Name:____________________________

Theory of Computation – Dr. Weiss
Extra Practice Exam Solutions

Directions:
Answer the questions as well as you can. Partial credit will be given, so show your work where appropriate. Try to be precise in your answers in order to maximize your points. Also make sure that your answers to pumping lemma questions are sufficiently clear so that I can tell that your reasoning is correct. Good luck.

Note: DFA = Deterministic Finite Automata, NFA = Nondeterministic Finite Automata
PDA = Push-Down Automata CFG=Context Free Grammar

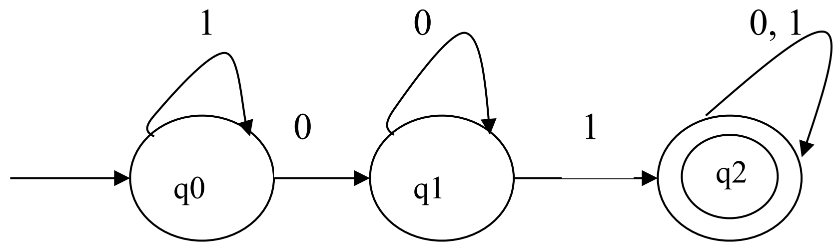
Here is the pumping lemma for regular languages:

If A is a regular language, then there is a number p (the pumping length) where, if s is any string in A of length at least p, then s may be divided into 3 pieces, $s = xyz$, satisfying the following conditions:

1. For each $i \geq 0$, $xy^iz \in$ A,
2. $|y| > 0$, and
3. $|xy| \leq p$

Name:______________________________

1. (17 points) Let M be the Deterministic Finite Automata (DFA) shown below



   a. Provide a <u>formal</u> description of M below (7 points)

      M = ({q0, q1, q2}, {0, 1}, δ, q0, {q2}) or alternatively

      Q = {q0, q1, q2}
      Σ = {0, 1}
      q0 = q0
      F = {q2}

      In either case, you need to specify δ as:

	0	1
q0	q1	q0
q1	q1	q2
q2	q2	q2

   b. In plain English, describe the language described by this DFA. To get full credit you need to provide reasonably succinct description that ignores irrelevant considerations. (3 points)

      The DFA shown above describes a language that <u>contains "01" as a substring</u>. Also give full credit for <u>contains one or more 0's followed by at least a single 1</u>.

   c. What is the regular expression that is equivalent to the DFA above (i.e., expresses the same language)? (4 points)

      $1^*00^*1\Sigma^*$ or $1^*0^+1(0\cup1)^*$ or $\Sigma^*01\Sigma^*$ (note that $\Sigma^*0\Sigma^*1\Sigma^*$ is not wrong, but not concise)

   d. (1 pt) The language described by M is a regular language (circle one): $\boxed{\text{True}}$ False

   e. Briefly justify your answers (2 points)
      A language generated by a DFA is by definition a regular language..

Name:______________________________

2. Actual Factuals (24 points) . Answer the following True/False and fill-in-the-blank questions. If you do not recall the answers, feel free to prove them for yourself (often not hard), but for this question I do not need to see any work. Note that a language is closed under an operation (e.g., union) if when you apply that operator to things in that language you get back something in that language.

a. A DFA is equivalent in expressive power to an NFA . $\boxed{\text{True}}$ False

b. Regular languages are closed under complement. $\boxed{\text{True}}$ False

c. Regular languages are closed under union. $\boxed{\text{True}}$ False

d. An NFA accepts a string w only if when w is presented to the NFA and all paths terminate in an accept state. True $\boxed{\text{False}}$

e. A DFA and NFA accept a string as soon as they enter an accept state. True $\boxed{\text{False}}$

f. You cannot build a DFA to recognize $\{0^{500}1^{40000} \cup 1^{1000}0^{200}\}$ True $\boxed{\text{False}}$

g. You cannot build an NFA to recognize the language 0^n1^n for $n \geq 0$. $\boxed{\text{True}}$ False

h. If an NFA has *n* states then you can always construct an equivalent DFA using at most 2*n* states. True $\boxed{\text{False}}$

i. The pumping lemma can be used to show that a language is regular. True $\boxed{\text{False}}$

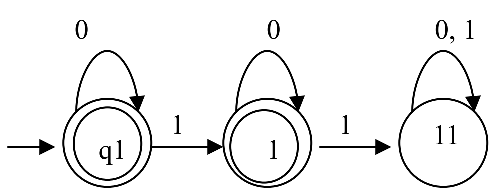
j. The minimum pumping length for the language {1010} is: ___5_______

k. The minimum pumping length for the language {1*} is: ___1_______

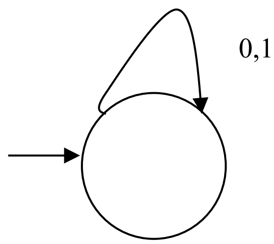
l. The minimum pumping length for the language {11*} is: ____0______

