos
n + m = S(n + l)
n + m = n + Sl

m = Sl ✓

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

product : ListNat → Nat

tidy : ListNat → Bool

takeEvens, *atEvens* : ListNat → ListNat

replaceAt, ~~*addAt*~~ : Nat → Nat → ListNat

addNat, *removeAt* : Nat → ListNat → ListNat

pwAdd : ListNat → ListNat → ListNat

RESPOSTA. **Não** repita as tipagens na resposta!

```
product [] = 1
product (x:xs) = x · (product xs)

addNat _ [] = []
addNat 0 xs = xs
addNat n (x:xs) = (n + x) : (addNat n xs)

removeAt _ [] = []
removeAt 0 (x:xs) = xs
removeAt

pwAdd _ [] = []
pwAdd [] xs = xs
pwAdd (x:xs) (y:ys) = (x+y) : (pwAdd xs ys)

atEvens [] = []
atEvens (x:(x':xs)) = x : (atEvens xs)

tidy [] = True
tidy x:(x':xs) = ● ev x = True ∧ tidy xs

replaceAt _ _ [] = []
replaceAt 0 n (x:xs) = n : xs
```

```
takeEvens [] = []

atEvens [] = []

ev 0 = True
ev SO = False
ev (SSn) = ev n
```

porque favorecer a direita?!

(9) **C**

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[\,a^{b\cdot c} = (a^b)^c\,]$:

(18) **T**

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

|8|

Indução.
Base:
Seja m natural.
Escolho - L.
Temos $0 \leq m$. [(≤)-min]. ✓
Passo indutivo:
Seja k natural t.l.q
$(\forall m)[k \leq m$ ou $m \leq k]$. (1) ✓
Seja m natural.
Caso $m = 0$:
Escolho - R.
Temos $0 \leq Sk$. [(≤)-min] ✓
Caso $m = Sm'$:
Aplicamos (1) em m'
logo $k \leq m'$ ou $m' \leq k$.
Caso $k \leq m'$:
Temos $Sk \leq Sm'$. [(≤)-suc]
Escolho - L.
Imediato.
Caso $m' \leq k$:
Similar. ✓

Só isso mesmo.

(8) **I** 1.?

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

4

EVEN : NAT → BOOL	ODD : NAT → BOOL	
EV 0 : TRUE	ODD 0 : FALSE	
EV S0 : FALSE	ODD S0 : TRUE	
EV SSN : EV N	ODD SSN : ODD N	

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

$$\textit{product} : \mathsf{ListNat} \to \mathsf{Nat} \qquad \textit{replaceAt}, \textit{addAt} : \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{ListNat}$$
$$\textit{tidy} : \mathsf{ListNat} \to \mathsf{Bool} \qquad \textit{addNat}, \textit{removeAt} : \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat}$$
$$\textit{takeEvens}, \textit{atEvens} : \mathsf{ListNat} \to \mathsf{ListNat} \qquad \textit{pwAdd} : \mathsf{ListNat} \to \mathsf{ListNat} \to \mathsf{ListNat}$$

RESPOSTA. **Não** repita as tipagens na resposta!

1) (∀ XS, YS : LISTNAT) [PRODUCT (XS ++ YS) = PRODUCT (XS) * PRODUCT (YS)]

2)

?!?!?

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

$(\leq) : Nat \times Nat \to Prop \iff (\exists k)[_ + k = _]$

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

$$\begin{aligned} \mathit{product} &: \mathsf{ListNat} \to \mathsf{Nat} & \mathit{replaceAt}, \mathit{addAt} &: \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat} \\ \mathit{tidy} &: \mathsf{ListNat} \to \mathsf{Bool} & \mathit{addNat}, \mathit{removeAt} &: \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat} \\ \mathit{takeEvens}, \mathit{atEvens} &: \mathsf{ListNat} \to \mathsf{ListNat} & \mathit{pwAdd} &: \mathsf{ListNat} \to \mathsf{ListNat} \to \mathsf{ListNat} \end{aligned}$$

RESPOSTA. **Não** repita as tipagens na resposta!

(9) C

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,\left[\,a^{b\cdot c} = (a^b)^c\,\right]$:

(18) T

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

2

Seja m, n nats.
Escolho lado esquerdo.
Por indução em m.
Base: $n \leq 0$
Seja l nat t.q. $n + l = 0$.
Como $n + l = 0$, logo $n = 0$ e $l = 0$.
Trivial. !'.
Passo indutivo: $(\forall i)(\forall n)[n \leq i \Rightarrow n \leq \text{succ } i]$
Seja i, n nats.
Suponho $n \leq i$.
Seja k nat t.q. $n + k = i$.
Calculamos:

$\text{succ}(n + k) = \text{succ } i$ [succ]
$n + \text{succ } k = \text{succ } i$ [add_succ]

Uso succ k como testemunho.

Se teria como demonstrar isso a partir dessa linha, tu teria demonstrado que $(\forall n)(\forall m)[n \leq m]$.

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

2 *product* : ListNat → Nat

tidy : ListNat → Bool

takeEvens, *atEvens* : ListNat → ListNat

replaceAt, *addAt* : Nat → Nat → ListNat

2 *addNat*, *removeAt* : Nat → ListNat → ListNat

2 *pwAdd* : ListNat → ListNat → ListNat

RESPOSTA. **Não** repita as tipagens na resposta!

```
{product [] = 1
{product (x:xs) = x * product xs

{addNat n [] = []
{addNat n (x:xs) = (x+n) : addNat n xs

{pwdAdd _ [] = []
{pwdAdd [] _ = []
{pwdAdd (x:xs) (y:ys) = (x+y) : pwdAdd xs ys
```

