Problem 1

Total: 30 points (15 points each)
Exercise 2.6. Give context-free grammars (CFGs) generating the following languages.

1. The set of strings over the alphabet $\Sigma = \{a, b\}$ with more a's than b's
   Solution 1:

$$\begin{aligned} S &\to A \mid MS \mid SM \mid SS \\ A &\to Aa \mid a \\ M &\to \epsilon \mid bMa \mid aMb \end{aligned}$$

   Explanation: Production rules for M generates equal number of a and b. Since the starting variable can only be eliminated using $S \to A$, the grammar ensures the generated string has one or more a than b. $S \to SS$ allows us to derive an arbitrary number of M and A in any desired structure, which can then be used to derive a desired string.
   Solution 2:

$$\begin{aligned} S &\to TaTaTbT \mid TaTbTaT \mid TbTaTaT \mid a \\ T &\to aTb \mid bTa \mid aT \mid \epsilon \end{aligned}$$

   Explanation: Production rules for the starting variable S ensure that the generated string has at least one more a than b. T generates at least as many a as b. Combined, the grammar generates strings with more a than b. This can be easily proved using induction on $\#a(s)$, or the number of a in s, which is left as an optional exercise.

2. The complement of the language $\{a^n b^n \mid n \geq 0\}$
   Solution:
   A is the complement of the language $\{a^n b^n \mid n \geq 0\}$.
   That is, A $= \{a^n b^m \mid n \neq m\} \cup \{(a \cup b)^* ba(a \cup b)^*\}$ (The left language contains strings in the correct order, i.e., a's followed by b's, but the number of a and b contained are uneven. The right language contains strings with the order of a and b mixed).
   Let $A_1 = \{a^n b^m \mid n \neq m\}$ and $A_2 = \{(a \cup b)^* ba(a \cup b)^*\}$
   The CFG that generates A_1 is:

$$\begin{aligned} S_1 &\to aS_1b \mid T \mid U \\ T &\to aT \mid a \\ U &\to Ub \mid b \end{aligned}$$

   The CFG that generates A_2 is:

$$\begin{aligned} S_2 &\to VbaV \\ V &\to VV \mid a \mid b \mid \epsilon \end{aligned}$$

   Therefore, the CFG that generates $A = A_1 \cup A_2$ is:

$$\begin{aligned} S &\to S_1 \mid S_2 \\ S_1 &\to aS_1b \mid T \mid U \\ T &\to aT \mid a \\ U &\to Ub \mid b \\ S_2 &\to VbaV \\ V &\to VV \mid a \mid b \mid \epsilon \end{aligned}$$

Problem 2

Total: 15 points
Exercise 2.9. Give CFG generating the following language:

$A = \{a^i b^j c^k \mid i = j \text{ or } j = k \text{ where } i, j, k \geq 0\}$

Is your grammar ambiguous? Why or why not? If yes, please provide an example of two different leftmost derivations that generate the same string.

Solution: We have discussed this in class. Notice $A = A_1 \cup A_2$ where $A_1 = \{a^i b^j c^k \mid i = j \text{ where } i, j, k \geq 0\}$ and $A_2 = \{a^i b^j c^k \mid j = k \text{ where } i, j, k \geq 0\}$. A rigorous proof of why A is *inherently ambiguous* is beyond what we have covered in class. An informal argument in one of the following directions is acceptable.

- $A_1 \cap A_2$ is NOT context-free (as per our discussion in class). There cannot be a CFG powerful enough to single out this non-context-free part of the language to allow a unique leftmost derivation.
- Construct CFGs for A_1 and A_2, which you can combine into a CFG for A. Then find an ambiguous string and show two different leftmost derivations of the string using your grammar that results in two different parse trees.

Problem 3

Total: 15 points
Exercise 2.14. Convert the following CFG into an equivalent CFG in Chomsky normal form using the procedure given in Theorem 2.9.
Please provide all intermediate steps with comments on how you transform from the grammar from one version to another (these steps are critical for your work to be graded).

$$
\begin{aligned}
&A \to BAB \mid B \mid \epsilon \\
&B \to 00 \mid \epsilon
\end{aligned}
$$

Solution:
Step 1: Introduce a new starting variable (since the grammar contains the original starting variable A on RHS of a production rule).

$$
\begin{aligned}
&S \to A \\
&A \to BAB \mid B \mid \epsilon \\
&B \to 00 \mid \epsilon
\end{aligned}
$$

Step 2: Eliminate ϵ-rules. In our grammar: $B \to \epsilon$ and $A \to \epsilon$

$$
\begin{aligned}
&S \to A \mid \epsilon \\
&A \to BAB \mid AB \mid BA \mid BB \mid B \\
&B \to 00
\end{aligned}
$$

Step 3: Eliminate unit production rules. In our grammar: $A \to B$ and $S \to A$

$$
\begin{aligned}
&S \to BAB \mid AB \mid BA \mid BB \mid 00 \mid \epsilon \\
&A \to BAB \mid AB \mid BA \mid BB \mid 00 \\
&B \to 00
\end{aligned}
$$

Step 4: Convert terminals in non-unit terminal production rules into variables. In our grammar: 0

$$
\begin{aligned}
S &\to BAB \mid AB \mid BA \mid TT \mid \epsilon \\
A &\to BAB \mid AB \mid BA \mid TT \\
B &\to TT \\
T &\to 0
\end{aligned}
$$

Step 5: Split long rules. In our grammar: $S \to BAB$ and $A \to BAB$

$$
\begin{aligned}
S &\to BU \mid AB \mid BA \mid BB \mid TT \mid \epsilon \\
A &\to BU \mid AB \mid BA \mid BB \mid TT \\
B &\to TT \\
T &\to 0 \\
U &\to AB
\end{aligned}
$$