1 Full Evaluation Step

Provide the full derivation for a single step of evaluation of

\( ((\lambda x.\lambda y. x y y) ((\lambda z. (\lambda a.\lambda b. b a))) ((\lambda c.c) (\lambda d.d))) \)

You don’t need to provide a derivation for the variable replacements \( e[v/x] \) – you can simply replace the variables.

2 Check for Zero Applications

Define a class of untyped lambda calculus terms \( App(n) = \lambda f.\lambda x.x^n x \). So \( App(0) = \lambda f.\lambda x.x \) and \( App(1) = \lambda f.\lambda x.x f x \) and \( App(4) = \lambda f.\lambda x.x(f(f(f(f x)))) \). Write an untyped lambda calculus expression \( g \) such that \( g App(0) = \lambda x.\lambda y.x \) and if \( n > 0 \) then \( g App(n) = \lambda x.\lambda y.y \).

3 Full Variable Replacements

Find the expression \( e' \), and show the full derivation, for \( (\lambda x.z (\lambda z.z) x)(\lambda w.w/z) = e' \).

4 Non-Typeability

Prove that there does not exist a \( \tau \) such that \( \lambda(x:\tau).x x \) is well typed.

5 Weakening in STLC

Let \( \Gamma \) and \( \Gamma' \) be partial functions from \( Vars \rightarrow T \). We define \( \Gamma \subseteq \Gamma' \) as \( \forall v \in Vars \) where \( \Gamma \) is defined on \( v \), then \( \Gamma' \) is defined on \( v \) and \( \Gamma(v) = \Gamma'(v) \).

Prove that if \( \Gamma \vdash e : \tau \) and \( \Gamma \subseteq \Gamma' \) then \( \Gamma' \vdash e : \tau \).

6 Full Typing Derivation

Find a type \( \tau \), and show the full derivation for \( \vdash \lambda(x:\text{Bool}).\lambda(y:\text{Int} \rightarrow \text{Bool}).\text{if } x \text{ then } y(S\;O) \text{ else } y\;O : \tau \).