NOTE: For questions 2 and 3, you will receive 2 marks for a BLANK answer to part (a), 1 mark for a BLANK answer to part (b) and 0.5 marks for a BLANK answer to part (c).

Name: ID:

---

1. (10 MARKS) **Note**: I put in explanations. You are not required to do this.

   (a) If L_1 is regular and L_2 is any language such that $L_2 \subseteq L_1$, then L_2 is regular. T F

   False: Take $L_2 = \emptyset$, $L_1 = \{0^n 1^n \mid n \geq 0\}$

   (b) If L_1 is regular and L_2 is context-free, then $L_1 \cap L_2$ is regular. T F

   False: Take $L_1 = \Sigma^*$, $L_2 = \{0^n 1^n \mid n \geq 0\}$

   (c) If M is a NFA with n states, then any DFA accepting $L(M)$ must have at least 2^n states. T F

   False: E.g., consider the case where M is *already* a DFA (every DFA is an NFA).

   (d) Every CNF grammar is unambiguous T F

   False: See the last question on problem set 2.

   (e) Which ONE of the following is an *unambiguous* grammar for (possibly empty) strings of balanced parentheses

   i. $S \to (S) \mid SS$
   ii. $S \to \epsilon \mid (S) \mid SS$
   iii. $S \to \epsilon \mid (S)S$
   iv. $S \to () \mid \epsilon \mid (S) \mid SS$
   v. None of the above

   (i), (ii) and (iv) are ambiguous. Why does (iii) work? Clearly every string generated by (iii) is balanced. Now let w be a balanced string of parentheses. The case $w = \epsilon$ follows immediately. Now suppose $w = (u)$. It must be the case that u is balanced, so a derivation that starts $S \Rightarrow (S) \Rightarrow \dots$ will work. Otherwise, since w must start with a (, there must be a *matching*) somewhere before the last symbol of w. This means that $w = (u)v$ where u and v are balanced, so a derivation that starts $S \Rightarrow (S) \Rightarrow \dots$ will work. We can use a simliar case analysis (plus induction on $|w)$ to show that (iii) is unambiguous.

2. (a) (5 MARKS) Use the pumping lemma to prove that

$$L = \{w \in \{(,)\}^* \mid w \text{ is balanced}\}$$

is not regular

Given n take, $x = (^n)^n$. Clearly $x \in L$ and $|x| \geq n$. Now, given u, v, w with (i) $uvw = x$, (ii) $|uv| \leq n$ and (iii) $v \neq \epsilon$, take $i = 0$. Then $uv^iw = uv^0zw \notin L$. This is because conditions (i) and (ii) imply that uv is an initial segment of the leading $(^n$ of x. Condition (iii) implies that $v = (^k$, $k \geq 1$. So $uv^0w = (^{n-k})^n$. Since $k \neq 0$, this means that $uv^0w \notin L$.

(b) (3 MARKS) For a string $w = w_1 \ldots w_k$, $k \geq 0$, define $w^R = w_k w_{k-1} \ldots w_1$. For any language L, define $L^R = \{w^R \mid w \in L\}$. Suppose L is regular. Is L^R always regular? If your answer is "yes", give a construction that proves this is the case. If your answer is "no", give a regular language L and prove that L^R is not regular.

Suppose $L = L(M)$ where $M = (Q, \Sigma, \delta, q_0, F)$. Define $M' = (Q \cup \{q_0'\}, \Sigma, \delta', q_0', \{q_0\})$, where δ' is defined as follows:

- $\delta'(q_0', \epsilon) = r$ for $r \in F$
- $\delta'(q, a) = p$ for every p, q, a such that $\delta(p, a) = q$.

It is easy to verify that $L(M') = L^R$.

(c) (2 Marks) Let

$$L = \{w \in \{0,1\}^* \mid w \text{ contains the same number of occurrences of the substrings 01 and 10}\}$$

For example $101 \in L$, but $1010 \notin L$. Prove that L is regular by giving a FA or regular expression that defines it, or use the pumping lemma or closure properties of regular languages to prove that it is not regular.

