CS311 Midterm Exam (winter 99)

Student Name (please print):

January 25, 1999

Problem 1. Prove that 3 is a factor of $n^3 - n + 3$ for all $n \geq 0$.

$\heartsuit$ *Solution.* By induction on n.

Basis. $n = 0$, $n^3 - n + 3 = 3$ is divisible by 3.

Hypothesis. 3 is a factor of $k^3 - k + 3$ for all $0 \leq k \leq n$.

Inductive Step. Want to show that 3 is a factor of $(n+1)^3 - (n+1) + 3$. Some elementary arithmetic gives

$$(n+1)^3 - (n+1) + 3 = \cdots = n^3 - n + 3 + 3(n^2 + 3n).$$

Now 3 is a factor of $n^3 - n + 3$ by induction hypothesis, and 3 is obviously a factor of $3(n^2 + 3n)$; therefore 3 is a factor of $n^3 - n + 3 + 3(n^2 + 3n) = (n+1)^3 - (n+1) + 3$.

$\square$

Problem 2. Give a recursive definition of the set of strings over $\{a, b, c\}$ that contains all and only those strings with the number of b's equal to the number of c's.

$\heartsuit$ *Solution.* Let $\mathbf{L}$ be the set of strings over $\{\mathbf{a}, \mathbf{b}, \mathbf{c}\}$ that contains all and only those strings with the number of b's equal to the number of c's. We define it recursively as follows:

Basis. $\lambda, a \in \mathbf{L}$ are the strings of length 0 and 1 in $\mathbf{L}$.

Recursive Step. If $w \in \mathbf{L}$, $length(w) = n$ and $w = xyz$ with $x, y, z \in \{\mathbf{a}, \mathbf{b}, \mathbf{c}\}^*$ (partition w in three arbitrary substrings) then $x\mathbf{a}yz$, $x\mathbf{a}y\mathbf{a}z$, $x\mathbf{b}y\mathbf{c}z$ and $x\mathbf{c}y\mathbf{b}z$ are elements of $\mathbf{L}$ of lengths $n+1$, $n+2$, $n+2$ and $n+2$ respectively.

Closure. A string w is an element of $\mathbf{L}$, $w \in \mathbf{L}$, if and only if w can be obtained by a finite number of recursive steps.

$\square$

Problem 3. Let

$$\begin{aligned} L_1 &= (abba)^* \\ L_2 &= \left((a \cup b)^5\right)^* \end{aligned}$$

Describe the strings that are in the language $L_1 \cap L_2$.

$\heartsuit$ *Solution.* L_1 contains strings of lengths multiple of 4 with the pattern $abbaabba\ldots$, while L_2 contains strings of lengths multiple of 5, with no specific pattern. Therefore $L_1 \cap L_2$ will contain strings of lengths multiple of $4 \times 5 = 20$ and with the pattern $abbaabba\ldots$:

$$L_1 \cap L_2 = \left((abba)^5\right)^*$$

$\square$

Problem 4. Prove the following identity

$$(a \cup b)^* = (a^* \cup ba^*)^*$$

Hint: Table 2.3.1 from page 48.

$\heartsuit$ *Solution.* The solution is based on identities 8 and 12, Table 2.3.1, page 48, textbook.

		Equality		*Identity Applied*
$(a \cup b)^*$	$=$	$(a^* \cup b)^*$	12.	$(u \cup v)^* = (u^* \cup v)^*$ with $u = a,\ v = b.$
	$=$	$(a^* \cup b(a^*)^*)^*$	12.	$(u \cup v)^* = (u \cup vu^*)^*$ with $u = a^*,\ v = b.$
	$=$	$(a^* \cup ba^*)^*$	8.	$(u^*)^* = u$ with $u = a.$

$\square$

Problem 5. Give a regular expression that represents the set of strings over $\{a, b, c\}$ in which the total number of (b's and c's) is three (i.e. $\#b + \#c = 3$).

$\heartsuit$ *Solution.* There are many possible regular expressions. For example

$$a^*((b \cup c)a^*)^3$$

$\square$

Problem 6. Give a regular grammar that generates the set of strings over $\{a, b, c\}$ in which the total number of (b's and c's) is three.

