Lund University
Computer Science
Lennart Andersson

Examination
DAT130, EDA145
2007–08–17, 14.00–18.00

Examination in Programming language theory

This exam has 6 problems, each worth 5 marks. For passing the exam at most 15 marks will be required.

The following texts may be used during the exam:
Nielson, Nielson, Semantics with Applications.
Andersson, Programming language theory, Lecture notes.

---

1 Prove by structural induction that

$$\mathcal{A}[\![a[y \mapsto a_0]]\!]\sigma = \mathcal{A}[\![a]\!](\sigma[y \mapsto \mathcal{A}[\![a_0]\!]\sigma])$$

for all a, a_0, y and σ. It suffices to consider the cases when a is a variable and a sum of two expressions.

2 Let

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

Show that $X\, X\, X = K$.

3 Assume that S_1 and S_2 are equivalent according to the natural operational semantics for **While**. Show that $S; S_1$ is equivalent to $S; S_2$.

4 Extend the **While** language with an iterate statement

`iterate` *S* `for` *a*

According to the informal semantics S should be executed the number of times given by the initial value of the arithmetic expression a. If the value is less than or equal to 0 it should not be executed. Specify the denotational semantics.

5 Functional languages often include a `let` expression

$$\texttt{let}\ x = e_0\ \texttt{in}\ e$$

The value of e_0 is bound to the name x which is then used when computing e. Define the compiling function $\mathcal{C}$ for the `let` expression.

6 Give an example of a monotone function f on a chain complete poset that is not continuous.