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

8

```
(∀x) x é par ⇔ Even x = True.
(∀x) x é impar ⇔ odd x = True.
(∀ n,m)  n ≤ m ⇔ leq n m = True.
leq : Nat → Nat → Bool.
  (Uma prova)

-- Even : Nat → Bool
Even 0 = True
Even (S n) = odd n
odd : Nat → Bool
odd 0 = False.
odd (S n) = Even n.
```

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

$$\begin{aligned} \mathit{product} &: \mathsf{ListNat} \to \mathsf{Nat} & \mathit{replaceAt}, \mathit{addAt} &: \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{ListNat} \\ \mathit{tidy} &: \mathsf{ListNat} \to \mathsf{Bool} & \mathit{addNat}, \mathit{removeAt} &: \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat} \\ \mathit{takeEvens}, \mathit{atEvens} &: \mathsf{ListNat} \to \mathsf{ListNat} & \mathit{pwAdd} &: \mathsf{ListNat} \to \mathsf{ListNat} \to \mathsf{ListNat} \end{aligned}$$

RESPOSTA. **Não** repita as tipagens na resposta!

(9) **C**

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,\left[a^{b \cdot c} = (a^b)^c\right]$:

(18) **T**

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

Indução.
Base:
Seja n.
Escolho 4.
Uso n como testemunho

mid-term!

Passo indutivo:
Seja k t.q. (∀n') [n' ≤ k ou k ≤ n'] [H.I] ✓
Seja n.
Separo em casos pela [H.I] com n':= n.
Caso n ≤ k.
Logo Sn ≤ Sk.
Como n ≤ Sn, logo Sn ≤ Sk (≤)-trans
Qed.
Caso k ≤ n.
Seja k' t.q. k + k' = n.
Separo em casos. Sn
Caso k' = 0.
Temos k = n, logo n ≤ Sk.
Qed.
Caso k' = Sk''.
S(k + k'') = n, logo, Sk + k'' = n.
Uso k'' como testemunho.
Qed.

Só isso mesmo.

→ Não seja mão-de-vaca com as linhas!

(8) **I**

PAR IMPAR

O INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

DATA EVEN:
M EVEN: $m = 2k$

DATA ODD:
M ODD: $m = 2k+1$

DATA $(\leq)$:
$\mathsf{IM} \leq m$: $m + k = m$
$\mathsf{IM} \leq m$: $m + k = n$

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

POSIÇÕES PAR TEM PAR
IMPAR TEM IMPAR

$product : \mathsf{ListNat} \to \mathsf{Nat}$ $\quad$ $replaceAt, addAt : \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{ListNat}$

$tidy : \mathsf{ListNat} \to \mathsf{Bool}$ $\quad$ $addNat, removeAt : \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat}$

$takeEvens, atEvens : \mathsf{ListNat} \to \mathsf{ListNat}$ $\quad$ $pwAdd : \mathsf{ListNat} \to \mathsf{ListNat} \to \mathsf{ListNat}$

PEGA PAR — PEGA POSIÇÕES DE PARES

[1,2,3] [100,200,50,500] [10,202,55]

RESPOSTA. **Não** repita as tipagens na resposta!

(9) C

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[a^{b\cdot c} = (a^b)^c]$:

0

INDUÇÃO EM c COM d

BASE:

CALCULAMOS:

$a^{b\cdot 0} = (a^b)^0$

$a^0 = (a^b)^0$

$1 = 1$ → concluiu que $1=1$?!

PASSO INDUTIVO:

$a^{b\cdot d} = (a^b)^d$?

$a^{b\cdot(c+d)} = (a^b)^{c+d}$??

$a^{(b\cdot c)+b} = (a^{b\cdot c})\cdot a^b$

(18) T

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

Só isso mesmo,

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

$$\begin{aligned} \mathit{product} &: \mathsf{ListNat} \to \mathsf{Nat} & \mathit{replaceAt}, \mathit{addAt} &: \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{ListNat} \\ \mathit{tidy} &: \mathsf{ListNat} \to \mathsf{Bool} & \mathit{addNat}, \mathit{removeAt} &: \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat} \\ \mathit{takeEvens}, \mathit{atEvens} &: \mathsf{ListNat} \to \mathsf{ListNat} & \mathit{pwAdd} &: \mathsf{ListNat} \to \mathsf{ListNat} \to \mathsf{ListNat} \end{aligned}$$

RESPOSTA. **Não** repita as tipagens na resposta!

product:

$[n, m] = n \cdot m$

$[n, m, _] = [n \cdot m, _]$

(9) **C**

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[a^{b\cdot c} = (a^b)^c]$:

Seja a,b,c: Nat. Demonstraremos que $a^{b\cdot c} = (a^b)^c$ com indução em c.

3

Base

Seja $c = 0$.

Calculamos

$a^{b\cdot 0} = (a^b)^0$

$a^0 = (a^b)^0$ [pela (·).1]

$1 = 1$

$1 = 1$

concluir 1=1

Passo indutivo

?!?

Seja $\varphi(k)$: Nat t. q. $\varphi(k)$: $a^{b\cdot k} = (a^b)^k$

Demonstraremos que $a^{b\cdot Sk} = (a^b)^{Sk}$

Calculamos

$a^{b\cdot Sk} = (a^b)^{Sk}$ [pela (·).2]

$a^{(b\cdot k)+b} = (a^b)^k + a^b$ [pela (^).2]

$a^{(b\cdot k)} + a^b = (a^b)^k + a^b$ [fator comum na igualdade]

$a^{b\cdot k} = (a^b)^k$ que é o exato $\varphi(k)$.

não parece

(18) **T** algo que vai convencer alguem sobre qualquer coisa...

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

Nenhuma das duas coisas é um cálculo.
São afirmações soltas, uma depois da outra.
E a primeira, nos dois "Calculos" é exatamente
a coisa que queres demonstrar!!.

