FLAC Assignment 7

Exercise 1 Give a Turing machine with at most 12 states that doubles a number in unary representation. You will lose points if you use extra states. It should be clear your solution is correct; give explanation if necessary.

Exercise 2 (a) Convert the following CFG into Chomsky Normal Form. Write down your steps.

$$
\begin{aligned}
S &\to \mathtt{a}A\mathtt{a} \mid \mathtt{b}B\mathtt{b} \mid \epsilon \\
A &\to C \mid \mathtt{a} \\
B &\to C \mid \mathtt{b} \\
C &\to CDA \mid \epsilon \\
D &\to A \mid B \mid \mathtt{ab}
\end{aligned}
$$

(b) Use Younger's Algorithm to to decide whether "ababa" is in the language. Write down the steps.

Exercise 3 We already know $\{\mathtt{a}^n\mathtt{b}^n\mathtt{c}^n \mid n \geq 0\}$ is not a context free language. Give a Turing machine that decides this language.

Exercise 4 We know following grammar is ambiguous. Please give some string in the language and show such that it has two different parse trees.

Here, <stmt> is the start symbol and terminals are: `else`, `basic_stmt`, `for_clause`, `if`, `boolexpr`, `then`, `blk`, `compound`.

```
<stmt>        → <uncond_stmt> | <cond_stmt>
<uncond_stmt> → basic-stmt | <for_stmt>| blk | compound
<for_stmt>    → for_clause <stmt>
<cond_stmt>   → <if_stmt> |<if_stmt> else <stmt>
<if_stmt>     → <if_clause> <uncond_stmt>
<if_clause>   → if boolexpr then
```

Exercise 5 In class we introduced a type of Turing Machine whose tape is two-way infinite, which means the machine can keep moving left or right indefinitely. Also the action that the machine can take is one of $\{L, R, N\}$.

In the book, the definition is slightly different. The tape of the Turing Machine is one-way infinite, which means there is a leftmost square of the tape and the machine cannot move left when at that position. In addition, the action the machine can take is one of $\{L, R\}$.

You task is to prove that a Turing Machine of the type defined in the textbook can simulate a Turing Machine of the type defined in class.

Exercise 6 (Bonus) Prove that any context-free language over alphabet size 1, for example $\Sigma = \{1\}$, is also regular language.

if the language is finite, its regular anyways
if the language is infinite, there will be a pumping length, let this be p
and for any string of length greater than p, we will have a value of lvxl
<=p such that a^(n+tlvxl) is in L. make a state corresponding to each
avlue of lvxl and one for strings of length less than the pumping length