

Nome: Θάνος

Gabarito

2022-12-07

Regras:

- I. Não vires esta página antes do começo da prova.
- II. Nenhuma consulta de qualquer forma.
- III. Nenhum aparelho ligado (por exemplo: celular, tablet, notebook, *etc.*).¹
- IV. Nenhuma comunicação de qualquer forma e para qualquer motivo.
- V. $(\forall x) [\text{Colar}(x) \implies \neg \text{Passar}(x, \text{FMC1})]$.²
- VI. Responda dentro das caixas indicadas, escrevendo em forma clara e facilmente legível.
- VII. Nenhuma prova será aceita depois do fim do tempo—mesmo se for atraso de 1 segundo.
- VIII. Respeite as restrições dos problemas que têm escolha.³
- IX. Escolha até um dos L, T.⁴

Esclarecimento: Tuas demonstrações precisam ser escritas na linguagem mid-level que temos elaborado na disciplina.⁵ Tuas definições devem utilizar apenas a sintaxe e a notação que temos utilizado na disciplina.

Dados:

```
data Nat                data Maybe α                data List α
  0 : Nat                Nothing : Maybe α                Nil  : List α
  S : Nat → Nat         Just   : α → Maybe α            Cons : α → List α → List α

data Either α β        data Tree α                data Dir
  Left  : α → Either α β    Tip    : α → Tree α                L : Dir
  Right : β → Either α β    Fork   : Tree α → Tree α → Tree α    R : Dir

(+) : Nat → Nat → Nat    (*) : Nat → Nat → Nat    (++) : List α → List α → List α
m + 0 = m                m * 0 = 0                [] ++ ys = ys
m + (S n) = S (m + n)    m * (S n) = n + (m * n)  (x:xs) ++ ys = x : (xs ++ ys)

(.) : (b → c) → (a → b) → (a → c)
(f . g) x = f (g x)

type Path = List Dir
```

Os teoremas: (+)-ass/com/id/inv, (·)-ass/com, (≤)-refl/trans/min/succ.

Boas provas!

¹Ou seja, *desligue antes* da prova.

²Se essa regra não faz sentido, melhor desistir desde já.

³Respostas violando essa regra (respondendo em mais questões) tirarão 0 pontos.

⁴Provas violando essa regra (com respostas em mais problemas) não serão corrigidas (tirarão 0 pontos).

⁵Não inclua os Dados/Alvo nem outros rascunhos no teu texto!

(38) **L**

(12) **L1.** Complete as igualdades seguintes **com algo interessante**:⁶

$$\begin{array}{ll} \text{reverse} \circ \text{reverse} = \text{id} & \text{map id} = \text{id} \\ \text{map } f \circ \text{map } g = \text{map } (f \circ g) & \text{filter } p \circ \text{map } f = \text{map } f \circ \text{filter } (p \circ f) \\ \text{map } f (xs \# ys) = \text{map } f xs \# \text{map } f ys & \text{sum} \circ \text{map } (n \cdot) = (n \cdot) \circ \text{sum} \end{array}$$

(6) **L2.** Defina recursivamente as funções: reverse, map, filter.

DEFINIÇÕES.

$\begin{array}{l} \text{reverse} : \text{List } \alpha \rightarrow \text{List } \alpha \\ \text{reverse } [] = [] \\ \text{reverse } (x :: xs) = \text{reverse } xs \# [x] \end{array}$	$\begin{array}{l} \text{map} : (\alpha \rightarrow \beta) \rightarrow \text{List } \alpha \rightarrow \text{List } \beta \\ \text{map } f [] = [] \\ \text{map } f (x :: xs) = [f x] \# \text{map } f xs \end{array}$
$\begin{array}{l} \text{filter} : (\alpha \rightarrow \text{Bool}) \rightarrow \text{List } \alpha \rightarrow \text{List } \alpha \\ \text{filter } p [] = [] \\ \text{filter } p (x :: xs) = \text{if } p x \text{ then } x :: (\text{filter } p xs) \text{ else } \text{filter } p xs \end{array}$	

(20) **L3.** Escolha **exatamente uma da primeira coluna do L1** para demonstrar.

