Extra Credit Programming Assignment

1 Introduction

This extra credit programming assignment is intended to be a fun project with which to end the quarter. There are two choices for this extra credit assignment:

1. implement some optimizations in your compiler; or
2. write a Cool program.

2 Optimizer Project

Your project should, in addition to code generation, perform some significant optimization of the code. This optimization should be well written, documented, and demonstrated. Two critical factors are

1. correctness (the optimizations don't result in incorrect programs); and
2. the percentage speed-up your optimized code achieves over coolc, as measured by a weighted sum of the instructions executed on spim over a suite of benchmarks.

To find out how many instructions a Cool program executes, run spim with the `-keepstats` option.

There are many possible optimizations to implement; see the ASU chapters 9 and 10 for ideas. Assuming your initial code generator is straightforward (like coolc's), then two directions that may yield significant improvement are (1) improving register usage and (2) specializing the implementation of the basic classes **Int** and **String**.

We have not implemented an optimization phase in coolc, so we have no skeleton code to give you—you are on your own. *Under absolutely no circumstances should you try optimization before your code generator is finished!!*

There is a `-O` flag that controls the global variable **cgen_optimize** (C++) and **Flags.cgen_optimize** (Java). You may use this flag to switch between generating normal code and optimized code.

3 Test Program Project

Write a test program that reveals bugs or strange behavior in Cool compilers (this can include bugs in an optimizing compiler if you chose to implement one). Test cases cannot be based, even loosely, on pre-existing Cool programs, such as those in the `examples` directory.