Consider any string w of this form. Assume without loss of generality that w starts with a 0. It is not hard to see that w must have the form $0\ldots01\ldots10\ldots0\ldots1\ldots10\ldots0$, i.e., each time we go from 0 to 1 we must eventually go back from 1 to 0. More specifically, in this case w must have the form $0(0\cup1)^*0$. So $L = L(R)$ where

$$R = 0(0\cup1)^*0 \cup 1(0\cup1)^*1 \cup \epsilon$$

i.e., any string that starts and ends with the same symbol will do.

3. (a) (5 Marks) Convert the grammar $S \to \epsilon \mid (S)S$ to CNF. Give each step of the CNF construction. If the step does not apply to this grammar, explain why.

Step 1: Introduce a new start symbol.

$$\begin{aligned} S_0 &\to S \mid \epsilon \\ S &\to \epsilon \mid (S)S \end{aligned}$$

Step 2: Elminate ϵ's (except in $S_0 \to \epsilon$.)

$$\begin{aligned} S_0 &\to S \mid \epsilon \\ S &\to (S)S \mid ()S \mid (S) \mid () \end{aligned}$$

Step 3: Elminate unit productions

$$\begin{aligned} S_0 &\to \epsilon \mid (S)S \mid ()S \mid (S) \mid () \\ S &\to (S)S \mid ()S \mid (S) \mid () \end{aligned}$$

Step 3: All RHS have at most 2 symbols

$$\begin{aligned} S_0 &\to \epsilon \mid (S_1 \mid (S_2 \mid (S_3 \mid () \\ S &\to (S_1 \mid (S_2 \mid (S_3 \mid () \\ S_1 &\to SS_2 \\ S_2 &\to)S \\ S_3 &\to S) \end{aligned}$$

Step 4: Replace RHS terminals by variables

$$\begin{aligned} S_0 &\to \epsilon \mid OS_1 \mid OS_2 \mid OS_3 \mid OC \\ S &\to OS_1 \mid OS_2 \mid OS_3 \mid OC \\ S_1 &\to SS_2 \\ S_2 &\to CS \\ S_3 &\to SC \\ O &\to (\\ C &\to) \end{aligned}$$

(b) (3 Marks) Give a grammar for the language

$$\{w \in \{0,1\}^* \mid w \text{ contains the same number of 0's and 1's.}\}$$

.

$S \to 0S1 \mid 1S0 \mid SS \mid \epsilon$

(Note: Getting an uambigious grammar is possible, but *much* too hard for an exam question)

(c) (2 Marks) Let G be a CNF grammar, and $w = w_1 \dots w_k \in L(G)$. How many steps must there be in the derivation of w? Give a justification for your answer.

Since terminals appear only in rules of the form $A \to a$, we can assume without loss of generality that the derivation has the form $S \Rightarrow^* A_1 A_2 \dots A_k \Rightarrow^k w_1 w_2 \dots w_k$, where the A_i's are all variables, i.e., the last k steps of the derivation are used to obtain each of the symbols in the string, one-at-a-time. To get to $A_1 A_2 \dots A_k$ from S, we can only use rules of the form $A \to BC$. What happens when we apply such a rule? Let $z \in V^*$ be the string in the derivation before we apply this step (how do we know that z only contains variables?) So $z = xAy$ for some $x, y \in V^*$. After we apply $A \to BC$, we will have $xABy$ as the derived string. In particular, the string *gets longer by one variable*. So starting from S, it will take $k - 1$ steps to get to $A_1 A_2 \dots A_k$. So in total there are $k - 1$ steps to get to $A_1 A_2 \dots A_k$ and k more steps to get to $w_1 w_2 \dots w_k$. So there are $2k - 1$ steps in total.