Part B

1. Consider the set of all strings of balanced parentheses of two types, round and square. An example of where these strings come from is as follows. If we take expressions in **C**, which use round parentheses for grouping and for arguments of function calls, and use square brackets for array indexes, and drop out everything but the parentheses, we get all strings of balanced parentheses of these two types. For example,

   `f(a[i]*(b[i][j],c[g(x)]),d[i])`

   becomes the balanced-parenthesis string `([]([][][()])[])`. Design a grammar for all and only the strings of round and square parentheses that are balanced.

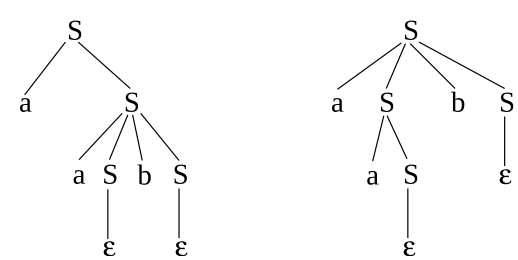
   Solution: $S \longrightarrow \varepsilon|SS|[S]|(S)$

2. Consider the following grammar:

   $$G = (\{S\}, \{a, b\}, S, \{S \longrightarrow aS|aSbS|\varepsilon\})$$

   (a) Show that G is ambiguous; that is, show that there is a string in $L(G)$ which has two *distinct* parse trees.

   Solution: The string aab has the following two parse trees:

   (b) Show that $L(G)$ is the set of all (and only) strings of a's and b's such that every prefix has at least as many a's as b's.

   Solution: Let "w has the property" mean that every prefix of the string w has at least as many a's as b's. Consider any G derivation $S \Rightarrow w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_n$ (each w_i, $i \leq n$, is a string over the alphabet $\{a, b, S\}$). We show by induction on n that each w_i, $1 \leq i \leq n$, has the property. BC: $n = 1$, so w_1 is obtained in one step, so w_1 must be $aS, aSbS$, or ε, and all three have the property. IS: Consider $w_n \Rightarrow w_{n+1}$, where w_n has the property. w_{n+1} is obtained by one of the three rules. In each case it is easy to see that the property is preserved. Since any $w \in L(G)$ is the result of one such derivation, *it* must have the property.

   Suppose on the other hand that w has the property. Then we show by induction on the length of w that $w \in L(G)$ by showing that it has a G derivation. BC: $|w| = 0$, so $w = \varepsilon$, so $S \Rightarrow \varepsilon$. IS: w has the property, $|w| = n + 1$, and the claim holds for all z with the prefix property such that $|z| \leq n$. We know that $w = ax$ (bec. w has the property by assumption). There are now two possibilities for x: it also has the property, so $S \Rightarrow x$ (by IH) and we are done, or x does not have the property, but since w has it, it follows that $x = y_1 b y_2$ where y_1, y_2 both have the property. Thus, w can be derived as follows: $S \Rightarrow aSbS \stackrel{*}{\Rightarrow} ay_1bS \stackrel{*}{\Rightarrow} ay_1by_2 = w$.

3. Let Σ be an alphabet. A **regular expression** over Σ is defined inductively as follows: any $a \in \Sigma$ is a regular expression, and so are ε and $\emptyset$. If R_1, R_2 are regular expressions, then so are $(R_1), R_1^*, R_1R_2, R_1 + R_2$.

   The semantics of regular expressions are defined inductively as follows: if $a \in \Sigma$, then $L(a) = \{a\}$. Also, $L(\varepsilon) = \{\varepsilon\}$, and $L(\emptyset) = \emptyset$. Furthermore, $L((R_1)) = L(R_1)$, $L(R_1^*) = L(R_1)^*$, $L(R_1R_2) = \{xy|x \in L(R_1) \text{ and } y \in L(R_2)\}$, and $L(R_1 + R_2) = L(R_1) \cup L(R_2)$.

   Show that a language L can be defined by a grammar of type 2 if it can be defined by a regular expression.

   That is, if L is a language ($L \subseteq \Sigma^*$, for some alphabet Σ), then: there exists a grammar G such that $L = L(G)$ if there exists a regular expression R such that $L = L(R)$.

   For example, the language $L = \{\varepsilon, ab, abab, ababab, abababab, \ldots\}$, is given by the regular expression $R = (ab)^*$, and by the type 2 grammar $G = (\{S\}, \{a, b\}, S, \{S \longrightarrow \varepsilon|abS\})$.

   Solution: We show by structural induction that any regular expression R has a grammar G of type 2 such that $L(R) = L(G)$. BC: R is $a, \varepsilon, \emptyset$. All three cases are trivial. IS: consider R^*, were R has a grammar G of type 2. Let G' contain the rules of G with the extra rule $S \longrightarrow SS|\varepsilon$. Then $L(G') = L(R^*)$. Consider R_1, R_2 where G_1, G_2 are the corresponding grammars of type 2. Let G have the rules of G_1, G_2 (we assume that the variable sets of G_1, G_2 are distinct so that there are no conflicts, and S_1, S_2 are their respective starting variables), and also the rules $S \longrightarrow S_1S_2$, where S is the start variables of G. Then $L(G) = L(R_1R_2)$. If we let $S \longrightarrow S_1|S_2$ instead, then $L(G) = L(R_1 + R_2)$.

4. Prove the Unique Readability Theorem for first order terms. That is, show that every $\mathcal{L}$-term can be uniquely parsed, given the language $\mathcal{L}$.

   Solution: As in the case of propositional formulas the URT follows from the following lemma.

   bf Lemma: If we assign weights to term symbols as follows: $n - 1$ to a function symbol f with arity n, and -1 to each variable, then any well-formed term has weight -1, and any proper initial segment of a well-formed term has weight ≥ 0.

   The proof of the lemma is very simple (and by structural induction, of course). BC: x a variable is term, and its weight is -1. IS: Consider $ft_1t_2 \ldots t_n$, where f has arity n. By IH the weights of $t_1, t_2, \ldots, t_n$ are -1 each, and $(n - 1) + n(-1) = -1$. On the other hand, any proper initial segment of $ft_1t_2 \ldots t_n$ starts with f, with weight $n - 1$, and any proper initial segment of t_i has weight ≥ 0, so any proper initial segment of $ft_1t_2 \ldots t_n$ has weight ≥ 0.

5. Read the article **"Program Development by Stepwise Refinement"** by Niklaus Wirth, posted on the course's web site (Jan 27 announcement).