Scanner grammar is

$$\begin{aligned} \text{id} &::= \text{letter}\,(\text{letter}|\,\text{digit}|\,\_) \\ \text{num} &::= \text{decimal}^* \end{aligned}$$

Factored Parser grammar:

Program ::= ClassDeclaration* eot

ClassDeclaration ::= class id { (

Visibility Access (

Type id (; | (ParameterList?) {Statement*})

|

void id (ParameterList?) {Statement*}

)

)* }

FieldDeclaration ::= Visibility Access Type id ;

MethodDeclaration ::= Visibility Access (Type | void) id (ParameterList?) {Statement*}

Visibility ::= (public | private)?

Access ::= static ?

Type ::= boolean | id (ε | []) | int (ε | [])

ParameterList ::= