CSCE 355, Assignment 3

1. For the ϵ-NFA of textbook Exercise 2.5.2,

	ϵ	a	b	c
$\rightarrow p$	$\{q, r\}$	$\emptyset$	$\{q\}$	$\{r\}$
q	$\emptyset$	$\{p\}$	$\{r\}$	$\{p, q\}$
$*r$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

find an equivalent NFA (without ϵ-moves) using the method explained in class. This is also Method 2 described in the COURSE NOTES, Section 10.4.

2. Do Exercise 2.5.3(a): Design an ϵ-NFA for the following language: the set of all strings consisting of zero or more `a`'s followed by zero or more `b`'s, followed by zero or more `c`'s. Try to use ϵ-transitions to simplify your design.

3. Do Problem 2.3 [here copied verbatim from pp. 81–82 of the textbook with minor corrections and annotations; this illustrates a proof by string induction and gets you thinking about citing the reasons for various proof steps]: Here is the transition function of a simple, deterministic automaton with start state A and accepting state B:

	0	**1**
A	A	B
B	B	A

We want to show that this automaton accepts exactly those strings with an odd number of 1's, or more formally:

$$\delta(A, w) = B \text{ if and only if } w \text{ has an odd number of 1's.} \tag{1}$$

Here, δ is the extended transition function of the automaton; that is, $\delta(A, w)$ is the state that the automaton is in after processing input string w. The proof of the statement above is an indunction on the length of w. Below, we give the proof with reasons missing. You must give a reason for each step, and then demonstrate your understanding of the proof by classifying your reasons into the following three categories:

A) Use of the inductive hypothesis.

B) Reasoning about properties of deterministic finite automata, e.g., that if string $s = yz$, then $\delta(q, s) = \delta(\delta(q, y), z)$.

C) Reasoning about properties of binary strings (strings of 0's and 1's), e.g., that every string is longer than any of its proper substrings.

Basis ($|w| = 0$):

1. $w = \epsilon$ because:
2. $\delta(A, \epsilon) = A$ because:
3. ϵ has an even number of 0's because:

Induction ($|w| = n > 0$):

4. There are two cases: (a) when $w = x1$ and (b) when $w = x0$ because:

Case (a):

5. In case (a), w has an odd number of 1's if and only if x has an even number of 1's because:
6. In case (a), $\delta(A, x) = A$ if and only if w has an odd number of 1's because:
7. In case (a), $\delta(A, w) = B$ if and only if w has an odd number of 1's because:

Case (b):

8. In case (b), w has an odd number of 1's if and only if x has an odd number of 1's because:
9. In case (b), $\delta(A, x) = B$ if and only if w has an odd number of 1's because:
10. In case (b), $\delta(A, w) = B$ if and only if w has an odd number of 1's because:

4. (a) Show that every regular language is recognized by an ϵ-NFA where out of each state there is *no more than one* ϵ-transition and *no more than one* non-ϵ-transition (i.e., a transition on a symbol from the alphabet).

   (b) Show that every regular language is recognized by an ϵ-NFA where out of each state there is *exactly one* ϵ-transition and *exactly one* non-ϵ-transition (i.e., a transition on a symbol from the alphabet). (A solution to this part is obviously also a solution to the previous part.)

5. Do Exercise 3.1.1(b,c): Write regexes for the following languages:

   b) The set of strings of 0's and 1's whose tenth symbol from the right end is 1.

   c) The set of strings of 0's and 1's with at most one pair of consecutive 1's.

6. (Optional) Do the following textbook exercises:

   Exercise 3.1.2(b): Write regular expressions for the following languages:

   b) The set of strings of 0's and 1's whose number of 0's is divisible by five.

   Exercise 3.1.3(a,b,c): Write regular expressions for the following languages:

   a) The set of all strings of 0's and 1's not containing 101 as a substring.

   b) The set of all strings with an equal number of 0's and 1's, such that no prefix has two more 0's than 1's, nor two more 1's than 0's.

   c) The set of strings of 0's and 1's whose number of 0's is divisible by five and whose number of 1's is even.

7. Write a regular expressions for the following languages over $\{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$:

   (a) The set of strings where no `a` appears after any `b` or `c`.

   (b) The set of strings where no `b` occurs anywhere after an `a` (not just continguously). This language is the complement of that given by the regex

$$(\mathtt{a}+\mathtt{b}+\mathtt{c})^*\mathtt{a}(\mathtt{a}+\mathtt{b}+\mathtt{c})^*\mathtt{b}(\mathtt{a}+\mathtt{b}+\mathtt{c})^* \ .$$

8. Do Exercise 3.2.3: Convert the following DFA to a regular expression, using the state-elimination technique of Section 3.2.2.

	0	1
$\rightarrow *p$	s	p
q	p	s
r	r	q
s	q	r

9. Do Exercise 3.2.4(c): Convert the following regex to an ϵ-NFA: $\mathbf{00}(\mathbf{0}+\mathbf{1})^*$.

10. Recall the DFA D in item 3 (Problem 2.3 of the textbook) that accepts a binary string iff it has an odd number of 1's:

	0	1
$\rightarrow A$	A	B
$*B$	B	A

   (a) Convert D into an equivalent clean ϵ-NFA using the clean-up procedure in class (add a new start state, a new final state, and some ϵ-transitions).

   (b) Use the state elimination method to convert D to a regular expression. Eliminate state A first, then B.

11. Same exercise as 10 above, except make A the final state (so that D accepts a string iff it has an *even* number of 1's).

12. (Optional) Recall the product DFA P that counts an even number of zeros and an odd number of ones:

	0	1
$\rightarrow EE$	OE	EO
OE	EE	OO
$*EO$	OO	EE
OO	EO	OE

   Use the state elimination method to convert P to a regular expression. (To control the complexity, you may wish to define names for intermediate regexes.)

13. Draw the transition diagram of an ϵ-NFA equivalent to the regex $(\mathtt{a}+\mathtt{bc})^*\mathtt{aa}$. You may (but are not required to) contract ϵ-transitions provided it is safe to do so.