Problems

These problems will be discussed on seminar in week 3.

---

1 Let

$$
\begin{aligned}
K &\triangleq \lambda x.\lambda y.x \\
S &\triangleq \lambda x.\lambda y.\lambda z.xz(yz) \\
X &\triangleq \lambda t.t\,K\,S\,K
\end{aligned}
$$

Show that $X\,(X\,X) = S$.

2 Find a combinator M such that $M\ I\ S\ S = M$. Hint: Rewrite the equation as a fixed point equation.

3 When there are repeated abstractions $\lambda x.\lambda y.\lambda z.M$ we may use the abbreviation $\lambda xyz.M$ Let

$$
\begin{aligned}
A &\triangleq \lambda abcdefghijklmnopqstuvwxyzr.r(thisisafixedpointcombinator) \\
B &\triangleq A\,A\,A\,A\,A\,A\,A\,A\,A\,A\,A\,A\,A\,A\,A\,A\,A\,A\,A\,A\,A\,A\,A\,A\,A\,A\,A
\end{aligned}
$$

Show that B is a fixed point combinator, i.e. $BF = F(BF)$ for any F.

4 (Exercise 3.4) We shall extend **While** with the statement `random(x)` and the idea is that its execution will change the value of x to be any positive natural number. Extend the natural semantics and the structural operational semantics to express this. Discuss whether this statement is superfluous when **While** is also extended with the `or` statement.

5 (Exercise 3.5) Consider an extension of **While** that in addition to the `par` statement also contains the construct `protect` S `end`. The idea is that the statement S has to be executed as an atomic entity so that, for example, `x:=1 par protect (x:=2; x:=x+2) end` only has two possible outcomes, **1** and **4**. Extend the structural operational semantics to express this.

6 (Exercise 3.15) Modify the syntax of `While` with blocks and procedures so that procedures take two *call by value* parameters, i.e. the kind of parameters used by Java for the primitive types.

$D_P ::=$ `proc` $p(x_1, x_2)$ `is` $S; D_P \mid \epsilon$

$S ::= \ldots \mid$ `call` $p(a_1, a_2)$

Procedure environments will now be elements of

$$\mathbf{Env}_P = \mathbf{Pname} \hookrightarrow (\mathbf{Var} \times \mathbf{Var} \times \mathbf{Stm} \times \mathbf{Env}_V \times \mathbf{Env}_P)$$

Use static name scopes. You may assume that procedures are not recursive. You have to provide new rules for procedure declarations and procedure calls.

7 Consider replacing the first rule in Table 3.2 with

$$\frac{\langle D_V, \sigma\rangle \to_D \sigma'}{\langle \texttt{var x:=a;} D_V, \sigma\rangle \to_D \sigma'[x \mapsto \mathcal{A}[\![\texttt{a}]\!]\sigma]}$$

Would this change the semantics? Either prove that it doesn't or exhibit an example where the semantics differ.