PROGRAMMING LANGUAGE CONCEPTS (COMP 2212)
SEMESTER TWO, 2018

Home

exercise sheet three: the toy language

This exercise sheet is intended to be completed over the next **two** weeks.

The aim of this exercise class is to use the Alex and Happy tools to define a grammar from scratch. In addition to this we will be writing an interpreter. By the end of the class next week you should be able to write an Alex file, a Happy file and a CEK evaluator for the Toy language.

The demonstrators in the lab will help answer any questions you have about the tutorials you are reading or if you have difficulty understanding the solutions to the exercises.

Task One

Consider the language λToy from Lecture 8. Use Alex to generate a lexer for this language.

Use the characters '->' for the function type symbol and use the character '\' in place of λ.

Task Two

Use Happy to generate a parser for the λToy language. Make sure that the type you define to represent ASTs of the language derives `show`.

Task Three

Write a Haskell program containing a main function that allows the user to repeatedly input a line of text. For each line inputted, parse the input as a λToy program. Print out the parsed AST value each time.

Task Four

If you have not already done so, write a makefile to compile your Haskell program. This should include calls to Alex and Happy to generate the lexer and parser also so that invoking `make` should compile your program from scratch.

Task Five

Add a function `eval` to your Haskell program that takes an abstract syntax tree and interprets it using a call-by-value semantics in the intended way. You may find it helpful to refer to Sheet 7 and Lecture 14 of COMP2209 on the topic of interpreters and CEK machines.

Modify your main function to call `eval` on your parsed input rather than just printing out the AST value. Print out the evaluated result of the inputted λToy program each time instead.