♡ *Solution.*

$$\begin{array}{lcl} S & \to & aS \mid bX \mid cX \\ X & \to & aX \mid bY \mid cY \\ Y & \to & aY \mid bZ \mid cZ \\ Z & \to & aZ \mid \lambda \end{array}$$

□

Problem 7. Let G be the grammar

$$\begin{array}{lcl} S & \to & aSb \mid B \\ B & \to & bB \mid b \end{array}$$

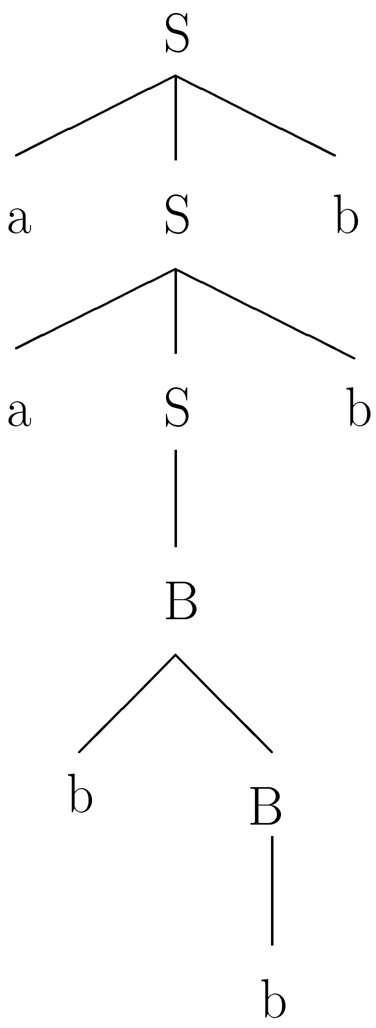
What is $L(G)$?

♡ *Solution.* $L(G) = \{a^n b^m | 0 \leq n < m\}$ □

Problem 8. Let G be the grammar in problem 7. Show the derivation tree for the string

$$aabbbb\ .$$

♡ *Solution.*



□

Problem 9. Let G be a regular grammar. Is the following assertion true or false ?

"For any $w \in L(G)$ with $S \stackrel{n}{\Rightarrow} w$ (i.e. S derives w using exactly n production rule applications) we have $length(w) \le n$.

You need to fully prove your answer.

♡ *Solution.* For this problem we need to undestand how the derivations work in a regular grammar. As explained at page 71/text, the sentential forms of a regular grammar have the form uA or u, where $u \in \Sigma^*$ and $A \in V$ (at most one nonterminal variable, A, can be present; if present it is the rightmost symbol in the string).

We claim that the number of terminal symbols (letters) in a sentential form is less than or equal to the number of derivation steps,

$$\text{For } S \stackrel{n}{\Rightarrow} w \text{ we have } \#letters(w) \leq n.$$

Note that, when counting the number of letters, we ignore the posible rightmost variable. For example, if $u \in \Sigma^*$, $length(u) = k$, and uA is a sentential form of the grammar, then $\#letters(uA) = k$. The length of the sentential form is at most $n+1$ (if a variable is present) and at most n for terminal symbols.

We prove the claim by induction after the number of derivation steps.

Basis. $n = 1$. We check all sentential forms derivable with one production rule application. Depending on the production rule we have: $S \to \lambda$ ($\#letters(\lambda) = 0 \leq 1$), $S \to a$ ($\#letters(a) = 1 \leq 1$), $S \to aA$ ($\#letters(aA) = 1 \leq 1$).

Hypothesis.

$$S \stackrel{k}{\Rightarrow} w \Longrightarrow \#letters(w) \leq k \quad \text{for all } k \leq n.$$

Induction Step.

Let w be a sentential form obtained with $n+1$ production rules applications, $S \stackrel{n+1}{\Rightarrow} w$. The last derivation is possible only if the second to last sentential form contained a variable, i.e.

$$S \stackrel{n}{\Rightarrow} uA \stackrel{1}{\Rightarrow} w.$$

By induction hypothesis we have that

$$\#letters(u) \leq n.$$

Depending on what production rule was applied for the last derivation, we have

Production	*Last derivation*	$w =$	$\#letters(w) =$
$A \to \lambda$	$uA \to u$	u	$\#letters(u) \leq n \leq n+1$
$A \to a$	$uA \to ua$	ua	$\#letters(u) + 1 \leq n+1$
$A \to aB$	$uA \to uaB$	uaB	$\#letters(u) + 1 \leq n+1$

We proved that the number of letters in any sentential form is less than or equal to the number of derivations

$$S \overset{n}{\Rightarrow} w \Longrightarrow \#letters(w) \leq n, \text{ for all } n.$$

In particular, if the sentential form is a string of the language $w \in \Sigma^*$ (i.e. a sentential form without any variable symbols) the number of letters equals its length and is less than or equal to the number of derivation steps

$$length(w) = \#letters(w) \leq n.$$

□