**CS 3350** Automata, Computability, and Formal Languages  
Spring 2019, Test 1

Last 4 digits of your UTEP ID number: ___________________  
\[ \begin{array}{c} \frac{100}{100} \end{array} \]

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 out 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!

30/30

1-3. Let us consider the automaton for recognizing signed binary integers. This automaton has 4 states: start (st), sign (si), integer (i), and error (e). Start is the starting state, integer is the only final state. The transitions are as follows:

- from st, any digit (0 or 1) leads to i, any sign (+ or −) leads to si;
- from si, any digit leads to i, any sign to e;
- from i, any digit (0 or 1) leads to i, any sign to e;
- from e, every symbol leads to e.

1. Trace, step-by-step, how this finite automaton will check whether the following two words (sequences of symbols) represent a valid Java signed integer:

- the word +0101 (which this automaton should accept) and
- the word 1+1 (which this automaton should reject).

2. Use the above tracing to find the parts \( x, y, \) and \( z \) of the word +0101 corresponding to the Pumping Lemma.

3. Write down the tuple \( <Q, \Sigma, \delta, q_0, F> \) corresponding to this automaton:

- \( Q \) is the set of all the states,
- \( \Sigma \) is the alphabet, i.e., the set of all the symbols that this automaton can encounter;
- \( \delta: Q \times \Sigma \rightarrow Q \) is the function that describes, for each state \( q \) and for each symbol \( s \), the state \( \delta(q, s) \) 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);
- \( q_0 \) is the starting state, and
- \( F \) is the set of all final states.
1. + 0 1 0 1 1 + 1 +

2. + 0 1 0 1

\[ Q = \{ s_I, s_I, s_e, \varepsilon \} \]
\[ \Sigma = \{ 0, 1, +, -3 \} \]
\[ Q_0 = s_I \]
\[ F = \{ s_I, \varepsilon \} \]
\[ \delta(s_I, 0) = i \]
\[ \delta(s_I, +) = e \]
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 $+$s and $1$s 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, $+$ or $1$ 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 $A \cup B$ 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 $1+1$ (that it should accept);
- test the intersection automaton on the example of the words $+01$ (that it should reject).
Automaton A

Intersection:
Intersection is also part of union.

Union:

Intersection:
Intersection is also part of union.
6. Use the general algorithm that we learned in class to design a non-deterministic finite automaton that recognizes the language \((a \cup b)^*c:\)

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

7. Use the general algorithm to transform the resulting non-deterministic finite automaton into a deterministic one.
8.9. Use a general algorithm to transform the following finite automaton into the corresponding regular expression.

- This automaton has two states: a state $s_1$ which is a start state, and another state $s_2$ which is a final state.
- In the state $s_1$, a and b lead to $s_2$, and c leads to $s_1$.
- In the state $s_2$, a and c lead to $s_1$, and b leads to $s_2$.

\[
R_{s_1 s_2} = R_{s_1 s_2} \cup C R_{s_1 s_1} R_{s_1 s_1} R_{s_1 s_2}^* \] 
\[
= \emptyset \cup C \emptyset \emptyset \] 
\[
= \emptyset
\]

\[
R_{s_1 f} = R_{s_1 f} \cup C R_{s_1 s_1} R_{s_1 s_1} R_{s_1 f}^* \] 
\[
= \emptyset
\]

\[
R_{s_2 f} = R_{s_2 f} \cup C R_{s_2 s_1} R_{s_1 s_1} R_{s_1 f}^* \] 
\[
= \emptyset
\]

\[
R_{s_2 s_2} = R_{s_2 s_2} \cup C R_{s_2 s_1} R_{s_1 s_1} R_{s_1 s_1} R_{s_2 s_2}^* \] 
\[
= \emptyset \cup C \emptyset \emptyset \] 
\[
= \emptyset
\]
Eliminating $S_2$:

$$R_{S_1, F} = R_{S_1, F} \cup \rho R_{S_1, S_4} R_{S_4, S_5} R_{S_5, F} = c^* (a \cup b) (b \cup (a \cup c (c^* \cup a \cup b))^*)$$

Regular expression for automaton
8-9. Use a general algorithm to transform the following finite automaton into the corresponding regular expression.

- This automaton has two states: a state $s_1$ which is a start state, and another state $s_2$ which is a final state.
- In the state $s_1$, a and b lead to $s_2$, and c leads to $s_1$.
- In the state $s_2$, a and c lead to $s_1$, and b leads to $s_2$.

$$R_{ss_1}^* = R_{ss_1} \cup (R_{ss_2} R_{s_2 s_1}^* R_{s_2 s_1})$$

$$= \emptyset \cup (\emptyset b^* (a \cup c))$$

$$= \emptyset \cup \emptyset = \emptyset$$

$$R_{sf}^* = R_{sf} \cup (R_{ss_2} R_{s_2 s_1}^* R_{s_2 s_1})$$

$$= \emptyset \cup (\emptyset)$$

$$= \emptyset$$

$$R_{s_1 s_1}^* = R_{s_1 s_1} \cup (R_{s_1 s_2} R_{s_2 s_1}^* R_{s_2 s_1})$$

$$= C \cup ((a \cup b) b^* (a \cup c))$$

$$R_{s_1 f}^* = R_{s_1 f} \cup (R_{s_1 s_2} R_{s_2 s_1}^* R_{s_2 s_1})$$

$$= \emptyset \cup ((a \cup b) b^* \emptyset)$$

$$= (a \cup b) b^*$$
S. Automaton from Problem 12 accepts the word aeb.

- Trace, step-by-step, that this word is indeed accepted by the given automaton.
- Trace that the words xyyz and xyyyyz are also accepted by this automaton.