Semantics of Programming Languages

(a) Consider the types given by the grammar below.

\[ T ::= \text{unit} \mid T_1 \to T_2 \]

Define the syntax and type system for a pure functional language over these types: a syntax of expressions \( e \) for variables, skip, functions, and function application, and typing rules defining a judgement \( \Gamma \vdash e : T \). State clearly what mathematical objects \( \Gamma \) ranges over, and what the binding is in your language. [5 marks]

(b) For each of the following, state whether it is true or false. For the true statements, give examples (instantiations for the existentially quantified variables); for the false statements, give proofs of their negations. For any inductive proofs, include statements of the kind of induction used and the induction hypothesis.

(i) \( \exists \Gamma_1, \Gamma_2, e, T_1, T_2. (\Gamma_1 \vdash e : T_1) \land (\Gamma_2 \vdash e : T_2) \land (T_1 \neq T_2) \)

(ii) \( \exists \Gamma, e, T_1, T_2. (\Gamma \vdash e : T_1) \land (\Gamma \vdash e : T_2) \land (T_1 \neq T_2) \)

(iii) \( \exists \Gamma, e, T. \Gamma \vdash e : T \)

(iv) \( \exists \Gamma, e, T. \Gamma \vdash e : T \) such that \( \Gamma \not\vdash e : T \) if the syntax and rules were interpreted concretely, instead of up to alpha equivalence. [14 marks]

(c) Discuss briefly whether alpha equivalence is needed to define type systems for ML-like and Java-like languages. [1 mark]