1. (a) 1101001. This is the binary representation of 105, for CSE 105.

   (b) $\{a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z\}$. This list is exhaustive because it is exactly the size of the alphabet. For a string of length one, we just need to pick exactly one character from the alphabet. This means we have $|\Sigma_2|$ choices, each one resulting in a string of length 1.

   (c) $5^3 = 125$ distinct strings of length 3 over Γ. To build a string of length 3, we have 5 choices for each position in the string. There are 3 positions we need to fill out, thus the calculation.

   (d) ε, x, y, z, 0, 1, xx, xy, xz, x0.
   Note: String order is assumed to be by length, then by the specified order.

   (e) $\varnothing$. This is an allowed set of string over each of the alphabets.
   Note: $\{\varepsilon\}$ is also an allowed set.

   (f) Σ_1^* is a valid set of string over Σ_1 and Γ.
   Note: Since $\Sigma_1 \subset \Gamma$, any infinite set of strings over Σ_1 would work.

2. (a) Let $\Sigma = \{a, b\}$. Consider $(\Sigma\Sigma)^*$, $\varepsilon \circ (\Sigma\Sigma)^*$, $\varepsilon \cup ((\Sigma\Sigma) \circ (\Sigma\Sigma)^*)$

   (b) False. Consider the regular expression $\varnothing^*$. This describes the language $\{\varepsilon\}$, which is a finite language.
   Note: The question did not specify whether or not the Kleene star is the outermost operation like in R^*. So, technically, something like $\emptyset \circ R_2^*$ also works

3. (a) There can be no such language. Informally, any string is a substring of itself, so any language A must be a subset of $\textsc{Substring}(A)$.
   More formally: consider this proof by contradiction. Let $A \neq \varnothing$, and assume (towards a contradiction) that $\textsc{Substring}(A) = \varnothing$. Since $A \neq \varnothing$, there exists $w \in A$. Now, consider $a = b = \varepsilon \in \Sigma_1^*$ in the string awb. Since a, b are empty strings, $awb = w$. Thus, $a, b \in \Sigma_1^*$ and $awb = w \in A$. Thus, it is shown that $w \in \textsc{Substring}(A)$, which contradicts the assumption that $\textsc{Substring}(A) = \varnothing$.

   (b) Consider $B = \Sigma^* \backslash \{1\}$, $C = \{\varepsilon, 1\}$. Now, we want to show that $B \circ C = \Sigma^*$. Let $w \in \Sigma^*$, then there are two cases: If $w \neq 1$, then let $u = w \in B, v = \varepsilon \in C$. We have $w = uv \in B \circ C$. Else, $w = 1$, then let $u = \varepsilon \in B, v = 1 \in C$. Again, we have $w = uv \in B \circ C$.
   The above shows $\Sigma^* \subseteq B \circ C$. Since B and C are languages over Σ, their set-wise concatenation is too, i.e. $B \circ C \subseteq \Sigma^*$. Thus $B \circ C = \Sigma^*$

   (c) Consider $L_1 = L_2 = \{1\}$. $L_1 \circ L_2 = \{11\} \neq L_1 = \{1\}$, and $|L_1 \circ L_2| = |L_1| = 1$

4. (a) Counterexample: $\delta(q_3, 0) = q_3$, but the formula states that $\delta(q_3, 0) = q_1$. The correct formula is

$$\delta(q_i, x) = \begin{cases} q_0 & \text{when } i = 0 \text{ and } x = 1 \\ q_3 & \text{when } 1 \leq i \leq 3 \text{ and } x = 1 \\ q_j & \text{when } j = (i+1) \bmod 3 \text{ and } 0 \leq i \leq 2 \text{ and } x = 0 \\ q_3 & \text{when } i = 3 \text{ and } x = 0 \end{cases}$$

   (b) $(1 \cup 000)^*$
   To show this is indeed the language of the automaton. First, notice that (using the definition of regular expressions), language described by the regular expression is the set all strings that have k slots (for some integer $k \geq 0$), each of which is either 1 or 000. This simplifies to the set of all strings whose zeros come in groups of three. To see why this equals the language recognized by the automaton:

   $\Longrightarrow$ Take an arbitrary string is in the language described by the regex, i.e. a series of concatenations of 1 or 000. The computation of the automaton on it starts in q_0, and each time after we read a single 1 or three 0s in a row, we will end up back in q_0. Thus, the computation must end in q_0 so the string will be accepted by the automaton.

   $\Longleftarrow$ On the other hand, if a string is not in the language described by the regex, then this string has zeros that do not come in groups of three. Consider the first such occurrence, we must have a 1 after two 3s or a 1 after one 3. In either case, the computation of the automaton on this string we will transition to q_3 at this point and never reach q_0 again, the only accept state. Thus, the automaton will reject the string.

(c) $F_{new} = \{q_1\}$. Before, ε was accepted, but now, it is not since q_0 is no longer an accept state. Similarly, before, 0 was rejected as the computation of the machine on it ended up in q_1, but now, q_1 is an accept state so we accept.