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

$product : \mathsf{ListNat} \to \mathsf{Nat}$ $\qquad$ $replaceAt, addAt : \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{ListNat}$

$tidy : \mathsf{ListNat} \to \mathsf{Bool}$ $\qquad$ $addNat, removeAt : \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat}$

$takeEvens, atEvens : \mathsf{ListNat} \to \mathsf{ListNat}$ $\qquad$ $pwAdd : \mathsf{ListNat} \to \mathsf{ListNat} \to \mathsf{ListNat}$

RESPOSTA. **Não** repita as tipagens na resposta!

```
product [] = 1
product (x:xs) = x * product xs

tidy [] = True
tidy (x:xs)
  | isOdd x = isOdd (length (x:xs)) ∧ tidy xs
  | otherwise = isEven (length (x:xs)) ∧ tidy xs

takeEvens [] = []
takeEvens (x:xs)
  | isEven x = x : takeEvens xs
  | otherwise = takeEvens xs

pwAdd (x:xs) (y:ys) = (x+y) : pwAdd xs ys
pwAdd _ _ = []

atEvens [] = []
atEvens (x:xs)
  | isEven (length (x:xs)) = x : atEvens xs
  | otherwise = atEvens xs

replaceAt 0 v (x:xs) = v : xs
replaceAt (S n) v (x:xs) = x : replaceAt n v xs

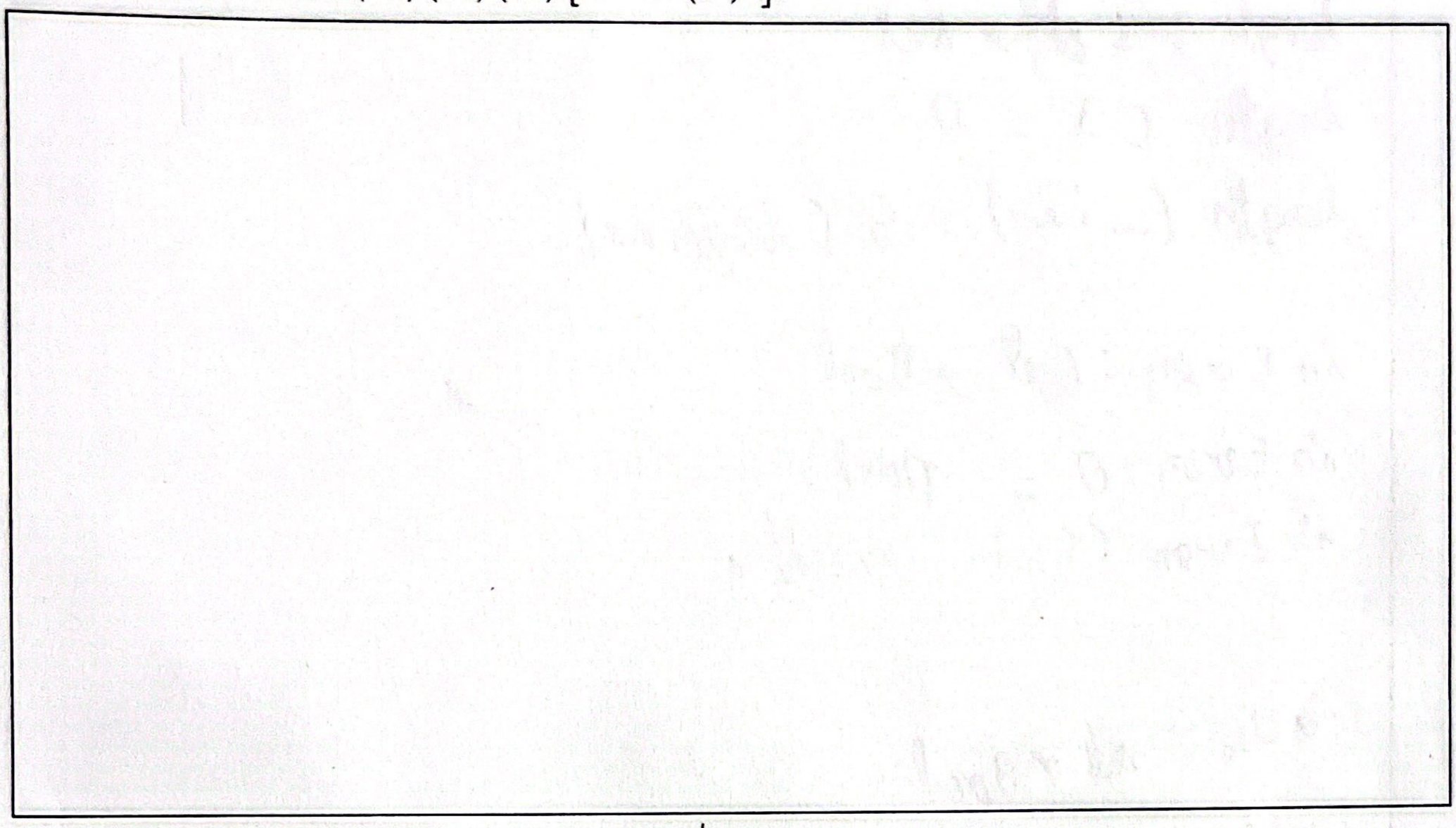
addAt 0 v (x:xs) = (v+v) : v : xs
addAt (S n) v (x:xs) = x : addAt n v xs

addNat _ [] = []
addNat v (x:xs) = (v+x) : addNat v xs

removeAt 0 (x:xs) = xs
removeAt (S n) (x:xs) = removeAt n xs
```

(9) C

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[\,a^{b\cdot c} = (a^b)^c\,]$:



(18) T

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m$ ou $m \leq n]$:

4

Seja n : Nat.
Indução.
  Base.
  Escolha R.
  Imediato Pela (≤).1.
  Passo indutivo.
  Suponha n ≤ m ∨ m ≤ n.
  Split n em casos.
  Caso O.. Escolha L. Imediato Pela (≤).1.
  Caso (S n):
    Escolha L. Imediato Pela (≤).2.

tá pensando na leq?

