2.30 a. Assume $A$ is context-free. Let $p$ be the pumping length given by the pumping lemma. We show that $s = 0^p1^p0^p1^p$ cannot be pumped. Let $s = uvxyz$.

If either $u$ or $y$ contain more than one type of alphabet symbol, $u^2xy^2z$ does not contain the symbols in the correct order and cannot be a member of $A$. If both $u$ and $y$ contain (at most) one type of alphabet symbol, $u^2xy^2z$ contains runs of 0's and 1's of unequal length and cannot be a member of $A$. Because $s$ cannot be pumped without violating the pumping lemma conditions, $A$ is not context-free.

d. Assume $A$ is context-free. Let $p$ be the pumping length from the pumping lemma. Let $s = a^p b^p a^p b^p$. We show that $s = uvxyz$ cannot be pumped. Use the same reasoning as in part (c).

2.42 Assume $Y$ is a CFL and let $p$ be the pumping length given by the pumping lemma. Let $s = 1^{p+1}#1^{p+2}# \cdots #1^{5p}$. String $s$ is in $Y$ but we show it cannot be pumped. Let $s = uvxyz$ satisfying the three conditions of the lemma. Consider several cases.

i) If either $u$ or $y$ contain #, the string $uv^2xy^2z$ has two consecutive $t_i$'s which are equal to each other. Hence that string is not a member of $Y$.

ii) If both $u$ and $y$ contain only 1s, these strings must either lie in the same run of 1s or in consecutive runs of 1s within $s$, by virtue of condition 3. If $v$ lies within the runs from $1^{p+1}$ to $1^{5p}$ then $uv^2xy^2z$ adds at most $p$ 1s to that run so that it will contain the same number of 1s in a higher run. Therefore the resulting string will not be a member of $Y$. If $v$ lies within the runs from $1^{5p+1}$ to $1^{5p}$ then $uv^0xy^0z$ subtracts at most $p$ 1s to that run so that it will contain the same number of 1s in a lower run. Therefore the resulting string will not be a member of $Y$.

The string $s$ isn't pumpable and therefore doesn't satisfy the conditions of the pumping lemma, so a contradiction has occurred. Hence $Y$ is not a CFL.

2.45 Suppose $A$ is context-free and let $p$ be the associated pumping length. Let $s = 1^{2p}0^p1^p2p$ which is in $A$ and longer than $p$. By the pumping lemma we know that $s = uvxyz$ satisfying the three conditions. We distinguish cases to show that $s$ cannot be pumped and remain in $A$.

- If $uv$ is entirely in the last two thirds of $s$, then $uv^2xy^2z$ contains 0s in its first third but not in its last third and so is not in $A$.

- Otherwise, $uv$ intersects the first third of $s$, and it cannot extend beyond the first half of $s$ without violating the third condition.
  - If $v$ contains both 1s and 0s, then $uv^2xy^2z$ contains 0s in its first third but not in its last third and so is not in $A$.
  - If $v$ is empty and $y$ contains both 0s and possibly 1s, then again $uv^2xy^2z$ contains 0s in its first third but not in its last third and is not in $A$.
  - Otherwise, either $v$ contains only 1s, or $v$ is empty and $y$ contains only 1s. In both cases, $uv^1+5pxy^{1+6p}z$ contains 0s in its last third but not in its first third and so is not in $A$.

A contradiction therefore arises and so $A$ isn't a CFL.
Problem 6.
(a) \( S(x) \). The language \( L = \{ a^n b^n | n \geq 3 \} \) is context-free.

Suppose that \( L \) were regular. Then
\( L^* \cap a^* b^* \) would be regular.

Notice that \( L^* \cap a^* b^* = L \), which is non-regular.

(b) I thought that the statement would be true, but it is in fact false.

Consider that \( L = \{ a^n b^n | n \geq 3 \} \) is context-free.

Now \( K = \{ w w r | w \in L \} \)
\[ = \{ a^n b^n b^n a^n | n \geq 3 \} \]
\[ = \{ a^n b^{2n} a^n | n \geq 3 \} \]

We can use Pumping Lemma to show that \( a^n b^{2n} a^n | n \geq 3 \) is not context-free.

