# Challenge

Spare Time Teaching

May 11, 2015

## Introduction

There is a beautiful symmetry between compilation and decompilation, unfortunately the latter is very hard to do. Therefore we have made a "toy" example.

## Problem

A proper combinator is a closed term following this grammar:

```
pc ::= λ x . pc
     | body
body ::= x
       | body body
```

Write a lambda term, `decompile`, that take a proper combinator and a number for how many arguments the it takes, and returns the number of applications in the combinator.

For example `S` takes three arguments and has three applications, whereas `K` takes two arguments and has zero applications.

## Example

```
S ≡ λ f g x . f x (g x) ≡ λ f g x . ((f x) (g x))
K ≡ λ x y . x

0 ≡ lam s z . z
1 ≡ lam s z . s z
2 ≡ lam s z . s (s z)
3 ≡ lam s z . s (s (s z))

decompile S 3 →* 3
decompile K 2 →* 0
```