Exam rules. The exam is open book and open notes—you can use any printed or handwritten material. However, no electronic devices are allowed. Anything with an on-off switch must be turned off. There is one exception: you can use a music player as long as nobody else can hear it and you just put it on shuffle and do not use its controls during the exam.

About the problems. In the following problems, a machine is “equivalent” to a regular expression if the machine accepts any word matching the regular expression and vice-versa, any word matching the regular expression is accepted by the machine. When you are asked for a machine you should give it by means of a diagram using circles for the states and arrows for the transitions. When you are asked for a regular expression, please give an expression built up from the basic operations shown in $(ab)^*(a|b)$. You may use abbreviations such as $[a−z]$ or $[0−9]$ (or any other interval of characters), but NOT the more complex abbreviations used in programming languages that allow regular expressions.

When you are asked for an FA, it is not required that there be a transition out of every state on every symbol; in other words, the convention of the textbook and JFLAP about the implicit presence of a “dead state” will be accepted.

Always assume that the alphabet consists of all the characters mentioned in the problem and no more.

Scoring. There are four problems, but the last one is easier than the first three. The first three are worth 30 points each and the fourth one is worth 10 points. You have 75 minutes so there should be plenty of time to check your work carefully. Please put your final answer on the exam sheet. Although it is almost impossible to carry out all the steps of these algorithms without errors, it’s possible to check the work and correct the errors, so that the final answer is correct. You may attach your work in the hope of partial credit, but if you do so, please staple it to the exam sheet and avoid loose pages or dog-ear attachments. (Hopefully I will provide a stapler.)
1. Find an NFA equivalent to the following regular expression: \((ab)^*(c(aa|bb))^*\).

Please remove \(\epsilon\) transitions so the answer has a reasonable size (such as six states); in other words, you are not required to present the result of a systematic algorithm (which introduces a lot of \(\epsilon\)-transitions). However, it is not required to remove those transitions—if you would rather present a large machine resulting from the algorithm, go ahead.

Answer:

![NFA Diagram]

\(q_0\) \(\rightarrow\) \(q_1\) \(\rightarrow\) \(q_2\) \(\rightarrow\) \(q_3\) \(\rightarrow\) \(q_4\) \(\rightarrow\) \(q_5\)
2. Find a regular expression equivalent to the following finite automaton. Please delete superfluous $\epsilon$'s.

Step 1:

Step 2:

Step 3 (using + instead of | in the diagram)

leading to the final answer $(aa|bb)(bb|ca)^*$
3. Find an FA equivalent to the following NFA:

![Original NFA Diagram]

**Answer:**

![Equivalent FA Diagram]
4. In the minimization algorithm, the middle part of the algorithm is to mark the inequivalent states. Is it always true that any two favorable (accepting) states are equivalent? If you say “yes”, explain your answer. If you say “no”, then give an example of a finite automaton in which there are two inequivalent favorable states.

Answer: No. All we need to do is arrange that there are two final states, and starting from one of them we can accept a, but starting from the other one we cannot accept a. For example: