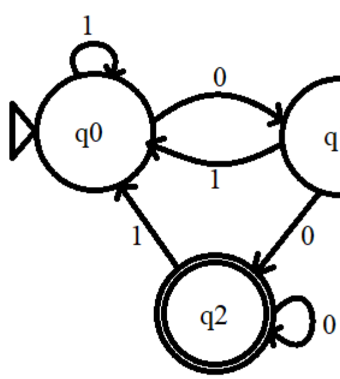
Due: Thursday, September 12

1. **Building NFAs**

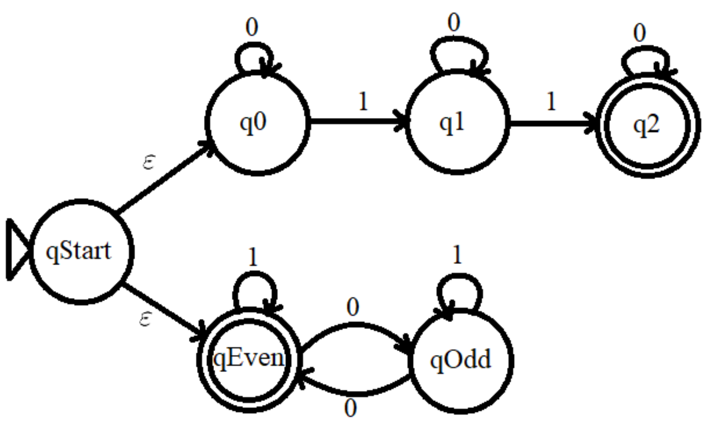
   Draw NFAs for the following languages, with the specified number of states. You may assume the alphabet is always $\{0, 1\}$.

   (a) The language $\{w \mid w \text{ ends in } 00\}$, 3 states.



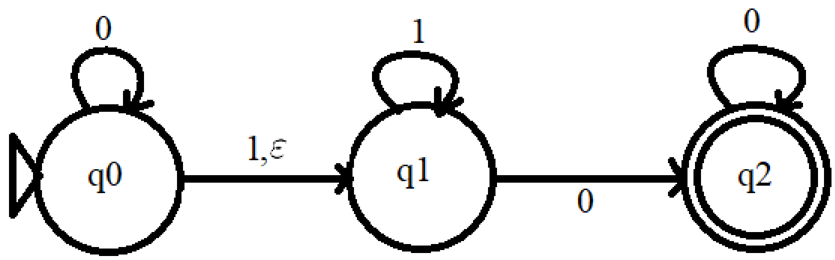
   - q_0: just saw a 1
   - q_1: just saw one 0
   - q_2: just saw two 0s

   (b) The language $\{w \mid w \text{ contains an even number of 0s or exactly two 1s}\}$, 6 states.



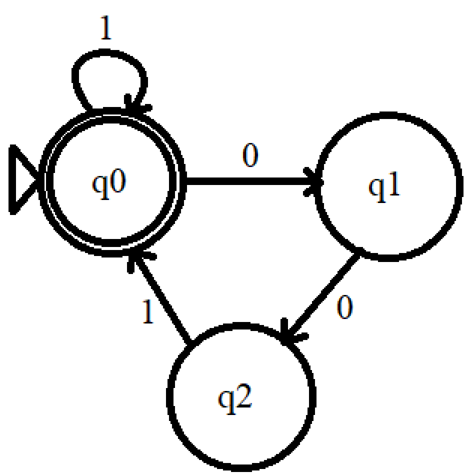
   - $qStart$: start state for union
   - q_0: zero 1s
   - q_1: one 1
   - q_2: two 1s
   - $qEven$: even 0s
   - $qOdd$: odd 0s

(c) The language $0^*1^*0^+$, 3 states.