m. PDA's include the power of non-determinism $\boxed{\text{True}}$ False

3. Given ∑ = {0,1}, draw the DFA that recognizes the language {w|w does not contain more than one "1"}. (7 points)



4. Given ∑ = {0,1}, draw the DFA that recognizes the language = the empty set. (4 points)

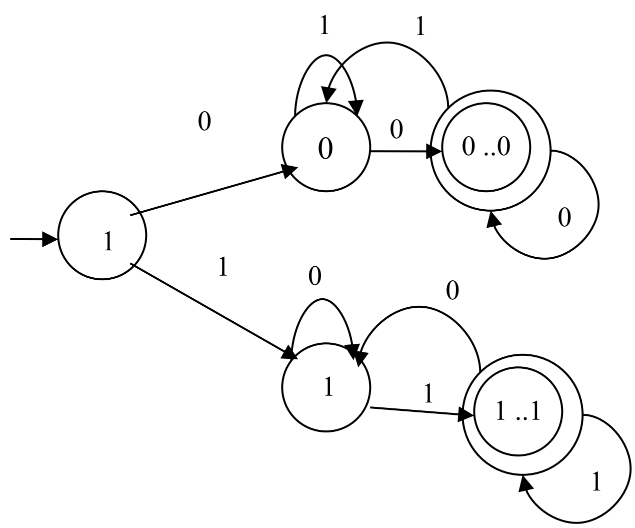

0,1

Name:____________________________

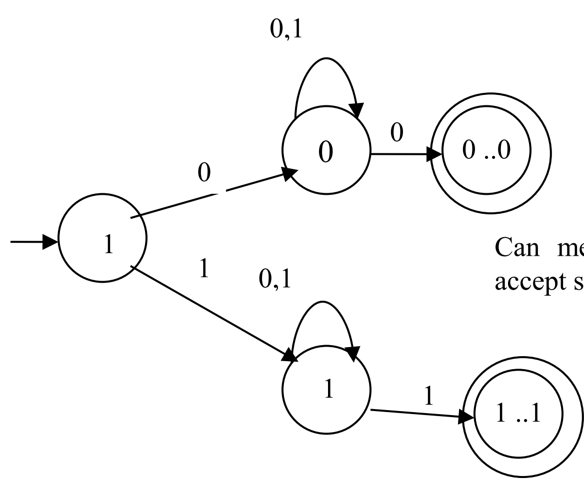
5. (18 points) For this problem you are given:
   $\Sigma = \{0, 1\}$
   L = {w| w begins and ends with the same symbol with total length at least 2}.

   Note: we make minimum length 2 so you do not have to cope with handling "0" or "1" since both do begin and end with the same symbol.

   a. Neatly draw the DFA that recognizes this language (8 points)



   b. Now draw the NFA that recognizes the same language and makes the most use of nondeterminism. I can check this objectively by checking the number of states and transitions (arcs). You should especially focus on minimizing number of transitions (arcs). You will lose points if you have extra ones. (6 points)



Can merge these two accept states

   c. Now write out the regular expression that expresses this language (4 points)

$0\Sigma^*0 \cup 1\Sigma^*1$

6. (22 points) Let L be the language of <u>all</u> binary strings with an equal number of 0's and 1's. Thus some of the elements that L would contain are: 01, 0011, 0101, 000111, 001011, etc.

   a. Is L regular? Circle one: Regular $\boxed{\text{Not Regular}}$ (1 points)
   I did this one for you since otherwise you would get the rest wrong! You even get a point.

   b. In the space below, informally explain why the language is not regular, without using any direct reference to the pumping lemma. If you recall, we provided such informal explanations in class for other non-regular languages. However, your reasoning should be clear. Hint: consider the properties of a finite automata (4 points)

   L is not regular because you could have all of the 0's before the 1's in which case you would need to count an unbounded number of 0's and you cannot do that with finite memory.

   c. Now <u>prove</u> your answer using the pumping lemma for regular languages, which is stated on the first sheet. Be clear. *I promise you this is not a hard problem and you have in fact seen the solution before.* (12 points)

   By the pumping lemma, there is a pumping length p where we can construct a string s that obeys the 3 conditions listed in the lemma. Let the string be 0^p1^p. By condition 3, y may only contain 0's. Thus, when we pump up to get xyyz, there will be more 0's than 1's. Thus, the language is not regular.

   d. The pumping lemma is a proof by contradiction and the trick is to find a "hard" string in the language that cannot be pumped. In this part I would like you to find a string in this language that <u>can</u> be pumped and then show exactly how it can be pumped (i.e., specify the values of x, y, and z that allow it to be pumped). (5 points)

   By the pumping lemma, there is a pumping length p where we can construct a string s that obeys the 3 conditions listed in the lemma. Let the string $s = (01)^p$ which therefore has a length 2p which is greater than p. We can break s up into x, y, and z as follows: $x = 01$, $y=01$, $z=01^{p-2}$ .Then xyyz is in L and so is any string that we pump. Thus, we have not proven that L is not regular (but we have not proven that it is regular either and hence we just answered one of the T/F questions).

Name:______________________________

7. Provide a Context Free grammar that generates the language 00*1*.

   S → AB

   A → 0A|0

   B → 1B|ε

8. Provide a context free grammar that generates L = {anbm: n ≠m}

   $S \rightarrow AS_1|S_1B$

   $S1 \rightarrow aS_1b \mid \varepsilon$

   $A \rightarrow aA|a$

   $B \rightarrow bB|b$

9. (Describe a PDA that accepts the following language: $L = \{0^m1^n: n \leq m\}$

   Start in a state that will take any leading 0's and push them onto the stack.

   When you see a 1, transition to a new state and start popping a 0 from the stack for each 1 that is seen. If any subsequent 0's are seen, immediately reject. If you run out of 0's to pop from the stack, then reject (this means n > m). When you complete processing the input, accept (you would have rejected before this point if the input did not belong to the language).

10. (8 points) A binary string has odd parity if the number of 1's is odd. Generate a DFA that accepts a string only if it has odd parity.