CMPS 130
Spring 2016
Homework Assignment 8 Solutions
Problems are from Martin 4th edition.

Chapter 4 (p.154): 26ab, 27, 28, 29d, 55
Chapter 5 (p.196): 1ab, 4a

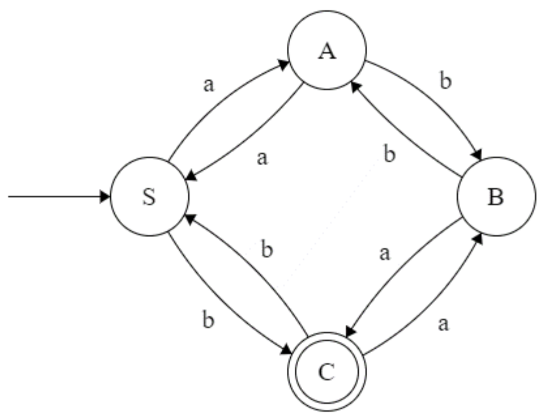
1. Problem 4.26ab
   In each part, draw an NFA (which might be a DFA) accepting the language generated by the CFG having the given productions.

   a. $\begin{cases} S \to aA \mid bC \\ A \to aS \mid bB \\ B \to aC \mid bA \\ C \to aB \mid bS \mid \lambda \end{cases}$

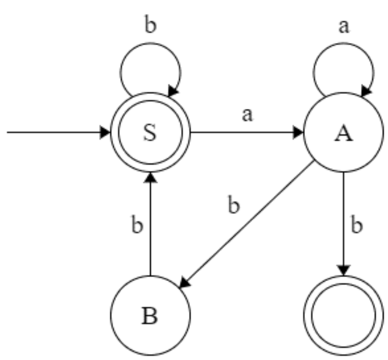
   b. $\begin{cases} S \to bS \mid aA \mid \lambda \\ A \to aA \mid bB \mid b \\ B \to bS \end{cases}$

   Solution:

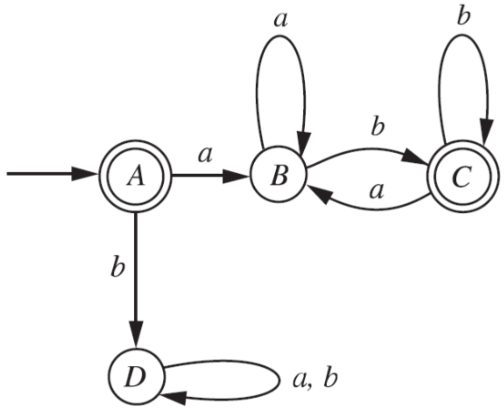
   a.



   b.



2. Problem 4.27
   Find a regular grammar generating the language $L(M)$, where M is the FA shown in Figure 4.33.



Solution:
In the following regular grammar, the start variable is A.

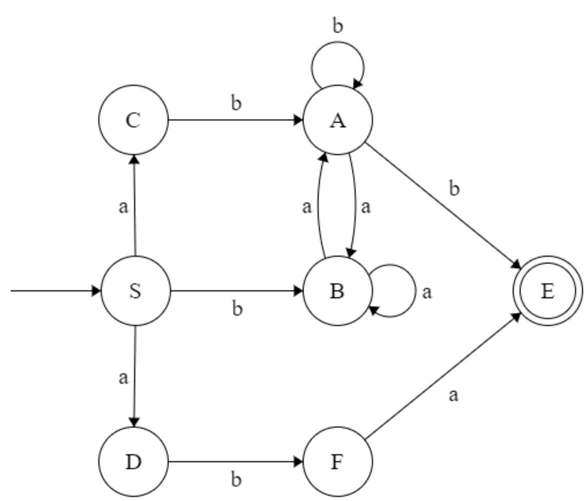
$$\begin{cases} A \to aB \mid bD \mid \lambda \\ B \to aB \mid bC \\ C \to bC \mid aB \mid \lambda \\ D \to aD \mid bD \end{cases}$$

