Semantics of Programming Languages
Exercise Sheet 6

Exercise 6.1  Type coercions

Adding and comparing integers and reals can be allowed by introducing implicit conversions: Adding an integer and a real results in a real value, comparing an integer and a real can be done by first converting the integer into a real. Implicit conversions like this are called coercions.

1. Modify, in the theory Types, the inductive definitions of taval and tbval such that implicit coercions are applied where necessary.
2. Extend the datatype com by a loop construct DO a TIMES c which executes the command c exactly a times, where a is an arbitrary arithmetic expression of integer type.
3. Adapt all proofs in the theory Types accordingly.

Hint: Isabelle already provides the coercion functions nat, int, and real.

Homework 6.1  Compiler optimization

Submission until Tuesday, December 3, 10:00am.

A common programming idiom is IF b THEN c, i.e., the else-branch consists of a single SKIP command.

1. Look at how the program IF Less (V "x") (N 5) THEN "y" ::= N 3 ELSE SKIP is compiled by ccomp and identify a possible compiler optimization.
2. Implement an optimized compiler (by modifying ccomp) which reduces the number of instructions for programs of the form IF b THEN c.
3. Extend the proof of comp_correct to your modified compiler.
Homework 6.2  Absolute Adressing

Submission until Tuesday, December 3, 10:00am. This homework is worth 5 bonus points.

The current instruction set uses relative addressing, i.e., the jump-instructions contain an offset that is added to the program counter. An alternative is absolute addressing, where jump-instructions contain the absolute address of the jump target.

Write a semantics that interprets the 3 types of jump instructions with absolute addresses. Write a function that converts a program from relative to absolute addressing. Show that the semantics match wrt. your conversion.

**definition** \( cnv_{to\_abs} :: \text{instr list} \Rightarrow \text{instr list} \)

**abbreviation**

\( exec_{abs} :: \text{instr list} \Rightarrow \text{config} \Rightarrow \text{config} \Rightarrow \text{bool} \) ("(\_ \vdash_a (\_ \rightarrow^*/ \_))" 50)

**theorem** "\( cnv_{to\_abs} P \vdash_a c \rightarrow^* c' \longleftrightarrow P \vdash c \rightarrow^* c' \)"