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

18!

product : ListNat → Nat
tidy : ListNat → Bool
takeEvens, *atEvens* : ListNat → ListNat
replaceAt, *addAt* : Nat → Nat → ListNat
addNat, *removeAt* : Nat → ListNat → ListNat
pwAdd : ListNat → ListNat → ListNat

RESPOSTA. **Não** repita as tipagens na resposta!

```
Product Nil = 1
Product (x : xs) = x * (Product xs)

tidy Nil = True
tidy (x : Nil) = even x
tidy (x : x' : xs) = even x && odd x' && tidy xs

TakeEvens Nil = Nil
TakeEvens (x : xs) =
    | even x = x : (TakeEvens xs)
    | otherwise = TakeEvens xs

AtEvens Nil = Nil
AtEvens (x : Nil) = x : Nil
AtEvens (x : x' : xs) = x : (AtEvens xs)

ReplaceAt m 0 (x : xs) = m : xs
ReplaceAt m _ Nil = Nil
ReplaceAt m (S m) (x : xs) = x : (ReplaceAt m m xs)
AddNat _ Nil = Nil
AddNat m (x : xs) = (m + x) : xs
```

```
AddAt m 0 (x : xs) = (m + x) : xs
AddAt _ _ Nil = Nil
AddAt m (S m) (x : xs) = x : (AddAt m m xs)
```

```
RemoveAt _ Nil = Nil
RemoveAt 0 (x : xs) = xs
RemoveAt (S m) (x : xs) = x : (RemoveAt m xs)
PwAdd (x : xs) (y : ys) = (x + y) : (PwAdd xs ys)
PwAdd _ _ = Nil
```

(9) **C**

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[a^{b\cdot c} = (a^b)^c]$:

(18) **T**

mid-level!

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

18

Seja n : Nat. ✓
Indução. ✓
Base:
Escolho - R.
Vlo n como testemunha.
Temos $0+m = m$ [Lemma (*)].
- Passo Indutivo
seja k : Nat t.q $m \leq k$ ou $k \leq m$.
Caso $m \leq k$.
Seja x : Nat t.q. $m+x = k$
Escolho - L.
Vlo Sx como testemunha
Calculamos:
$m + Sx = S(m+x)$ [+1.1] k.
$= Sk$. ✓

Caso $k \leq m$:
Seja x : nat t.q $k+x = m$ ✓
Caso $x = 0$:
temos $k = m$. [+.I]
Vlo 0 como testemunha.
Calculamos
$m + S_0 = S(m+0)$ [+'.2]
$= Sm$ [+.I]
$= Sk$ [k=m]
Caso $x = Sy$:
Escolho - R
Vlo y como testemunha.
Calculamos.
$Sk + y = S(k+y)$ [Lemma (!)]
$= k + Sy$ [+.2 ⇐]
$= m$ [H.I]

✓ ✓ ✓

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

0

$$\begin{aligned} \mathit{product} &: \mathsf{ListNat} \to \mathsf{Nat} & \mathit{replaceAt}, \mathit{addAt} &: \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat} \\ \mathit{tidy} &: \mathsf{ListNat} \to \mathsf{Bool} & \mathit{addNat}, \mathit{removeAt} &: \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat} \\ \mathit{takeEvens}, \mathit{atEvens} &: \mathsf{ListNat} \to \mathsf{ListNat} & \mathit{pwAdd} &: \mathsf{ListNat} \to \mathsf{ListNat} \to \mathsf{ListNat} \end{aligned}$$

RESPOSTA. **Não** repita as tipagens na resposta!

ReplaceAt
Add Nt
n + m + [] = [42,55,m,619]

addNat, RemoveAt:
m+m+[] = [42,55,m+m,619]
m-m+[] = [42,55,m-m,619]

PWAdd
[X,Y,W] ♣ + [X',Y',W',Z] = [X+X', Y'+Y+W+W']

Product
[n] = n
[Sn] = Sn

Tidy
[] = True
[] = False

TakeEvens
[P,I,P',I',P''] = [P,P',P'']

estude muito o gabarito!

(9) **C**

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[\,a^{b\cdot c} = (a^b)^c\,]$:

(18) **T**

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[\,n \leq m \text{ ou } m \leq n\,]$:

0 Sejam n, m Nat.

Escolho L.

Calculando:

$$Sn \leqq Sm \;[L2]$$
$$n \leqq m \;[L1]$$

■

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

product : ListNat → Nat

tidy : ListNat → Bool

takeEvens, *atEvens* : ListNat → ListNat

replaceAt, *addAt* : Nat → Nat → ListNat

addNat, *removeAt* : ~~(Nat → ListNat → ListNat)~~

pwAdd : ListNat → ListNat → ListNat

RESPOSTA. **Não** repita as tipagens na resposta!

Product :

Product nill = ∅

Product x:xs = x · Product xs

tidy :

tidy [] = (?)

tidy x:xs = ∈ v x : tidy xs

Pw Add

Pw Add x n:xs = Pw Add x

Pw Add x:xs n = Pw Add x

Pw Add x n = [x+n]

Pw Add x [] = []

?!

Esta errada

Veja o gabarito!

(9) C

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,\left[\,a^{b\cdot c} = (a^b)^c\,\right]$:

(18) T

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

2

Seja n: Nat.

Indução em m.

Base:

Seja m: Nat

temos n ≤ 0 ou 0 ≤ n — undeclared — não temos isso. Isso é nosso alvo!

logo n + K = 0 ou 0 + K = n ?

Escolho-L ?!

Imediato.

Passo Indutivo:

Seja m: Nat ← assim m é arbitrário, nada legal!

temos n ≤ Sm ou Sm ≤ n

Escolho-L

Como n ≤ m, logo n ≤ Sm.

Imediato.