DEMONSTRAÇÃO DA _____.

$$(\text{map } f \circ \text{map } g) \ell \stackrel{?}{=} \text{map } (f \circ g) \ell.$$

Por indução no ℓ .

CASO $[]$.

$$\begin{array}{ll} (\text{map } f \circ \text{map } g) [] & \\ = \text{map } f (\text{map } g []) & ((\circ).1) \\ = \text{map } f [] & (\text{map}.1) \\ = [] & (\text{map}.1) \end{array}$$

$$\begin{array}{ll} \text{map } (f \circ g) [] & \\ = [] & (\text{map}.1) \end{array}$$

CASO $(x :: xs)$.

$$\begin{array}{ll} (\text{map } f \circ \text{map } g) (x :: xs) & \\ = \text{map } f (\text{map } g (x :: xs)) & ((\circ).1) \\ = \text{map } f (g x :: \text{map } g xs) & (\text{map}.2) \\ = f (g x) :: \text{map } f (\text{map } g xs) & (\text{map}.2) \\ = (f \circ g) x :: \text{map } f (\text{map } g xs) & ((\circ).1) \\ = (f \circ g) x :: \text{map } (f \circ g) xs & (\text{HI}) \\ = \text{map } (f \circ g) (x :: xs) & (\text{map}.2) \end{array}$$

$$(\text{reverse} \circ \text{reverse}) \ell \stackrel{?}{=} \text{id } \ell.$$

Calculamos.

$$\begin{array}{ll} (\text{reverse} \circ \text{reverse}) \ell & \\ = \text{reverse } (\text{reverse } \ell) & ((\circ).1) \\ = \ell & ((\text{rev-rev})) \\ = \text{id } \ell & ((\text{id}.1)) \end{array}$$

$$\text{map } f (xs \# ys) \stackrel{?}{=} \text{map } f xs \# \text{map } f ys.$$

Por indução no xs .

CASO $[]$.

$$\begin{array}{ll} \text{map } f ([] \# ys) & \\ = \text{map } f ys & ((\#).1) \end{array}$$

$$\begin{array}{ll} \text{map } f [] \# \text{map } f ys & \\ = [] \# \text{map } f ys & (\text{map}.1) \\ = \text{map } f ys & ((\#).1) \end{array}$$

CASO $(x :: xs)$.

$$\begin{array}{ll} \text{map } f ((x :: xs) \# ys) & \\ = \text{map } f (x : (xs \# ys)) & ((\#).2) \\ = f x : \text{map } f (xs \# ys) & (\text{map}.2) \\ = f x : (\text{map } f xs \# \text{map } f ys) & (\text{HI}) \\ = (f x : \text{map } f xs) \# \text{map } f ys & ((\#).2) \\ = \text{map } f (x : xs) \# \text{map } f ys & (\text{map}.2) \end{array}$$

⁶DEFINIÇÃO. Chamamos algo de *interessante* sse Thanos acha tal algo interessante.

(56) **T**

(7) **T1.** Escreva a regra de inferência que corresponde à indução do tipo `Tree α`.

$$\frac{(\forall x : \alpha) [\varphi(\text{Tip } x)] \quad (\forall l, r : \text{Tree } \alpha) [\varphi(l) \ \& \ \varphi(r) \implies \varphi(\text{Fork } l \ r)]}{(\forall t : \text{Tree } \alpha) [\varphi(t)]} \text{IND}_{\varphi}^{\text{Tree } \alpha}$$

(5) **T2.** Escreva **apenas nomes e tipagens** para os destrutores do tipo `Tree α`.⁷

`tip : Tree α → α` `lchild, rchild : Tree α → Tree α`

(4) **T3.** Defina o que precisa para o `Tree` virar um Functor.⁸

DEFINIÇÃO.

