Teoria de Categorias & Programação Funcional
2023.2

Hugo Nobrega

Lista de Exercícios 2

As entregas podem ser feitas em duplas, mas lembre-se que
não poderá haver repetição de duplas em listas diferentes!
Entregue todas as questões marcadas com * até
04/12 às 23:59

Questão 1. Em aula fizemos a seguinte definição:

```
data Natural = Zero | Suc Natural
  deriving (Eq, Show)
```

o que faz com que o seguinte aconteça no REPL:

```
> Zero == Zero
True
> Zero == Suc Zero
False
> Suc . Suc . Suc . Suc $ Zero
Suc (Suc (Suc (Suc Zero)))
```

*** a.** Modifique a definição de `Natural` e faça as novas definições necessárias para que o seguinte passe a acontecer no REPL:

```
> Zero

> Suc Zero
#
> Suc . Suc . Suc . Suc $ Zero
####
> -- etc!
```

Dica: lembre-se de que você pode digitar

```
> :i Show
```

no REPL para obter mais informações sobre a classe de tipos `Show`.

b. Modifique a definição de `Natural` e faça as novas definições necessárias para que o seguinte passe a acontecer no REPL:

```
> Zero
0
> Suc Zero
1
> Suc . Suc . Suc . Suc $ Zero
4
> -- etc!
```

c. Usando o construtor de tipos `Maybe` para implementar a ideia de "operação parcial", faça a definição da operação de subtração entre naturais em Haskell.

d. Faz sentido tentarmos definir uma instância de `Semigroup` para `Natural`? Se sim, defina uma. Se não, justifique por quê.

*** e.** Faz sentido tentarmos definir uma instância de `Functor` para `Natural`? Se sim, defina uma. Se não, justifique por quê.

Questão 2. Além de funções e (construtores de) tipos, *classes de tipos* (*type classes*) também podem ser definidas em Haskell usando a palavra-chave `class`, como no exemplo abaixo:

```
class Foo x where
  f :: x -> x -> x
  g :: x -> x
  h :: x

  g = f h
```

No REPL:

```
> :i Foo
type Foo :: * -> Constraint
class Foo x where
  f :: x -> x -> x
  g :: x -> x
  h :: x
  {-# MINIMAL f, h #-}
```

Note que uma "implementação padrão" foi fornecida para `g`, o que significa que instâncias não precisam (mas podem) fornecer suas próprias implementações:

```haskell
instance Foo Integer where
  f = (*)
  g = (50 +)
  h = 157

instance Foo String where
  f = (++)
  h = "Hugo"
```

Em matemática, um *anel* é uma estrutura com assinatura:

- um universo
- duas operações binárias (que aqui denotaremos $\#$ e $\star$)
- uma operação unária (que aqui denotaremos $\underset{\#}{inv}$)
- duas constantes (que aqui denotaremos z e u)

satisfazendo os seguintes axiomas:

1. "$\#$ é associativa"
2. "$\#$ é comutativa", ou seja:

$$\forall x, y \ \ (x \# y = y \# x)$$

3. "z é um elemento neutro para $\#$", ou seja:

$$\forall x \ \ (x \# z = x = z \# x)$$

4. "a operação $\underset{\#}{inv}$ é inversa em relação a $\#$" no seguinte sentido:

$$\forall x \ \ \left[(x \# \underset{\#}{inv}(x)) = z = (\underset{\#}{inv}(x) \# x)\right]$$

5. "$\star$ é associativa"
6. "u é um elemento neutro para $\star$", ou seja:

$$\forall x \ \ (x \star u = x = u \star x)$$

7. "$\star$ é distributiva em relação a $\#$", ou seja:

$$\forall x, y, z \ \ [x \star (y \# z) = (x \star y) \# (x \star z)]$$
$$\forall x, y, z \ \ [(y \# z) \star x = (y \star x) \# (z \star x)]$$

*** a.** Dê uma definição para a classe de tipos dos anéis em Haskell e defina instâncias de `Natural` (definido aula) e `Integer` para essa classe. Você consegue satisfazer todos os axiomas com as suas instâncias?

