FP 2005-2006, Tussentoets

March 10, 2006, 13.00-15.00

The exam consists of 6 multiple choice questions (1 point each) and two open questions (2 points each). A wrong multiple choice answer will give a negative result (-0.25 point), whereas omitting the answer results in 0 points. Therefore, guessing is not recommended. Hand in the second page, with choices made and open questions answered. Put the right answer to the multiple choice questions in the corresponding box.

1. Which expressions are equivalent, i.e., can replace each other in any context?

   (a) $takeWhile\ p.dropWhile\ p$ and $dropWhile\ p$
   (b) $takeWhile\ p.dropWhile\ p$ and id
   (c) $takeWhile\ p.dropWhile\ q$ and $dropWhile\ q.takeWhile\ p$
   (d) $takeWhile\ p.takeWhile\ q$ and $takeWhile\ (\lambda x \rightarrow p\ x \wedge q\ x)$

2. What is the result of $foldr\ ((*).(*2))\ 2\ [2,2,2]$?

   (a) 16
   (b) 32
   (c) 64
   (d) 128

3. What is the type of $foldr\ map$?

   (a) $[a] \rightarrow [a \rightarrow a] \rightarrow [a]$
   (b) $[a] \rightarrow [[a \rightarrow a]] \rightarrow [a]$
   (c) $[a] \rightarrow [[a \rightarrow a] \rightarrow [a]]$
   (d) $[[a]] \rightarrow [a \rightarrow a] \rightarrow [a]$

4. Which of the following holds for the search trees (from Chapter 5)?

   (a) Elements can only occur once
   (b) You cannot search in an empty tree
   (c) You cannot delete an element that is not in the tree.
   (d) The time to find an element depends on the way the tree was constructed.

5. Someone who want to program list comprehension free, translates the right-hand side of the following definition into comprehension-free code. Which of the alternatives is the correct one?

$$segs\ xs = [\,] : [\,t \mid i \leftarrow inits\ xs, t \leftarrow tails\ i, not\ (null\ t)\,]$$

(a)
$$[\,] : concat\ (map\ f\ (inits\ xs))$$
$$\textbf{where}\ f\ i = concat\ (map\ g\ (tails\ i))$$
$$\textbf{where}\ g\ t = \textbf{if}\ not\ (null\ t)\ \textbf{then}\ [\,]\ \textbf{else}\ [t]$$

(b)
$$[\,] : concat\ (map\ f\ (inits\ xs))$$
$$\textbf{where}\ f\ i = concat\ (map\ g\ (tails\ i))$$
$$\textbf{where}\ g\ t = \textbf{if}\ not\ (null\ t)\ \textbf{then}\ [t]\ \textbf{else}\ [\,]$$

(c)
$$[\,] : concat\ (map\ f.map\ g)\ (inits\ xs)$$
$$\textbf{where}\ f\ i = tails\ i$$
$$g\ t = \textbf{if}\ null\ t\ \textbf{then}\ [\,]\ \textbf{else}\ [t]$$

(d)
$$[\,] : filter\ (not.null).concat\ (map\ f.map\ g)\ (inits\ xs)$$
$$\textbf{where}\ f\ i = tails\ i$$
$$g\ t = [t]$$

6. The function $intersperse :: a \rightarrow [a] \rightarrow [a]$ puts its first argument between all the elements of a non-empty list. Thus *intersperse* `'a' "xyz"` results in `"xayaz"`. Which definition is correct?

(a) $intersperse\ a\ as = foldr\ (\lambda e\ r \rightarrow (e : a : r))\ [\,]\ as$

(b) $intersperse\ a\ as = tail.concat.map\ (\lambda x \rightarrow [a, x])\ \$\ as$

(c) $intersperse\ a\ as = foldl\ (\lambda r\ e \rightarrow (a : e : r))\ [\,]\ as$

(d) $intersperse\ a\ as = tail\ [(a : e) \mid e \leftarrow as]$

7. Given the type

$$\textbf{data}\ Tree = Node\ Tree\ Int\ Tree$$
$$\mid\ Leaf$$

Write a function *sortedPathsOnly* that efficiently produces all paths from the root to the leafs that are increasing

8. Give the type and the definition of the function *foldl*. Use the function *foldl* to define a function $list2Int :: [Int] \rightarrow Int$ which maps a sequence of digits (represented as integers i, with $0 \leqslant i < 10$) to the *Int* value represented by these digits. Use the function in a program *readAndPrintTheSquare* that reads a line from the terminal which contains characters representing digits, and prints the square of the value represented by these digits. You may use the functions $char2Digit :: Char \rightarrow Int$, $show :: Int \rightarrow String$ and the standard *IO*-functions from the Helium prelude.

Name:

Student nr:

Bachelor program: Inf/CKI/...

Answers to the mutiple choice questions:

1	2	3	4	5	6

QUESTION 7, the function *sortedPathsOnly*:

(see other side/ zie andere zijde)

QUESTION 8, the functions *foldl*, *list2Int* and *readAndPrintAnInt* and their types: