**Q1 What is the difference between *basic* regular expressions and *extended* regular expressions?**

Basic regular expressions are those defined by the grammar of c, 1, 0, +, … - they present the formal concept of regular expressions.

Extended regular expressions are those found in most modern program languages, with extended features for matching patterns in strings, such as a {n}, a +, etc. Some of these features are simply syntactic sugar over features found in basic regular expressions. Extended expression can build up on basic expression as an expression.

**Q2 What is the language recognised by the regular expressions $(0^{*})^{*}$.**

L((0*)*) = {[]}

Remember * always introduces the empty string

L(0) = {}, but because of * we get L(0**) = {[]}

**Q3 Review the first handout about sets of strings and read the second handout. Assuming the alphabet is the set {*a*, *b*}, decide which of the following equations are true in general for arbitrary languages *A*, *B* and *C*:**

| | |
|---|---|
| $(A \cup B)\ @\ C =^{?} A@C \cup B@C$ | equal -> (A+B)C = A·C +B·C |
| $A^{*} \cup B^{*} =^{?} (A \cup B)*$ | not equal -> left regex doesn't support ABABABABA |
| $A^{*}@A^{*} =^{?} A*$ | equal->A*@A*=A(*+*)=A* |
| $(A \cap B)\ @\ C =^{?} (A@C) \cap (B@C)$ | not equal, try A={[a]},B={[]} and C={[a],[]} |

**Q4 Given the regular expressions $r_1$ = 1 and $r_2$ = 0 and $r_3$ = *a*. How many strings can the regular expressions $r_1^{*}$, $r_2^{*}$ and $r_3^{*}$ each match?**

r1*: empty string

r2*: empty string

r3*: infinite

**Q5 Give regular expressions for (a) decimal numbers and for (b) binary numbers. Hint: Observe that the empty string is not a number. Also observe that leading 0s are normally not written—for example the JSON format for numbers explicitly forbids this. So 007 is not a number according to JSON.**

Decimal: 0 + (1+2+3+4+5+6+7+8+9) · (1+2+3+4+5+6+7+8+9+0)*

Binary: 0 + 1·(1+0)*

**Q6 Decide whether the following two regular expressions are equivalent $(1+a)^* \equiv^? a^*$ and $(a \cdot b)^* \cdot a \equiv^? a \cdot (b \cdot a)^*$.**

$(1+a)^* \equiv^? a^*$ Yes, they are equal. Both match the empty string.

$(a \cdot b)^* \cdot a \equiv^? a \cdot (b \cdot a)^*$ Yes, they are equal.

**Q7 Given the regular expression $r = (a \cdot b + b)^*$. Compute what the derivative of $r$ is with respect to $a$, $b$ and $c$. Is $r$ nullable?**

r is nullable
der a (r) =(b) ·(a·b +b)*; der b(r) = (a·b +b)*; der c (r) = 0

**Q8 Give an argument for why the following holds: if $r$ is nullable then $r^{\{n\}} \equiv r^{\{..n\}}$.**

8/

$[\,] \in L(r)$

$L(r^{\{n\}}) = \bigcup_{0 \leq n} L(r)^n$

$L(r^{\{..n\}}) = L(r)^0 \cup L(r)^1 \cup \ldots \cup L(r)^n$

if $[\,] \in A$, then $A^n \subset A^{n+m}$ for every $0 <= m$

Eg. $A = \{a, b, c, [\,]\}$ from HW1

Therefore, $L(r)^0 \subset L(r)^n, \ldots, L(r)^{n-1} \subset L(r)^n$

$$
\begin{aligned}
L(r^{\{..n\}}) &= L(r)^0 \cup L(r)^1 \cup \ldots \cup L(r)^n \\
&= L(r)^n \\
&= L(r^{\{n\}})
\end{aligned}
$$

$\therefore r^{\{n\}} \equiv r^{\{..n\}}$

---

$A = \{a, b, [\,]\}$

$$
\begin{aligned}
A^2 &= 2^0 + 2^1 + 2^2 \\
&= 1 + 2 + 4 \\
&= 7
\end{aligned}
$$

$\{a, b, [\,]\}$

$= \{a, b, aa, ab, ba, bb, [\,]\}$

**Q9 Define what is meant by the derivative of a regular expression with respect to a character. (Hint: The derivative is defined recursively.)**

$$
\begin{array}{lll}
der\,c\,(\mathbf{0}) & \stackrel{\text{def}}{=} & \mathbf{0} \\
der\,c\,(\mathbf{1}) & \stackrel{\text{def}}{=} & \mathbf{0} \\
der\,c\,(d) & \stackrel{\text{def}}{=} & \text{if } c = d \text{ then } \mathbf{1} \text{ else } \mathbf{0} \\
der\,c\,(r_1 + r_2) & \stackrel{\text{def}}{=} & der\,c\,r_1 + der\,c\,r_2 \\
der\,c\,(r_1 \cdot r_2) & \stackrel{\text{def}}{=} & \text{if } nullable(r_1) \\
 & & \text{then } (der\,c\,r_1) \cdot r_2 + der\,c\,r_2 \\
 & & \text{else } (der\,c\,r_1) \cdot r_2 \\
der\,c\,(r^*) & \stackrel{\text{def}}{=} & (der\,c\,r) \cdot (r^*)
\end{array}
$$

**Q10 Assume the set *Der* is defined as *Der c A* = {*s* | *c* :: *s* □ *A*}. What is the relation between *Der* and the notion of derivative of regular expressions?**

Der c A is for a set of world, and der c (a) is for the regular expressions

L(der c r) = Der c (L(r))

**Q11 Give a regular expression over the alphabet {*a*, *b*} recognising all strings that do not contain any substring *bb* and end in *a*.**

r = (a*ba)* · a + (a*ab)* · a

**Q12 Do (*a*+*b*)□ ·*b*$^+$ and (*a*□ ·*b*$^+$)+(*b*□ ·*b*$^+$)define the same language?**

No. For example, abbab

**Q13 Define the function *zeroable* by recursion over regular expressions.**

$$
\begin{array}{lll}
zeroable(\mathbf{0}) & \stackrel{\text{def}}{=} & true \\
zeroable(\mathbf{1}) & \stackrel{\text{def}}{=} & false \\
zeroable(c) & \stackrel{\text{def}}{=} & false \\
zeroable(r_1 + r_2) & \stackrel{\text{def}}{=} & zeroable(r_1) \wedge zeroable(r_2) \\
zeroable(r_1 \cdot r_2) & \stackrel{\text{def}}{=} & zeroable(r_1) \vee zeroable(r_2) \\
zeroable(r^*) & \stackrel{\text{def}}{=} & true
\end{array}
$$

**Q14 Give a regular expression that can recognise all strings from the language {*a*$^n$ |□*k*.*n*=3*k*+1}.**

r = a · (a^3)*

**Q15 Give a regular expression that can recognise an odd number of *a*s or an even number of *b*s.**

r = a·(aa)* + bb·(bb)*