

# SmallC Formal Type Checking and Type Inference Rules

April 3, 2025

## 1 Introduction

This document presents the formal type checking and type inference rules using the syntax used in lecture.

## 2 Preliminaries

**2.1. Environments.** Type rules define judgments that make use of a context  $\Gamma$  (Stylized as  $G$ ). The rules show two operations on environments:

- $G(x)$  means to look up the type of  $x$  mapped to in the context  $G$ . This operation is undefined if there is no mapping for  $x$  in  $G$ .
- The second operation is written  $G[x : t]$ . It defines a new environment that is the same as  $G$  but maps  $x$  to  $t$ . It thus overrides any prior mapping for  $x$  in  $G$ .

**2.2. Constraints.** In addition to the context, we also will need to keep track of the constraints we should use when we try and infer types of variables.

- A constrain set is a set of bindings of one type to another. We will use the syntax  $t_1 : t_2$  to indicate that type  $t_1$  should be the same as type  $t_2$ .
- To keep types consistent, we may use a placeholder type to refer to an expression. You can see more in the below constraints rules. These placeholder types will look like  $t$  or  $t_n$ . If two types use the same  $n$  value, then they are considered the same type.

**2.3. Syntax.** In this document, we have simplified the presentation of the syntax, so it may not correspond exactly to the files that your checker/inferencer will read in. For example, we write `while  $e$   $s$`  to represent the syntax of a while-loop, where  $e$  is the guard and  $s$  is the body. This corresponds to `While of expr * stmt` in the `ast.ml` file. Hopefully the connection between what we show here at that file is clear enough from context.

**2.4. Error conditions.** The semantics here defines only *correct* evaluations. It says nothing about what happens when, say, you have a type error. For example, for the rules below there is no type  $t$  for which you can prove the judgment  $\Gamma \vdash 1 + \text{true} \multimap t$ . In your actual implementation, erroneous programs will cause an exception to be raised, as indicated in the project README.

### 3 Type Checking Rules

$$\begin{array}{c}
\text{int } \frac{}{G \vdash n : \text{int}} \quad \text{bool } \frac{}{G \vdash b : \text{bool}} \quad \text{value } \frac{}{G \vdash \text{read}() : t} \quad \text{var-lookup } \frac{G(x) = t}{G \vdash x : t} \\
\\
\text{add } \frac{G \vdash e_1 : \text{int} \quad G \vdash e_2 : \text{int}}{G \vdash e_1 + e_2 : \text{int}} \quad \text{sub } \frac{G \vdash e_1 : \text{int} \quad G \vdash e_2 : \text{int}}{G \vdash e_1 - e_2 : \text{int}} \quad \text{mult } \frac{G \vdash e_1 : \text{int} \quad G \vdash e_2 : \text{int}}{G \vdash e_1 * e_2 : \text{int}} \\
\\
\text{div } \frac{G \vdash e_1 : \text{int} \quad G \vdash e_2 : \text{int}}{G \vdash e_1 / e_2 : \text{int}} \quad \text{pow } \frac{G \vdash e_1 : \text{int} \quad G \vdash e_2 : \text{int}}{G \vdash e_1 \wedge e_2 : \text{int}} \\
\\
\text{and } \frac{G \vdash e_1 : \text{bool} \quad G \vdash e_2 : \text{bool}}{G \vdash e_1 \&\& e_2 : \text{bool}} \quad \text{or } \frac{G \vdash e_1 : \text{bool} \quad G \vdash e_2 : \text{bool}}{G \vdash e_1 || e_2 : \text{bool}} \\
\\
\text{equal } \frac{G \vdash e_1 : t \quad G \vdash e_2 : t}{G \vdash e_1 == e_2 : \text{bool}} \quad \text{not-equal } \frac{G \vdash e_1 : t \quad G \vdash e_2 : t}{G \vdash e_1 != e_2 : \text{bool}} \quad \text{greater-equal } \frac{G \vdash e_1 : t \quad G \vdash e_2 : t}{G \vdash e_1 >= e_2 : \text{bool}} \\
\\
\text{less-equal } \frac{G \vdash e_1 : t \quad G \vdash e_2 : t}{G \vdash e_1 <= e_2 : \text{bool}} \quad \text{greater } \frac{G \vdash e_1 : t \quad G \vdash e_2 : t}{G \vdash e_1 > e_2 : \text{bool}} \quad \text{less } \frac{G \vdash e_1 : t \quad G \vdash e_2 : t}{G \vdash e_1 < e_2 : \text{bool}}
\end{array}$$

