**CS323 Documentation**

About 2 pages

1. **Problem Statement**

*As a continuation of all of our previous projects, we are finishing off the compiler by modifying our syntax analysis files to generate assembly code for our Rat32F files.*

1. **How to use your program**

*We have provided an object code compiled file titled* ***main****. In any linux terminal, you may simply type* ***./main*** *to run our code. There will be no issues with dependencies or anything since this file is compiled. This will bring up a command line interface where you can pick text files to convert into assembly. This will be able to find text files so long as they are in your current working directory.*

1. **Design of your program**

*This program modifies the existing syntax analysis code in order to generate lines of assembly. This works by adding instructions to where they would fit in the analysis. For example, if our syntax analyzer finds an “assign” statement, when we know that after the right side of the = is analyzed, we should use POPM to pop the value on the stack into the desired memory location. We did this everywhere necessary in order to properly generate assembly code. For the symbol table, we used a hashmap that maps identifiers to memory locations, starting at 7000 and incrementing for every identifier.*

1. **Any Limitation**

*None*

1. **Any shortcomings**

*None*