CS 5500 – The Structure of a Compiler
HW #4

Helpful Hints

In order to do the required type checking for MIPL programs, you're going to have to "set" type information in various nonterminal functions in order to "communicate" that information to other functions. This may be a bit obscure when processing expressions, which are broken down into **<simple expr>, <term>, <factor>, <add op lst>, <op expr>,** and **<mult op lst>** (not to mention **<variable>, <idx var>, <const>,** and **<bool const>**).

It should be fairly obvious what type a constant or a variable is (look up the latter in your symbol table!). But here are some guidelines for determining the type for some of the other productions:

<expr> → <simple expr> <opExpr>

The type of <expr> is BOOLEAN if <opExpr> is not ε. Otherwise, the type of <expr> is the type of <simple expr>.

<opExpr> → <rel op> <simple expr> | ε

The type of <opExpr> is the type of <simple expr>. If the ε production is applied, then the type of <opExpr> is not applicable.

<simple expr> → <term> <add op lst>

The type of <simple expr> is the type of <term>.

<add op lst> → <add op> <term> <add op lst> | ε

The type of <add op lst> is the type of <term>. If the ε production is applied, then the type of <add op lst> is not applicable. Note: No other function may be interested in a type returned by <add op lst>.

<term> → <factor> <mult op lst>

The type of <term> is the type of <factor>.

<mult op lst> → <mult op> <factor> <mult op lst> | ε

The type of <mult op lst> is the type of <factor>. If the ε production is applied, then the type of <mult op lst> is not applicable. Note: No other function may be interested in a type returned by <mult op lst>.

<factor> → <sign> <variable> | <const> | (<expr>) | not <factor>

The type of <factor> is the type of the <variable>, the <constant>, or the <expr> if one of those productions is applied. If not <factor> is applied, the type of <factor> is BOOLEAN.