Simon Fraser University
School of Computing Science

CMPT 383

Assignment 4 (Functional Programming)

Due date: December 1, 2005

1. Imagine a language of expressions for representing integers defined by the syntax rules: (a) zero is an expression, (b) if `e` is an expression, then so are `succ(e)` and `pred(e)`.

   An evaluator reduces expressions in this language by applying the following rules repeatedly until no longer possible:

   ```
   succ(pred(e)) = e
   pred(succ(e)) = e
   ```

   Given the expression `succ(pred(succ(pred(pred(zero)))))`, write a **`reduction sequence`**. In how many ways can the reductions be applied to this expression? Do they all lead to the same final result?

2. Suppose a date is represented by a triple `(d,m,y)` of three integers, where `d` is the day, `m` is the month, and `y` is the year. Define a function **`age`** that takes two dates, the first being the current date, and the second being the birthdate of some person `P`, and return the age of `P` as a whole number of years.

3. Define a function **`convert`**`::Nat→Integer` that converts a natural number to an integer.

4. Define the function that **`split`**s a list of numbers into two lists: positive ones (including zero) and negative ones. For example

   ```
   ? split [3,-1,0,5,-2]
   ([3,0,5],[-1,-2])
   ```

5. The function `filter` takes a Boolean function `p` and a list `xs` and return that sublist of `xs` whose elements satisfy `p`. For example,

   ```
   ? filter even [1,2,4,5,32]
   [2,4,32]
   ```

   This function `filter` can be defined in terms of `concat` and `map`:

   ```
   filter p xs = concat · map box
             where box = …
   ```

   Give the definition of **`box`**.

6. The functions `takeWhile` and `dropWhile` are similar to `take` and `drop` except that they both take a boolean function as first argument instead of a natural number. The value `takeWhile p xs` is the longest initial segment of `xs` all of whose elements satisfy `p`. For example:

   ```
   ? takeWhile even [2,4,6,1,5,6]
   [2,4,6]
   ```

   The value `dropWhile p xs` gives what remains; for example:

   ```
   ? dropWhile even [2,4,6,1,5,6]
   [1,5,6]
   ```

   Give recursive definitions of **`takeWhile`** and **`dropWhile`**.

7. Define the function **`palindrome`** that verifies if a string is palindrome. A string is a palindrome if it reads the same in the forward and in the backward direction. For example:

   ```
   ? palindrome “madam”
   True
   ```

8. Write a program to **`convert`** a whole number of pence into words. For example, the number 3649 should convert to “thirty-six pounds and forty-nine pence”.

   ```
   ? convert 3649
   Thirty-six pounds and forty-nine pence
   ```

9. An integer `x` can be represented by a pair of integers `(y,z)` with `x=10*y+z`. For example, `27` can be represented by `(2,7)`, `(3,-3)`, and `(1,17)`, among others. Among possible representations we can choose one in which `abs z<5` and `abs y` is as small as possible (subject to `abs z≤5`). Define a function **`reprint`**, so that `reprint x` returns this canonical representation.

   ```
   ? reprint 27
   (3,-3)
   ```

10. Suppose that there are tab stops at every 8 spaces. Write a function that will take a string as an argument and return as a result a string that is equivalent to the input string, except that whenever two or more spaces can be replaced by a tab, this is done. You may assume that the input string contains no tabs, newlines, or other whitespace characters other than spaces.

    For example, “`1234?????01??4??7890123??6`” should be transformed into “`1234\t?01??4\t7890123??6`” where “`\t`” denotes a tab and “`?`” represents a space. (If we show the position of tab stops by “^”, then the input is “`1234????^?01??4??^7890123?^?6`”.)

    The first tab replaces 4 spaces while the other tab replaces 2 spaces. A third tab could replace the single space in the 24th position of the input, except that we only use a tab to replace 2 or more spaces. The 2 spaces between 1 and the 4 cannot be replaced with a tab, because the 4 is not positioned after a tab stop. The other 2 spaces, near the end of the input, remain because only 1 can be replaced by a tab.