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

$(\forall n)(\forall m)[n \leq m \text{ ou } m \leq n]$

Se n ou $m \Rightarrow$ Even, $2 \mid n$ ou $2 \mid m$

Se n ou $m \Rightarrow$ Odd, $\neg\, 2 \mid n$ ou $\neg\, 2 \mid m$

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

$$\begin{aligned} \mathit{product} &: \mathsf{ListNat} \to \mathsf{Nat} & \mathit{replaceAt}, \mathit{addAt} &: \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{ListNat} \\ \mathit{tidy} &: \mathsf{ListNat} \to \mathsf{Bool} & \mathit{addNat}, \mathit{removeAt} &: \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat} \\ \mathit{takeEvens}, \mathit{atEvens} &: \mathsf{ListNat} \to \mathsf{ListNat} & \mathit{pwAdd} &: \mathsf{ListNat} \to \mathsf{ListNat} \to \mathsf{ListNat} \end{aligned}$$

RESPOSTA. **Não** repita as tipagens na resposta!

(9) **C**

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,\left[a^{b\cdot c} = (a^b)^c\right]$:

(18) **T**

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

Sejam n, m : Nat

Se $n \leq m \Rightarrow n < m$ ou $n = m$

Se $m \leq n \Rightarrow m < n$ ou $m = n$.

(succ)

$Sn \leq Sm \Leftrightarrow n \leq m$

Portanto: $n \leq m$ ou $m \leq n$.

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

```
Leq : Nat → Nat → Nat
  Leq 0 _ = true
  Leq Sn 0 = False
  Leq Sn Sm = Leq n m
```

```
Ev : Nat → Prop
  Ev 0 = true
  Ev S0 = False
  Ev SSn = Ev n
```

```
Od : Nat → Prop
  Od 0 = False
  Od S0 = true
  Od SS0 = False
  Od SSSn = Od Sn
```

por que tantas??

7

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

$product : \mathsf{ListNat} \to \mathsf{Nat}$ $\qquad$ $replaceAt, addAt : \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{ListNat}$

$tidy : \mathsf{ListNat} \to \mathsf{Bool}$ $\qquad$ $addNat, removeAt : \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat}$

$takeEvens, atEvens : \mathsf{ListNat} \to \mathsf{ListNat}$ $\qquad$ $pwAdd : \mathsf{ListNat} \to \mathsf{ListNat} \to \mathsf{ListNat}$

RESPOSTA. **Não** repita as tipagens na resposta!

(9) C

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[\,a^{b\cdot c} = (a^b)^c\,]$:

(18) T

mid-level!!

Por que um nome tão feio?

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

obrigatório um $(\forall n)$ aqui!!

Sejam n, m : Nats.

Indução em n.

-- caso base $n = 0$

Left.

Escolho m como testemunha.

Está nos dadas!

-- passo indutivo $[n \leq m \text{ ou } m \leq n \Rightarrow Sn \leq m \text{ ou } m \leq Sn]$

Suponha $n \leq m$ ou $m \leq n$. (1)

Separo em casos pela (1).

-- caso $n \leq m$

Seja θ t.q. $m = n + \theta$.

Separo em casos por θ.

-- $\theta = 0$

Right: $m \leq Sn$

Escolho $S0$ como testemunha.

Calculemos:

$m + S0 = n + S0 = S(n+0)$

$= Sn$.

-- $\theta = S\theta'$

Left.

Calculemos:

$m = n + S\theta'$ (escolha de θ)

$= S(n + \theta')$ (+).2

$= S(\theta' + n)$ (+ comm)

$= \theta' + Sn$ (+).2

$= Sn + \theta'$ (+ comm)

$Sn \leq m$

Está nos dadas!

-- caso $m \leq n$: $m \leq n \leq Sn$.

Seja p t.q. $n = m + p$

Separo em casos por p.

-- $p = 0$.

Right: $m \leq Sn$

Calculemos:

$Sn = S(m + 0)$

$= m + S0$ (+).2

Escolho $S0$ como testemunha.

Está nos dadas.

-- $p = Sp'$

Calculemos:

$Sn = S(m + Sp')$ (escolha de p)

Só isso mesmo.

$= m + SSp'$ (+).2

Right. $m \leq Sn$

Está nos dadas! ■

16

~~$m = n + S\theta'$~~

~~$m = S(n + \theta)$~~

~~$m = Sn + \theta'$~~

~~$Sn = n + p$~~

~~$Sn \leq Sn$~~

~~$n \leq m$~~

~~$m = n + p$~~

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

$product : \mathsf{ListNat} \to \mathsf{Nat}$ ✓ — $replaceAt, addAt : \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{ListNat} \to$ ListNat

$tidy : \mathsf{ListNat} \to \mathsf{Bool}$ ✓ — $addNat, removeAt : \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat}$

$takeEvens, atEvens : \mathsf{ListNat} \to \mathsf{ListNat}$ ✓ — ✓ $pwAdd : \mathsf{ListNat} \to \mathsf{ListNat} \to \mathsf{ListNat}$

15

RESPOSTA. **Não** repita as tipagens na resposta!

isEven e isOdd na Lemnata

```
product [] = 50
product (x:xs) = x * product xs

pwAdd (x:xs) (y:ys) = (x+y) : pwAdd xs ys
pwAdd _ _ = []

oddNat n (x:xs) = (n+x) : oddAt n xs
oddNat _ [] = []

takeEvens [] = []
takeEvens (x:xs)
    | isEven x = x : takeEvens xs
    | otherwise = takeEvens xs

atEvens [] = []
atEvens [x] = ?
atEvens (x:(y:ys)) = x : atEvens ys

tidy (x:(y:ys)) = isEven x && isOdd y && tidy ys
tidy [] = True
tidy [x] = isEven x
    | isEven x = True
    | otherwise False
```

```
replaceAt _ _ [] = []
replaceAt n 0 (x:xs) = n : xs
replaceAt n (Sm) (x:xs)
    | m = 0 = m : xs ?
    | otherwise replaceAt n
        replaceAt n m (x:xs)
```

esse recursão "não vai andar na lista"

```
oddAt _ _ [] = []
oddAt n 0 (x:xs) = (n+x) : xs
oddAt n (Sm) (x:xs)
    | m = 0 = (x+n) : xs
    | otherwise
        oddAt n m (x:xs)
```

