# Assignment 3

The third exercise is about extending the FLAE interpreter presented in the lecture and understanding the difference between first class functions and first order functions. Definitions of the base languages used in this assignment are in the `definition` package.

## Task 1: Preprocess Lets (4 points)

We do not need the "let" construct, when we have functions as values. Taking this into consideration, define a preprocessor for FLAE such that the support for "let" is replaced with a combination of App and Fun. (cf. PLAI book, Section 6.3, link below).

Link to the book: http://cs.brown.edu/~sk/Publications/Books/ProgLangs/2007-04-26/plai-2007-04-26.pdf

Your code has to be in functional style, i.e. avoid mutable `var`s and imperative constructs.

## Task 2: Preprocess Multi-Argument Functions (6 points)

Multi-argument functions are another feature that can be implemented by means of a preprocessor. For the purpose of this exercise, we call the language which extends FLAE with multiple-arguments MFLAE. You can also take a look at the testcases to understand its functionality. Considering this extension of FLAE, define a preprocessor such that the support for multiple-argument functions is replaced with currying and single-argument functions.

Your code has to be in functional style, i.e. avoid mutable `var`s and imperative constructs.

## Task 3: First-Order vs. First-Class Scala (4 points)

Implement two functions in Scala that make use of first-class functions and could not be written nicely with first-order functions only, and two test cases *for each* function.

## Task 4: First-Order vs. First-Class MFAE (4 points)

Implement two functions *in the language MFAE* that make use of first-class functions and could not be written nicely with first-order functions only, and two test cases *for each* function.

To define MFAE functions use Scala's val definitions, similar to the example already given in the project template. Note that the example does not yet use first-class functions. You may use examples similar to ones from task 3.