Statements can modify the context. So  $\rightarrow G'$  means the context updated the environment to  $G'$ .

$$\begin{array}{c}
\text{seq } \frac{G \vdash s_1 : () \rightarrow G' \quad G' \vdash s_2 : () \rightarrow G''}{G \vdash s_1; s_2 : () \rightarrow G''} \quad \text{assign } \frac{G \vdash e : t}{G \vdash x = e : () \rightarrow G[x : t]} \\
\\
\text{if } \frac{G \vdash e_1 : \text{bool} \quad G \vdash s_1 : () \rightarrow G' \quad G \vdash s_2 : () \rightarrow G''}{G \vdash \text{if } e \ s_1 \ s_2 : () \rightarrow G' \cup G''} \quad \text{for } \frac{G \vdash e_1 : \text{int} \quad G \vdash e_2 : \text{int} \quad G \vdash s : () \rightarrow G'}{G \vdash \text{for } x \ e_1 \ e_2 \ s : () \rightarrow G'} \\
\\
\text{while } \frac{G \vdash e : \text{bool} \quad G \vdash s : () \rightarrow G'}{G \vdash \text{while } e \ s : () \rightarrow G'} \quad \text{print } \frac{G \vdash e : t}{G \vdash \text{printf}(e) : () \rightarrow G}
\end{array}$$

## 4 Type Inference Rules

$$\begin{array}{c}
\text{int} \frac{}{G \vdash n : \text{int} \dashv \{\}} \quad \text{bool} \frac{}{G \vdash b : \text{bool} \dashv \{\}} \quad \text{value} \frac{}{G \vdash \text{read}() : t \dashv \{\}} \quad \text{var-lookup} \frac{G(x) = t}{G \vdash x : t \dashv \{\}} \\
\\
\text{add} \frac{G \vdash e_1 : t_1 \dashv C_1 \quad G \vdash e_2 : t_2 \dashv C_2}{G \vdash e_1 + e_2 : \text{int} \dashv \{t_1 : \text{int}, t_2 : \text{int}\} \cup C_1 \cup C_2} \quad \text{sub} \frac{G \vdash e_1 : t_1 \dashv C_1 \quad G \vdash e_2 : t_2 \dashv C_2}{G \vdash e_1 - e_2 : \text{int} \dashv \{t_1 : \text{int}, t_2 : \text{int}\} \cup C_1 \cup C_2} \\
\\
\text{mult} \frac{G \vdash e_1 : t_1 \dashv C_1 \quad G \vdash e_2 : t_2 \dashv C_2}{G \vdash e_1 * e_2 : \text{int} \dashv \{t_1 : \text{int}, t_2 : \text{int}\} \cup C_1 \cup C_2} \quad \text{div} \frac{G \vdash e_1 : t_1 \dashv C_1 \quad G \vdash e_2 : t_2 \dashv C_2}{G \vdash e_1 / e_2 : \text{int} \dashv \{t_1 : \text{int}, t_2 : \text{int}\} \cup C_1 \cup C_2} \\
\\
\text{pow} \frac{G \vdash e_1 : t_1 \dashv C_1 \quad G \vdash e_2 : t_2 \dashv C_2}{G \vdash e_1 \wedge e_2 : \text{int} \dashv \{t_1 : \text{int}, t_2 : \text{int}\} \cup C_1 \cup C_2} \quad \text{and} \frac{G \vdash e_1 : t_1 \dashv C_1 \quad G \vdash e_2 : t_2 \dashv C_2}{G \vdash e_1 \text{ -- } e_2 : \text{bool} \dashv \{t_1 : \text{bool}, t_2 : \text{bool}\} \cup C_1 \cup C_2} \\
\\
\text{or} \frac{G \vdash e_1 : t_1 \dashv C_1 \quad G \vdash e_2 : t_2 \dashv C_2}{G \vdash e_1 \text{ + } e_2 : \text{bool} \dashv \{t_1 : \text{bool}, t_2 : \text{bool}\} \cup C_1 \cup C_2} \quad \text{equal} \frac{G \vdash e_1 : t_1 \dashv C_1 \quad G \vdash e_2 : t_2 \dashv C_2}{G \vdash e_1 == e_2 : \text{bool} \dashv \{t_1 : t_2\} \cup C_1 \cup C_2} \\
\\
\text{not-equal} \frac{G \vdash e_1 : t_1 \dashv C_1 \quad G \vdash e_2 : t_2 \dashv C_2}{G \vdash e_1 != e_2 : \text{bool} \dashv \{t_1 : t_2\} \cup C_1 \cup C_2} \quad \text{greater-equal} \frac{G \vdash e_1 : t_1 \dashv C_1 \quad G \vdash e_2 : t_2 \dashv C_2}{G \vdash e_1 >= e_2 : \text{bool} \dashv \{t_1 : t_2\} \cup C_1 \cup C_2} \\
\\
\text{greater} \frac{G \vdash e_1 : t_1 \dashv C_1 \quad G \vdash e_2 : t_2 \dashv C_2}{G \vdash e_1 > e_2 : \text{bool} \dashv \{t_1 : t_2\} \cup C_1 \cup C_2} \quad \text{less-equal} \frac{G \vdash e_1 : t_1 \dashv C_1 \quad G \vdash e_2 : t_2 \dashv C_2}{G \vdash e_1 <= e_2 : \text{bool} \dashv \{t_1 : t_2\} \cup C_1 \cup C_2} \\
\\
\text{less} \frac{G \vdash e_1 : t_1 \dashv C_1 \quad G \vdash e_2 : t_2 \dashv C_2}{G \vdash e_1 < e_2 : \text{bool} \dashv \{t_1 : t_2\} \cup C_1 \cup C_2}
\end{array}$$

