

8 Semantics of Programming Languages (nk480)

(a) Suppose we have a language with booleans, integers, and *mutable* variables:

$$e ::= n \mid e_0 + e_1 \mid e_0 < e_1 \mid \text{true} \mid \text{false} \mid \text{if } e \text{ then } e_1 \text{ else } e_2 \\ \mid x \mid \text{var } x = e_0 \text{ in } e_1 \mid x := e$$

(i) Give a grammar for the *values* of this language. [1 mark]

(ii) What mathematical object should be used to represent a store  $\sigma$  (which tracks which values each variable has)? [1 mark]

(iii) Give a reasonable operational semantics for this language, as a transition relation. (You may assume the existence of a substitution operation  $\{v/x\}e$ .)

$$\langle \sigma; e \rangle \rightsquigarrow \langle \sigma'; e' \rangle$$

This semantics should ensure (though you need not prove) that for any configuration  $\langle \sigma; e \rangle$ , it is either of the form  $\langle \sigma; v \rangle$  with no further transitions, or otherwise it has at most one transition  $\langle \sigma; e \rangle \rightsquigarrow \langle \sigma'; e' \rangle$ . In addition to the formal rules, give an explanation of the reduction rules you define for variable declarations  $\text{var } x = e_0 \text{ in } e_1$  and assignments  $x := e$ .

[8 marks]

(b) (i) Define a reasonable set of types for this programming language. [1 mark]

(ii) Explain what a typing context should look like for this language. [1 mark]

(iii) Define a set of typing rules for this programming language, which should ensure type safety. [7 marks]

(iv) State (but do not prove) the progress and type preservation theorems for this language. [1 mark]