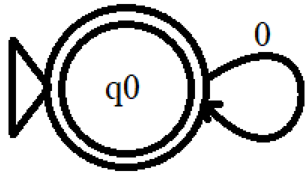
- q_0: 0^* (not accepting because concatenated)
- q_1: 1^* (not accepting because concatenated)
- q_2: 0^+

(d) The language $1^*(001^+)^*$, 3 states.



- q_0: 1^* (also represents 1^+ when reached from q_2)
- q_1: 0
- q_2: 0

(e) The language 0^*, 1 state.

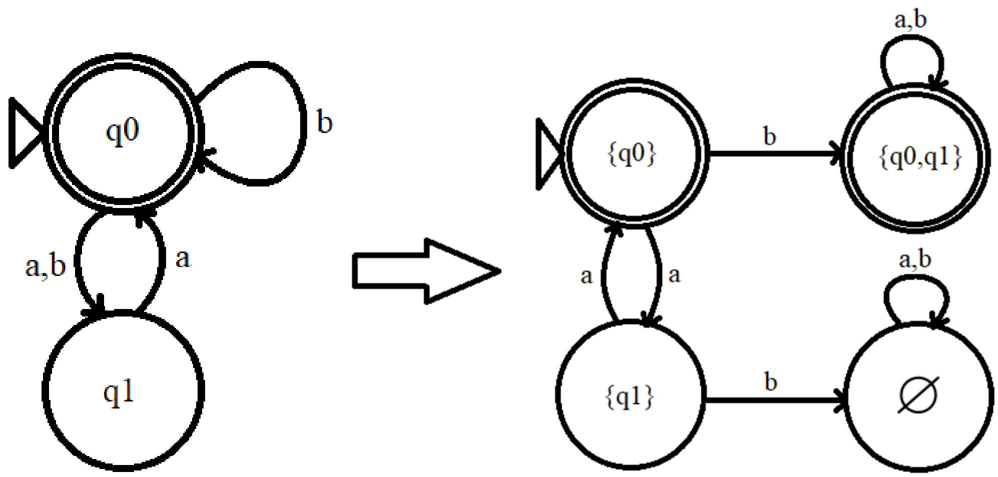


- q_0: 0^*

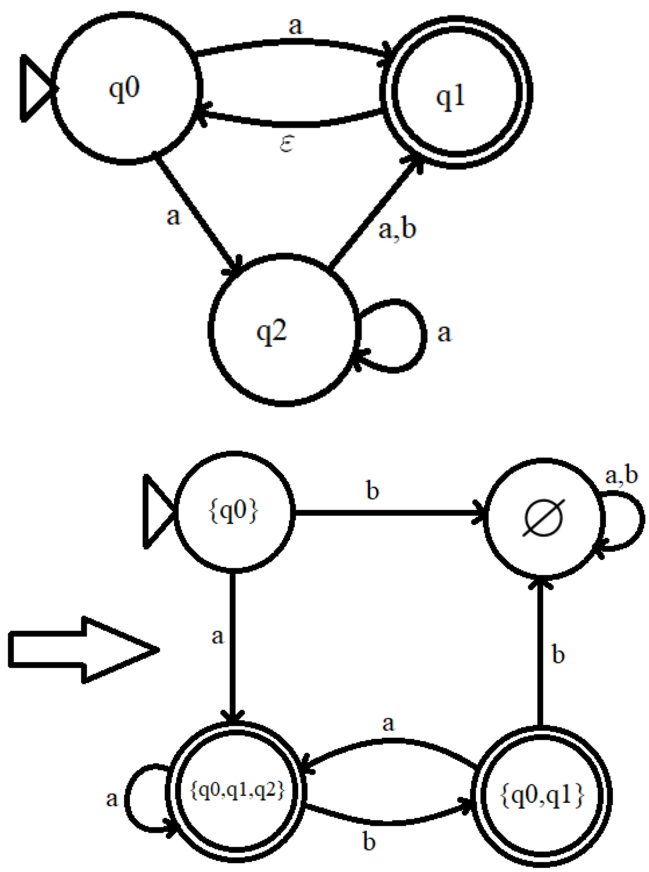
2. **Subset Construction**

Turn the following NFAs into DFAs using the subset construction. It should be clear from your drawing of a DFA how the subset construction was applied.

