Simple Compiler v3 Assignment 5 – ARM Back End

Be sure to fully review and comply with the SimpleO Language Specification document for this and all future SimpleCompilerv3 assignments.

In this assignment, be sure to incorporate the new demo files into your working directory. The test cases have also been reorganized into ARM and x86 test cases (they are identical between the two architectures, though).

Using the information provided in the lecture and the following link, implement the ARM backend code into your compiler. Once implemented, all test cases should pass for x86 and ARM. Please note that there are different shell scripts to run for each as shown in the lecture demo.

https://medium.com/@tunacici7/aarch64-procedure-call-standard-aapcs64-abi-calling-conventions-machine-registers-a2c762540278

****All provided test cases must work without error or warning to receive full credit.** These test cases are not comprehensive and there are likely edge cases that may or may not work that are not covered by the test cases. However, the test cases provide sufficient coverage for the functionality requested (and they align with the SimpleO Language Specification document). The test cases directory is cumulative and later assignments include all test cases from the previous assignments to make sure you don't break functionality as you develop your compiler. You may develop your own test cases, though I will only be using the supplied test cases to evaluate your submissions for each assignment.

****Do not modify the MakeFile, shell scripts, file names, or directory structure.** Any modification can lead to it not working in my environment when I grade it and you will not receive credit if I cannot evaluate your compiler. **Only Ubuntu 22.04 LTS is supported.** No other environments have been validated and are not supported.

Be sure to follow a professional coding standard, including the naming of your variables, etc. You should use descriptive names that are appropriate. Nonsensical or whimsical names will lose points.

Additionally, you must attest and add the following honor statement to your Canvas submission (in the Canvas submission comments):

I, <state your name>, attest that this submission is my own work. I have not worked with anyone else on my implementation and have not used AI or other unpermitted sources to help me with my solution, in accordance with our definition of class cheating shown in Lecture 1.

Any submission that does not include this attestation will not receive any credit.