1.[5 pts] For each of the following CFG, use set notation to define the language generated by the grammar

1.1) $S \longrightarrow aSbb \mid A$
$A \longrightarrow cA \mid c$
L = {a^n c^m b^(2n) | n ≥ 0, m ≥ 1}

1.2) $S \longrightarrow aS \mid bB \mid \varepsilon$
$B \longrightarrow aB \mid bS \mid bC$
$C \longrightarrow aC \mid \varepsilon$
L = {w ∈ {a, b}*}

2. [5 pts] Let G be the following grammar:
$S \longrightarrow SAB \mid \varepsilon$
$A \longrightarrow aA \mid a$
$B \longrightarrow bB \mid \varepsilon$

2.1) Give a leftmost derivation of abbaab (fill in the table to show both derivation and the rules)
Derivation Rule applied

Derivation	Rule Applied
$S \Rightarrow SAB$	S -> SAB
$SABAB$	S -> SAB
$ABAB$	S -> e
aBAB	A -> a
abBAB	B -> bB
abbBAB	B -> bB
abbAB	B -> e
abbaAB	A -> aA
abbaaB	A -> a
abbaabB	B -> bB
abbaab	B -> e

2.2) Give two leftmost derivation of aa Derivation Rule applied

ONE:

Derivation	Rule Applied
$S \Rightarrow SAB$	S -> SAB
AB	S -> e
aAB	A -> aA
aaB	A -> a
aa	B -> e

TWO:

Derivation	Rule Applied
$S \Rightarrow SAB$	S -> SAB
SABAB	S -> SAB
ABAB	S -> e
aBAB	A -> a
aAB	B -> e
aaB	A -> a
aa	B -> e

2.3) Is this grammar ambiguous?

The grammar is ambiguous because there are 2 different leftmost derivations of the string 'aa'. Several other strings can be taken to prove this idea that the grammar is in fact, ambigous

2.4) Is this language L(G) regular? If yes, give a regular expression for the language.

No, this language is not regular. This language could also be stated as a^n b^n where n >= 0. This is not regular because a dfa/nfa could not be created that demonstrates this language because it would require an infinite number of states. N in this case can be infinite and there would be no way to count the number of a's that are there to make sure that they are equal to the number of b characters.

3. [10 pts] Follow the procedure discussed in class (or in textbook pp.128), convert the following CFG into Chomsky Normal Form.

$S \longrightarrow aTc \mid bc$
$T \longrightarrow ST \mid \varepsilon$

Converting to CNF:

- Make a new start state
  - X -> S
  - S -> aTc | bc
  - T -> ST | ε
- Find Nullable Vars and Remove Epsilons
  - X -> S | ε
  - S -> aTc | bc | ac
  - T -> ST | T | S
- Eliminate unit rules
  - X -> aTc | bc | ac | ε
  - S -> aTc | bc | ac
  - T -> ST | aTc | bc | ac
- Ensure RHS is all Vars, or Single Terminals
  - X -> ATC | BC | AC | ε
  - S -> ATC | BC | AC
  - T -> ST | ATC | BC | AC
  - A -> a
  - B -> b
  - C -> c
- Reduce RHS so only 2 Vars
  - X -> JC | BC | AC | ε
  - S -> JC | BC | AC
  - T -> ST | JC | BC | AC
  - A -> a
  - B -> b
  - C -> c
  - J -> AT