(a)



(b)



Note that there are four unreachable states.

3. **Reversing a Language and Building a Binary Adder**

If A is a language, we define its reverse, $\mathit{reverse}(A)$, as the language of all strings in A written in reverse, i.e. $\mathit{reverse}(A) = \{w^{\mathcal{R}} \mid w \in A\}$. For example, if the string '00110' is in language A, then the string '01100' is in $\mathit{reverse}(A)$.

(a) Prove that the set of regular languages are closed under the reverse operation, i.e., prove that if language A is regular, then $\mathit{reverse}(A)$ is also regular. (Hint: if A is regular, then there is an NFA M_1 that recognizes it. How would you change that NFA into a new machine M_2 that recognizes the reverse?)

Take the machine M_1 that recognizes language A. Make its start state q_0 into an accepting state, and its accepting states F into non-accepting states. Add a new starting state $qStart$ and connect this state with ε transitions to the formerly accepting states in F. Finally, reverse the direction of all of the remaining transitions in the machine. This new machine will start in $qStart$ and only accept a string that can follow a path from a state in F to the state q_0, i.e. a string that originally would have followed a path from q_0 to an accepting state in F.

(b) Define the alphabet

$$\Sigma_3 := \left\{ \begin{bmatrix}0\\0\\0\end{bmatrix}, \begin{bmatrix}0\\0\\1\end{bmatrix}, \begin{bmatrix}0\\1\\0\end{bmatrix}, \begin{bmatrix}1\\0\\0\end{bmatrix}, \begin{bmatrix}0\\1\\1\end{bmatrix}, \begin{bmatrix}1\\0\\1\end{bmatrix}, \begin{bmatrix}1\\1\\0\end{bmatrix}, \begin{bmatrix}1\\1\\1\end{bmatrix} \right\}.$$

The alphabet Σ_3 contains eight size-3 columns of 0s and 1s. A string of symbols in Σ_3 thus builds threes rows of 0s and 1s. Consider each row to be a binary number and define the language

$$B = \{w \in \Sigma_3 \,|\, \text{the bottom row of } w \text{ is the sum of the top two rows}\}.$$

For example,

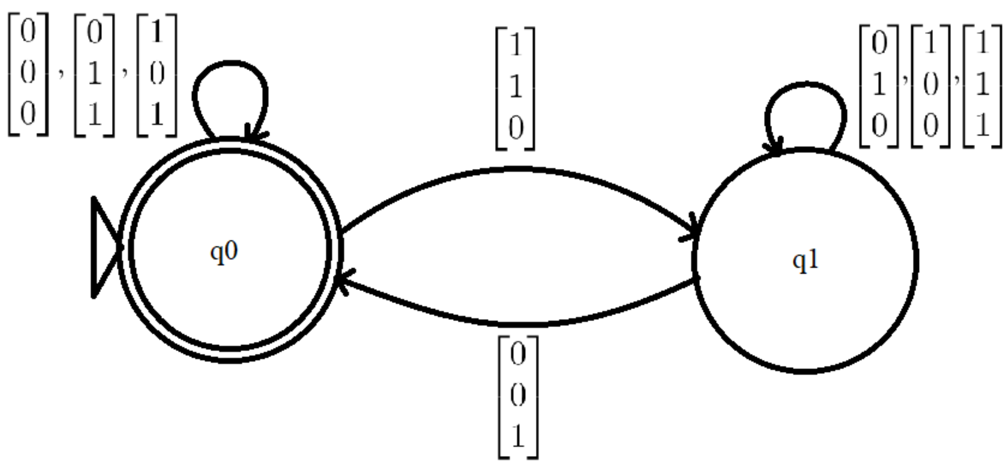
$$011 + 001 = 100 \quad \text{but} \quad 01 + 00 \neq 11,$$

so

$$\begin{bmatrix}0\\0\\1\end{bmatrix}\begin{bmatrix}1\\0\\0\end{bmatrix}\begin{bmatrix}1\\1\\0\end{bmatrix} \in B \quad \text{but} \quad \begin{bmatrix}0\\0\\1\end{bmatrix}\begin{bmatrix}1\\0\\1\end{bmatrix} \notin B.$$

