Homework 1

Due: Monday, February 16, 2026 — in class or uploaded electronically

Note. You are free to return your answers in the form of Lean code that type-checks or proves the required computations. Each problem is worth equal weight.

Problem 1 — α-Equivalence

Find out for each of the following λ-terms whether it is α-equivalent, or not, to $\lambda x.\ x\ (\lambda x.\ x)$:

(a) $\lambda y.\ y\ (\lambda x.\ x)$,
(b) $\lambda y.\ y\ (\lambda x.\ y)$,
(c) $\lambda y.\ y\ (\lambda y.\ x)$.

Problem 2 — Substitution

Give the results of the following substitutions:

(a) $(\lambda x.\ y\ (\lambda y.\ x\ y))\ [y := \lambda z.\ z\ x]$,
(b) $((x\ y\ z)\ [x := y])\ [y := z]$,
(c) $((\lambda x.\ x\ y\ z)\ [x := y])\ [y := z]$,
(d) $(\lambda y.\ y\ y\ x)\ [x := y\ z]$.

Problem 3 — Church Numerals

We define the λ-terms `zero`, `one`, `two` (the first three so-called *Church numerals*), and the λ-terms `add` and `mult` (which mimic addition and multiplication of Church numerals) by:

$$\begin{aligned}
\texttt{zero} &:= \lambda f\ x.\ x,\\
\texttt{one} &:= \lambda f\ x.\ f\ x,\\
\texttt{two} &:= \lambda f\ x.\ f\ (f\ x),\\
\texttt{add} &:= \lambda m\ n\ f\ x.\ m\ f\ (n\ f\ x),\\
\texttt{mult} &:= \lambda m\ n\ f\ x.\ m\ (n\ f)\ x.
\end{aligned}$$

(a) Show that $\texttt{add one one} \twoheadrightarrow_\beta \texttt{two}$.

You may use Lean to do so by implementing these functions and using `#eval` (show your code).

Problem 4 — Normal Forms and Infinite Reductions

Let M be a λ-term with the following properties:

1. M has a β-normal form.
2. There exists a reduction path $M \equiv M_0 \to_\beta M_1 \to_\beta M_2 \to_\beta \cdots$ of infinite length.

(a) Prove that every M_i has a β-normal form.

Problem 5 — Strong Normalisation

Prove the following: if $M\ N$ is strongly normalising, then both M and N are strongly normalising.

Problem 6 — Self-Referential Terms

(a) Construct a λ-term M such that $M =_\beta \lambda x\ y.\ x\ M\ y$.

(b) Construct a λ-term M such that $M\ x\ y\ z =_\beta x\ y\ z\ M$.

Problem 7 — Simple Typability

Investigate for each of the following λ-terms whether they can be typed with a simple type. If so, give a type for the term and the corresponding types for x and y. If not, explain why.

(a) $x\ x\ y$,
(b) $x\ y\ y$,
(c) $x\ y\ x$,
(d) $x\ (x\ y)$,
(e) $x\ (y\ x)$.

As usual, in the affirmative cases you may provide a Lean function with the required type.

> **Note.** In the problems below you may use Lean for quick computations, but make sure that you write **at least one full derivation** by hand.

Problem 8 — Inhabitation (Empty Context)

Find inhabitants of the following types in the **empty context**, by giving appropriate derivations.

(a) $(\alpha \to \alpha \to \gamma) \to \alpha \to \beta \to \gamma$,
(b) $((\alpha \to \gamma) \to \alpha) \to (\alpha \to \gamma) \to \beta \to \gamma$.

Alternatively, show Lean code that type-checks with the above types.

Problem 9 — Inhabitation (Non-Empty Context)

Find a term of type τ in context Γ, with:

(a) $\tau \equiv (\alpha \to \beta) \to \alpha \to \gamma, \quad \Gamma \equiv x : \alpha \to \beta \to \gamma,$

(b) $\tau \equiv \alpha \to (\alpha \to \beta) \to \gamma, \quad \Gamma \equiv x : \alpha \to \beta \to \alpha \to \gamma,$

(c) $\tau \equiv (\alpha \to \gamma) \to (\beta \to \alpha) \to \gamma, \quad \Gamma \equiv x : (\beta \to \gamma) \to \gamma.$

Alternatively, write Lean functions with the corresponding signature, e.g. for the first one:

```
def problem9a (x :   →   →  ) : ( → ) →   →   := sorry
```

Give appropriate derivations.