COMS22201: Semantics Revision

(Nielson and Nielson)

csxor@bristol.ac.uk

1 Essential Reading

Chapter 1

1	Introduction to semantics	(1)
1.1	Overview of operational/axiomatic/denotational semantics	(1-7)
1.2	The While language	(7-9)
1.3	Denotational semantics of arithmetics and Booleans	(9-14)
1.4	(Free) Variables and substitutions	(15-17)

Chapter 2

2	Introduction to operational semantics	(19)
2.1	Natural (operational) semantics	(19-30 only)
2.2	Structural operational semantics	(32-39 only)

NOTE: You may omit the semantic functions $\mathcal{S}_{ns}$ (p.31) and $\mathcal{S}_{sos}$ (p.31) and the associated equivalence result (§2.3).

Chapter 4

4	Introduction to denotational semantics	(85)
4.1	(Direct Style) denotational semantics	(85-89 only)

NOTE: You may omit requirements on the fixpoint, fixpoint theory, existence proofs, equivalence results, locations and continuations (p.90-131).

Chapter 6

6 Introduction to axiomatic semantics	(169)
6.2 Partial Correctness Assertions	(175-176;178-183 only)
6.4 Total Correctness Assertions	(191-194 only)

NOTE: You may omit the discussion on direct proofs of program correctness (p.169-175), the subtleties of the assertion language (p.177) and the associated soundness/completeness results (§6.3 and p.194-196).

But you will be expected to do partial and total correctness proofs in the exam. You may write proofs as a tree or as an annotated program tableau (as shown in the lecture slides).

For total correctness assertions you may either write $\{P\}S\{\Downarrow Q\}$ as in the book or $[P]S[Q]$ as in the slides.

Where book uses the notation $\Rightarrow$ in an axiomatic side condition, it should be understood as meaning logical entailment $\models$ (and not confused with the operational semantic arrows in any way).

2 Key Semantic Definitions

In the exam, you will be required to know the standard semantic definitions of program statements by heart. These are succinctly summarised in the following five tables in the book:

Natural Table 2.1	(20)
Structural Table 2.2	(33)
Denotational Table 4.1	(86)
Partial Table 6.1	(178)
Total Table 6.2	(192)

You will be expected to work with these definitions and adapt them to give semantics to other program language constructs such as other arithmetic operators, non-deterministic choice, parallel execution, other types of loop, and side-effects. You should be able to interleave the various semantic styles, as done in the book for example, where it embeds the denotational semantics of expressions within the operational semantics of commands.

You will also be expected write semantic rules for arithmetic and Boolean expressions in the five styles above (as done in the lecture slides). Note that the book only gives a denotational semantics for such expressions:

Arithmetic Table 1.1	(13)
Boolean Table 1.2	(14)

2.1 Key Proof Techniques

Although you will NOT be required to write an inductive proof in the exam, for convenience, a summary of the different types of inductive proof and examples of their use are given below:

structural induction (basic expressions/composite expressions) (11)
as used in Lemma 1.11 (15)

Induction on shape of derivation trees (axioms/rules) (28)
as used in Theorem 2.9 (29)

Induction on length of derivation sequences (zero/nonzero) (37)
as used in Lemma 2.19 (37)

2.2 Optional Reading

To help with exam preparation, you are advised to read about the extensions of the While language discussed in the book in sections 2.4, 4.5 and 6.4. But please note that the exam NOT require you to give a semantics for procedure or function calls.

You are advised to familiarise yourself with the mathematical notation and definitions in Appendix A of the book (although the sections on functions and relations are much more important than those on predicates and transitions systems).

Appendices B,C and D contain Miranda implementations that could help you understand the denotational and operational semantics of While.