# Compilers

CMPT 432

## Project Four - 200 points

**Project**

1. Projects one, two, and three working perfectly. [50 points]
2. Write a code generator that takes your AST and generates 6502a machine code (found in 6502a-instruction-set.pdf) for our language grammar found on our class web site at https://www.labouseur.com/courses/compilers/grammar.pdf. [150 points]

**Notes and Requirements**

- As with the projects before this one, include verbose output functionality that traces the stages of the parser including the construction of the symbol table and type checking actions.
- When you detect an error report it in helpful detail including where it was found.
- The generated code must conform to the 6502a instructions set specified on our class web site and execute on SvegOS.
- If you're feeling up to it, consider adding one or more of the following for extra credit and extra coolness: code optimization (ask me about it), non-value-returning procedures (sub program call and return), value-returning functions (sub program call and return), integer arrays

**Other Requirements**

Create a plethora of test programs that cause as many different types of errors as you can in order to thoroughly test your code. (You have been thinking about code coverage and edge cases, right?) Include **tons** of test cases that show it working as well. Write up your testing results in a document in your Git repo.

Your code must … [−∞ if not]

- separate structure from presentation.
- be professionally formatted.
- use and demonstrate best practices.
- make me proud to be your teacher.

**Labs**

Lab 8 focuses on some of the components of this project. Lab 9 will help you prepare for the final exam.

**Submitting Your Work**

Make many commits to GitHub. Commit early and often. If you're not already doing this, you've probably failed this course by now.

**Grading Details**

25 points - Lex and Parse still fully working
25 points - Semantic Analysis still fully working
30 points - single scope sequence code generation
30 points - multi-scope sequence code generation
35 points - alternation (if) code generation
35 points - repetition (while) code generation
20 points - advanced cases code generation