Problem 1.

Give a regular expression, simplified to the best of your abilities, for the language of **all** strings of a's, b's, and c's where a is never immediately followed by b.

Solution

$$((a^+c^+)^*b^*c^*)^*a^*$$

Reasoning
Given any a, it must be followed by another a, a c or be the end of the string. Thus, the first part of the expression $(a^+c^+)*$ covers the case of a string of a's which is not at the end of the string and thus must be followed by a c.
The case of a string of a's terminating the string is covered at the end of the expression.
Note that with the expression $(a^+c^+)^*a^*$ we have the set of strings, which if they are not empty, start in a and only contain a and c.
We simply follow our optional a's (followed by c's) with as many b's and c's as we want.
Note that they may be in any order, as all of our our groups are zero or more inside another group of zero or more.
Further note that it is perfectly possible for a b to be followed by an a, this is fine and does not violate the terms of the problem.
Thus, although c's are required following occurences of a in the string, we can add individual c's in at any time with c^* (group 3 in the examples below).

Examples (note groupings are not necessarily unique)
Subscripts denote which group is used to generate each substring $((a^+c^+)^*_1b^*_2c^*_3)^*a^*_4$.

$$a : a_4$$
$$b : b_2$$
$$ac : (ac)_1$$
$$acacbaaa : (ac)_1(ac)_1b_2(aaa)_4$$
$$cccccbbb : (ccccc)_3(bbb)_2$$
$$acbacb : (ac)_1b_2(ac)_1b_2$$
$$cbacba : c_3b_2(ac)_1b_2a_4$$

Problem 2.

Give a regular expression, simplified to the best of your abilities, for the language of **all** strings of a's, b's, and c's that contain an even number of b's.

Solution

$$(a + c + (b(a + c)^* b))^*$$

Reasoning

If we read strings from this language left to right, any b which we find must be followed by another b separated only be optional a's and c's. That is to say that, we can assume there will be no b's between them, else we would have chosen the closer pair.
Thus the problem can be thought of as the strings $(a + c)^*$, with optional substrings surrounded by b's.

Examples (note groupings are not necessarily unique)
Subscripts denote which group is used to generate each substring $(a_1 + c_1 + (b(a + c)^*_3 b)_2)^*$

$$\begin{aligned}
aaaccc &: (aaa)_1(ccc)_1 \\
baaccaab &: (b(aaccaa)_3 b)_2 \\
bbbbbb &: (bb)_2(bb)_2(bb)_2 \\
aacbbbbabbacac &: (aa)_1(c)_1(bb)_2(bb)_2(a)_1(bb)_2(a)_1(c)_1(a)_1(c)_1
\end{aligned}$$

Problem 3.

Simplify (if possible) the expression $(a+b+c)^*(a+b)^*$, then describe as concisely as you can in English the language it defines.

Solution

$$(a+b+c)^*$$

Note that + takes the union of the left and right, in this case just the characters, thus it behaves like an 'or' operator. By this logic, we know that $(a+b) \subset (a+b+c)$. Thus any string in $(a+b)*$ could instead be created by another application of $(a+b+c)*$.

This expression defines a language of string containing the characters $\{a, b, c\}$ (in any order, combination and quantity).

Problem 4.

Simplify (if possible) the expression $(a+b)^*c^*(a+b)^*$, then describe as concisely as you can in English the language it defines.

Solution

Cannot be simplified.

This expression defines the language containing the characters $\{a, b, c\}$ where all characters c must be neighbors of each other, with strings $\{a, b\}^*$ on either side. Note that the given expression describes exactly that, with no redundancy, and no part can be removed without removing one of those properties.

Problem 5.

Define a DFA, simplified to the best of your abilities, for the language of **all** strings of a's, b's, and c's where a is never immediately followed by b.

Solution

$$M = (\{s_0, s_1, s_2\}, \{a, b, c\}, \delta, s_0, \{s_0, s_1\})$$

Where $\delta : \{s_0, s_1, s_2\} \times \{a, b, c\} \rightarrow \{s_0, s_1, s_)\}$ is defined as follows:

δ	a	b	c
s$_0$	s$_1$	s$_0$	s$_0$
s$_1$	s$_1$	s$_2$	s$_0$
s$_2$	s$_2$	s$_2$	s$_2$

As can be observed from the graph below, there are two final states which are considered valid, the first s_0 (also the initial state), is for any string which does not end in a. The second, s_1, represents that the most recent character input was an a, note that the only arrows pointing to s_1 are for an input of a. A string ending in a is still valid, so s_1 is also a final state. That only leaves s_2 which serves as a trap state marking the string as invalid. This state can only be reached from s_1 by an input of b, thus represents an input of an a immediately followed by b.

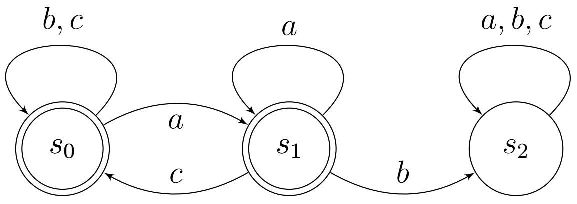

Figure 1

Problem 6.

Define a DFA, simplified to the best of your abilities, that recognizes the language

$$L = \{w \in \{a, b\}^* : |w|_a \bmod 3 = 0\}.$$

Solution

$$M = (\{s_0, s_1, s_2\}, \{a, b\}, \delta, s_0, \{s_0\})$$

Where $\delta : \{s_0, s_1, s_2\} \times \{a, b\} \to \{s_0, s_1, s_)\}$ is defined as follows:

δ	a	b
s$_0$	s$_1$	s$_0$
s$_1$	s$_2$	s$_1$
s$_2$	s$_0$	s$_2$

Note that we needn't concern ourselves with the number of b's we find in our strings. The only determining factor is the number of a's found. Thus we have three states, corresponding to the modulo remainders (respect to 3) for the number of a's. Each a will progress to the next state (effectively counting modulo 3). Any b's found along the way form a self loop, and have no impact on the result. The only valid final state is s_0 which represents $|w|_a \bmod 3 = 0$.

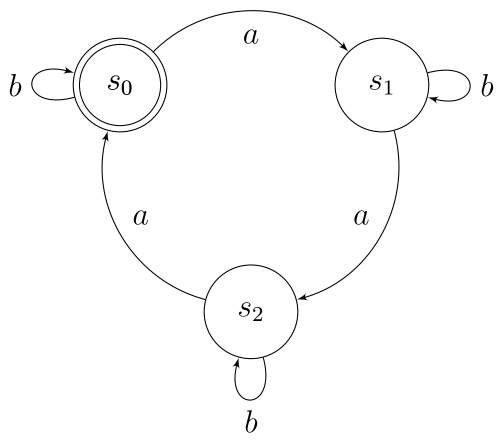

Figure 2