Haskell Hwk3

Martin Kozeny
CSCI 4501: Programming Language Structure
Spring 2011 University of New Orleans

March 23, 2011

1 Haskell code

```
1  import Text.Regex.Base
2  import Text.Regex.Posix
3  import System.IO
4  import System.Environment
5  import Data.Char
6  import Data.List(isPrefixOf)
7
8  --here is used code for splitting a string
9  --published on http://hackage.haskell.org/packages/archive/MissingH/1.0.0/doc/html/src/Data-List-Utils.html#split
10 startswith :: Eq a => [a] -> [a] -> Bool
11 startswith = isPrefixOf
12
13
14 spanList :: ([a] -> Bool) -> [a] -> ([a], [a])
15 spanList _ [] = ([],[])
16 spanList func list@(x:xs) =
17     if func list
18        then (x:ys,zs)
19        else ([],list)
20     where (ys,zs) = spanList func xs
21
22 breakList :: ([a] -> Bool) -> [a] -> ([a], [a])
23 breakList func = spanList (not . func)
24
25 split :: Eq a => [a] -> [a] -> [[a]]
26 split _ [] = []
27 split delim str =
28     let (firstline, remainder) = breakList (startswith delim) str
29         in
30         firstline : case remainder of
31                                   [] -> []
32                                   x -> if x == delim
33                                        then [] : []
34                                        else split delim
35                                                (drop (length delim) x)
36
37 --end of using code for splitting
38
39
40 --this function controls by reg. expression
41 --if is the input string email address
42 isEmail :: String -> Bool
43 isEmail text = if (text =~ "[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+.[a-zA-Z]{2,4}")
44                  then True
45                 else False
```

```
46  --this function only controls if input string contains '@'
47  containsAt :: String -> Bool
48  containsAt [] = False
49  containsAt (x:xs) = if x == '@'
50                              then True
51                          else
52                              containsAt xs
53
54  --this function takes as parameter list of strings
55  --and send recursively every element of this list to function 'convertToListOfPairs'
56  --before that this string is split into elements by space delimiter and controlled
57  --in function 'controlList'
58  nameList :: [String] -> [(String,[String])]
59  nameList [] = []
60  nameList (x:xs) = convertToListOfPairs (controlList(split " " x)) : (nameList xs)
61
62
63  --this function recursively control, if input list does not contain elements
64  --in which ',' appears; if some string contains that character,
65  --this function split it and controls if next string after ',' is not an empty string
66  --if yes, than empty string is omitted, else controls if string is not empty itself,
67  --else is string put back to list
68  controlList :: [String] -> [String]
69  controlList [] = []
70  controlList (x:xs) = if (length (split "," x)) > 1
71                           then (omitEmptyString (split "," x)) ++ (controlList xs)
72                       else if x == ""
73                           then controlList xs
74                           else x : (controlList xs)
75
76  --this function only controls if in input list of strings is not empty string
77  --if yes, then it is omitted
78  omitEmptyString :: [String] -> [String]
79  omitEmptyString [] = []
80  omitEmptyString (x:xs) = if x == ""
81                           then omitEmptyString xs
82                           else x:(omitEmptyString xs)
83
84
85  --this function creates recursively a tuple of string and list of strings
86  --by going through this list and controling if actual element does not contain '@'(or is not email address)
87  --if so, then is concatenated to first string in tuple,
88  --if not then is added to second element of tuple - list of mail address
89  convertToListOfPairs :: [String] -> (String,[String])
90  convertToListOfPairs [] = ("",[])
91  convertToListOfPairs (x:xs) = if not (isEmail x)
92                           then (x ++ " " ++ (fst (convertToListOfPairs xs)), snd (convertToListOfPairs xs))
93                           else (fst (convertToListOfPairs xs), x: (snd (convertToListOfPairs xs)) )
94
95
96  --this function format first element of input tuple to html source
97  formatNames :: [(String,[String])] -> String
98  formatNames [] = ""
99  formatNames (x:xs) = if (length (fst x) > 0)
100                          then "<li>" ++ (fst x) ++ (formatMails (snd x)) ++ "</li>" ++ (formatNames xs)
101                      else ""
102
103
104 --this function format element of input list to html source as link to email address
105 formatMails :: [String] -> String
106 formatMails [] = ""
107 formatMails (x:xs) = " , <a href='mailto:" ++ x ++ "'>" ++ x ++ "</a>" ++ (formatMails xs)
```

```
108
109 --this function initializes generating html source
110 generateHTML :: String -> String
111 generateHTML text = "<ul>" ++ (formatNames (nameList (split "\n" text))) ++ "</ul>"
112
113
114 --main action for calling(and compiling)
115 main = do htmlEmailDirectory
116
117
118 --action for reading input parameters
119 --(first is name of source text file and second is name of target html file)
120 --and calling function for generating html from provided source file
121 htmlEmailDirectory :: IO ()
122 htmlEmailDirectory = do
123     (inputFile:outputFile:_) <- getArgs
124     text <- readFile inputFile
125     writeFile outputFile (generateHTML text)
126     return ()
```