3. Problem 4.28
   Draw an NFA accepting the language generated by the grammar with productions

$$\begin{cases} S \to abA \mid bB \mid aba \\ A \to b \mid aB \mid bA \\ B \to aB \mid aA \end{cases}$$

Solution:
The given grammar is not regular, but it is close to regular. In particular, the right hand side of each production contains at most one variable, and each such variable is the rightmost symbol. In such a case, the grammar can easily be converted to a regular grammar by inserting some new variables. We then turn the grammar into an NFA.

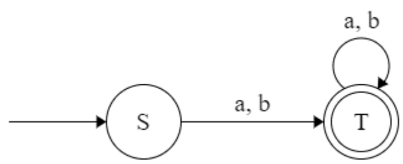
$$\begin{cases} S \to aC \mid bB \mid aD \\ A \to bE \mid aB \mid bA \\ B \to aB \mid aA \\ C \to bA \\ D \to bF \\ E \to \lambda \\ F \to aE \end{cases}$$



4. Problem 4.29d
   Each of the following grammars, though not regular, generates a regular language. In each case, find a regular grammar generating the language.

$$\text{d.}\quad \begin{cases} S \to AB \\ A \to aAa \mid bAb \mid a \mid b \\ B \to aB \mid bB \mid \lambda \end{cases}$$

Solution:
We recognize the productions for A as generating odd length palindromes, and those for B as generating arbitrary strings in $\{a, b\}^*$. Therefor S generates all strings of the form: an odd length palindrome concatenated with an arbitrary string. But all strings $x \in \{a, b\}^*$ with $|x| \geq 1$ are of this form, since the first character in x is itself an odd length palindrome. Hence the above grammar generates the language $L = \{ x \in \{a, b\}^* \mid |x| \geq 1 \}$, which is regular since it is accepted by the following DFA.



We now read off the following regular grammar for L.

$$\begin{cases} S \to aT \mid bT \\ T \to aT \mid bT \mid \lambda \end{cases}$$

5. Problem 4.55
   For alphabets Σ_1 and Σ_2, a *homomorphism* $f\colon \Sigma_1^* \to \Sigma_2^*$ was defined in Exercise 3.53 by the requirement that $f(xy) = f(x)f(y)$ for all strings $x, y \in \Sigma_1^*$. Show that if $f\colon \Sigma_1^* \to \Sigma_2^*$ is a homomorphism and $L \subseteq \Sigma_1^*$ is a context-free language, then $f(L) \subseteq \Sigma_2^*$ is also a CFL.

Solution:
Since $L \subseteq \Sigma_1^*$ is a CFL, there exists a CFG $G_1 = (V_1, \Sigma_1, S_1, P_1)$ generating it. We define a CFG on Σ_2 by $G_2 = (V_2, \Sigma_2, S_2, P_2)$, where $V_2 = V_1$ and $S_2 = S_1$. Before we define the set of productions P_2, we first extend the homomorphism $f\colon \Sigma_1^* \to \Sigma_2^*$ to a homomorphism $\overline{f}\colon (\Sigma_1 \cup V_1)^* \to (\Sigma_2 \cup V_2)^*$ as follows.

$$\overline{f}(\sigma) = f(\sigma) \text{ for all } \sigma \in \Sigma_1$$
$$\overline{f}(A) = A \text{ for all } A \in V_1$$

$\overline{f}(\alpha)$ is now uniquely defined for all $\alpha \in (\Sigma_1 \cup V_1)^*$ by the requirement that it be a homomorphism. Specifically, if $\alpha = \alpha_1 \alpha_2 \cdots \alpha_n$ where each $\alpha_i \in \Sigma_1 \cup V_1$ $(1 \leq i \leq n)$, we define

$$\overline{f}(\alpha) = \overline{f}(\alpha_1)\overline{f}(\alpha_2) \cdots \overline{f}(\alpha_n)$$

We define P_2 as follows: for each production $A \to \alpha$ in P_1, we create a corresponding production $\overline{f}(A) \to \overline{f}(\alpha)$ and place it in P_2. The CFG $G_2 = (V_2, \Sigma_2, S_2, P_2)$ is now fully defined.

