CSE 311: Foundations of Computing (Spring, 2015)

Homework 6 Out: Fri, 15-May. Due: Friday, 22-May, **before class (1 pm)** on **Gradescope**

Additional directions: You should write down carefully argued solutions to the following problems. Your first goal is to be complete and correct. A secondary goal is to keep your answers simple and succinct. The idea is that your solution should be easy to understand for someone who has just seen the problem for the first time. (Re-read your answers with this standard in mind!) You may use any results proved in lecture (without proof). Anything else must be argued rigorously. Make sure you indicate the specific steps of your proofs by induction.

1. Mmmmmm… induction (15 points)

Assume that a chocolate bar consists of n squares arranged in a rectangular pattern. In any one step, you can break the whole bar, or any piece you have formed, along a single horizontal or vertical line.

(a) Determine how many breaks you must make in order to break the bar into n separate squares.

(b) Use strong induction to prove your answer correct.

2. Not this again (20 points)

Suppose the word size on our computer is large enough so that all numbers modulo M fit into a single register. Consider the following pseudo-code:

FastMod(a, k):

If $k == 0$,

return 1;

else if ($k \bmod 2 == 1$)

return ($a *$ FastMod$(a, k-1) \bmod M$);

else

return FastMod($(a * a) \bmod M, k/2$);

Let $R(k)$ be the number of multiplications that occur when FastMod(a, k) is called. Using strong induction, prove that $R(k) \leq 2\log_2(k+1)$.

[You will need to remember that $\log_2(ab) = \log_2(a) + \log_2(b)$.]

3. Reversal of power (20 points)

Let Σ be a finite alphabet. Recall that reversal of strings is defined by:

$$\varepsilon^R = \varepsilon$$
$$(wa)^R = aw^R \text{ for } w \in \Sigma^* \text{ and } a \in \Sigma$$

Let us also define the **concatenation powers** of a string: For an integer $j \geq 0$, and $w \in \Sigma^*$, we define

$$w^0 = \varepsilon$$
$$w^{j+1} = w \cdot w^j$$

Where $\cdot$ denotes concatenation of strings.

Using structural induction, prove that for all $w \in \Sigma^*$ and $i \geq 0$, it holds that $(w^R)^i = \left(w^i\right)^R$.

[Hint: Although not strictly necessary, you might want to first prove that for any two strings $x, y \in \Sigma^*$, you have $(xy)^R = y^R x^R$.]

4. Regular expressions (20 points)

For each of the following, construct regular expressions that match the given set of strings.

(a) The set of all binary strings that have at least two 0's separating every occurrence of a 1.

(b) The set of all binary strings that end with a 0 and have odd length, or start with a 1 and have even length.

(c) The set of all binary strings that don't contain 100.

(d) The set of all binary strings that contain at least three o's and at most two 1's.

5. Proving correctness (20 points)

Consider the following recursive definition of a binary tree:

- $\circ$ is a tree
- If L and R are trees, then $\mathrm{Tree}(x, L, R)$ is a tree.

Consider all the following recursive function insert defined by

$$\mathrm{insert}(\circ, y) = \mathrm{Tree}(y, \circ, \circ)$$

$$\mathrm{insert}(\mathrm{Tree}(x, L, R), y) = \begin{cases} \mathrm{Tree}(x, L, R) & \text{if } y = x \\ \mathrm{Tree}(x, \mathrm{insert}(L, y), R) & \text{if } y < x \\ \mathrm{Tree}\big(x, L, \mathrm{insert}(R, y)\big) & \text{if } y > x \end{cases}$$

Finally, we define a function sorted which tests if a tree is in sorted order:

$$\mathrm{sorted}(\circ) = \mathrm{True}$$

$$\mathrm{sorted}\big(\mathrm{Tree}(x, \circ, \circ)\big) = \mathrm{True}$$

$$\mathrm{sorted}\big(\mathrm{Tree}(x, \circ, \mathrm{Tree}(y, L_1, R_1))\big) = (x < y) \wedge \mathrm{sorted}\big(\mathrm{Tree}(y, L_1, R_1)\big)$$

$$\mathrm{sorted}\big(\mathrm{Tree}(x, \mathrm{Tree}(y, L_0, R_0), R)\big) = (y < x) \wedge \mathrm{sorted}\big(\mathrm{Tree}(y, L_0, R_0)\big) \wedge \mathrm{sorted}\big(\mathrm{Tree}(x, \circ, R)\big)$$

Problem:

Prove that if $\mathrm{sorted}(T)$ is true, then also $\mathrm{sorted}(\mathrm{insert}(T, y))$ is true for all trees T and numbers $y \in \mathbb{N}$.

6. Extra credit: Deleting a node from a tree

Your goal is now to define a recursive function $\text{delete}(T, x)$ that deletes the node with value x from a sorted binary search tree T.

(a) [Olga] Uh-oh! Our notion of "sorted" is too weak to allow for deletion. Give an example showing that deletion can fail on a "sorted" tree.

(b) Define a new recursive function $\text{ProperlySorted}(T)$ that is true precisely when T satisfies the property: For every node with value x, all the left children have value less than x, and all the right children have value greater than x.

(c) Now define a recursive function $\text{delete}(T, x)$ that deletes the value x from T if it exists. Use structural induction to prove that

$$\text{ProperlySorted}(T) \Rightarrow \text{ProperlySorted}\big(\text{delete}(T, x)\big).$$