

Cálculo de Programas

Algebra of Programming

Lic. Ciências da Computação (3º ano)
Lic. em Engenharia Informática (3º ano)
UNIVERSIDADE DO MINHO

2026/27 — Ficha (Worksheet) nr.º 5

1. Mostre que a propriedade de cancelamento da exponenciação *Show that the cancellation property*

$$\text{ap} \cdot (\bar{f} \times \text{id}) = f \tag{F1}$$

corresponde à definição

is nothing but the definition

$$\text{curry } f \ a \ b = f \ (a, b)$$

quando se escreve $\text{curry } f$ em lugar de $\bar{f}$.

once curry f is written instead of $\bar{f}$.

2. Mostre que a definição *pointwise* de uncurry se pode obter também de (F1) fazendo $f := \text{uncurry } g$, introduzindo variáveis e simplificando. *Show that the pointwise definition of uncurry can also be obtained from (F1) by instantiating $f := \text{uncurry } g$, introducing variables and simplifying.*

3. Prove a igualdade *Prove the equality*

$$\overline{f \cdot (g \times h)} = \overline{\text{ap} \cdot (\text{id} \times h)} \cdot \bar{f} \cdot g \tag{F2}$$

usando as leis das exponenciais e dos produtos.

using the laws of products and exponentials.

4. É dada a definição *Let flip be defined by*

$$\text{flip } f = \widehat{\bar{f}} \cdot \text{swap} \tag{F3}$$

de acordo com:

according to:

$$\begin{array}{ccccccc} (C^B)^A & \cong & C^{A \times B} & \cong & C^{B \times A} & \cong & (C^A)^B \\ f & \mapsto & \hat{f} & \mapsto & \hat{f}.\text{swap} & \mapsto & \widehat{\bar{f}} \cdot \text{swap} = \text{flip } f \end{array}$$

Mostre que flip é um isomorfismo por ser a sua própria inversa:

Show that it is an isomorphism because it is its own inverse:

$$\text{flip} (\text{flip } f) = f \tag{F4}$$

Mostre ainda que:

Furthermore show:

$$\text{flip } f \ x \ y = f \ y \ x$$

5. Mostre que

Show that

$$\text{junc} \cdot \text{unjunc} = \text{id} \quad (\text{F5})$$

$$\text{unjunc} \cdot \text{junc} = \text{id} \quad (\text{F6})$$

se verificam, onde

hold for

$$A^{B+C} \begin{array}{c} \xrightarrow{\text{unjunc}} \\ \cong \\ \xleftarrow{\text{junc}} \end{array} A^B \times A^C \quad \left\{ \begin{array}{l} \text{junc}(f, g) = [f, g] \\ \text{unjunc } k = (k \cdot i_1, k \cdot i_2) \end{array} \right. \quad (\text{F7})$$

6. O código que se segue, escrito em Haskell, implementa a noção de ciclo-for, onde b é o corpo (“body”) do ciclo e i é a sua inicialização:

The following Haskell code implements a for-loop where b is the loop-body and i is its initialization:

$$\left\{ \begin{array}{l} \text{for } b \ i \ 0 = i \\ \text{for } b \ i \ (n+1) = b \ (\text{for } b \ i \ n) \end{array} \right. \quad (\text{F8})$$

Tem-se pois o tipo:

We thus have type:

$$\text{for} : (a \rightarrow a) \rightarrow a \rightarrow \mathbb{N}_0 \rightarrow a.$$

Mostre que $\text{for } b \ i$ satisfaz o diagrama que se segue, para um dado g :

Show that $\text{for } b \ i$ fits in the following diagram, for some g :

$$\begin{array}{ccc} \mathbb{N}_0 & \begin{array}{c} \xrightarrow{\text{out}} \\ \cong \\ \xleftarrow{\text{in}} \end{array} & 1 + \mathbb{N}_0 \\ \text{for } b \ i \downarrow & & \downarrow \text{id} + \text{for } b \ i \\ A & \xleftarrow{g} & 1 + A \end{array}$$

onde

where

$$\left\{ \begin{array}{l} \text{in} = [\text{zero}, \text{succ}] \\ \text{zero } _ = 0 \\ \text{succ } n = n + 1 \end{array} \right. \quad (\text{F9})$$

Sugestão: faça $g = [g_1, g_2]$ e resolva o problema em ordem a g_1 e g_2 .

Hint: make $g = [g_1, g_2]$ and solve the problem for g_1 and g_2 .

