Announcements

Lab 6
April 2014

<table>
<thead>
<tr>
<th>Su</th>
<th>Mo</th>
<th>Tu</th>
<th>We</th>
<th>Th</th>
<th>Fr</th>
<th>Sa</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19: Lab 6 Due</td>
</tr>
<tr>
<td>20</td>
<td>21</td>
<td>22: Quiz 6</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
</tr>
</tbody>
</table>

Leaderboard
http://csci3155.cs.colorado.edu/leaderboard.php

retest

```scala
def test(re: RegExpr, chars: List[Char], sc: List[Char] => Boolean): Boolean
```

This helper function will see if a prefix of `chars` matches the regular expression `re`. If there is a prefix match, then the success continuation `sc` is called with the remainder of `chars` that has yet to be matched. That is, the success continuation `sc` captures “what to do next if a prefix of `chars` successfully matches `re`.” If `test` discovers a failure to match, then it can “return false early.”
\[ \angle (\#) = \left\{ \begin{array}{ll}
\end{array} \right\} \]

```
def test(re: RegExpr, chars: List[Char], sc: List[Char] => Boolean): Boolean =
(re, chars) match {
  case (RNoString, _) => false
  case (REmptyString, _) => sc(chars)
  case (RConcat(re1, re2), _) => test(re1, chars, { ... fill in here ... })
  ...
}
```