(9) **C**

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,\left[a^{b \cdot c} = (a^b)^c\right]$:

(18) **T**

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[\overbrace{n \leq m \text{ ou } m \leq n}]$:

7

Troca.
Seja m : Not.
Indução.
Base:
Escolho-L.
Testemunho m.
Calculamos:
0 + m = m + 0 [(+).com]
= m [(+).1]

P.I:
Seja k : Not tq. k ≤ m ou m ≤ k.
Caso k ≤ m.

Caso k ≤ m.
Seja m' : Not tq. k + m' = m.
Seja m' : Not tq. k + S m' = m.
Calculamos: k + S m' = S k + m', pela
def + succ, que S(m' + k) = S m'
Calculamos:
S(m' + k) = m' + S k [(+).2 ←]

mid-level!! → Pra quê?! Teu alvo é simétrico nos n, m!

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

$product : \mathsf{ListNat} \to \mathsf{Nat}$ — $replaceAt, addAt : \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat}$

$tidy : \mathsf{ListNat} \to \mathsf{Bool}$ — $addNat, removeAt : \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat}$

$takeEvens, dropEvens : \mathsf{ListNat} \to \mathsf{ListNat}$ — $pwAdd : \mathsf{ListNat} \to \mathsf{ListNat} \to \mathsf{ListNat}$

RESPOSTA. **Não** repita as tipagens na resposta!

por que olhar no tamanho da lista?

esse formato é bugado.

```
Ev : Nat → Bool
Ev 0 = true
Ev S0 = False
Ev SSn = Ev n

atEvens
atEvens [] = []
atEvens (n:ns) = if (Ev len ns) then n:atEvens ns
                 else atEvens ns.

Product
Product [] = 1
Product n:ns = n * Product ns

len : ListNat → Nat
len [] = 0
len _:l = S (len l)

takeEvens
takeEven [] = []
takeEven (n:ns) = if (Ev n) then n:takeEven ns
                  else takeEven ns

pwAdd
pwadd _ [] = []
pwadd [] _ = []
pwadd (n:ns) (m:ms) = (n+m) : pwAdd ns ms

tidy
tidy [] = true
tidy n:ns = ((Ev len ns) = false) = Ev n) && tidy ns

addNat
addNat _ [] = []
addNat n (k:ks) = (n+k) : (addNat n ks)
```

(9) **C**

9

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[a^{b\cdot c} = (a^b)^c]$:

Sejam a, b nats
Indução
Base (c = 0)
calculamos:
$a^{b\cdot 0} = a^0$ [(e).1]
$= S0$ [(A).1]
$= (a^b)^0$ [(A).1 ⇐]

Passo indutivo
seja c: nat t.q $a^{bc} = (a^b)^c$ [Hi]

Calculamos:
$(ab)^{Sc} = (a^b)^c \cdot a^b$ [(A).2]
$= a^{bc} \cdot a^b$ [Hi]
$= a^{bc+b}$ [Lema1]
$= a^{b\cdot Sc}$ [(.).2 ⇐]

QED

(18) **T**

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

Só isso mesmo.

(8) **I** 2 S [1,2,7] 0 2 [2,3] 1,4,3
(4,5]

INTERNALIZE a relação $(\leq)$: Nat × Nat → Prop e os predicados Even, Odd : Nat → Prop:

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

product : ListNat → Nat
tidy : ListNat → Bool
takeEvens, *atEvens* : ListNat → ListNat
replaceAt, *addAt* : Nat → Nat → ListNat
addNat, *removeAt* : Nat → ListNat → ListNat
pwAdd : ListNat → ListNat → ListNat

RESPOSTA. **Não** repita as tipagens na resposta!

```
PRODUCT [] = 0
PRODUCT (x:xs) = x * PRODUCT xs

EV : NAT -> BOOL
EV [] = TRUE
EV S(Sn) = EV n

TIDY [] = FALSE   TRUE
TIDY (x:xs) = x : TIDY xs   TYPE ERROR
              ELSE FALSE

TAKEEVENS [] = FALSE   TYPE ERROR
TAKEEVENS (x:xs) = IF EV x THEN x : TAKEEVENS xs
                   ELSE TAKEEVENS xs

ATEVENS [n] = [n]
ATEVENS [] = []
ATEVENS (x:xs) = x : addNAt(S 0 ATEVENS xs)   ?!!.

REPLACEAT _ n [] = [n]   []
REPLACEATSSn m (x:xs) = x : REPLACE n m xs
REPLACEAT 0 m (x:xs) = m : xs

ADDAT _ [] = []
ADDAT 0 m (x:xs) = (x+m) : xs
ADDAT Sn m (x:xs) = x : ADDAT n m xs
```

$(a^b)^c = a^{b \cdot c}$ $\quad$ $a^{b+k} = a^b \cdot a^k$

(9) C

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[a^{b \cdot c} = (a^b)^c]$:

9

Sejam a, b : NAT
Indução.
Base : $a^{b \cdot 0} = (a^b)^0$
Calculamos :
$a^{b \cdot 0} = a^0$ ((1).1)
$= S0$ ((1).1)
$(a^b)^0 = \underline{S0}$ ((1).1)
Logo $a^{b \cdot 0} = (a^b)^0$;
Como indutivo : $(\forall k \mid [a^{b \cdot k} = (a^b)^k] \Rightarrow$
$a^{b \cdot Sk} = \underline{(a^b)^{Sk}}]$
Sejá k : nat t.q $a^{b \cdot k} = (a^b)^k$

Calculemos :
$a^{b \cdot Sk} = a^{(b \cdot k)+b}$ ((+).7)
$= a^{b \cdot k} \cdot a^b$ [POW_LEMA]
$= (a^b)^k \cdot a^b$ [H.I]
$= (a^b)^{k+b}$ [POW_LEMA]
$= \underline{(a^b)^{Sk}}$ [+((1).2].