2 Test scripts

In some tests were used more spaces between tokens for really extreme testing, but these spaces are not shown in inputs presented in this paper because of well formatting. Input files with more spaces are submitted to BlackBoard. For testing it is necessary to give name of input text file as a first argument and name of output html file as a second argument.

2.1 Test1

2.1.1 Input

```
1 Jaime Nino jaime@cs.uno.edu, jnino@uno.edu, ninosalcedo.jaime@gmail.com
2 Curry Haskell cHaskell@functional.com, haskell.curry@gmail.com
```

2.1.2 Output

```
1 <ul>
2         <li>Jaime Nino , <a href='mailto:jaime@cs.uno.edu'>jaime@cs.uno.edu</a>
3         , <a href='mailto:jnino@uno.edu'>jnino@uno.edu</a> , <a
4                 href='mailto:ninosalcedo.jaime@gmail.com'>ninosalcedo.jaime@gmail.com</a></li>
5         <li>Curry Haskell , <a href='mailto:cHaskell@functional.com'>cHaskell@functional.com</a>
6         , <a href='mailto:haskell.curry@gmail.com'>haskell.curry@gmail.com</a></li>
7 </ul>
```

2.2 Test2

2.2.1 Input

```
1 Jaime Nino jaime@cs.uno.edu,jnino@uno.edu,ninosalcedo.jaime@gmail.com
2 Curry Haskell cHaskell@functional.com,haskell.curry@gmail.com
```

2.2.2 Output

```
1 <ul>
2         <li>Jaime Nino , <a href='mailto:jaime@cs.uno.edu'>jaime@cs.uno.edu</a>
3         , <a href='mailto:jnino@uno.edu'>jnino@uno.edu</a> , <a
4                 href='mailto:ninosalcedo.jaime@gmail.com'>ninosalcedo.jaime@gmail.com</a></li>
5         <li>Curry Haskell , <a href='mailto:cHaskell@functional.com'>cHaskell@functional.com</a>
6         , <a href='mailto:haskell.curry@gmail.com'>haskell.curry@gmail.com</a></li>
7 </ul>
```

2.3 Test3

2.3.1 Input

```
Martin Kozeny mkozeny@uno.edu , martin.kozeny@ample.cz
Jan Hradek ample@ample.cz , jan.hradek@ample.cz
Michal Janousek michal.janousek@ample.cz , michal.janousek@gmail.com
```

2.3.2 Output

```
<ul>
        <li>Martin Kozeny , <a href='mailto:mkozeny@uno.edu'>mkozeny@uno.edu</a>
        , <a href='mailto:martin.kozeny@ample.cz'>martin.kozeny@ample.cz</a></li>
        <li>Jan Hradek , <a href='mailto:ample@ample.cz'>ample@ample.cz</a>
        , <a href='mailto:jan.hradek@ample.cz'>jan.hradek@ample.cz</a></li>
        <li>Michal Janousek , <a href='mailto:michal.janousek@ample.cz'>michal.janousek@ample.cz</a>
        , <a href='mailto:michal.janousek@gmail.com'>michal.janousek@gmail.com</a></li>
</ul>
```

2.4 Test4

2.4.1 Input

```
Martin Kozeny mkozeny@uno.edu , martin.kozeny@ample.cz
Jan Hradek ample@ample.cz , jan.hradek@ample.cz
Michal Janousek michal.janousek@ample.cz , michal.janousek@gmail.com
```

