Homework 6: Number Theory, Lists, REGEX, and CFGs!

Due date: Friday, August 2nd at 11:59 PM
If you work with others (and you should!), remember to follow the collaboration policy outlined in the syllabus.
In general, you are graded on your work's clarity and accuracy. Your solution should be clear enough that someone in the class who had not seen the problem before would understand it.
We sometimes describe approximately how long our explanations are. These are intended to help you understand approximately how much detail we expect. You can have longer explanations, but explanations significantly longer than necessary may receive deductions.

1. 737 [12 points]

Show that for all positive integers n, if $7|2^n - 1$ then $3|n$.

Hint: You will need the division theorem, and cases on the remainder.

2. List Me By a Mile [20 points]

Recall the definition of lists of numbers from lecture:

> **Basis Step**: $\texttt{nil} \in$ **List**
>
> **Recursive Step**: for any $a \in \mathbb{Z}$, if $L \in$ **List**, then $a :: L \in$ **List**.

For example, the list $[1, 2, 3]$ would be created recursively from the empty list as $1 :: (2 :: (3 :: \texttt{nil}))$. We will consider "::" to associate to the right, so $1 :: 2 :: 3 :: \texttt{nil}$ means the same thing.

The next two problems use three recursively-defined functions. The first is length, which calculates the length of the list. It is defined recursively as follows:

$$\begin{array}{rcll} \text{length}(\texttt{nil}) & := & 0 & \\ \text{length}(a :: L) & := & 1 + \text{length}(L) & \forall a \in \mathbb{Z}, \forall L \in \textbf{List} \end{array}$$

The second function, concat, which concatenates two lists into a single list, is defined by:

$$\begin{array}{rcll} \text{concat}(\texttt{nil}, R) & := & R & \forall R \in \textbf{List} \\ \text{concat}(a :: L, R) & := & a :: \text{concat}(L, R) & \forall a \in \mathbb{Z}, \forall L, R \in \textbf{List} \end{array}$$

For example, we get $\text{concat}(1 :: 2 :: \texttt{nil}, 3 :: \texttt{nil}) = 1 :: 2 :: 3 :: \texttt{nil}$ from these definitions.

The third function, positives, which returns only the positive numbers in the list, is defined by:

$$\begin{array}{rcll} \text{positives}(\texttt{nil}) & := & \texttt{nil} & \\ \text{positives}(a :: L) & := & \text{positives}(L) & \text{if } a \leq 0 \quad \forall a \in \mathbb{Z}, \forall L \in \textbf{List} \\ \text{positives}(a :: L) & := & a :: \text{positives}(L) & \text{if } a > 0 \quad \forall a \in \mathbb{Z}, \forall L \in \textbf{List} \end{array}$$

For example, from these definitions, we get $\text{positives}(-1 :: 2 :: -3 :: \texttt{nil}) = 2 :: \texttt{nil}$.

(a) Write a chain of equalities, citing the appropriate definitions, showing that

$$\text{concat}(1 :: 2 :: \texttt{nil}, 3 :: \texttt{nil}) = 1 :: 2 :: 3 :: \texttt{nil}$$

(b) Write a chain of equalities, citing the appropriate definitions, showing that

$$\text{positives}(1 :: -2 :: 3 :: \texttt{nil}) = 1 :: 3 :: \texttt{nil}$$

(c) Use structural induction to prove that

$$\forall L \in \textbf{List}\,(\text{length}(\text{positives}(L)) \leq \text{length}(L))$$

3. Constructing Regular Expressions (Online) [20 points]

For each of the following languages, construct a regular expression that matches exactly the given set of strings.

You will submit (and check!) your answers online at https://grin.cs.washington.edu/. Think carefully before entering a submission; you only have 5 guesses. Because these are auto-graded, we will not award partial credit.

NOTE: We highly recommend you take a screenshot of your answer for each problem. Grin may be buggy and we may end up needed to grade these by hands, so please keep a record of your solution.

(a) Strings over $\{0, 1, 2\}$ where every 2 comes (directly) before a 0 or 1.

(b) Strings over $\{a, b, c\}$ where every a is immediately followed by a b or c.

(c) Binary strings that start with 0 and have length congruent to 2 (mod 3).

(d) Binary strings with at least three 1's **or** at most two 0's.

(e) **[EXTRA CREDIT]** Strings over $\{0, 1, 2\}$ that start with 0 and have odd length, where ever occurence of a 2 is followed (directly) after by a 0 or 1 .

4. Context Is Everything. Except for Context-Free Grammars (Online) [20 points]

For each of the following languages, construct a context-free grammar that generates exactly the given language.

You will submit (and check!) your answers online at https://grin.cs.washington.edu/. Think carefully before entering a submission; you only have 5 guesses. Because these are auto-graded, we will not award partial credit.

NOTE: We highly recommend you take a screenshot of your answer for each problem. Grin may be buggy and we may end up needed to grade these by hands, so please keep a record of your solution.

(a) Binary strings that are palindromes, where the first character of the string must be a '1'.

(b) All strings over $\{0, 1, 2\}$ of the form $x2y$, with $x, y \in \{0, 1\}^*$ and y a subsequence of x^R (i.e., it is x^R with some characters possibly removed), where x^R is defined as the reverse of x.

Note: "Subsequence" is not the same thing as "substring".

(c) All strings in the form of $\{a^x b^y a^{2x+y} : x, y \geq 0\}$

(d) Binary strings matching the regular expression $000(1 \cup 01 \cup 001)^*$.

(e) **[EXTRA CREDIT]** The set of all binary strings with an odd number of 0's and an even number of 1's [0 Points].

5. Feedback [1 point]

Please keep track of how much time you spend on this homework and answer the following questions. This can help us calibrate future assignments and future iterations of the course, and can help you identify which areas are most challenging for you.

- How many hours did you spend working on this assignment (excluding any extra credit questions, if applicable)? Report your estimate to the nearest hour.
- Which problem did you spend the most time on?
- Any other feedback for us?