Claim: $f(L) = L(G_2)$, and hence $f(L)$ is a CFL as required.
We show that $f(L) \subseteq L(G_2)$ and $f(L) \supseteq L(G_2)$, establishing the claim.

Proof that $f(L) \subseteq L(G_2)$: Let $y \in f(L)$. Then $y = f(x)$ for some $x \in L$. Since L is generated by the grammar G_1, there exists a derivation: $S_1 \Rightarrow_{G_1}^{*} x$. Apply the homomorphism $\overline{f}$ to every string (first, last, intermediate) in this derivation to obtain: $\overline{f}(S_1) \Rightarrow_{G_2}^{*} \overline{f}(x)$. This is a valid derivation in G_2 because of the way we defined P_2. Since $\overline{f}(S_1) = S_1 = S_2$ and $\overline{f}(x) = f(x) = y$, we have a derivation $S_2 \Rightarrow_{G_2}^{*} y$, showing that $y \in L(G_2)$.

Proof that $f(L) \supseteq L(G_2)$: Let $y \in L(G_2)$. Then there exists a derivation $S_2 \Rightarrow_{G_2}^{*} y$. We must show that there exists an $x \in L$ for which $f(x) = y$, and hence $y \in f(L)$. By our definition of P_2, each of the productions used in this derivation is of the form $\overline{f}(A) \to \overline{f}(\alpha)$ for some production $A \to \alpha$ in P_1. The above derivation can therefore be written as $\overline{f}(\beta) \Rightarrow_{G_2}^{*} \overline{f}(\xi)$ for some $\beta, \xi \in (\Sigma_1 \cup V_1)^*$ with $\overline{f}(\beta) = S_2$ and $\overline{f}(\xi) = y$. It must be that $\beta = S_1$ since $\overline{f}(S_1) = S_1 = S_2$. Also since y consists of terminal symbols only, the same must be true of ξ since $\overline{f}$, as we've defined it, takes variables to variables and terminals to terminals. Thus $\xi \in \Sigma_1^*$. Accordingly we rename ξ to x. We now have $x \in \Sigma_1^*$ for which $f(x) = \overline{f}(x) = y$. It remains only to show that $x \in L$. Take the derivation $\overline{f}(\beta) \Rightarrow_{G_2}^{*} \overline{f}(\xi)$ and remove $\overline{f}$ from every string (first, last, intermediate) to create a sequence of valid derivation steps in G_1. This is possible because of the way we defined P_2. We now have a derivation $\beta \Rightarrow_{G_1}^{*} \xi$, which is $S_1 \Rightarrow_{G_1}^{*} x$. Since $L = L(G_1)$, this shows that $x \in L$, as required.

6. Problem 5.1ab
   a. For the PDA in Table 5.4 (accepting AnBn), trace the sequence of moves made for the input strings *ab*, *aab*, and *abb*. (Note: q_3 is the sole accepting state.)

Move Number	State	Input	Stack Symbol	Move(s)
1	q_0	a	Z_0	(q_1, aZ_0)
2	q_1	a	a	(q_1, aa)
3	q_1	b	a	(q_2, λ)
4	q_2	b	a	(q_2, λ)
5	q_2	λ	Z_0	(q_3, Z_0)
	all other combinations			$\emptyset$

Solution:

ab: $(q_0, ab, Z_0) \vdash (q_1, b, aZ_0)$
$\vdash (q_2, \lambda, Z_0)$
$\vdash (q_3, \lambda, Z_0)$ accepting configuration

aab: $(q_0, aab, Z_0) \vdash (q_1, ab, aZ_0)$
$\vdash (q_1, b, aaZ_0)$
$\vdash (q_2, \lambda, aZ_0)$ non-accepting configuration

abb: $(q_0, abb, Z_0) \vdash (q_1, bb, aZ_0)$
$\vdash (q_2, b, Z_0)$ die without consuming input string

b. For the PDA in Table 5.6 (accepting SimplePal), trace the sequence of moves made for the input strings *bacab* and *baca*. (Note: q_2 is the sole accepting state.)