(c) True. We show that \( L \cup K \) is not context-free by contradiction.

Suppose \( L \cup K \) is context-free.

Since \( L \) is assumed to be regular, \( L \) is also regular.

Hence, the language \( \overline{L} \) is context-free.

But, what is \( (L \cup K) \cap \overline{L} \)?

You can show that since \( L \cap K = \emptyset \) by assumption,
we have \( (L \cup K) \cap \overline{L} = K \).

\( \therefore \), \( K \) is context-free,
which contradicts the assumption that \( K \) is not context-free.
Problem 7.  
(a) Consider the following context-free grammar $G$:  
$$ S \to XS_0 | S_1 | \$$  
$$ X \to 0 | 1 $$  
Can see that $G$ generates the given language.  
(b) The given language $L = \{ a^m b^n c^k \mid a^m \leq b^n \leq c^k \}$  
is not context-free.  
We may consider the language $L \cap a^* b^* c^* = \{ a^m b^n c^k \mid 0 \leq m \leq n \leq k \}$ and use Pumping Lemma to show that the resulting language is not context-free.  
Here, we apply Pumping Lemma directly on $L$.  
Suppose that $L$ were context-free, let $n$ be the Pumping Lemma constant.  
Consider $z = a^n b^n c^n \in L$ with $|z| = 3n \geq n$.  
Consider all possible $u, v, w, x, y \in \{a, b, c\}^*$  
such that $z = u v w x y$ and $|v x| \leq n$, $|v x| > 1$.  

Since $|v x| \leq n$, it is not possible that the string $v x$ contains both the symbols $a$ and the symbol $c$, as these symbols are separated by $b$ in $w$.  

Case 1: The string $v x$ does not contain the symbol $c$.  
In this case, we may consider $i = 0$.  
As $|v x| \geq 1$, it follows that $a(x)$  
$$ a( u^0v^0w^0x^0y) \geq a( u^0v^0w^0x^0y) $$  
$$ a( u^0v^0w^0x^0y) \geq a( u^0v^0w^0x^0y) $$  

Case 2: The string $v x$ contains the symbol $c$.  
In this case, we may consider $i = 1$.  
As $|v x| \geq 1$, it follows that $c(x)$  
$$ c( u^0v^0w^0x^0y) \geq c( u^0v^0w^0x^0y) $$  
$$ c( u^0v^0w^0x^0y) \geq c( u^0v^0w^0x^0y) $$
Case 2: The string \( vx \) does not contain the symbol \( c \). In this case, we may consider \( i = 2 \).

As \( |vx| \geq 1 \), it follows that either
\[
\#(uv^2wx^2y) > \#(uv^2wx^2y)
\]
\[
\#_c(uv^2wx^2y) > \#_c(uv^2wx^2y)
\]
and therefore \( uv^2wx^2y \notin L \).

Combining both cases, \( i = 2 \), we can conclude that \( L \) is not context-free.
We call $M$ such a Turing Machine. $M = (Q, \Sigma, \Gamma, \delta, q_0, \text{accept}, \text{reject})$ can be the following:

- $Q := \{q_0, q_1, q_2, q_3, q_4, q_f\}$
- $\Sigma := \{0, 1, X, L\}$
- $\Gamma := \{0, 1, X, R\}$
- $\text{accept} := q_f$
- $\text{reject} := q_0$

The transition function $\delta$ is given by:

