CS 3350 Automata, Computability, and Formal Languages
Spring 2019, Test 2

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 grammar, but on the second part, you correctly traced the new grammar, you will get full credit for the second part.

Good luck!  

10/10

1. A student who gets only As and Bs is classified as cum laude during the commencement when his or her GPA is 3.5 is above, i.e., if he/she has at least as many As as Bs. For example, students with records ABAB or ABA qualify. Prove that the language consisting of all sequences of As and Bs that make the student qualified is not regular.

$L = \{ B^n A^k : n = 0,1,2,... ; k \geq n \}$

Assume $L$ is regular. Then by pumping lemma $\exists p \forall w \in L \ w \geq p \ w = xyz \ \text{len}(y) > 0, \ \text{len}(xy) \leq p, \ x \ y^i \ z \in L$. Let's take the word $w = B^n A^n$ . Its length is $2p \geq p$, so $w$ can be represented as $xyz$ where $\text{len}(y) \leq p$, so $xy^i z$ consist of $B$. So when we go from $xyz$ to $xy^2z$, we add $B$ but we don't add $A$. So in $x y^2 z$ we have more $B$ than $A$, so $x y^2 z$ is not in $L$. But by pumping lemma, $x y^2 z$ is in $L$. We get a contradiction, so our assumption is wrong and $L$ is not regular.
2. Design a finite automaton for recognizing binary sequences that have even number of 0s and even number of 1s. Assume that the input strings contain only symbols 0 and 1. The easiest is to have 4 states:

- the desired (final) state ee in which we have even number of 0s and even number of 1s,
- the state eo, in which we have even number of 0s and odd number of 1s, and
- similarly defined states oe (odd-even) and oo (odd-odd).

You just need to describe transitions between these states, and which states are final. Show, step-by-step, how your automaton will accept the string 1010.
3. Use a general algorithm to transform the finite automaton from Problem 2 into a context-free grammar (CFG). Show, step-by-step, how this CFG will generate the word 1010.

\[
E_e = ee \\
E_o = eo \\
O_e = oe \\
O_o = oo \\
E_e \rightarrow \epsilon \\
E_e \rightarrow 1E_o \\
E_e \rightarrow \emptyset O_e \\
E_o \rightarrow 1E_e \\
E_o \rightarrow \emptyset O_o \\
O_e \rightarrow 1O_o \\
O_e \rightarrow \emptyset E_e \\
O_o \rightarrow 1O_e \\
O_o \rightarrow \emptyset E_o \\
E_e \rightarrow 1E_o \rightarrow 1\emptyset O_o \rightarrow 1\emptyset 1O_e \rightarrow 1\emptyset 1\emptyset E_e \rightarrow 1\emptyset 1\emptyset
\]
4-5. Use a general algorithm to translate the CFG from Problem 3 into an appropriate push-down automaton. Explain, step-by-step, how this automaton will accept the word 1010.
6-7. Use a general algorithm to translate the CFG from Problem 3 into Chomsky normal form.

Prelim: \(S_0 \rightarrow Ec\)

step 0: \(E_0 \rightarrow 1\)
- \(O_c \rightarrow \emptyset\)
- \(S_0 \rightarrow \varepsilon\)

step 1: \(S_0 \rightarrow 1E_0\)
- \(S_0 \rightarrow \emptyset O_c\)

step 2: \(V_1 \rightarrow 1\)
- \(V_\emptyset \rightarrow \emptyset\)
- \(E_c \rightarrow V_1 E_0\)
- \(E_c \rightarrow V_\emptyset O_c\)
- \(E_0 \rightarrow V_1 E_c\)
- \(E_0 \rightarrow V_\emptyset O_o\)
- \(O_o \rightarrow V_1 O_o\)
- \(O_o \rightarrow V_\emptyset E_c\)
- \(O_o \rightarrow V_\emptyset E_0\)
- \(S_0 \rightarrow V_1 E_0\)
- \(S_0 \rightarrow V_\emptyset O_c\)
8. Use the general stack-based algorithms to show:

- how the compiler will transform a Java expression $8 - 4 - 2$ into inverse Polish notation, and
- how it will compute the value of this expression.

\[
\begin{align*}
8 - 4 - 2 - & \quad 9 - 2 - \\
\end{align*}
\]
9-10. Use a general algorithm for transforming PDA into CFG to design a CFG that corresponds to the following pushdown automaton. This automaton has two states: the starting a-state $s_a$ and the final b-state $s_b$. The transitions are:

- From $s_a$ to $s_a$, the transition is $a, \varepsilon \rightarrow 1$
- From $s_a$ to $s_b$, the transition is $b, 1 \rightarrow \varepsilon$.
- From $s_b$ to $s_b$, the transition is $b, 1 \rightarrow \varepsilon$.

Show, step by step, how the word $ab$ will be generated by the resulting grammar.

\[
S \rightarrow A_{s_a} s_b \\
A_{s_a} s_b \rightarrow a A_{s_a} s_a b \\
A_{s_a} s_a \rightarrow \varepsilon
\]
11 (for extra credit). For the context-free grammar from the Problem 3, show how the word 1010 can be represented as $uvwyz$ in accordance with the pumping lemma.

$u = \varepsilon$

$v = 1010$

$x = \varepsilon$

$y = \varepsilon$

$z = \varepsilon$

$uvxyz = 1010$