CMPSC 461: Programming Language Concepts
Assignment 1 Solution

Problem 1 [8pt] Add parentheses to the following lambda terms so that the grouping of sub-terms becomes explicit. For example, the term $\lambda x.\ x\ \lambda y.\ y$ with parentheses is $\lambda x.\ (x\ (\lambda y.\ y))$.

a) (4pt) $\lambda x.\lambda y.\lambda z.\ x\ y\ z$

Solution: $\lambda x.\ (\lambda y.\ (\lambda z.\ ((x\ y)\ z)))$

b) (4pt) $\lambda x.\lambda y.\ y\ \lambda x.\ x$

Solution: $\lambda x.\ (\lambda y.\ (y\ (\lambda x.\ x)))$

Problem 2 [6pt] Fully evaluate the following λ-term so that no further β-reduction is possible:

$$((\lambda x\ y.\ x)\ (\lambda y.\ y))\ y$$

Solution:

$$\begin{aligned}
& ((\lambda x\ y.\ x)\ (\lambda y.\ y))\ y \\
= {} & ((\lambda x.\ (\lambda y.\ x))\ (\lambda y.\ y))\ y && \text{(desugar term } \lambda x\ y.\ x)) \\
= {} & (\lambda y.\ (\lambda y.\ y))\ y && (\beta - \text{reduction}) \\
= {} & (\lambda u.\ (\lambda v.\ v))\ y && (\alpha - \text{reduction}) \\
= {} & \lambda v.\ v && (\beta - \text{reduction})
\end{aligned}$$

Problem 3 [10pt] Recall that under Church encoding, we have the following definitions:

$$\begin{aligned}
\texttt{TRUE} &\triangleq \lambda t\ f.\ t \\
\texttt{FALSE} &\triangleq \lambda t\ f.\ f \\
\texttt{IF} &\triangleq \lambda b\ t\ f.\ b\ t\ f
\end{aligned}$$

Show that $(\texttt{IF TRUE FALSE TRUE}) = \texttt{FALSE}$ under such encoding.

Solution:

$$\begin{aligned}
& (\texttt{IF TRUE FALSE TRUE}) \\
= {} & (\lambda b\ t\ f.\ b\ t\ f)\ \texttt{TRUE FALSE TRUE} && (\text{definition of } \texttt{IF}) \\
= {} & \texttt{TRUE FALSE TRUE} && (\beta - \text{reduction}) \\
= {} & (\lambda t\ f.\ t)\ \texttt{FALSE TRUE} && (\text{definition of } \texttt{TRUE}) \\
= {} & \texttt{FALSE} && (\beta - \text{reduction})
\end{aligned}$$

Problem 4 [10pt] Using the pure λ-calculus, we can encode a pair of λ-terms as

$$\texttt{PAIR} \triangleq \lambda a\ b\ p.\ (p\ a\ b)$$

where PAIR takes three λ-terms a, b, p and applies p to a and b. Try to define two functions LEFT and RIGHT in λ-calculus so that given a pair, they return the first term and the second term in the pair respectively. For example, LEFT $(\texttt{PAIR}\ x\ y)$ should evaluate to x under your encoding. (Hint: you can reuse the definition of TRUE and FALSE, which select one term if two terms are given).

Solution:

$$\texttt{LEFT} \triangleq \lambda p.\ (p\ \texttt{TRUE}) = \lambda p.\ (p\ (\lambda t\ f\ .\ t))$$
$$\texttt{RIGHT} \triangleq \lambda p.\ (p\ \texttt{FALSE}) = \lambda p.\ (p\ (\lambda t\ f\ .\ f))$$

Problem 5 [16pt] In the lecture, we have used the pure λ-calculus to construct natural numbers and encoded some operations on them. In such encoding, a number n is encoded as a function $\lambda f\ z.\ f^n\ z$ (denoted as $\underline{n}$). If you get stuck in any question, try to proceed by assuming the previous function is properly defined. You can reuse any encoding given in lectures and previous questions.

a) (4pt) Define a function ISZERO in λ-calculus, so that given a number $\underline{n}$, it returns TRUE (the encoding of true) if $\underline{n} = \underline{0}$; FALSE (the encoding of false) if $\underline{n} \neq \underline{0}$. (Hint: try to define two terms so that applying the second term multiple times to the first term returns FALSE; otherwise, the application returns TRUE).
Solution:
$\texttt{ISZERO} \triangleq \lambda n.\ n\ (\lambda f.\ \texttt{FALSE})\ (\texttt{TRUE})$

b) (4pt) Define a function PRED in λ-calculus, so that given a number $\underline{n}$, the function returns its predecessor, assuming the predecessor of $\underline{0}$ is $\underline{0}$ (Hint: follow the idea in Problem 4a. You might need to use PAIR).
Solution:
$\texttt{PRED} \triangleq \lambda n.\ \texttt{RIGHT}\ (n\ (\lambda p.\ \texttt{PAIR}\ (\texttt{SUCC}\ (\texttt{LEFT}\ p))\ (\texttt{LEFT}\ p))\ (\texttt{PAIR}\ \underline{0}\ \underline{0}))$

c) (4pt) Use your encoding of PRED to define a subtraction function MINUS, so that MINUS $\underline{n_1}\ \underline{n_2}$ returns $\underline{n_1 - n_2}$ when $n_1 \geq n_2$, and $\underline{0}$ otherwise.
Solution:
$\texttt{MINUS} \triangleq \lambda n_1\ n_2.\ n_2\ \texttt{PRED}\ n_1$

d) (4pt) Based on what you have encoded so far, and the Boolean encodings in the lecture, define an equality test, called EQUALS on two numbers, so that it returns TRUE when two numbers are equal, and FALSE otherwise.
Solution:
$\texttt{EQUALS} \triangleq \lambda n_1\ n_2.\ \texttt{AND}\ (\texttt{ISZERO}\ (\texttt{MINUS}\ n_1\ n_2))\ (\texttt{ISZERO}\ (\texttt{MINUS}\ n_2\ n_1))$