Computer Engineering 175
Project Guide

1 A Simple C Compiler

What is Simple C? Simple C is a subset of the C programming language. It has few built-in types, a simplified declaration syntax, a streamlined set of operators, and only simple control constructs.

Is every Simple C program a C program? Yes, Simple C is a true subset of C, so every legal Simple C program is a legal C program. Every Simple C program can be compiled using a traditional C compiler.

Why use a subset of C? Although Pascal is a smaller language and perhaps a better choice for building a first compiler, most students are familiar with the syntax and semantics of C, C++, or Java.

How will I implement my compiler? Your compiler will be implemented in C++, and will generate code for an Intel processor running the Linux operating system.

Why use C++ for implementation? The principles of abstraction, encapsulation, and reuse of tested code (specifically the Standard Template Library) are key in building a large project such as your compiler.

2 Grading

How will my compiler be graded? Your compiler will be graded in several phases. Each phase will have a weight assigned. After each phase, your may continue with your implementation or use the solution provided.

How will each assignment be graded? Grading will be fully automated. Therefore, your compiler ***must*** produce output exactly as indicated in the assignment and online examples. Incorrectly formatted output that is otherwise correct will receive a zero. Example inputs and outputs will be available on Camino for comparison.

How will I submit each assignment? Assignments will be submitted through Camino. A late submission will be ***penalized one point*** for each minute late.

What do I submit? You must submit a `tar` file containing your project directory, which must be named `phase`*n*, and contains a `Makefile` that will produce an executable file called `scc`. The following steps will be used to compile your assignment:

1. `tar xf` *submission*
2. `cd phase`*n*
3. `rm -f *.o scc core`
4. `make`

A shell script will be provided for you to verify that your submission is in the proper format, compiles correctly, and produces correctly formatted output. ***It is your responsibility to verify that your submission is correct.*** An incorrectly submitted assignment, even if otherwise correct, will receive a zero.

How will my compiler work? Your compiler will read input from the standard input, write valid output to the standard output, and write error messages to the standard error: `scc <` *input-file* `>` *output-file* `2>` *error-file*. Your compiler ***must*** work on the Linux machines in the Engineering Computing Center.