Milestone #3: Static Single Assignment

Overview
For this milestone, you will augment your compiler to use virtual registers for all local variables and parameters. This will require modifications to the LLVM instruction generation to ensure that all virtual register use is in static single-assignment (SSA) form.

It is recommended that you follow the algorithm of Braun, et. al., to construct LLVM in SSA form with virtual registers as discussed in lecture and described in the paper linked from the course website.

Out-of-SSA
Note that the translation from LLVM to assembly must be updated to include translating “out-of-SSA” by replacing the phi instructions with an appropriate set of copy operations. The most direct approach is to create a new register for each phi, place in each predecessor a copy into that new register from the corresponding operand, and then replace the phi with a copy from the new register into the target of the original phi instruction.

Command-line Options
The output of your compiler should default to generating assembly output based on the LLVM representation with local variables in virtual registers (this milestone). You should also add a -llvm command-line option so that you can view the LLVM representation.

You will also want to maintain the option to use the LLVM representation with stack allocation (from milestone #2, via the -stack option) to allow comparisons between the two representations at the end of the course. Your compiler should support command-line option(s) to dictate which representation is emitted (e.g., without options, default assembly output based on virtual register-based LLVM; -stack, assembly output based on the stack-based LLVM; -llvm, llvm output based on the register-based LLVM; -llvm -stack, llvm output based on the stack-based LLVM).