1. (a) Give a CFG generating the language \( L = \{a^m b^m \mid 0 \leq n \leq m \leq 3n\} \).

\[
S \rightarrow aSb \mid aSbb \mid aSbbb \mid \epsilon
\]

(b) Is your grammar ambiguous? Why or why not?
Yes, it is ambiguous. There are two different parse trees for deriving the same string \( aabbb \):

![Parse Trees](image-url)
2. Give the state diagram of a PDA that recognizes the language

\[ M = \{ a^i b^j \mid i, j \geq 0 \text{ and } j < 2i \} \]

Note: your PDA is allowed to push more than one symbol onto the stack at a time; if it pushes a string \( a_1 a_2 \ldots a_k \), then the symbol \( a_1 \) will become the top of the stack.

Idea: for each input symbol \( a \), push two \( a \)'s onto the stack. When see the first \( b \), start popping \( a \)'s. Accept while the stack remains nonempty. Reject if the stack becomes empty.

3. (a) Complete the definition: A CFG \( G = (V; \Sigma; R, S) \) is in **Chomsky Normal Form** if each rule in \( R \) is of the form

\[ A \rightarrow BC \]

or

\[ A \rightarrow a. \]

where \( A \in V, B, C \in V - \{S\}, a \in \Sigma. \)

In addition, we allow the rule \( S \rightarrow \epsilon. \)

3. (b) Convert the following CFG into an equivalent CFG in Chomsky Normal Form. (The start variable is \( A \).)

\[ A \rightarrow BAB \mid B \mid 000 \]

\[ B \rightarrow 00 \mid AB \]

Solution:

\[ S \rightarrow BX \mid ZZ \mid AB \mid ZY \]

\[ A \rightarrow BX \mid ZZ \mid AB \mid ZY \]

\[ B \rightarrow ZZ \mid AB \]

\[ X \rightarrow AB \]

\[ Y \rightarrow ZZ \]

\[ Z \rightarrow 0 \]

The new start variable is \( S \).
4. Use the Pumping Lemma to show that the following language is not context-free:

\[ L = \{ w \mid w \in \{ a, b, c \}^* \text{ and } w \text{ has equal number of } a \text{'s, } b \text{'s, and } c \text{'s} \} \]

Suppose \( L \) is a CFL. Then the Pumping Lemma holds for \( L \) with some pumping length \( p \). Consider the string \( s = a^p b^p c^p \). Clearly, \( s \in L \) and \( |s| > p \).

By the Pumping Lemma, there must be a partitioning of \( s = uvxyz \) such that \( |vy| > 0, |vxy| \leq p, \) and \( uv^i xy^i z \in L \) for every \( i \geq 0 \). Consider all possible cases: Since \( |vxy| \leq p \), \( v \) and \( y \) contain at most two different (consecutive) symbols: type 1 and type 2. Hence, if we take the string \( uvxyzz \), then it will contain more symbols of type 1 or of type 2, while it will still contain the same number of symbols of the remaining type (type 3). Hence, \( uvxyzz \notin L \). A contradiction.

5. Complete the following high-level description of a Turing machine recognizing the language

\[ L = \{ w \mid w \in \{ a, b, c \}^* \text{ and } w \text{ has equal number of } a \text{'s, } b \text{'s, and } c \text{'s} \} \]

On input \( w \):

(a) If \( w = \epsilon \), then Accept.

(b) Scan the input and find the leftmost \( a \). Replace it by \( X \). If no \( a \) is found, Reject.

(c) Scan the input and find the leftmost \( b \). Replace it by \( Y \). If no \( b \) is found, Reject.

(d) Scan the input and find the leftmost \( c \). Replace it by \( Z \). If no \( c \) is found, Reject.

(e) Scan the input and check if any \( a \text{'s, } b \text{'s, or } c \text{'s are left. If none, then Accept}

Otherwise go to Step (b)