`fmap : (α → β) → Tree α → Tree β`
`fmap f (Tip x) = Tip (f x)`
`fmap f (Fork l r) = Fork (fmap f l) (fmap f r)`

(16) **T4.** Levando em consideração os exemplos de uso no quadro, defina recursivamente as funções:

(2×) `forks, tips : Tree α → Nat` `flat : Tree α → List α`
(5×) `search : α → Tree α → List Path` `fetch : Path → Tree α → Maybe α.`

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

```
forks (Tip _) = 0
forks (Fork l r) = S (forks l + forks r)

tips (Tip _) = S 0
tips (Fork l r) = tips l + tips r

flat (Tip x) = [x]
flat (Fork l r) = flat l ++ flat r

search w (Tip x) = if w = x then [[]] else []
search w (Fork l r) = let (lpaths, rpaths) = (search w l, search w r)
                      in map (L ::) lpaths ++ map (R ::) rpaths

fetch (L :: ds) (Fork l _) = fetch ds l
fetch (R :: ds) (Fork _ r) = fetch ds r
fetch [] (Tip x) = Just x
fetch _ _ = Nothing
```

⁷Versões parciais/unsafe!

⁸Com tipagem; e **sem** demonstrar as leis necessárias!

(24) **T5.** Demonstre **exatamente uma das**:

(16) (i) $\text{tips} = \text{S} \circ \text{forks}$;

(24) (ii) $\text{flat} \circ \text{fmap } f = \text{map } f \circ \text{flat}$.

Podem considerar dados quaisquer dos teoremas da **L1**.

DEMONSTRAÇÃO.

(i) $\text{tips } t \stackrel{?}{=} (\text{S} \circ \text{forks}) t$.

Por indução na t .

CASO (Tip x).

$$\begin{aligned} & (\text{S} \circ \text{forks}) (\text{Tip } x) \\ &= \text{S} (\text{forks} (\text{Tip } x)) && ((\circ).1) \\ &= \text{S } \mathbf{O} && ((\text{forks}).1) \\ &= \text{tips} (\text{Tip } x) && ((\text{tips}).1) \end{aligned}$$

CASO (Fork $l r$).

$$\begin{aligned} & \text{tips} (\text{Fork } l r) \\ &= \text{tips } l + \text{tips } r && ((\text{tips}).2) \\ &= (\text{S} \circ \text{forks}) l + (\text{S} \circ \text{forks}) r && (\text{HI}) \\ &= \text{S} (\text{forks } l) + \text{S} (\text{forks } r) && ((\circ).1) \\ &= \text{S} (\text{S} (\text{forks } l) + \text{forks } r) && ((+).2) \\ &= \text{S} (\text{forks } r + \text{S} (\text{forks } l)) && ((+)-\text{com.}) \\ &= \text{S} (\text{S} (\text{forks } r + \text{forks } l)) && ((+).2) \\ &= \text{S} (\text{S} (\text{forks } l + \text{forks } r)) && ((+)-\text{com.}) \\ &= \text{S} (\text{forks} (\text{Fork } l r)) && (\text{forks}.2) \\ &= (\text{S} \circ \text{forks}) (\text{Fork } l r) && ((\circ).1) \end{aligned}$$

(ii) $(\text{flat} \circ \text{fmap } f) t \stackrel{?}{=} (\text{map } f \circ \text{flat}) t$.

Por indução na t .

CASO (Tip x).

$$\begin{aligned} & (\text{flat} \circ \text{fmap } f) (\text{Tip } x) \\ &= \text{flat} (\text{fmap } f (\text{Tip } x)) && ((\circ).1) \\ &= \text{flat} (\text{Tip } (f x)) && (\text{fmap}.1) \\ &= [f x] && (\text{flat}.1) \end{aligned}$$

$$\begin{aligned} & (\text{map } f \circ \text{flat}) (\text{Tip } x) \\ &= \text{map } f (\text{flat} (\text{Tip } x)) && ((\circ).1) \\ &= \text{map } f [x] && (\text{flat}.1) \\ &\stackrel{\text{SUG}}{=} \text{map } f (x :: []) \\ &= f x :: \text{map } f [] && (\text{map}.2) \\ &= f x :: [] && (\text{map}.1) \\ &\stackrel{\text{SUG}}{=} [f x] \end{aligned}$$

CASO (Fork $l r$).

