Use the `size(1)` command to determine the size of your compiler. How does the size change at run time? Try to characterize the magnitude of the change in storage requirements with a change in the size of the MINILAX program being compiled.

2) Generate a long (several thousand lines) MINILAX program by replicating some block. Make a temporary modification to the lexical analyzer to count the tokens in your program. Compile and assemble this program, use `wc(1)` to obtain statistics on the source and assembly files, and use `size(1)` to determine the size of the object program.

3) Use profiling tools such as `prof(1)` and `gprof(1)` to find out where your compiler is spending its time as it processes your long program. Can you relate the profile data to the various sizes that you obtained in answer to question (2)?

4) If the profile data shows any areas that require a disproportionate amount of time, make a proposal for changing the compiler to reduce this requirement.

5) Point out at least two situations in which the compiler generates obviously poor code, and propose means of improving the code quality. Try to assess the cost of your proposals in term of compiler development effort.

The purpose of this assignment is to introduce you to the process of compiler quality assessment. It is worth 15 points, and is due at the beginning of the lecture on April 29.