Our Project Grammar

```
Program             ::== Block $
Block               ::== { StatementList }
StatementList       ::== Statement StatementList
                    ::== ε
Statement           ::== PrintStatement
                    ::== AssignmentStatement
                    ::== VarDecl
                    ::== WhileStatement
                    ::== IfStatement
                    ::== Block
PrintStatement      ::== print ( Expr )
AssignmentStatement ::== Id = Expr
VarDecl             ::== type Id
WhileStatement      ::== while BooleanExpr Block
IfStatement         ::== if BooleanExpr Block

Expr                ::== IntExpr
                    ::== StringExpr
                    ::== BooleanExpr
                    ::== Id
IntExpr             ::== digit intop Expr
                    ::== digit
StringExpr          ::== " CharList "
BooleanExpr         ::== ( Expr boolop Expr )
                    ::== boolval
Id                  ::== char
CharList            ::== char CharList
                    ::== space CharList
                    ::== ε

type                ::== int | string | boolean
char                ::== a | b | c ... z
space               ::== the space character
digit               ::== 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

boolop              ::== == | !=
boolval             ::== false | true
intop               ::== +
```

Curly braces denote new scope.

= is assignment.

== is test for equality.

Comments are bounded by /* and */ and ignored by the lexer.