(18) T

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

$$\begin{aligned} \mathit{product} &: \mathsf{ListNat} \to \mathsf{Nat} & \mathit{replaceAt}, \mathit{addAt} &: \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{ListNat} \\ \mathit{tidy} &: \mathsf{ListNat} \to \mathsf{Bool} & \mathit{addNat}, \mathit{removeAt} &: \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat} \\ \mathit{takeEvens}, \mathit{atEvens} &: \mathsf{ListNat} \to \mathsf{ListNat} & \mathit{pwAdd} &: \mathsf{ListNat} \to \mathsf{ListNat} \to \mathsf{ListNat} \end{aligned}$$

RESPOSTA. **Não** repita as tipagens na resposta!

```
Product :
  Product [] = 0
  Product (x:xs) = x . Product x
```

(9) **C**

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[a^{b \cdot c} = (a^b)^c]$:

Sejam a, b naturais.
Indução na c. ✓
base:
Calculamos:

$a^{b.c} = a^{b.0}$ (hipótese do caso)

$= a^0$ ((0).1)

$= S0$ ((1).1)

$(a^b)^c = (a^b)^0$ (hipótese do caso)

$= S0$ ((1).1)

✓

Passo indutivo:
Quero mostrar que
$(\forall K)[a^{b.K} = (a^b)^K \Rightarrow a^{b.SK} = (a^b)^{SK}]$
Seja K natural.
Suponha $a^{b.K} = (a^b)^K$ (Hi) ✓
Calculamos:

$a^{b.SK} = a^{(b.K)+b}$ ((•).2)

$= (a^b)^{K+b}$ (~~Hi~~)

aqui não aparece o $a^{b.k}$ como subexpressão para conseguir fazer nela!!.

4

(18) **T**

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

Natural × Natural → Proposição
(≤): Menor ou igual a...
Even: Até ... Odd: Alternadamente ...

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

$$\begin{aligned} \mathit{product} &: \mathsf{ListNat} \to \mathsf{Nat} & \mathit{replaceAt}, \mathit{addAt} &: \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{ListNat} \\ \mathit{tidy} &: \mathsf{ListNat} \to \mathsf{Bool} & \mathit{addNat}, \mathit{removeAt} &: \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat} \\ \mathit{takeEvens}, \mathit{atEvens} &: \mathsf{ListNat} \to \mathsf{ListNat} & \mathit{pwAdd} &: \mathsf{ListNat} \to \mathsf{ListNat} \to \mathsf{ListNat} \end{aligned}$$

RESPOSTA. **Não** repita as tipagens na resposta!

(9) C

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[a^{b \cdot c} = (a^b)^c]$:

Conjunto de a, conjunto de b conjunto de c
(∀a) (∀b) (∀c)

a ao b x c é igual á a ao b x c

(18) T

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

product : ListNat → Nat

tidy : ListNat → Bool

takeEvens, *atEvens* : ListNat → ListNat

replaceAt, *addAt* : Nat → Nat → ListNat

addNat, *removeAt* : Nat → ListNat → ListNat

pwAdd : ListNat → ListNat → ListNat

RESPOSTA. **Não** repita as tipagens na resposta!

```
product : ListNat → Nat
product [] = 1
product (S x :xs) = S (product x :xs)
tidy : ListNat → Bool
tidy [] = False
tidy (SS x :xs) = tidy x :xs
pwAdd : ListNat → ListNat → ListNat
pwAdd [] ys = xs
pwAdd (S x :xs) (y :ys) = S x : pwAdd (x :ys)
ReplaceAt : Nat → Nat → ListNat → ListNat
ReplaceAt n m [] = []
ReplaceAt n (S m) (x :xs) = n m ReplaceAt (S m xs)
remAt : Nat → ListNat → ListNat
remAt n [] = []
remAt (S m) x :xs =      remAt(  )
```

```
AddAt : Nat → Nat → ListNat → ListNat
AddAt n x :[] = []
AddAt n + S m x :xs =
```

(9) **C**

4

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[\,a^{b\cdot c} = (a^b)^c\,]$:

Sejam a, b, c naturais, ✓
Por indução em c, ✓
Base:
Calculamos:
$a^{b\cdot 0} \equiv a^0 \quad (\cdot 1)$
$\equiv 1 \quad (11)$
$(a^b)^0 = 1 \quad (11)$. ✓
Passo Indutivo:
Seja k um natural, tal que, $a^{b\cdot k} = (a^b)^k$ (HI),
Calculemos:
$a^{b\cdot 5k} \equiv a^{(b\cdot k)+b} \quad (\cdot 2)$
$\equiv (a^b)^{k+b} \quad (HI)$
$\equiv (a^b)^{3k} \quad (\cdot 2^t)$

a^{bk} não aparece como subexpressão para focar nele!

