| %{ #include "y.tab.h" %}  %% "int" { return INT; } "float" { return FLOAT; } [a-zA-Z\_][a-zA-Z0-9\_]\* { return IDENTIFIER; } [0-9]+ { return INTEGER; } [0-9]+\.[0-9]\*([eE][+-]?[0-9]+)? { return FLOATING\_POINT; } "+" { return PLUS; }  "-" { return MINUS; } "\*" { return MULTIPLY; } "/" { return DIVIDE; } "=" { return ASSIGN; } ";" { return SEMICOLON; } [ \t\n] { /\* ignore whitespace \*/ } . { /\* catch all for any other character \*/ } %% |
| --- |

| %% %class Lexer %unicode %public %type Token  %{  private Symbol symbol(int type) {  return new Symbol(type, yychar, yyline);  } %}  INT = "int" FLOAT = "float" ID = [a-zA-Z\_][a-zA-Z0-9\_]\* INT\_LIT = [0-9]+ FLOAT\_LIT = [0-9]+\.[0-9]\*([eE][+-]?[0-9]+)? WHITESPACE = [ \t\n]  %%  {INT} { return symbol(sym.INT); } {FLOAT} { return symbol(sym.FLOAT); } {ID} { return symbol(sym.ID); } {INT\_LIT} { return symbol(sym.INT\_LIT); } {FLOAT\_LIT} { return symbol(sym.FLOAT\_LIT); } "+" { return symbol(sym.PLUS); } "-" { return symbol(sym.MINUS); } "\*" { return symbol(sym.MULTIPLY); } "/" { return symbol(sym.DIVIDE); } "=" { return symbol(sym.ASSIGN); } ";" { return symbol(sym.SEMICOLON); } {WHITESPACE} { /\* skip whitespace \*/ } . { /\* catch all for any other character \*/ |
| --- |