1. [20 points] Give a CFG for the language \( L \), where \( L = \{0^i 1^j \mid i < j \text{ or } i > 2j \} \). \textbf{Hint:} Consider the cases where \( i < j \) and \( i > 2j \) separately.

2. The following CFG \( G \) generates the same language accepted by the regular expression \((10)^*0(10 + 01)^*\):

\[
S \rightarrow A0B \\
A \rightarrow AC \mid \epsilon \\
B \rightarrow BD \mid \epsilon \\
C \rightarrow 10 \\
D \rightarrow C \mid E \\
E \rightarrow 01
\]

\( a) \) [8 points] Draw a parse tree for \( G \) yielding the string 101000110.

\( b) \) [7 points] List all of the steps in the rightmost derivation of the string 00110.

\( c) \) [5 points] List all of the steps in the leftmost derivation of the string 10001.

3. [10 points] Construct a PDA accepting the language over \( \{0, 1\} \) consisting of all strings such that no prefix has more 1s than 0s.

4. [10 points] Convert the following CFG to a PDA:

\[
S \rightarrow 0S1 \mid A \\
A \rightarrow 1A0 \mid S \mid \epsilon
\]

5. [20 points] Consider the following PDA \( P = (\{q, p\}, \{0, 1\}, \{X, Z\}, \delta, q_0, Z_0, F) \), where \( \delta \) is illustrated by the following diagram:

(Note that \( P \) accepts by empty stack.) Convert \( P \) to a CFG. You don’t need to include any useless productions in your final CFG.

6. [20 points] If PDAs are modified so that the extra storage consists only of a finite stack (rather than an infinite one), it turns out that they can only accept regular languages. Demonstrate that this is the case by describing in detail how to convert a PDA with a finite stack to an NFA with \( \epsilon \)-transitions.

Assume that your starting PDA \( P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F) \). \( P \) accepts by final state, and its stack can hold at most \( k \) symbols, for some arbitrary constant \( k > 0 \). Assume that attempting to add symbols to a full stack will cause execution to die.