Programming Language Grammar (EBNF)

Legend:
terminals non-terminals operators

Program Structure
```
program = { function } ;
```

Function Definitions
```
function = "func" ID "(" [ parameters ] ")" block ;
parameters = ID { "," ID } ;
```

Blocks
```
block = "[" { statement } "]" ;
```

Statements
```
statement =
          let_stmt
        | assign_stmt
        | if_stmt
        | while_stmt
        | return_stmt
        | print_stmt
        | expr_stmt
        ;

let_stmt = "let" ID ";" ;
assign_stmt = ID "=" expression ";" ;
expr_stmt = expression ";" ;
if_stmt = "if" expression block "else" block ;
while_stmt = "while" expression block ;
return_stmt = "return" expression ";" ;
print_stmt = "print" expression ";" ;
```

Expression (Pratt)
```
expression = logic_or ;
logic_or = logic_and { "|" logic_and } ;
logic_and = equality { "&" equality } ;
equality = relational { ( "==" | "!=" ) relational } ;
relational = additive { ( "<" | ">" ) additive } ;
additive = multiplicative { ( "+" | "-" ) multiplicative } ;
multiplicative = unary { ( "*" | "/" ) unary } ;
unary = ( "!" | "-" ) unary | primary ;
primary =
```

```
            INT
            | BOOL
            | ID
            | function_call
            | "(" expression ")"
            ;
function_call = ID "(" [ arguments ] ")" ;
arguments = expression { "," expression } ;
```