FUNCTIONAL PROGRAMMING, 2013-2014 SPRING, FINAL EXAM

100 minutes May 28, 2014

Id	Full Name	Signature

Q 1	Q 2	Q 3	Total
/ 45	/ 30	/ 25	/ 100

1. Consider the following functions. (Note: The symbol '␣' represents a space. For any question, you can use the functions described in the previous questions even if you haven't answered that question.)

```
foo1 :: String -> String
foo1 []        = []
foo1 (x:xs)
  | x == '␣'  = []
  | otherwise = x : foo1 xs
```

```
foo2 :: String -> String
foo2 []        = []
foo2 (x:xs)
  | x == '␣'  = x:xs
  | otherwise = foo2 xs
```

(a) What is the result of the expression `foo1` ‘‘`functional␣programming`’’? What does the function `foo1` do?

(b) What is the result of the expression `foo2` ‘‘`functional␣programming`’’? What does the function `foo2` do?

(c) Write the function `lstrip` that removes leading spaces from a string.
For example: `lstrip` ‘‘`␣␣␣␣ab␣␣c␣def␣␣␣`‘‘ should produce ‘‘`ab␣␣c␣def␣␣␣`‘‘.

(d) Fill in the template below for the function `rstrip` that removes trailing spaces from a string.
For example: `rstrip` ‘‘`␣␣␣␣ab␣␣c␣def␣␣␣`‘‘ should produce ‘‘`␣␣␣␣ab␣␣c␣def`‘‘. Use the `lstrip` and `reverse` functions.

```
rstrip = _________________________________________
```

(e) Write the function `split` that splits a string into a list of words, where words are separated by a space.
For example: `split ‘‘␣␣␣␣ab␣␣c␣def␣␣‘‘` should produce `[‘‘ab’’, ‘‘c‘‘, ‘‘def’’]`.
Use the functions defined in (a) - (d).

(f) Write the function `join` that concatenates a list of strings by inserting a separator between its elements.
For example, `join ‘‘ ‘‘ [‘‘functional’’, ‘‘programming’’]` should produce `‘‘functional programming’’`,
whereas `join ‘‘ --> ‘‘ [‘‘A’’, ‘‘B’’, ‘‘C’’]` should produce `‘‘A --> B --> C’’`.

(g) Fill in the template below for the function `normalize` that removes extra spaces from a string.
For example: `normalize ‘‘␣␣␣␣ab␣␣c␣def␣␣␣‘‘` should produce `‘‘ab␣c␣def‘‘`. Use the `split` and `join` functions.

```
normalize = ____________________________________
```

Id	Full Name	Signature

Q 1	Q 2	Q 3	Total
/ 45	/ 30	/ 25	/ 100

2. What are the results of the following expressions? Briefly explain your answer.

   (a) `(map (==4) . filter even) [1 ..]`

   (b) `(take 5 . map (==4) . filter even) [1 ..]`

   (c) `foldr (||) False ((map (==4) . filter even) [1 ..])`

   (d) `foldl (||) False ((map (==4) . filter even) [1 ..])`

3. Consider the function given below. (Note: The function `getLine` reads a string from the standard input.)

```
empty = do line <- getLine
           return (length line == 0)
```

   (a) What is the type of the function `empty`?

   (b) The function `repeatIO` has the following signature: `repeatIO :: IO Bool -> IO () -> IO ()`.
   Write the definition of this function so that `repeatIO test oper` has the effect of repeating `oper` until the condition `test` is true.

   (c) Give an example about how `repeatIO` can be called using `empty` and explain how that example would behave.
   (Note: Your example doesn't have to carry out a meaningful operation.)