CS 3350 Automata, Computability, and Formal Languages
Fall 2019, Test 1

Last 4 digits of your UTEP ID number: ________________________

General comments:

• you are allowed up to 5 pages of handwritten notes;
• if you need extra pages, place the last 4 digits of ID number on each extra page;
• the main goal of most questions is to show that you know the corresponding algorithms; so, if
  you are running of time, just follow the few first steps of the corresponding algorithm;
• each question will be graded on its own merit; so, for example, if when answering to the first
  part of the question, you got a wrong automaton, but on the second part, you correctly traced
  the new automaton, you will get full credit for the second part.

Good luck!

1-3. Let us consider an automaton for recognizing identifiers in Java, i.e., strings that start with a
letter, followed by letters, digits, or an underscore \_ . For simplicity, assume that the only letter is b,
and the only digit is 0. signed binary integers. This automaton has 3 states: start (s), identifier (i), and
error (e). Start is the starting state, identifier is the only final state. The transitions are as follows:

• from the state s, symbol b leads to i, symbols 0 and _ lead to e;
• from i, any symbol leads to i; and
• from e, every symbol leads to e.

1. Trace, step-by-step, how this finite automaton will check whether the following two strings are
valid Java identifiers:

• the word b_0 (which this automaton should accept) and
• the word 0b (which this automaton should reject).

2. Use the above tracing to find the parts x, y, and z of the word bb_0 corresponding to the Pumping
Lemma. Check that the “pumped” word xyyz will also be accepted by this automaton.

3. Write down the tuple <Q, Σ, δ, q₀, F> corresponding to this automaton:

• Q is the set of all the states,
• Σ is the alphabet, i.e., the set of all the symbols that this automaton can encounter;
• δ: Q x Σ → Q is the function that describes, for each state q and for each symbol s, the state
  δ(q, s) to which the automaton that was originally in the state q moves when it sees the symbol
  s (you do not need to describe all possible transitions this way, just describe two of them);
• q₀ is the staring state, and
• F is the set of all final states.
1. \[ b \rightarrow 10 \]
   \[ st \rightarrow i \rightarrow 1 \]
   \[ accepted \]

2. \[ b \rightarrow b \rightarrow 0 \]
   \[ x = b \text{ (before 1st rep)} \]
   \[ y = b \text{ (between 1st and 2nd rep)} \]
   \[ z = -0 \text{ (after the 2nd rep)} \]

   By pumping \( y = xyz \), it also shows it can be accepted.

3. \[ Q = \{ s, i, e \} \]
   \[ \Sigma = \{ b, o, \rightarrow, 1 \} \]
   \[ \delta(s, b) = i \]
   \[ \delta(s, o) = e \]
   \[ q_0 = s \]
   \[ F = \{ i, f \} \]
4-5. Let A be the automaton described in Problem 1-3. Let B be an automaton that accepts all the strings that contain only b and _ but not any other symbols. This automaton has two states: the start state which is also a final state, and the sink state. The transitions are as follows:

- from the start state, b or _ leads back to the start state, 0 leads to the sink;
- from the sink state, any symbol leads back to the sink.

4. Use the algorithm that we had in class to describe the following two new automata:

- the automaton that recognizes the union A U B of the two corresponding languages, and
- the automaton that recognizes the intersection of the languages A and B.

5. Test these two new automata step-by-step on the following words:

- test the union automaton on the example of the word 0b (that it should accept);
- test the intersection automaton on the example of the words b_0 (that it should reject).
6. Use the general algorithm that we learned in class to design a non-deterministic finite automaton that recognizes the language \( b(0 \cup b \cup \_\_\_)^* \):

- first, describe the automata for recognizing \( b \), \( 0 \), and \( \_\_\_ \);
- then, combine them into the automata for recognizing the union \( 0 \cup b \), \( 0 \cup b \cup \_\_\_ \) and the Kleene star \( (0 \cup b \cup \_\_\_)^* \);
- finally, combine the automaton for \( b \) with an automaton for the Kleene star into an automaton for recognizing the desired composition of the two languages.

7. Use the general algorithm to transform the resulting non-deterministic finite automaton into a deterministic one.
\((0v+bv-)\)
8-9. Use a general algorithm to transform the finite automaton B from Problem 4-5 into the corresponding regular expression.

\[
R^*_{ns,s^+} = R_{ns,s^+} \cup (R_{ns,s^+} R^*_{sink,sink} R_{sink,s^+}) \\
= \wedge \cup (\emptyset \ldots) = \wedge
\]

\[
R^*_{s^+,s^+} = R_{s^+,s^+} \cup (R_{s^+,s^+} R^*_{sink,sink} R_{sink,s^+}) \\
= b_1 \cup \emptyset (b_1 \wedge \emptyset^* \emptyset) = b_1 \cup \emptyset = b_1
\]

\[
R^*_{s^+,nf} = R_{s^+,nf} \cup (R_{s^+,sink} R^*_{sink,sink} R_{sink,nf}) \\
= \wedge \cup (b_1 \wedge \emptyset^* \emptyset) = \wedge \cup \emptyset = \wedge
\]
\( \mathcal{L} = S+ \)

\[ \rightarrow (b, \cdot)^* \]

\[ \rightarrow (s) \rightarrow (n) \rightarrow (f) \]

\[
R'_{ns, nf} = R_{ns, nf} \cup (R_{ns, st} \ R^*_{st, st} \ R_{st, nf}) \\
= \emptyset \cup (b, \cdot)^* \ = (b, \cdot)^*
\]
10) By contradiction, let's assume that \( L \) is regular. Then by pumping lemma,\( \exists p \geq p \geq p \rightarrow \exists x, y, z \text{ s.t. } w = xyz \& \text{len}(y) > 0 \& \text{len}(xy) \leq p \).

In particular, this should be true for the following word: \( xy \in L \).
\[
w = b^p0^p = b \ldots b, 0 \ldots 0 = xy, \text{len}(xy) \leq p
\]
x is in the first half, so \( y \) only has \( b \)'s. So when we go from \( w = xy \) to \( xy, \) we add \( b \)'s, but we did not add any \( 0 \)'s, so now we have more \( b \)'s than \( 0 \)'s, so \( xy \notin L \). Contradiction shows that our assumption is wrong, so \( L \) is not regular.
10. Prove that the language $L$ of all the words that have equal number of $b$'s and $0$'s is not regular.

Prove by contradiction using pumping lemma.

$L = \{w \in \{b, 0\}^* \mid p \leq 0 \}$ has as many

$b$'s as $0$'s.

Let $L$ be a regular language, there exist an integer $p > 0$, and a word $w \in L$ that we can write as

$w = xyz$ \quad \text{where} \quad \text{len}(y) \geq p, \text{len}(xy) \leq p$

for every $i \in \mathbb{N}$ in $xy^iz \in L$.

Let's use $\overline{b6 \ldots b000 \ldots 0}$

Assume that it is regular then there is an integer $p > 0$ as stated in pumping lemma. Set $w = \overline{b^p0^p}$

That $\overline{b^p0^p} = xyz$ where $\text{len}(y) \geq p$, $\text{len}(xy) \leq p$ and for every $i \in \mathbb{N}$ $xy^iz \in L$.

Note that $xy$ is made up of $b$'s so $y$ is made up of $b$'s. Since $\text{len}(y) \geq p$ there are at least one $b$. So when we pump $b$'s according to pumping lemma, we don't add any $0$'s. So it's not the same as $b$'s.