Show that B is regular. (Hint: show that *reverse*(B) is regular, and by the previous problem conclude that *reverse*(*reverse*$(B)) = B$ is also regular.)

We will create a machine that recognizes *reverse*(B), which will prove that *reverse*(B) is a regular language. Then by problem 3(a), we know that B must also be regular. We consider *reverse*(B) so that we see the 1's place first. Our machine will have two states: q_0 is the 'carry 0' state, and q_1 is the 'carry 1' state. We get:



4. **Intersection and Complement**

(Hint: DFAs are easier to use for the following proofs.)

(a) Prove that the set of regular languages is closed under the complement operation, i.e., prove that if language A is regular, then its complement $\bar{A}$ is also regular.

If language A is regular, then it has a **DFA** M_1 that recognizes it. Take this machine and swap its accepting and non-accepting states. So, any state that was accepting in M_1 will no longer be an accepting state, and any state that was not accepting in M_1 will now be accepting. This new machine now recognizes the language $\bar{A}$.

We can see this because we are dealing with DFAs, and so the path followed on a given input is unique. So, if M_1 accepted a string w_1, the new machine will end in the same state, but reject since that state is no longer accepting. A similar thing happens for a string that would originally have been rejected by M_1. Since we have built a DFA that recognizes $\bar{A}$, we have proven that $\bar{A}$ is regular.

(b) Prove that the set of regular languages is closed under the intersection operation, i.e., prove that if languages A and B are regular, then $A \cap B$ is also regular.

Proof #1: Since A and B are both regular languages, they are recognized by some DFAs M_1 and M_2 respectively. We will now build a new machine M_3 that recognizes their intersection. The process is as follows (note that this construction is nearly identical to the union DFA built in class, only differing in the last step where we define the accepting states of this new intersection DFA):

- The states of M_3 will represent ordered pairs of the states of M_1 and M_2.
- The alphabet for machine M_3 will be the combined alphabets of machines M_1 and M_2.
- The transitions of machine M_3 will directly simulate machines M_1 and M_2 simultaneously. In other words, there will be a transition from state (q_i, q_n) to state (q_j, q_m) on symbol a in machine M_3 if and only if there was a transition from q_i to q_j on symbol a in machine M_1 and there was a transition from q_n to q_m on symbol a in machine M_2.
- The starting state of machine M_3 will be the one corresponding to the ordered pair of the starting states of M_1 and M_2 together.
- The accepting states of machine M_3 will be the set of the ordered pairs where both states in the pair were accepting states of their respective machine. In other words, state (q_i, q_j) will be accepting in machine M_3 if and only if q_i was an accepting state of machine M_1 and q_j was an accepting state of machine M_2. (Note that the union machine accepted any pair where at least one of the states in the pair was accepting. This intersection machine only accepts if both of the states in the pair are accepting.)

Proof #2: The closure of regular languages under the intersection operation can also be proved using De Morgan's laws. In particular, recall that the intersection of two sets can be written as

$$A \cap B = \overline{(\bar{A} \cap \bar{B})}.$$

Note that in part (a) of this question, we proved that regular languages are closed under the complement operation, and in class we proved that they are closed under the union operation. So, if A and B are both regular languages, then so are $\bar{A}$ and $\bar{B}$. This means that the union of these two, $\bar{A} \cap \bar{B}$ is also regular. The complement of this regular language, $\overline{(\bar{A} \cap \bar{B})}$, is still regular. Hence the intersection of A and B must be regular.