Handout 4 (Sulzmann & Lu Algorithm)

So far our algorithm based on derivatives was only able to say yes or no depending on whether a string was matched by regular expression or not. Often a more interesting question is to find out *how* a regular expression matched a string? Answering this question will help us with the problem we are after, namely tokenising an input string, that is splitting it up into its "word" components. The algorithm we will be looking at was designed by Sulzmann & Lu in a rather recent paper. A link to it is provided at KEATS, in case you are interested.[1]

In order to give an answer for how a regular expression matched a string, Sulzmann and Lu introduce *values*. A value will be the output of the algorithm whenever the regular expression matches the string. If not an error will be raised. Since the first phase of the algorithm is identical to the derivative based matcher from the first coursework, the function *nullable* will be used to decide whether as string is matched by a regular expression.

Algorithm by Sulzmann, Lexing

[1] In my humble opinion this is an interesting instance of the research literature: it contains a very neat idea, but its presentation is rather sloppy. Students and I found several rather annoying typos in their examples and definitions.