Coursework 1

This coursework is worth 5% and is due on 13 October at 16:00. You are asked to implement a regular expression matcher and submit a document containing the answers for the questions below. You can do the implementation in any programming language you like, but you need to submit the source code with which you answered the questions. However, the coursework will *only* be judged according to the answers. You can submit your answers in a txt-file or pdf.

Disclaimer

It should be understood that the work submitted represents your own effort. You have not copied from anyone else. An exception is the Scala code I showed during the lectures, which you can use.

Tasks

The task is to implement a regular expression matcher based on derivatives. The implementation should be able to deal with the usual (basic) regular expressions

$$\varnothing, \epsilon, c, r_1 + r_2, r_1 \cdot r_2, r^*$$

but also with the following extended regular expressions:

$[c_1 c_2 \ldots c_n]$	a range of characters
r^+	one or more times r
$r^?$	optional r
$r^{\{n,m\}}$	at least n-times r but no more than m-times
$\sim r$	not-regular expression of r

In the case of $r^{\{n,m\}}$ we have the convention that $0 \leq n \leq m$. The meaning of these regular expressions is

$$\begin{array}{rcl} L([c_1 c_2 \ldots c_n]) & \stackrel{\text{def}}{=} & \{[c_1], [c_2], \ldots, [c_n]\} \\ L(r^+) & \stackrel{\text{def}}{=} & \bigcup_{1 \leq i} . L(r)^i \\ L(r^?) & \stackrel{\text{def}}{=} & L(r) \cup \{[]\} \\ L(r^{\{n,m\}}) & \stackrel{\text{def}}{=} & \bigcup_{n \leq i \leq m} . L(r)^i \\ L(\sim r) & \stackrel{\text{def}}{=} & \mathbb{A} - L(r) \end{array}$$

whereby in the last clause the set $\mathbb{A}$ stands for the set of *all* strings. So $\sim r$ means 'all the strings that r cannot match'. We assume ranges like $[a\text{-}z0\text{-}9]$ are a shorthand for the regular expression

$$[abcd \ldots z01 \ldots 9] \,.$$

Be careful that your implementation of *nullable* and *der* satisfies for every r the following two properties:

- $nullable(r)$ if and only if $[] \in L(r)$
- $L(der\, c\, r)) = Der\, c\, (L(r))$

Important! Your implementation should have explicit cases for the basic regular expressions, but also for the extended regular expressions. That means do not treat the extended regular expressions by just translating them into the basic ones. See also Question 2, where you asked to give the rules for *nullable* and *der*.

Question 1 (unmarked)

What is your King's email address (you will need it in Question 2)?

Question 2 (marked with 2%)

This question does not require any implementation. From the lectures you have seen the definitions for the functions *nullable* and der. Give the rules for the extended regular expressions:

$$\begin{array}{ll}
nullable([c_1 c_2 \ldots c_n]) & \stackrel{\text{def}}{=} \; ? \\
nullable(r^+) & \stackrel{\text{def}}{=} \; ? \\
nullable(r^?) & \stackrel{\text{def}}{=} \; ? \\
nullable(r^{\{n,m\}}) & \stackrel{\text{def}}{=} \; ? \\
nullable(\sim r) & \stackrel{\text{def}}{=} \; ? \\
\\
derc([c_1 c_2 \ldots c_n]) & \stackrel{\text{def}}{=} \; ? \\
derc(r^+) & \stackrel{\text{def}}{=} \; ? \\
derc(r^?) & \stackrel{\text{def}}{=} \; ? \\
derc(r^{\{n,m\}}) & \stackrel{\text{def}}{=} \; ? \\
derc(\sim r) & \stackrel{\text{def}}{=} \; ?
\end{array}$$

Question 3 (marked with 1%)

Implement the following regular expression for email addresses

$$([a\text{-}z0\text{-}9\_.\!-]^+) \cdot @ \cdot ([a\text{-}z0\text{-}9.\!-]^+) \cdot . \cdot ([a\text{-}z.]^{\{2,6\}})$$

and calculate the derivative according to your email address. When calculating the derivative, simplify all regular expressions as much as possible, but at least apply the following six simplification rules:

$$\begin{array}{lcl} r \cdot \varnothing & \mapsto & \varnothing \\ \varnothing \cdot r & \mapsto & \varnothing \\ r \cdot \epsilon & \mapsto & r \\ \epsilon \cdot r & \mapsto & r \\ r + \varnothing & \mapsto & r \\ \varnothing + r & \mapsto & r \end{array}$$

Write down your simplified derivative in the "mathematicical" notation using parentheses where necessary.

Question 4 (marked with 1%)

Consider the regular expression $/ \cdot * \cdot (\sim ([a\text{-}z]^* \cdot * \cdot / \cdot [a\text{-}z]^*)) \cdot * \cdot /$ and decide wether the following four strings are matched by this regular expression. Answer yes or no.

1. `"/**/"`
2. `"/*foobar*/"`
3. `"/*test*/test*/"`
4. `"/*test/*test*/"`

Question 5 (marked with 1%)

Let r_1 be the regular expression $a \cdot a \cdot a$ and r_2 be $(a^{\{19,19\}}) \cdot (a^?)$. Decide whether the following three strings consisting of as only can be matched by $(r_1^+)^+$. Similarly test them with $(r_2^+)^+$. Again answer in all six cases with yes or no.

These are strings are meant to be entirely made up of as. Be careful when copy-and-pasting the strings so as to not forgetting any a and to not introducing any other character.

1. `"aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa`
   `aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa`
   `aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa"`
2. `"aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa`
   `aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa`
   `aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa"`
3. `"aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa`
   `aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa`
   `aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa"`