1. (a) The language of all words over the alphabet \{1, 2, 3\} of length at least one and at most two.

(b) \(L = \{1, 2, 3, 11, 12, 13, 21, 22, 23, 31, 32, 33\}\)

(c) \(|L_1| = \sum_{i=m}^{n} |\Sigma|_i|^n\)

(Note that the “big sigma” here is the standard arithmetic sum operator.)

The above solution yields full marks. However, this is just a geometric series, for which the sum easily can be stated in closed form. See e.g. http://en.wikipedia.org/wiki/Geometric_series, or note the following.

Assuming \(r \neq 1:\)

\[
\left(\sum_{i=m}^{n} r^i\right) + r^{n+1} = r^m + \sum_{i=m+1}^{n+1} r^i = r^m + r \sum_{i=m}^{n} r^i
\]

Thus:

\[
\left(\sum_{i=m}^{n} r^i\right)(1 - r) = r^m - r^{n+1}
\]

giving

\[
\sum_{i=m}^{n} r^i = \frac{r^m - r^{n+1}}{1 - r}
\]

Finally, substituting \(|\Sigma|_1|\) for \(r\), we conclude

\[
|L_1| = \frac{|\Sigma|_1|^m - |\Sigma|_1|^{n+1}}{1 - |\Sigma|_1|}
\]

when \(|\Sigma|_1| \neq 1. If \(|\Sigma|_1| = 1, we have \(|L_1| = n - m + 1.\)

Strictly speaking, we should also note that neither of the above formulations cover the case when \(\Sigma = \emptyset\) and \(m = 0\). This is because 0\(^0\) is undefined. However, 0\(^0\) = \{\(\epsilon\}\), which means \(|L_1| = 1\) (for any \(n \geq 0\).

(d) \(|L_1| = \sum_{i=0}^{4} |\Sigma|_i^i = \sum_{i=0}^{3} 3^i = 1 + 3 + 9 + 27 + 81 = 121\)

or

\(|L_1| = \frac{3^0 - 3^{i+1}}{1 - 3} = \frac{1 - 273}{-2} = 121\)

[Marking: 5 each, for a total of 20]

2. (a) \(L_3 = \{\epsilon, b, cc\} \cap \{a, b, c\} = \{b\}\)

(b) \(L_4 = \{a, b, c\} \{\epsilon, b, cc\} = \{a, b, c, ab, bb, cb, acc, bcc, ccc\}\)

(c) \(L_5 = \emptyset\)

[Marking: 5 each, for a total of 15]

3. (a) DFA A

![DFA Diagram]

DFA A
b)

<table>
<thead>
<tr>
<th>$w$</th>
<th>$w \in L(A)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\epsilon$</td>
<td>yes</td>
</tr>
<tr>
<td>$ababa$</td>
<td>no</td>
</tr>
<tr>
<td>$ababba$</td>
<td>yes</td>
</tr>
<tr>
<td>$aabbaabba$</td>
<td>no</td>
</tr>
</tbody>
</table>

(c) $\hat{\delta}_A(0, bab) = \hat{\delta}_A(\delta_A(0, b), ab)$ def. $\hat{\delta}_A$

$= \hat{\delta}_A(2, ab)$ because $\delta_A(0, b) = 2$

$= \hat{\delta}_A(\delta_A(2, a), b)$ def. $\hat{\delta}_A$

$= \hat{\delta}_A(3, b)$ because $\delta_A(2, a) = 3$

$= \hat{\delta}_A(\delta_A(3, b), \epsilon)$ def. $\hat{\delta}_A$

$= \hat{\delta}_A(1, \epsilon)$ because $\delta_A(3, b) = 1$

$= 1$ def. $\hat{\delta}_A$

(d) $L(A)$ contains all words over $\{a, b\}$ in which the number of $a$’s and the number of $b$’s both are even or both are odd. But that’s the same as saying all the words over $\{a, b\}$ containing an even number of symbols. Which in turn suggests there is a DFA with fewer states that accepts the same language. (Can you find it?)

[Marking: 10 each, for a total of 40]

4. We need to count the number of $a$’s modulo 3, i.e. we need to keep track of whether the remainder when we divide the total number of $a$’s seen so far by 3 is 0, 1, or 2. Thus we need 3 states. They are named 0, 1, and 2 below, to indicate said remainder. When any symbol other than $a$ is read, the machine does not change state as the number of $a$’s seen remain unchanged. 0 should be the accepting state because a remainder of 0 indicates that the number of $a$’s seen is a multiple of 3. Note that 0 is a multiple of 3. Thus the empty string is accepted, and the accepting state is thus also the initial state.

[Marking: 25]