Program Design and Data Structures, 20.0 c

Course code: 1DL201, Report code: DL201, 67%, DAG, NML,
week: 44 - 02 Semester: Autumn 2017
week: 03 - 11 Semester: Spring 2018

LAB ASSIGNMENT 9

This information is not available in English. Now showing the Swedish version.

After this lab you should be able to ...

- Work with higher-order functions on lists.

Instructions

- Read the slides for Lecture 14 (Currying, Partial Function Application, Higher-Order Functions on Lists).
- For this lab assignment, you are not allowed to use recursion or list comprehension when solving the following questions. Instead, use the higher-order functions on lists (map, filter, foldr, foldl) that were discussed in Lecture 14.
- For this lab assignment, you are not allowed to define (named) auxilliary functions. Instead, use anonymous functions (lambda expressions) where necessary.
- *Hint*: Every function you are asked to write below can be solved with one line of Haskell code.
- Remember to write function specifications for all (non-anonymous) functions that you write. Do this *before* you write your Haskell code. Also remember to follow the other parts of our coding convention (for identifiers and indentation), and to provide a variant for each recursive function.

The task

PROBLEM 1

Write a function sizeList :: [String] -> [Int] that, given a list of strings, returns a list of integers where every integer is the length of the corresponding string.

Example: sizeList ["En", "kul", "uppgift"] == [2, 3, 7]

PROBLEM 2

Write a function multiples :: Int -> [Int] -> [Int] such that multiples n xs returns a list of those integers in xs that are multiples of n (i.e., evenly divisible by n). You may assume that n is non-zero.

Example: multiples 7 [55, 98, 45, 28, 58, 8, 35] == [98, 28, 35]

PROBLEM 3

Write a function myConcat :: [String] -> String that, given a list of strings, returns the strings concatenated (in order) into one string.

Example: myConcat ["En", "kul", "uppgift"] == "Enkuluppgift"

Do *not* use the function concat from the Haskell Prelude to define myConcat.

PROBLEM 4

Consider the following function:

```
bowman poole (floyd:hal) =
  if null hal
  then floyd
  else poole floyd (bowman poole hal)
```

Explain what happens when the expression bowman (*) [2.5, 2.0, 0.2] is evaluated. You do not have to give a step-by-step description, but you need to show that you understand how the evaluation is performed. What is the value of this expression? (Try to answer the question without running the code. Run the code in GHCi only to check your answer, or if you are stuck.)

WHEN YOU ARE DONE

When you are done with all problems (the explorations below are optional), raise your hand or approach an assistant to have your solution graded.

If you pass this lab at least 30 minutes early and other groups are still working on it, we ask you to **help one other group**. Do not simply share your solution with them, but try to understand the (partial) solutions that they have developed so far (which may be different from yours) and the difficulties that they have. Assist them in coming up with their own solutions. Once the group that you are helping completes the lab assignment, you may leave. (If it is still early, the group that received your help should then stay to help another group.)

EXPLORATIONS

If you have finished the lab exercises and are waiting to be graded, or if you wish to explore programming in Haskell further at any time, or want to get some extra practice, have a look at the Explorations page.

Exploration problems are optional and should only be attempted after other problems are completed.
You don't have to show these answers to the lab assistants for grading.