(18) **T**

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[\,n \leq m \text{ ou } m \leq n\,]$:

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

$product : \mathsf{ListNat} \to \mathsf{Nat}$ $\qquad$ $replaceAt, addAt : \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat}$

$tidy : \mathsf{ListNat} \to \mathsf{Bool}$ $\qquad$ $addNat, removeAt : \mathsf{Nat} \to \mathsf{ListNat} \to \mathsf{ListNat}$

$takeEvens, atEvens : \mathsf{ListNat} \to \mathsf{ListNat}$ $\qquad$ $pwAdd : \mathsf{ListNat} \to \mathsf{ListNat} \to \mathsf{ListNat}$

RESPOSTA. **Não** repita as tipagens na resposta!

```
ev : Nat -> Bool
ev 0 = true
ev (S n) = od n

od : Nat -> Bool
od 0 = False
od (S n) = ev n

product Nil = 1
product (x:xs) = x * product xs

addNat n Nil = Nil
addNat n (x:xs) = (n + x) : addNat n xs

pwAdd l Nil = Nil
pwAdd Nil l = l
pwAdd (x:xs) (y:ys) = (x+y) : pwAdd xs ys

removeAT n Nil = Nil
removeAT 0 (x:xs) = xs
removeAT (S n) (x:xs) = x : removeAT n xs

replaceAT n n Nil = Nil
replaceAT n 0 (x:xs) = n : xs
replaceAT n (S n) (x:xs) = x : replaceAT n n xs

takeEvens Nil = Nil
takeEvens (x:xs)
  | ev x = x : takeEvens xs
  | otherwise = takeEvens xs

tidy Nil = True
tidy (x:xs)
  | ev (length xs) = (od x) ∧ (tidy xs)
  | otherwise = (ev x) ∧ (tidy xs)

length : ListNat -> Nat
length Nil = 0
length (x:xs) = S (length xs)

atEvens Nil = Nil
atEvens (x:xs)
  | ev (length xs) = atEvens xs
  | otherwise = x : atEvens xs
```

como defeza o length da "tidy"?

por que favorecer a direita ?!

(9) C

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[a^{b\cdot c} = (a^b)^c]$:

2

indução.

Base $(\forall b, c)[0^{b\cdot c} = (0^b)^c]$:

Sejam b, c : Nat.

Calculamos:

$0^{b\cdot c} = 0 \quad (\neq 0) \leftarrow$?!

$= 0^b$?

$= (0^b)^c$?

Passo indutivo $(\forall b, c)[m^{b\cdot c} = (m^b)^c] \Rightarrow (\forall b, c)\,[(Sm)^{b\cdot c} = (Sm^b)^c]$:

Suponha $(\forall b, c)\,[m^{b\cdot c} = (m^b)^c]$. (1)

Sejam b, c : Nat.

. Aplico (1) em b e c, logo $m^{b\cdot c} = (m^b)^c$.

Calculamos:

$(Sm)^{b\cdot c} = 1 + m^{b\cdot c}$

$= 1 + (m^b)^c$

$=$

(18) T

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

Só isso mesmo.

(8) **I**

INTERNALIZE a relação $(\leq) : \mathsf{Nat} \times \mathsf{Nat} \to \mathsf{Prop}$ e os predicados $\mathrm{Even}, \mathrm{Odd} : \mathsf{Nat} \to \mathsf{Prop}$:

(18) **R**

Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

product : ListNat → Nat
tidy : ListNat → Bool
takeEvens, *atEvens* : ListNat → ListNat
replaceAt, *addAt* : Nat → Nat → ListNat
addNat, *removeAt* : Nat → ListNat → ListNat
pwAdd : ListNat → ListNat → ListNat

RESPOSTA. **Não** repita as tipagens na resposta!

```
product [] = 1
product (x:xs) = x * (product xs)

tidy [] = True
tidy (x:xs)

addNat n [] = [n]
addNat n (x:xs) = (n+x):(addNat n xs)

pwAdd (x:xs) (y:ys) = (x+y):(pwAdd xs ys)
pwAdd ys [] = ys
pwAdd [] ys = ys

takeEvens [] = []
takeEvens (x:xs)
  | even x = x:(takeEvens xs)
  | otherwise = takeEvens xs

atEvens [] = []
atEvens [x] = [x]
atEvens (x0:x1:xs) = x0:(atEvens xs)

addAt _ [] = []
```

(9) C

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[a^{b \cdot c} = (a^b)^c]$:

(18) T

10

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

Seja n : Nat.
Indução.
Base:
Basta demonstrar $0 \leq n$. Trivial.
not really!
Passo indutivo:
Seja k : Nat t.q. $n \leq k$ ou $k \leq n$. ✓
Caso $n \leq k$:
Basta demonstrar que $n \leq Sk$.
Seja i : Nat t.q. $n + i = k$. ✓
Calculamos:
$Sk = S(n+i)$ (pela escolha de i)
$= n + (Si)$ [(+).2]
Vou (Si) com fundamento. ✓

Caso $k \leq n$:
Basta demonstrar que $Sk \leq n$.
Como $k \leq n$, logo $Sk \leq Sn$.
Como $n \leq Sn$, logo $Sk \leq n$.
(Pela transitividade de $\leq$) ■
robou aqui!

Só isso mesmo.

(9) **C**

DEMONSTRAÇÃO DE $(\forall a)\,(\forall b)\,(\forall c)\,[a^{b \cdot c} = (a^b)^c]$:

Sejam a,b,c : Nat.

Indução em c.

Base : $[a^{b.0} \stackrel{?}{=} (a^b)^0]$

Calculemos:

$a^{b.0} = a^0$ ((·).1)

$(a^b)^0 = S0.$ ((^).1)

$= S0.$ ((^).1)

Passo indutivo: ~~$[a^{b.Sk} ? (a^b)^{Sk}]$~~ $[a^{b.k} = (a^b)^k \stackrel{?}{\Rightarrow} a^{b.Sk} = (a^b)^{Sk}]$ (∀k)

Seja ~~Suponha~~ k : Nat.

Calculemos:

$a^{b.Sk} = a^{(b.k)+b}$ ((·).2)

$= a^{b.k} \cdot a^b$ (Exp-add)

$= (a^b)^k \cdot a^b$ (H.I)

$= (a^b)^{Sk}$. ((^).2←)

(18) **T**

DEMONSTRAÇÃO DE $(\forall n)\,(\forall m)\,[n \leq m \text{ ou } m \leq n]$:

Só isso mesmo.