Semantics of Programming Languages
Supervision 1

Supervisor: Joe Isaacs (josi2).

All work should be submitted in PDF form 36 hours before the supervision to the email `josi2@cam.ac.uk`. If you have any questions on the course please include these at the top of the supervision work and we can talk about them in the supervision.

Please give both the original and modified code for any exercises that requires code to be changed. The extensions are encouraged but should only be attempted after the main work has been attempted.

Exercises 2.4

- **Ex1** Give indication of why you think this program is correct. Does this compute factorial for all inputs.
- **Ex3** Indicate which rules are used at each stage.
- **Ex4**
- **Ex8**
- **Ex9**
- **Ex10**
- **Extension: Ex11**

Exercises 3.4

This is a lot of work, but each of the proof only require a few cases so please try to complete each case thoroughly. I would prefer some of the proofs done thoroughly instead all of then done not so well.

- What are the two types of induction described and for each case write the mathematical statement.
- What is lemma 9?
- What are the inversion lemmas and why are they useful? Write out the inversion lemma for

$$\langle e, s\rangle \to \langle \hat{e}, \hat{s}\rangle$$

  in the case that

$$e = \textbf{ if } e_1 \textbf{ then } e_2 \textbf{ else } e_3$$

- In your own words what do the following theorems mean, also give a symbol definition of each rule and which type of induction would you use:
  - Determinacy
  - Progress

- Type Preservation
- Safety
- Uniqueness of Typing

For example Determinacy

$$\text{if } \langle e, s\rangle \to \langle e_1, s_1\rangle \text{ and } \langle e, s\rangle \to \langle e_2, s_2\rangle \text{ then } \langle e_1, s_1\rangle = \langle e_2, s_2\rangle$$

would be written as

$$\forall e, s, e_1, e_2, s_1, s_2 . \langle e, s\rangle \to \langle e_1, s_1\rangle \wedge \langle e, s\rangle \to \langle e_2, s_2\rangle \Rightarrow \langle e_1, s_1\rangle = \langle e_2, s_2\rangle$$

Since the only universally quantified ∀ variables are e, s it seems the best induction choice would be structural induction.

What is the different between typeability and type checking?

- **Ex12**

  Only try the **if** case, assuming all other cases.

- **Ex13**

  You only need consider the rule or rules which cause determinacy to be lost.

- **Ex14**

  Only do the **seq** case.

- **Ex16**

  Don't prove any cases, but instead consider how to proof structure will differ when compared to **Ex15**

- **Extension: Ex17**

  Just give a proof outline.

- How would you go proving **safety**?