ECE 468

Problem Set 1: Regular expressions and finite automata

1. With football season upon us, your boss wants you to write a program that scrapes sports websites for information about football players. Write a regular expression that captures information about a football player, in the following format:

   <First name> <Last name>, <Height>, <Weight>, <Position>

   Where first and last names start with capital letters and are followed by 0 or more other letters, height is a numeric height in feet, followed by an apostrophe, then a numeric height in inches, followed by a quotation mark, weight is a number followed by either "lb" or "kg" and position is one of the following abbreviations: QB, OL, RB, TE, WR, DL, LB, S or CB.

   Answer: There are many possible regexes that could match the given description. Here's one:

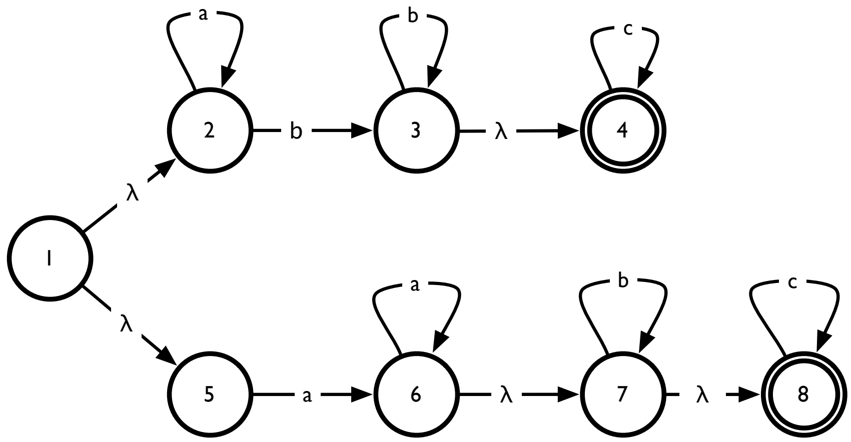
   $[A\text{--}Z][a\text{--}z]*\ [A\text{--}Z][a\text{--}z]*,\ [0\text{--}9]'[1\text{--}2][0\text{--}9]'',\ [0\text{--}9]+(lb|kg),\ (QB|OL|RB|TE|WR|DL|LB|S|CB)$

   Note that this regex allows for heights like $5'18''$, which are clearly not possible. A better (but longer) regex, would only allow the inches component of height to be 1–12.

2. Give a *non-deterministic* finite automaton that matches the following regular expression:

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

   Answer: There are many possible NFAs that could capture this regex, but here's one.



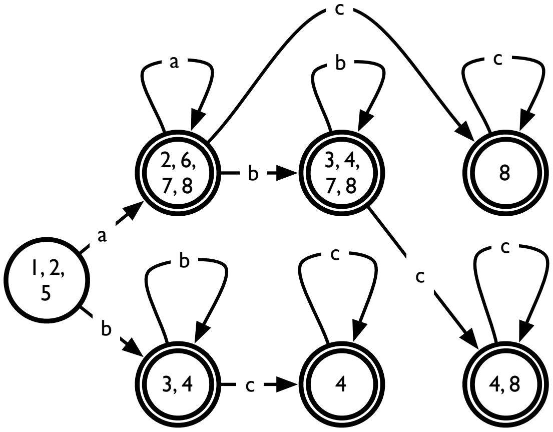
3. Give a *deterministic* version of the finite automaton, using the construction we described in class. You only need to show the state transition diagram.

   Answer:

   This is the state transition table:

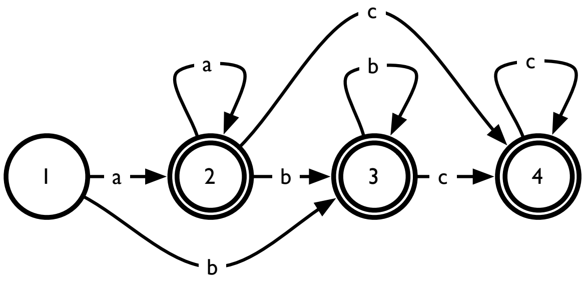
State(s)	a	b	c	Final?
1, 2, 5	2, 6, 7, 8	3, 4	Err	No
2, 6, 7, 8	2, 6, 7, 8	3, 4, 7, 8	8	Yes
3, 4	Err	3, 4	4	Yes
3, 4, 7, 8	Err	3, 4, 7, 8	4, 8	Yes
8	Err	Err	8	Yes
4	Err	Err	4	Yes
4, 8	Err	Err	4, 8	Yes

   This is what the graphical DFA looks like:



4. Derive the reduced DFA. Show both the graphical representation of the automaton and the state transition diagram.

   Answer: We start by putting all the final states together into a single final state. We then see that the two states {3, 4} and {3, 4, 7, 8} behave differently from the other four final states. The latter four loop back to the merged final state on a ‘c’, while the former two go to error. We can then split {2, 6, 7, 8} apart from that group of four because it does not go to error on an ‘a’. At this point, we cannot split apart any other states, so we are done. The resulting DFA has four states, instead of seven (note the new state numbering):



And the transition table is:

State(s)	a	b	c	Final?
1	2	3	Err	No
2	2	3	4	Yes
3	Err	3	4	Yes
4	Err	Err	4	Yes