CSE 331: Software Design & Engineering Fall 2024

Problem Set 8

Due: Thursday, November 21st, 11pm

This worksheet contains only the *written* parts of HW8. The coding instructions will be released tomorrow and be due Monday (11/25). The coding portion of this assignment is longer than in recent weeks, so you should **aim to finish the written part early (Wednesday night is a good goal)** to have plenty of time to work on the code.

Submission

After completing all parts below, submit your solutions as a PDF on Gradescope under "**HW8 Written**".

Don't forget to check that the submitted file is up-to-date with all written work you completed!

Make sure your work is legible and scanned clearly if you handwrite it, or compiled correctly if you choose to use LaTeX. Match each HW problem to the page with your work when you turn in. If your work is not readable or pages are not assigned correctly, you will receive a point deduction.

In the coding portion of this assignment, you will implement the client and server parts of an application that allows users to edit images made up of squares and to save them to and load them from a server.

In this written portion of the work, we will define images mathematically and the operations we will use to edit them. We begin, here, with our mathematical definition of an image, which we call a "Square". It is a tree-like inductive data type.

Squares

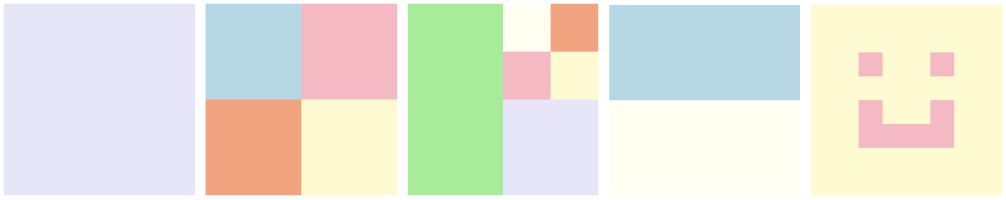
The next problem makes use of a few inductive types. First, a Color is one of the following:

$$\textbf{type}\ \text{Color} \quad := \quad \text{pink} \mid \text{orange} \mid \text{yellow} \mid \text{green} \mid \text{blue} \mid \text{purple} \mid \text{white}$$

Then, a Square is a tree, where each node is either a "solid" (leaf) node of a single color or a "split" (internal) node that breaks the square into four quadrants, each of which can be any square:

$$\begin{aligned} \textbf{type}\ \text{Square} \quad := \quad & \text{solid}(\text{color} : \text{Color}) \\ \mid \quad & \text{split}(\text{nw} : \text{Square},\ \text{ne} : \text{Square},\ \text{sw} : \text{Square},\ \text{se} : \text{Square}) \end{aligned}$$

For example, the following images all represent squares:



We will also need a way to refer to one of the children of a split square. We can do so as follows:

$$\textbf{type}\ \text{Dir} \quad := \quad \text{NW} \mid \text{NE} \mid \text{SW} \mid \text{SE}$$

With that, we define a path to be a list of directions, describing how to get to the node, starting from the root:

$$\textbf{type}\ \text{Path} \quad := \quad \text{List}\langle\text{Dir}\rangle$$

The following function, find retrieves the subtree at the given path:

$$\text{find} : (\text{Path}, \text{Square}) \to \text{Square}$$

$$\begin{aligned} &\text{find}(\text{nil},\ S) && := S \\ &\text{find}(x :: L,\ \text{solid}(c)) && := \text{undefined} \\ &\text{find}(\text{NW} :: L,\ \text{split}(\text{nw}, \text{ne}, \text{sw}, \text{se})) && := \text{find}(L,\ \text{nw}) \\ &\text{find}(\text{NE} :: L,\ \text{split}(\text{nw}, \text{ne}, \text{sw}, \text{se})) && := \text{find}(L,\ \text{ne}) \\ &\text{find}(\text{SW} :: L,\ \text{split}(\text{nw}, \text{ne}, \text{sw}, \text{se})) && := \text{find}(L,\ \text{sw}) \\ &\text{find}(\text{SE} :: L,\ \text{split}(\text{nw}, \text{ne}, \text{sw}, \text{se})) && := \text{find}(L,\ \text{se}) \end{aligned}$$

Think through this function, maybe with some examples, before starting the first task.

Task 1 – Couldn't Square Less [20 pts]

(a) Define a mathematical function

$$\text{replace} : (\text{Path}, \text{Square}, \text{Square}) \to \text{Square}$$

that returns the second Square (the last argument to the function) except with the subtree at the given path replaced by the first Square (the second argument).

To make this definition simpler, we can assume in our notation that, if an expression contains a call to a function whose result is undefined, then the entire expression is undefined. (This matches the behavior of the function if we think of returning undefined as throwing an exception.)

(b) As a simple check, prove by calculation that $\text{find}(\text{nil}, \text{replace}(\text{nil}, T, S)) = T$.

(c) As a better check, prove that $\text{find}(L, \text{replace}(L, T, S)) = T$ (or $=$ undefined) for all paths L and square S and T. Your proof should be by structural induction **on the Square** S.

Feel free to cite the fact that you proved in part(b).

Recall the assumption about undefined that we described in part (a). You can cite this in your proof, but make sure that you also state the function call that will result in undefined (making the whole expression undefined).

For this one proof only, if you do a proof by cases with 4 or more cases, all of which are substantially similar, we will allow you to do only two of the cases and say that the others "are analogous" without writing them out. Assuming that is true (they really are analogous), then you will get full credit for that proof by cases.