<table>
<thead>
<tr>
<th>State</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>X</th>
<th>0</th>
<th>1</th>
<th>Y</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_0$</td>
<td>$(q_0, X, R)$</td>
<td>$(q_0, X, R)$</td>
<td>$(q_0, Y, R)$</td>
<td>$(q_0, Y, R)$</td>
<td>$(q_f, X, R)$</td>
<td>$(q_f, Y, R)$</td>
<td>$(q_f, X, R)$</td>
<td>$(q_f, Y, R)$</td>
<td>$(q_f, X, R)$</td>
<td>$(q_f, Y, R)$</td>
</tr>
<tr>
<td>$q_1$</td>
<td>$(q_1, Y, L)$</td>
<td>$(q_1, Y, L)$</td>
<td>$(q_1, Y, L)$</td>
<td>$(q_1, Y, L)$</td>
<td>$(q_1, Y, L)$</td>
<td>$(q_1, Y, L)$</td>
<td>$(q_1, Y, L)$</td>
<td>$(q_1, Y, L)$</td>
<td>$(q_1, Y, L)$</td>
<td>$(q_1, Y, L)$</td>
</tr>
<tr>
<td>$q_2$</td>
<td>$(q_2, 0, L)$</td>
<td>$(q_2, 0, L)$</td>
<td>$(q_2, 0, L)$</td>
<td>$(q_2, 0, L)$</td>
<td>$(q_2, 0, L)$</td>
<td>$(q_2, 0, L)$</td>
<td>$(q_2, 0, L)$</td>
<td>$(q_2, 0, L)$</td>
<td>$(q_2, 0, L)$</td>
<td>$(q_2, 0, L)$</td>
</tr>
<tr>
<td>$q_4$</td>
<td>$(q_4, 0, R)$</td>
<td>$(q_4, 0, R)$</td>
<td>$(q_4, 0, R)$</td>
<td>$(q_4, 0, R)$</td>
<td>$(q_4, 0, R)$</td>
<td>$(q_4, 0, R)$</td>
<td>$(q_4, 0, R)$</td>
<td>$(q_4, 0, R)$</td>
<td>$(q_4, 0, R)$</td>
<td>$(q_4, 0, R)$</td>
</tr>
<tr>
<td>$q_f$</td>
<td>(accept)</td>
<td>(accept)</td>
<td>(accept)</td>
<td>(accept)</td>
<td>(accept)</td>
<td>(accept)</td>
<td>(accept)</td>
<td>(accept)</td>
<td>(accept)</td>
<td>(accept)</td>
</tr>
</tbody>
</table>

The state diagram of $M$ is reported below.
SOLUTION: It is easy to see that we can simulate any DFA on a Turing machine with stay put instead of left. The only non-trivial modification is to add transitions from state in $F$ to $q_{\text{accept}}$ upon reading a blank, and from states outside $F$ to $q_{\text{reject}}$ upon reading a blank.

Next, we start with a Turing machine $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ with stay put instead of left, and show how we can construct a DFA $(Q', \Sigma', \delta', q'_0, F)$ that recognizes the same language. The intuition here is that $M$ cannot move left and cannot read anything it has written on the tape as soon as it moves right, and therefore it has essentially only one-way access to its input, much like a DFA.

First, we modify $M$ as follows; note that these changes do not affect the language it recognizes.

- Add a new symbol so that $M$ never writes blanks on the tape; instead, $M$ writes the new symbol when it's going to write blanks, and we extend the transition function so that upon reading this new symbol, it behaves as though it read a blank.
- When $M$ transitions into $q_{\text{reject}}$ or $q_{\text{accept}}$, the reading head moves right (and never stays put).

Set $Q' = Q$, $\Sigma' = \Sigma$, $q'_0 = q_0$, and consider the transition function:

$$\delta'(q, \sigma) = \begin{cases} 
q, & \text{if } q \in \{q_{\text{accept}}, q_{\text{reject}}\} \\
n_{\text{reject}}, & \text{if } M \text{ starting at state } q \text{ and reading } \sigma \text{ keeps staying put.} \\
q', & \text{where } q' \text{ is the state the } M \text{ enters when it first moves right upon starting at state } q \text{ and reading } \sigma.
\end{cases}$$

(for $q \in Q$ and $\sigma \in \Sigma$). Observe that there are finitely many state-alphabet pairs, $M$ either ends up either staying put and looping, or eventually moves right, and thus $\delta'$ is well-defined. Finally, we define $F$ to be the set containing $q_{\text{accept}}$ and all states $q \in Q, q \neq q_{\text{accept}}, q_{\text{reject}}$ such that $M$ starting at $q$ and reading blanks, eventually enters $q_{\text{accept}}$. 