2.31 Assume $B$ is regular and get its pumping length $p$ from the pumping lemma. Let $s = 0^p1^p0^p$. Because $s \in B$, it can be split $s = uvxyz$ satisfying the conditions of the lemma. We consider several cases.

i) If both $v$ and $y$ contain only 0's (or only 1's), then $uv^2xy^2z$ has unequal numbers of 0s and 1s and hence won't be in $B$.

ii) If $v$ contains only 0s and $y$ contains only 1s, or vice versa, then $uv^2xy^2z$ isn't a palindrome and hence won't be in $B$.

iii) If both $v$ and $y$ contain both 0s and 1s, condition 3 is violated so this case cannot occur.

iv) If one of $v$ and $y$ contain both 0s and 1s, then $uv^2xy^2z$ isn't a palindrome and hence won't be in $B$.

Hence $s$ cannot be pumped and contradiction is reached. Therefore $B$ isn't regular.

2.32 Assume $C$ is regular and get its pumping length $p$ from the pumping lemma. Let $s = 1^p3^p2^p4^p$. Because $s \in C$, it can be split $s = uvxyz$ satisfying the conditions of the lemma. By condition 3, $vxy$ cannot contain both 1s and 2s, and cannot contain both 3s and 4s. Hence $uv^2xy^2z$ doesn't have equal number of 1s and 2s or of 3s and 4s, and therefore won't be a member of $C$, so $s$ cannot be pumped and contradiction is reached. Therefore $C$ isn't regular.

3.8 b. "On input string $w$:
1. Scan the tape and mark the first 0 which has not been marked. If there is no unmarked 0, go to stage 5.
2. Continue scanning and mark the next unmarked 0. If there is not any on the tape, reject. Otherwise, move the head to the front of the tape.
3. Scan the tape and mark the first 1 which has not been marked. If there is no unmarked 1, reject.
4. Move the head to the front of the tape and repeat stage 1.
5. Move the head to the front of the tape. Scan the tape for any unmarked 1s. If none, accept. Otherwise, reject."

c. "On input string $w$:
1. Scan the tape and mark the first 0 which has not been marked. If there is no unmarked 0, go to stage 5.
2. Continue scanning and mark the next unmarked 0. If there is not any on the tape, accept. Otherwise, move the head to the front of the tape.
3. Scan the tape and mark the first 1 which has not been marked. If there is no unmarked 1, accept.
4. Move the head to the front of the tape and repeat stage 1.
5. Move the head to the front of the tape. Scan the tape for any unmarked 1s. If none, reject. Otherwise, accept."
3.15 b. For any two Turing-recognizable languages \(L_1\) and \(L_2\), let \(M_1\) and \(M_2\) be the TMs that recognize them. We construct a NTM \(M'\) that recognizes the concatenation of \(L_1\) and \(L_2\):

"On input \(w\):
1. Nondeterministically cut \(w\) into two parts \(w = w_1w_2\).
2. Run \(M_1\) on \(w_1\). If it halts and rejects, reject.
3. Run \(M_2\) on \(w_2\). If it accepts, accept. If it halts and rejects, reject."

If there is a way to cut \(w\) into two substrings such \(M_1\) accepts the first part and \(M_2\) accepts the second part, \(w\) belongs to the concatenation of \(L_1\) and \(L_2\) and \(M'\) will accept \(w\) after a finite number of steps.
Problem 3 (a) \[ L = \{ a^i b^j c^k \mid (i,j,k \geq 0) \text{ and } i \geq j \lor i \geq k \} \]  

Obvious typographical error.

Can verify that \[ L = \{ a^i b^j c^k \mid (i,j,k \geq 0) \text{ and } i \geq j \} \cup \{ a^i b^j c^k \mid (i,j,k \geq 0) \text{ and } i \geq k \}. \]

Each of the component languages is easily seen to be context-free (CFLs).

Hence \( L \) is context-free.

What is \( \overline{L} = \{ a, b, c^* \setminus L \} \)?

We can see that \[ \overline{L} = \{ a^i b^j c^k \mid i \leq j \text{ and } i \leq k \}. \]

We show that \( \overline{L} \) is not context-free by using closure properties of context-free and pumping lemma.

Suppose that \( \overline{L} \) were context-free. Then:

\[ \overline{L} \cap a^* b^* c^* = \left( a^* b^* c^* \cup \{ a^i b^j c^k \mid i \leq j \text{ and } i \leq k \} \right) \cap a^* b^* c^* \]

would be context-free.

The resulting language is \( \{ a^i b^j c^k \mid i \leq j \text{ and } i \leq k \} \) which would be context-free can be shown to be non-context-free by using pumping lemma.

Suppose that \( \{ a^i b^j c^k \mid i \leq j \text{ and } i \leq k \} \) were context-free. Let \( n \) be the pumping lemma constant.

Consider \( z = a^n b^n c^{n+1} \in \{ a^i b^j c^k \mid i \leq j \text{ and } i \leq k \} \) with \( |z| = n + 2(n+1) = 3n + 2 \)
Then, a case analysis (see lecture notes examples, this homework).

Can show a contradiction.

Homework 3: let \( a \leq y \) \( a \leq a \) is not context-free, so \( T \) cannot be context-free.

1b) Similar to part (a).

Problem 4.

(a) True.
Since any finite language is regular, \( F \) is regular.
\( \overline{F} \) is regular.
Now, \( L - F = L \cap \overline{F} \) is context-free.

(b) True.
Suppose that \( L - F \) were context-free.
Then, we notice that:
\( L \cap \overline{F} \) is finite — since \( F \) is a finite language.
so \( L \cap \overline{F} \) is regular, hence context-free.
Now, \( (L - F) \cup (L \cap \overline{F}) \) is context-free, why? Should be regular.
This contradicts our assumption that \( L \) is not context-free.
(c) True.

Suppose that $L UF$ were context-free.

Then, we notice that:

$$F - L \text{ is finite} \quad \text{since } F \text{ is finite}$$

Now, consider

$$\left( L UF \right) - \left( F - L \right) = L$$

Supposed to be context-free

finite

Should be context-free

by part (a).

This contradicts to the assumption

that $L$ is not context-free.

Problem 5.

Idea: Let $M_i$ be a PDA accepting $L_i$ for $i = 1, 2$.

We construct a PDA $M$ that accepts $L_1 \cup L_2$.

The PDA $M$ simulates $M_i$ on the first part of the

input: pushing every symbol it reads on the stack

until it guesses (nondeterministically) that it

has reached the "middle" of the input.

After that, $M$ simulates $M_{2 - i}$ on the remaining part of

the input step: popping the stack for every symbol it reads.

If the stack is empty at the end of the input and

both $M_i$ and $M_{2 - i}$ accept, the PDA $M$ accepts.

If something goes wrong, for example, popping when the stack is

empty or getting to the end of the input prematurely, the PDA $M$

rejects on that branch of the computation.

Can you construct the transition function explicitly?