$$\begin{aligned} & (\text{flat} \circ \text{fmap } f) (\text{Fork } l r) \\ &= \text{flat} (\text{fmap } f (\text{Fork } l r)) && ((\circ).1) \\ &= \text{flat} (\text{Fork} (\text{fmap } f l) (\text{fmap } f r)) && (\text{fmap}.2) \\ &= \text{flat} (\text{fmap } f l) ++ \text{flat} (\text{fmap } f r) && (\text{flat}.2) \\ &= (\text{flat} \circ \text{fmap } f) l ++ (\text{flat} \circ \text{fmap } f) r && ((\circ).1) \\ &= (\text{map } f \circ \text{flat}) l ++ (\text{map } f \circ \text{flat}) r && (\text{HI de } (l)) \\ &= (\text{map } f \circ \text{flat}) l ++ (\text{map } f \circ \text{flat}) r && (\text{HI de } (r)) \\ &= \text{map } f (\text{flat } l) ++ \text{map } f (\text{flat } r) && ((\circ).1) \\ &= \text{map } f (\text{flat } l ++ \text{flat } r) && (\text{map-distr}(++)) \\ &= \text{map } f (\text{flat} (\text{Fork } l r)) && (\text{flat}.2) \\ &= (\text{map } f \circ \text{flat}) (\text{Fork } l r) && ((\circ).1) \end{aligned}$$

Obs: o $(\text{map-distr}(++))$ está na **L1**.

Só isso mesmo.

LEMMATA

rev-rev.

$$\text{rev}(\text{rev } \ell) = \ell$$

DEMONSTRAÇÃO.

Por indução na ℓ .

CASO []:

$$\begin{aligned} \text{rev}(\text{rev } []) & \\ &= \text{rev } [] \\ &= [] \end{aligned}$$

CASO $(x :: xs)$:

$$\begin{aligned} \text{rev}(\text{rev } (x :: xs)) & \\ &= \text{rev}(\text{rev } xs \# [x]) && (\text{rev.2}) \\ &= \text{rev } [x] \# \text{rev}(\text{rev } xs) && ((\text{rev-}(\#))) \\ &= \text{rev } [x] \# xs && (\text{HI}) \\ &\stackrel{\text{sug}}{=} \text{rev } (x :: []) \# xs \\ &= (\text{rev } [] \# [x]) \# xs && (\text{rev.2}) \\ &= ([] \# [x]) \# xs && (\text{rev.1}) \\ &= [x] \# xs && ((\#).1) \end{aligned}$$

nil-idR.

$$\ell \# [] = \ell$$

DEMONSTRAÇÃO.

Por indução na ℓ .

CASO []: $[] \# [] = []$.

CASO $(x :: xs)$: $(x :: xs) \# [] = x :: (xs \# []) = x :: xs$.

(#)-ass.

$$(xs \# ys) \# zs = xs \# (ys \# zs)$$

DEMONSTRAÇÃO.

Por indução na xs .

CASO []:

$$\begin{aligned} [] \# (ys \# zs) & \\ &= ys \# zs \\ ([] \# ys) \# zs & \\ &= ys \# zs \end{aligned}$$

CASO $(x :: xs)$:

$$\begin{aligned} (x :: xs) \# (ys \# zs) & \\ &= x :: (xs \# (ys \# zs)) && (\#.2) \\ &= x :: ((xs \# ys) \# zs) && (\text{HI}) \\ &= (x :: (xs \# ys)) \# zs && (\#.2) \\ &= ((x :: xs) \# ys) \# zs && (\#.2) \end{aligned}$$

rev-(#).

$$\text{rev}(xs \# ys) = \text{rev } ys \# \text{rev } xs.$$

DEMONSTRAÇÃO.

Por indução na xs .

CASO []:

$$\begin{aligned} \text{rev}([] \# ys) & \\ &= \text{rev } ys && ((\#).1) \\ \text{rev } ys \# \text{rev } [] & \\ &= \text{rev } ys \# [] && (\text{rev}.1) \\ &= \text{rev } ys && ((\text{nil-idR})) \end{aligned}$$

CASO $(x :: xs)$:

$$\begin{aligned} \text{rev}((x :: xs) \# ys) & \\ &= \text{rev}(x :: (xs \# ys)) \\ &= \text{rev}(xs \# ys) \# [x] && (\text{rev.2}) \\ &= (\text{rev } ys \# \text{rev } xs) \# [x] && (\text{HI}) \\ &= \text{rev } ys \# (\text{rev } xs \# [x]) && ((\#)\text{-ass.}) \\ &= \text{rev } ys \# (\text{rev}(x :: xs)) && ((\#)\text{-ass.}) \end{aligned}$$