7. Na sequência da questão anterior, codifique (F8) em Haskell bem como *As follow up of the previous question, encode (F8) in Haskell as well as*

$$f = \pi_2 \cdot aux \textbf{ where } aux = \text{for } \langle \text{succ} \cdot \pi_1, \text{mul} \rangle (1, 1) \quad (\text{F10})$$

e inspecione o seu comportamento. Que *and inspect its behavior. Which function is f?*
função f é essa?

8. Mostre que $(a+)$ dada a seguir é um ciclo for b i (F8) para um dado b e um dado i — descubra quais: *Show that $(a+)$ given next is a for-loop for b i (F8) for b and i to be calculated:*

$$\begin{cases} a + 0 = a \\ a + (n + 1) = 1 + (a + n) \end{cases} \quad (\text{F11})$$

9. Qualquer função $k = \text{for } f$ i pode ser codificada em sintaxe C escrevendo *Any function $k = \text{for } f$ i can be encoded in the syntax of C by writing:*

```
int k(int n) {
    int r=i;
    int j;
    for (j=1; j<n+1; j++) {r=f(r);}
    return r;
};
```

Escreva em sintaxe C as funções $(a*) = \text{for } (a+) 0$ e outros catamorfismos de naturais de que se tenha falado nas aulas da UC. *Encode function $(a*) = \text{for } (a+) 0$ in C and other catamorphisms that have been discussed in the previous classes.*

10. **Questão prática** — Este problema não irá ser abordado em sala de aula. Os alunos devem tentar resolvê-lo em casa e, querendo, publicarem a sua solução no canal **#geral** do Slack, com vista à sua discussão com colegas. Dão-se a seguir os requisitos do problema. ***Open assignment** — This assignment will not be addressed in class. Students should try to solve it at home and, wishing so, publish their solutions in the **#geral** Slack channel, so as to trigger discussion among other colleagues. The requirements of the problem are given below.*

Problem requirements: The solution given for a previous problem,

$$\text{store } c = \text{take } 10 \cdot \text{nub} \cdot (c:) \quad (\text{F12})$$

calls the standard function

$$\text{nub} :: (Eq \ a) \Rightarrow [a] \rightarrow [a]$$

available from the Data.List library in Haskell.

After inspecting the standard implementation of this function, define f so that

$$\text{nub} = [\text{nil}, \text{cons}] \cdot f.$$

is an alternative to the standard definition, where $\text{nil} _ = []$ and $\text{cons } (h, t) = h : t$. Check that store c (F12) works properly once the standard nub is replaced by yours.

Important: Structure your solution across the $f \cdot g$, $\langle f, g \rangle$, $f \times g$, $[f, g]$ and $f + g$ combinators available from library *Cp.hs*. Use **diagrams** to plan your solution, in which you should avoid re-inventing functions over lists already available in the Haskell standard libraries.

□

11. **Questão prática** — Este problema não irá ser abordado em sala de aula. Os alunos devem tentar resolvê-lo em casa e, querendo, publicarem a sua solução no canal **#geral** do Slack, com vista à sua discussão com colegas. Dão-se a seguir os requisitos do problema.

Open assignment — This assignment will not be addressed in class. Students should try to solve it at home and, wishing so, publish their solutions in the **#geral** Slack channel, so as to trigger discussion among other colleagues. The requirements of the problem are given below.

Problem requirements: The following function

$$\begin{aligned} \text{func} &:: \text{Eq } a \Rightarrow b \rightarrow [(a, b)] \rightarrow (a \rightarrow b) \\ \text{func } b &= (\text{maybe } b \text{ id} \cdot) \cdot \text{flip lookup} \end{aligned}$$

“functionalizes” a finite list of (key, value) pairs by converting it to a function from keys to values. The first parameter provides a default value for keys that cannot be found in the list.

As example, let us have a list of people (where the key is some numeric id),

$$a = [(140999000, \text{"Manuel"}), (200100300, \text{"Mary"}), (000111222, \text{"Teresa"})]$$

their nationalities (if known),

$$b = [(140999000, \text{"PT"}), (200100300, \text{"UK"})]$$

and places of residence (if known):

$$c = [(140999000, \text{"Braga"}), (200100300, \text{"Porto"}), (151999000, \text{"Lisbon"})]$$

Using only func , $\langle f, g \rangle$, π_1 , π_2 , map and nub , write a Haskell expression representing the following data aggregation:

<i>Id</i>	<i>Name</i>	<i>Country</i>	<i>Residence</i>
140999000	<i>Manuel</i>	<i>PT</i>	<i>Braga</i>
200100300	<i>Mary</i>	<i>UK</i>	<i>Porto</i>
000111222	<i>Teresa</i>	<i>?</i>	<i>-</i>
151999000	<i>(Unknown)</i>	<i>?</i>	<i>Lisbon</i>

□