Ayan Ahmad
K19002255

6CCS3CFL – COURSEWORK 1

OPTIONAL QUESTIONS

Question 1:

Email: ayan.ahmad@kcl.ac.uk
I am currently studying in London and will be studying on campus, in person face to face teaching.

Question 2:

I am a web developer, I started off with basic HTML and JS and CSS. Then I gained some experience in Adobe ColdFusion. I have made a few large business applications in PHP. I tend to focus more on the backend. Although I am extremely comfortable with frontend too.

I have decent experience in JavaScript (ReactJS and NodeJS) and am remarkably familiar with MySQL and MongoDB. I also have a decent amount of experience in Python, Java, C++, Flutter and Swift.

Through the courses at KCL, I have also worked in languages like: PDDL, Haskell, Ruby on Rails, Prolog, and a few more

I am currently learning Scala through this course and am thoroughly enjoying it. Lastly, I am learning Go for my own projects.

MANDATORY QUESTIONS

Question 3:

Nullable Expressions
$nullable([c1, c2, \ldots, cn\,]) \stackrel{\text{def}}{=} \ false$
$nullable(r^{+}) \ \stackrel{\text{def}}{=} \ nullable(r)$
$nullable(r^{?}) \ \stackrel{\text{def}}{=} \ true$
$nullable\left(r^{\{n\}}\right) \stackrel{\text{def}}{=} \ if\ n == 0$ $then\ true$ $else\ nullable(r)$
$nullable(r^{\{..m\}}) \ \stackrel{\text{def}}{=} \ true$
$nullable(r^{\{n..\}}) \ \stackrel{\text{def}}{=} \ if\ n == 0$ $then\ true$ $else\ nullable(r)$
$nullable(r^{\{n..m\}}) \ \stackrel{\text{def}}{=} \ if\ n == 0$ $then\ true$ $else\ nullable(r)$
$nullable(\sim r) \stackrel{\text{def}}{=} \ -nullable(r)$

Ayan Ahmad
K19002255

Der Expression
$der\ c\ ([c1, c2, \ldots, cn\ ]) \overset{\text{def}}{=}\ if\ c\ \in [c1, c2 \ldots cn]$ then 1 else 0
$der\ c\ (r+) \overset{\text{def}}{=}\ der\ c\ (r)\ .\ r^{*}$
$der\ c\ (r?\ ) \overset{\text{def}}{=}\ der\ c\ (r)$
$der\ c\ \left(r^{\{n\}}\right) \overset{\text{def}}{=} if\ n == 0\ then\ 0\ else\ der\ c\ (r)\ .\ \left(r^{n-1}\right)$
$der\ c\ \left(r^{\{..m\}}\right) \overset{\text{def}}{=}\ if\ m \leq 0\ then\ 0\ else\ der\ c\ (r)\ .\ \left(r^{\{..m-1\}}\right)$
$der\ c\ (r^{\{n...\}}) \overset{\text{def}}{=}\ if\ n == 0$ $then\ der\ c\ (r)\ .(r^{\{n..\}})$ $else\ der\ c\ (r)\ .\ (r^{\{n-1..\}})$
$der\ c\ (r^{\{n..m\}}) \overset{\text{def}}{=}\ if\ n == 0$ $then\ der\ c\ (r)\ .\ (r^{\{..m-1\}})$ $else\ der\ c\ (r)\ .\ (r^{\{n-1..m-1\}})$
$der\ c\ (\sim r) \overset{\text{def}}{=} \sim der\ c\ (r)$

Question 4

$$nullable\big(CFUN(f)\big)\ \overset{\text{def}}{=}\ false$$
$$der\ c\ \big(CFUN(f)\big) \overset{\text{def}}{=} if\ \big(f(c)\big)\ then\ ONE\ else\ ZERO$$

Definitions:
In order to implement CFUN, I have added some functions for CHAR, RANGE and ALL to translate their outputs to Booleans.

$c \overset{\text{def}}{=} CFUN(\ (ch) => \ ch == c)$
$[c1, c2, \ldots, cn] \overset{\text{def}}{=}\ CFUN(\ (ch) => if\ ch\ \in [c1, c2 \ldots cn]$ then **true** else **false**)
$ALL\ \overset{\text{def}}{=} CFUN(\ (\_) => \ true)$

Code:

```
case class CFUN(f: Char => Boolean) extends Rexp
/* CFUN translations */
def CHAR(c : Char) = CFUN((ch : Char) => c == ch)
def RANGE(s: Set[Char]) = CFUN((ch : Char) => s.contains(ch))
def ALL = CFUN ((_ : Char) => true)
/* End CFUN translations */
```

Ayan Ahmad
K19002255

Question 5

Email: ayan.ahmad@kcl.ac.uk

Regular Expression without running and simplifications:

```
SEQ(PLUS(RANGE(('a' to 'z').toSet ++ ('0' to '9').toSet + '_' + '.' + '-')),
SEQ(CHAR('@'), SEQ(PLUS(RANGE(('a' to 'z').toSet ++ ('0' to 9').toSet + '.' +
'-')), SEQ(CHAR('.'), BETWEEN(RANGE(('a' to 'z').toSet + '.'),2,6)))))
```

Refer to attached code for implementation and methodology.

Regular Expression after ders and after applying simplifications:

$$r = (([a - z0 - 9. -]^{*} \cdot . \cdot [a - z.]^{\{2..6\}}) + [a - z.]^{\{0..4\}}) + [a - z.]^{\{..1\}}$$

Question 6:

"/**/"	yes
"/*foobar*/"	yes
"/*test*/test*/"	no
"/*test/*test*/"	yes

Question 7:

R1 = (r1+)+

R2 = (r2+)+

<table>
<tr><td>“aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa”</td><td><table><tr><td>R1</td><td>R2</td></tr><tr><td>Yes</td><td>Yes</td></tr></table></td></tr>
<tr><td>“aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa”</td><td><table><tr><td>R1</td><td>R2</td></tr><tr><td>No</td><td>No</td></tr></table></td></tr>
<tr><td>“aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa”</td><td><table><tr><td>R1</td><td>R2</td></tr><tr><td>No</td><td>Yes</td></tr></table></td></tr>
</table>