Swansea University
Department of Computer Science

U Berger
14 March 2011

CS_191 Functional Programming I

Programming Laboratory 4

Lists and higher-order functions

In this Programming Laboratory session at least one of the two Exercises must be solved. As always, you may work in pairs.

You may download and use the file `fp1-lab110314.hs` from the course web page which contains the Haskell programs below.

When your solutions are complete, please show them to a lab supervisor for assessment. You are expected to be able to explain your solutions and run the functions you defined with some test data.

Exercise 1 We model a English-French dictionary by a list of pairs of strings

```
type Dict = [(String,String)]
```

In the file `fp1-lab110314.hs` you find an example of a dictionary. The following program `getF` takes a dictionary d and a word w and computes the list of all entries (e, f) in d such that w coincide with e.

```
getF :: Dict -> String -> Dict
getF d w = [ (e,f) | (e,f) <- d, w == e]
```

Try it out using the example dictionary $d1$.

Write a modified program `get :: Dict -> String -> Dict` that allows to use a dictionary in both directions and accepts incomplete words as inputs.

More precisely, `get` should takes a dictionary d and a word w and compute the list of all entries (e, f) in d such that w is an initial segment of e or of f.

You may use the following program for that purpose:

```
isinit :: String -> String -> Bool
isinit s1 s2 = l1 <= l2 && and [s1 !! i == s2 !! i | i <- [0..(l1-1)]]

  where  l1 = length s1
         l2 = length s2
```

Exercise 2 Use the higher-order function `map` to write a program that computes the list of lengths of a list of words.

For example, `["three","four","five"]` should result in `[5,4,4]`.