2.5 Output

```
<ul>
        <li>Martin Kozeny , <a href='mailto:mkozeny@uno.edu'>mkozeny@uno.edu</a>
        , <a href='mailto:martin.kozeny@ample.cz'>martin.kozeny@ample.cz</a></li>
        <li>Jan Hradek , <a href='mailto:ample@ample.cz'>ample@ample.cz</a>
        , <a href='mailto:jan.hradek@ample.cz'>jan.hradek@ample.cz</a></li>
        <li>Michal Janousek , <a href='mailto:michal.janousek@ample.cz'>michal.janousek@ample.cz</a>
        , <a href='mailto:michal.janousek@gmail.com'>michal.janousek@gmail.com</a></li>
</ul>
```

2.6 Test5

2.6.1 Input

```
Dominik Hasek czech goalie #39 dominik@hasek.cz
Jaromir Jagr czech forward #68 jaromir@jagr.cz
Vaclav Klaus czech prezident vaclav@klaus.cz
Petr Necas czech prime minister petr@necas.cz
```

2.6.2 Output

```
<ul>
        <li>Dominik Hasek czech goalie #39 , <a
                href='mailto:dominik@hasek.cz'>dominik@hasek.cz</a></li>
        <li>Jaromir Jagr czech forward #68 , <a
                href='mailto:jaromir@jagr.cz'>jaromir@jagr.cz</a></li>
        <li>Vaclav Klaus czech prezident , <a
                href='mailto:vaclav@klaus.cz'>vaclav@klaus.cz</a></li>
        <li>Petr Necas czech prime minister , <a
                href='mailto:petr@necas.cz'>petr@necas.cz</a></li>
</ul>
```

2.7 Test6

2.7.1 Input

```
Petr Koukal center forward #42 petr.koukal@hcpce.cz , petr.koukal@gmail.com
Jan Kolar left forward #48 jan.kolar@hcpce.cz , kolarjan@seznam.cz
Jan Stary right forward #23 jan.stary@hcpce.cz , stary.j@centrum.cz , j.stary@post.cz
```

2.7.2 Output

```
<ul>
        <li>Petr Koukal center forward #42 , <a
                href='mailto:petr.koukal@hcpce.cz'>petr.koukal@hcpce.cz</a> , <a
                href='mailto:petr.koukal@gmail.com'>petr.koukal@gmail.com</a></li>
        <li>Jan Kolar left forward #48 , <a
                href='mailto:jan.kolar@hcpce.cz'>jan.kolar@hcpce.cz</a> , <a
                href='mailto:kolarjan@seznam.cz'>kolarjan@seznam.cz</a></li>
        <li>Jan Stary right forward #23 , <a
                href='mailto:jan.stary@hcpce.cz'>jan.stary@hcpce.cz</a> , <a
                href='mailto:stary.j@centrum.cz'>stary.j@centrum.cz</a> , <a
                href='mailto:j.stary@post.cz'>j.stary@post.cz</a></li>
</ul>
```

2.8 Test7

2.8.1 Input

```
Martin Kozeny mkozeny@uno.edu martin.kozeny@ample.cz ,kozenmar@fel.cvut.cz
Jan Hradek ample@ample.cz , jan.hradek@ample.cz
Michal Janousek michal.janousek@ample.cz michal.janousek@gmail.com
```

2.8.2 Output

```
<ul>
        <li>Martin Kozeny , <a href='mailto:mkozeny@uno.edu'>mkozeny@uno.edu</a>
        , <a href='mailto:martin.kozeny@ample.cz'>martin.kozeny@ample.cz</a> ,
        <a href='mailto:kozenmar@fel.cvut.cz'>kozenmar@fel.cvut.cz</a></li>
        <li>Jan Hradek , <a href='mailto:ample@ample.cz'>ample@ample.cz</a>
        , <a href='mailto:jan.hradek@ample.cz'>jan.hradek@ample.cz</a></li>
        <li>Michal Janousek , <a href='mailto:michal.janousek@ample.cz'>michal.janousek@ample.cz</a>
        , <a href='mailto:michal.janousek@gmail.com'>michal.janousek@gmail.com</a></li>
</ul>
```