Assignment #2

Name: _____ ID: _____

This assignment has **4** questions, for a total of **25** marks.

Recall the following acronyms: SOS (structural operational semantics), COS (contextual operational semantics), SM (small step), BG (big step), CBV (call by value), CBN (call by name).

Question 1: **Typing derivation** ..... 6 marks

Show the typing derivation of these terms, with the following environment $\Gamma = f : \mathbb{N} \to \mathbb{N}$

- $t_1 = f\ (3 + 5) : \mathbb{N}$ [3]

- $t_2 = f\ ((\lambda x : \mathbb{N}.\, x + 2)5) : \mathbb{N}$ [3]

Question 2: **Encoding** ..... 7 marks

For each of the following constructs, create an encoding in STLC. Show that your encodings behave as the related construct by showing the reductions of your encoding using COS-SM-CBV. The intended semantics for the constructs is given after each construct in text.

- sequencing: $t ::= \cdots \mid t; t'$. Semantics: t is evaluated first, then t' is evaluated. [1]

- let-in: $t ::= \cdots \mid let\ x = t\ in\ t'$. Semantics: t is evaluated into a value v and then t' is evaluated for v in place of x. [1]

- arrays of length 4: $t ::= \cdots \mid [t, t, t, t]$. Values include arrays of values: $v ::= \cdots \mid [v, v, v, v]$. (no semantics for this case) [1]

- array field access: $t ::= \cdots \mid t.i \quad (i \in 0..3)$. Semantics: for $i \in 0..3$ we have that $[v_0, v_1, v_2, v_3].i$ returns v_i (show the encodings for at least two cases of i). [2]

- array update: $t ::= \cdots \mid t.i = t \quad (i \in 0..3)$. Semantics: for $i \in 0..3$ we have that $[v_0, v_1, v_2, v_3].2 = v$ returns $[v_0, v_1, v, v_3]$ (show the encodings for at least two cases of i). [2]

Question 3: **Named functions** . . . . . 6 marks

Consider extending STLC with named function, call this language STLCN. A program is no longer a term, but a collection of named functions ($P ::= \varnothing \mid P; F$). A named function defines a function name, a parameter of a certain type and the function body ($F ::= f(x : \tau) : \tau \mapsto t$). A function body is a term. Terms now must include new constructs to call other functions.

Define the COS judgements for STLCN. Define the primitive reduction rules for STLCN as well as evaluation contexts. Define the typing judgements for STLCN, starting from how to determine when a program is well typed. Define the typing rules for STLCN.

Some primitive reductions and typing rules from STLCN will be similar to those of STLC. To avoid duplicates, name all the STLC rules that have analogous ones and show how to convert two of them to the new forms only.

STLCN should still be safe, i.e., it should not get stuck trying to call a function with a parameter of the wrong type, or calling a function that does not exist. However, STLCN is not normalising: there may be functions that mutually call each other and thus diverge.

Question 4: **Private and public functions** . . . . . 6 marks

Extend STLC so that each program not only defines a list of named functions, but also a list of imports and exports. Imports are signatures (names and types) of functions that the program does not define. Exports are names of functions that the program defines. Change the typing rule for programs to reflect this.

Define the linking two programs ($P_1 + P_2$), which yields the program obtained by joining the two. Linking is only possible when both programs are well-typed and when they fulfill each other's exports with their own imports.