Move Number	State	Input	Stack Symbol	Move(s)
1	q_0	a	Z_0	(q_0, aZ_0)
2	q_0	b	Z_0	(q_0, bZ_0)
3	q_0	a	a	(q_0, aa)
4	q_0	b	a	(q_0, ba)
5	q_0	a	b	(q_0, ab)
6	q_0	b	b	(q_0, bb)
7	q_0	c	Z_0	(q_1, Z_0)
8	q_0	c	a	(q_1, a)
9	q_0	c	b	(q_1, b)
10	q_1	a	a	(q_1, λ)
11	q_1	b	b	(q_1, λ)
12	q_1	λ	Z_0	(q_2, Z_0)
	all other combinations			$\emptyset$

Solution:

bacab: $(q_0, bacab, Z_0) \vdash (q_0, acab, bZ_0)$
$\vdash (q_0, cab, abZ_0)$
$\vdash (q_1, ab, abZ_0)$
$\vdash (q_1, b, bZ_0)$
$\vdash (q_1, \lambda, Z_0)$
$\vdash (q_2, \lambda, Z_0)$ accepting configuration

baca: $(q_0, baca, Z_0) \vdash (q_0, aca, bZ_0)$
$\vdash (q_0, ca, abZ_0)$
$\vdash (q_1, a, abZ_0)$
$\vdash (q_1, \lambda, bZ_0)$ non-accepting configuration

7. Problem 5.4a
Consider the PDA in Table 5.8 (accepting Pal), and for each of the following languages over $\{a, b\}$, modify it to obtain a PDA accepting the language.

Move Number	State	Input	Stack Symbol	Move(s)
1	q_0	a	Z_0	$(q_0, aZ_0), (q_1, Z_0)$
2	q_0	a	a	$(q_0, aa), (q_1, a)$
3	q_0	a	b	$(q_0, ab), (q_1, b)$
4	q_0	b	Z_0	$(q_0, bZ_0), (q_1, Z_0)$
5	q_0	b	a	$(q_0, ba), (q_1, a)$
6	q_0	b	b	$(q_0, bb), (q_1, b)$
7	q_0	λ	Z_0	(q_1, Z_0)
8	q_0	λ	a	(q_1, a)
9	q_0	λ	b	(q_1, b)
10	q_1	a	a	(q_1, λ)
11	q_1	b	b	(q_1, λ)
12	q_1	λ	Z_0	(q_2, Z_0)
	all other combinations			$\emptyset$

a. The language of even-length palindromes.

Solution:
Remove the second pair from lines 1-6 in the above table to obtain:

Move Number	State	Input	Stack Symbol	Move(s)
1	q_0	a	Z_0	(q_0, aZ_0)
2	q_0	a	a	(q_0, aa)
3	q_0	a	b	(q_0, ab)
4	q_0	b	Z_0	(q_0, bZ_0)
5	q_0	b	a	(q_0, ba)
6	q_0	b	b	(q_0, bb)
7	q_0	λ	Z_0	(q_1, Z_0)
8	q_0	λ	a	(q_1, a)
9	q_0	λ	b	(q_1, b)
10	q_1	a	a	(q_1, λ)
11	q_1	b	b	(q_1, λ)
12	q_1	λ	Z_0	(q_2, Z_0)
	all other combinations			$\emptyset$