Um *corpo* é uma estrutura com assinatura obtida da assinatura dos anéis adicionando-se uma operação parcial unária (que aqui denotaremos $\underset{\star}{inv}$), satisfazendo os seguintes axiomas:

1.–7. todos os axiomas de anéis

8. "$\star$ é comutativa", ou seja:

$$\forall x, y \;\; (x \star y = y \star x)$$

9. $\forall x \;\; (x \neq z \implies x \in \mathrm{dom}(\underset{\star}{inv}))$

10. "$\underset{\star}{inv}$ é uma operação inversa de $\star$", ou seja:

$$\forall x \;\; (x \neq z \quad \implies \quad x \star \underset{\star}{inv}(x) = u = \underset{\star}{inv}(x) \star x)$$

b. Novamente usando `Maybe` para implementar a noção de "operação parcial", dê uma definição para a classe de tipos dos corpos em Haskell e defina instâncias de `Bool` e `Float` para essa classe. Você consegue satisfazer todos os axiomas com as suas instâncias?

Questão 3. Implemente o algoritmo de Euclides em Haskell, com assinatura `Integer -> Integer -> Integer`. (Vamos estipular que o mdc de 0 e 0 é 0). Lembre-se que `Integer` também inclui valores negativos!

***Questão 4.** Defina uma função de tipo `[a] -> [a]` em Haskell, diferente da função identidade e da função "constante com saída lista vazia", e prove (i.e., com um texto, fora do Haskell) que sua função é uma transformação natural do funtor "lista" para ele mesmo.

Não é permitido fazer um apelo a "teoremas de graça"!

Lembrete: a notação `[a] -> [a]` quer dizer que a sua função deve funcionar para listas de quaisquer tipos (mas, em qualquer chamada, o tipo da saída deve ser sempre o mesmo da entrada).

Questão 5. Nesta questão, use a nossa definição de `ArvBin` vista em aula.

a. Faça uma função de tipo `ArvBin a -> [a]` que produza uma lista contendo exatamente os valores dos nós não-folha da árvore dada (em qualquer ordem).

b. Faça uma função de tipo `ArvBin a -> Integer`, que indica quantos nós não-folha tem a árvore dada na entrada.

*** c.** Faça uma função de tipo `a -> ArvBin a -> Bool`, que indica se o valor de tipo `a` dado na entrada está em algum nó da árvore dada na entrada ou não.

*** d.** Vamos chamar de "arrumada" uma árvore binária na qual, para qualquer valor em qualquer nó da árvore, tenhamos que esse valor é

- maior ou igual a todos os valores na subárvore filha da esquerda do nó; e
- menor ou igual a todos os valores na subárvore filha da direita do nó.

Faça uma função que indique (com um `Bool`) se a árvore de tipo `ArvBin a` dada na entrada é "arrumada". Note que você vai precisar exigir algo sobre `a`.

Questão 6. Prove que

$$A \xleftarrow{\pi_0} X \xrightarrow{\pi_1} B \text{ é um produto em uma categoria } \mathcal{C}$$
$$\text{sse}$$
$$A \xrightarrow{\pi_0} X \xleftarrow{\pi_1} B \text{ é um coproduto na categoria dual } \mathcal{C}^{\text{op}}.$$

Questão 7. Prove que nas seguintes categorias todos os pares de objetos têm um produto.

*** a.** A categoria das matrizes "Mat" (objetos são os naturais, setas são as matrizes de números reais, dom é "quantidade de colunas", cod é "quantidade de linhas" e composta definida por $A \circ B = A \cdot B$, sendo $\cdot$ o produto usual de matrizes).

b. Categoria dos monoides "Mon" (objetos são os monoides, setas são os homomorfismos de monoides, o restante como usual para funções).

c. A categoria "Cat" de todas as categorias.

d. Dada uma assinatura qualquer, a categoria das estruturas dessa assinatura.

e. Categoria da lógica proposicional "Prop", que é a categoria poset construída a partir da seguinte pré-ordem: os elementos são as fórmulas da lógica proposicional (sem quantificadores), e a pré-ordem é dada por

$$\varphi \preceq \psi \quad \text{sse} \quad \varphi \to \psi \text{ é uma tautologia (i.e., é sempre verdadeira)}$$

Questão 8. Prove que nas seguintes categorias todos os pares de objetos têm um coproduto.

*** a.** Mat

b. Cat

c. Prop

***Questão 9.** Seja $G = (V, A)$ um grafo direcionado e considere a categoria "vértices e passeios" construída a partir de G como visto em sala.

Prove que os únicos isomorfismos desta categoria são as setas identidade de cada objeto.

