1. (20 pts). convert to Chomsky Normal Form

a)

$S \rightarrow AaBb$
$A \rightarrow aB$
$B \rightarrow b$

Step 1.) create a new start state

$S_0 \rightarrow S$
$S \rightarrow AaBb$
$A \rightarrow aB$
$B \rightarrow b$

Step 2.) NA

Step 3.) Remove Unit Rule $S_0 \rightarrow S$

$S_0 \rightarrow AaBb$
$S \rightarrow AaBb$
$A \rightarrow aB$
$B \rightarrow b$

Step 4a.) add additional Variable and Rule $A_1 \rightarrow a$

$S_0 \rightarrow AA_1BB$
$S \rightarrow AA_1BB$
$A \rightarrow AB$
$A_1 \rightarrow a$
$B \rightarrow b$

Step 4b.) add additional Variable and Rule $A_2 \rightarrow AA_1, B_1 \rightarrow BB$

$S_0 \rightarrow A_2B_1$
$S \rightarrow A_2B_1$
$A \rightarrow A_1B$
$A_1 \rightarrow a$
$A_2 \rightarrow AA_1$

$B_1 \to BB$
$B \to b$

b)

$S \to aAbA$
$A \to aB|bB|\epsilon$
$B \to A|c$

Step 1.) create a new start state

$S_0 \to S$
$S \to aAbA$
$A \to aB|bB|\epsilon$
$B \to A|c$

Step 2a.) eliminate ϵ move allowed From A and fix all rules which map to A

$S_0 \to S$
$S \to aAbA|a\epsilon bA|a\epsilon b\epsilon|aAb\epsilon$
$A \to aB|bB$
$B \to A|c|\epsilon$

Step 2b.) eliminate ϵ moves allowed From B and fix all rules which map to B

$S_0 \to S$
$S \to aAbA|abA|ab|aAb$
$A \to aB|bB|a|b$
$B \to A|c|$

Step 3.) Remove Unit Rule $S_0 \to S$

$S \to aAbA|abA|ab|aAb$
$S \to aAbA|abA|ab|aAb$
$A \to aB|bB|a|b$
$B \to A|c|$

Step 4a.) add additional Variable and Rule $A_1 \to a, B_1 \to b$

$S \to aAbA|abA|ab|aAb$
$S \to aAbA|abA|ab|aAb$
$A \to aB|bB|a|b$
$B \to A|c|$
$A_1 \to a$
$B_1 \to b$

Step 4b.) Replace the A in $B \to A$ with what A maps to

$S \to A_1AB_1A|A_1B_1A|A_1B_1|A_1AB_1$
$S \to A_1AB_1A|A_1B_1A|A_1B_1|A_1AB_1$
$A \to A_1B|B_1B|a|b$
$B \to A_1B|B_1B|a|b|c|$
$A_1 \to a$
$B_1 \to b$

step 5.) perform step 4 again with two new variable rules: Q T

$S \to QT|A_1T|A_1B_1|QB_1$
$S \to QT|A_1T|A_1B_1|QB_1$
$A \to A_1B|B_1B|a|b$
$B \to A_1B|B_1B|a|b|c|$
$A_1 \to a$
$B_1 \to b$
$Q \to A_1A$
$T \to B_1A$
———————————————-

———————————————

2. Show that if G is a CFG in Chomsky Normal Form, then for any string w in the language of G of length 1 or greater, then there are exactly 2n - 1 steps required to derive w where n is the size of the string.

We will show this with an inductive proof on the size n of any word w in G

BASE CASE: let n = 1

Then we know the word is 1 character long. In this case the start state will map directly to the output string. This takes exactly 1 step = 2(1) - 1

Inductive Hypothesis:
for all words of size n = k:

Suppose all words of k length can be derived in 2(k) - 1 steps, then any word w of k+1 length can be derived in 2(k+1) - 1 steps

Consider any word in G. Since we know G is in Chomsky Normal Form, we can be certain that every 'single' character in string w is ultimately derived in a single step. We also know that Every mapping is either to a single character in w or to a pair of variables which can either map to another pair of variables or a single character. There are no epsilon moves.

So then, we know from our supposition that every word of length (i=1 ... i = k) characters is derivable in 2(i) - 1 moves. Then for a word with i+1 characters' moves = (2(i+1) - 1) moves. This is true for every integer i from 1 to infiniti, because as the length increases by one exactly one of the final derivation steps (we will let it be represented as $S \to e$ where e is an element of w) from the word of length i must be converted to a mapping to a pair variables which are not terminals. one of these variables will then map to the the terminal e, while the other will map to the freshly added element of the word of length i+1. This results in the addition of 2 new steps.

Since (2(i+1)-1) subtract (2(i)-1 for all i from 1 to infiniti we have proven the inductive step as required, and as the base case has already been proved. We know that for all words of G they can be derived in exactly (2(n) - 1) steps s.t. n is the length of the word.

3. ____

4. GIVE UNAMBIGUOUS CFG'S FOR THE FOLLOWING LANGUAGES

a)
the language where every prefix of a word has at least as many a's as b's

This is easy enough to accomplish by just forcing the rule of where the start state maps to.

we can say $S \to abS|aS|baS|\epsilon$

This guarantees there will be as many or more a's in every prefix to S

However this CFG is ambiguous because there is more than one way to derive some word aba S

We fix this by adding 2 new states

$S_1 \rightarrow aS$
$S_2 \rightarrow bS$
$S_3 \rightarrow S_3|\epsilon$

where S_3 maps additionally to everything not in the prefix

and changing the rule for S to

$S \rightarrow S_1S_2S_3|S_2S_1S_3|S_1S_3$

b)

the language where every word has at exactly as many a's as b's

This is easy enough to accomplish by just forcing the rule of where the start state maps to to include either start with an 'a' and contain a single 'b' or start with a 'b' and contain a single 'a'. where any other similar substring can be to the string so long as it does not create an ambiguous construction

we can say $S \rightarrow aSbS|bSaS$

c)

This is the same as for part a except we do not consider the prefix as a separate part of the word. in other words we must be able to guarantee that there are at least the number of a's as b's in the entire word w in G without ambiguity

We can accomplish this by simply having S_3 map back to S. so that we have the CFG
$S_1 \rightarrow aS$
$S_2 \rightarrow bS$
$S_3 \rightarrow S|\epsilon$

$S \rightarrow S_1S_2S_3|S_2S_1S_3|S_1S_3$

where S is the start state

5. To show that the language in exercise 2.9 is inherently ambiguous we must show that there is no way to construct a grammar which satisfies the language and provides exactly one

way to derive each string in the language.

To accomplish this consider that the language accepts words with equal numbers of a's and b's OR equal numbers of b's and c's.

Since every grammar has to allow for the possibility of both instances potentially occurring for each word tested to see if it is accepted. Then we can deduce that every word accepted by A which can be accepted on the grounds that it meets both conditions ($i = j$ AND $j = k$) can be counted two different ways. If it can be counted or derived two different ways, then there are at least two different derivations for such words, which proves that language A is inherently ambiguous.

lets say a word has 10 a's, 10 b's, and 10 c's. Then the grammar must include a leftmost derivation which constructs the word by ignoring the number of a's before it begins matching b's and c's, AND it must also possess a different leftmost derivation which ignores the number of c's after the number of matching a's and b's. This shows that Language A is inherently ambiguous as required.