Statements can modify the context. So  $\rightarrow G'$  means the context updated the environment to  $G'$ .

$$\begin{array}{c}
\text{seq} \frac{G \vdash s_1 : () \dashv C_1 \rightarrow G' \quad G' \vdash s_2 : () \dashv C_2 \rightarrow G''}{G \vdash s_1; s_2 : () \dashv C_1 \cup C_2 \rightarrow G''} \quad \text{assign} \frac{G \vdash e : t \dashv C}{G \vdash x = e : () \dashv \{G(x) : t\} \cup C \rightarrow G[x : t]} \\
\\
\text{if} \frac{G \vdash e_1 : t \dashv C_1 \quad G \vdash s_1 : () \dashv C_2 \rightarrow G' \quad G \vdash s_2 : () \dashv C_3 \rightarrow G''}{G \vdash \text{if } e \text{ } s_1 \text{ } s_2 : () \dashv \{t : \text{bool}\} \cup C_1 \cup C_2 \cup C_3 \rightarrow G' \cup G''} \\
\\
\text{for} \frac{G \vdash e_1 : t_1 \dashv C_1 \quad G \vdash e_2 : t_2 \dashv C_2 \quad G \vdash s : () \dashv C_3 \rightarrow G'}{G \vdash \text{for } x \text{ } e_1 \text{ } e_2 \text{ } s : () \dashv \{t_1 : \text{int}, t_2 : \text{int}, G(x) : \text{int}\} \cup C_1 \cup C_2 \cup C_3 \rightarrow G'} \\
\\
\text{while} \frac{G \vdash e : t \dashv C_1 \quad G \vdash s : () \dashv C_2 \rightarrow G'}{G \vdash \text{while } e \text{ } s : () \dashv \{t : \text{bool}\} \cup C_1 \cup C_2 \rightarrow G'} \quad \text{print} \frac{G \vdash e : t \dashv C}{G \vdash \text{printf}(e) : () \dashv C \rightarrow G}
\end{array}$$