Questão 10. Prove que os seguintes pares de objetos nas categorias dadas são isomorfos:

a. Em Cat, a categoria poset dos naturais divisores de 30 com a relação de divisibilidade, e a categoria poset dos subconjuntos de $\{0, 1, 2\}$ com a relação de subconjunto.

b. Em Mon, o monoide das palavras finitas de um alfabeto de 1 letra com a operação de concatenação e constante "palavra-vazia", e o monoide dos naturais com operação de soma e constante 0.

c. Em Mon, o monoide dos números reais com operação de soma e constante 0, e o monoide dos números reais estritamente positivos, com operação de multiplicação e constante 1.

d. Em Set, o conjunto $\wp(\mathbb{N})$ [o conjunto das partes de $\mathbb{N}$, i.e., o conjunto cujos elementos são todos os subconjuntos de $\mathbb{N}$] e o conjunto de todas as sequências infinitas de 0s e 1s.

Questão 11. Considere a categoria Pos onde os objetos são todos os posets (conjuntos dotados de uma ordem parcial, i.e., uma relação reflexiva, antissimétrica e transitiva) e as setas são as funções não-decrescentes, i.e., que preservam ordem (como vimos em sala, isso é simplesmente o que "preservação de estrutura" quer dizer neste caso!).

a. Prove que, dados posets $(X, \leqslant_X)$ e $(Y, \leqslant_Y)$, um produto deles em Pos é dado por

$$(X, \leqslant_X) \xleftarrow{\pi_0} (X \times Y, \leqslant) \xrightarrow{\pi_1} (Y, \leqslant_Y)$$

onde $X \xleftarrow{\pi_0} (X \times Y) \xrightarrow{\pi_1} Y$ é o produto cartesiano usual de Set, e $\leqslant$ é definida por

$$(x_0, y_0) \leqslant (x_1, y_1) \quad \text{sse} \quad x_0 \leqslant_X x_1 \text{ e } y_0 \leqslant_Y y_1.$$

*** b.** Prove que em Pos temos o seguinte isomorfismo pra qualquer par de conjuntos A, B:

$$(\wp(A), \subseteq) \times (\wp(B), \subseteq) \quad \simeq \quad (\wp(A \uplus B), \subseteq)$$

onde $\wp$ é "conjunto das partes de" e $\uplus$ é "união disjunta".

Dica: o que está do lado esquerdo de $\simeq$ é (o objeto de) um produto; portanto, para provar que vale o isomorfismo, basta provar que o lado direito ...

Questão 12 (Versão "hardcore" da Questão 11)**.**

a. Prove que, dados posets $(X, \leqslant_X)$ e $(Y, \leqslant_Y)$, um produto deles em Pos é dado por

$$(X, \leqslant_X) \xleftarrow{\pi_0} (X \times Y, \leqslant) \xrightarrow{\pi_1} (Y, \leqslant_Y)$$

onde $X \xleftarrow{\pi_0} (X \times Y) \xrightarrow{\pi_1} Y$ é qualquer produto de X e Y em Set, e $\leqslant$ é definida por

$$a \leqslant b \quad \text{sse} \quad \pi_0(a) \leqslant_X \pi_0(b) \ \text{ e } \ \pi_1(a) \leqslant_Y \pi_1(b)$$

b. Prove que em Pos temos o seguinte isomorfismo pra qualquer par de conjuntos A, B:

$$(\wp(A), \subseteq) \times (\wp(B), \subseteq) \quad \simeq \quad (\wp(A + B), \subseteq)$$

onde

$$\wp(A) \xleftarrow{\pi_0} (\wp(A) \times \wp(B)) \xrightarrow{\pi_1} \wp(B) \qquad \text{e} \qquad A \xrightarrow{i_0} (A + B) \xleftarrow{i_1} B$$

são respectivamente qualquer produto e qualquer coproduto em Set.

***Questão 13.** Sejam $F, G : \mathcal{C} \to (P, \leqslant)$ funtores, onde $(P, \leqslant)$ é uma categoria poset.

Prove que qualquer função $\eta : \mathcal{O}(\mathcal{C}) \to \mathcal{A}(P, \leqslant)$ que satisfaça $\forall X \in \mathcal{O}(\mathcal{C})\ (\eta(X) : F(X) \leqslant G(X))$ é uma transformação natural entre F e G.