Constructing Deterministic Finite Automata (DFA) – Test Yourself

If you're starting with a regular expression (RE), you should first construct the NFA. It's much easier to do this intermediate step than to go straight to the DFA from the RE. (Remember to learn the six rules for constructing a NFA.)

Now, to construct the DFA, there are two functions you need to know:

1. The **ε-closure** function takes a state and returns the set of states reachable from it based on (one or more) ε-transitions. Note that this will always include the state itself. We should be able to get from a state to any state in its ε-closure without consuming any input.

2. The function **move** takes a state and a character, and returns the set of states reachable by <u>one</u> transition on this character.

Both of these functions can be generalized to apply to sets of states by taking the union of the application to individual states, e.g. if A, B and C are states, move({A,B,C},'a') = move(A,'a') U move(B,'a') U move(C,'a').

Now, to construct the DFA, build up a table. The first column in your table should be labeled "DFA states", then one column for each symbol in your input alphabet. The first cell in your table will have the ε-closure of the start state of the NFA. This new merged state will also be the start state of your DFA. The other cells in that row will have the results of **move** from that merged state for each symbol, e.g.:

DFA states	**symbol$_1$**	**symbol$_2$**	**…**	**symbol$_n$**
ε-closure (NFA start state)	**move** (**ε-closure** (NFA start state), symbol$_1$)	**move** (**ε-closure** (NFA start state), symbol$_2$)	…	**move** (**ε-closure** (NFA start state), symbol$_n$)

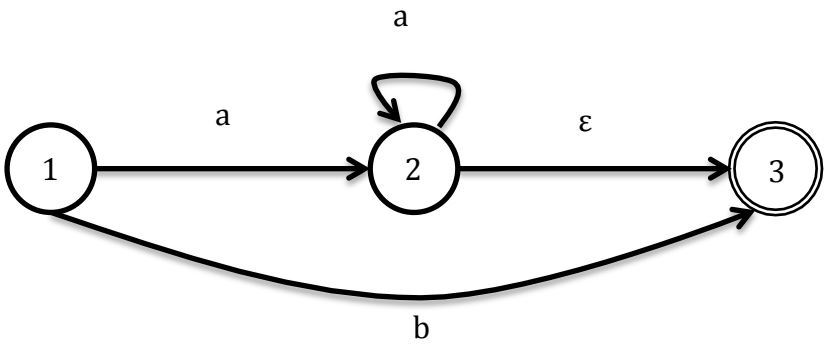
Each of the results of the **move** functions is a new combined state that becomes part of the DFA… so for each new state, add a line to this table, and perform the **move** function on this new state for each input symbol. Keep repeating this process until *all* the states that have been created by the **move** functions have a corresponding row in the table – this makes the table complete. It is then a straightforward exercise to translate the table into a diagram – remember to clearly label:

1. The start state
2. The states themselves
3. The transitions
4. The accepting states

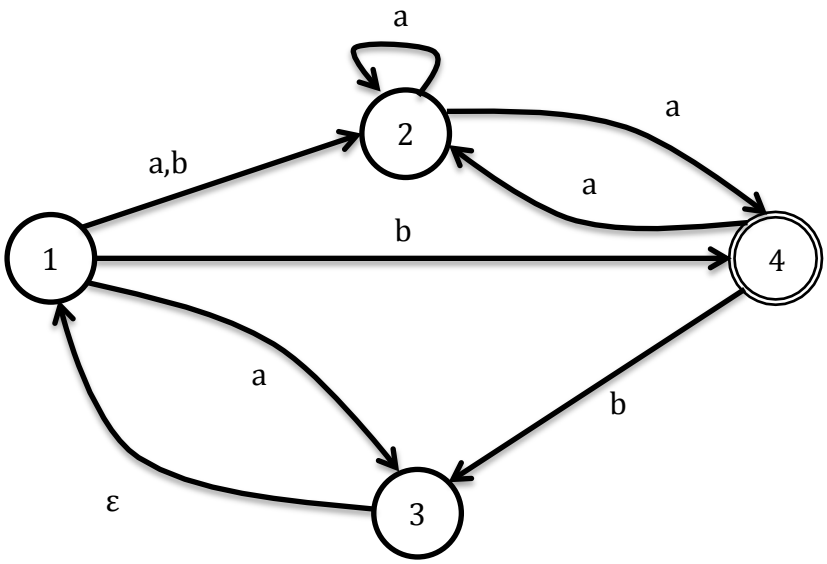
Note that the accepting states are *any* states that include an accepting state from the NFA.

You should be able to construct DFAs for any of the NFAs constructed on the *Test Yourself: NFAs* sheet. In addition, try these:

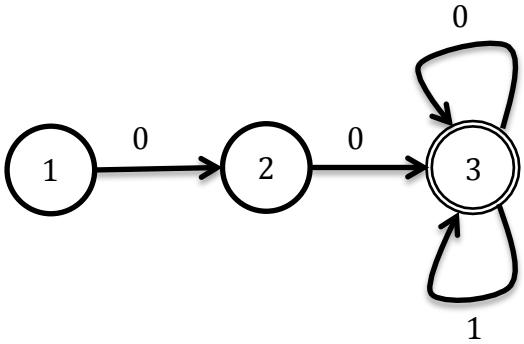
1.



2.



3.



In addition to creating the DFAs for these NFAs:

1. Identify the features in each that mean that these are *not* DFAs
2. What regular expressions do each of these NFAs represent?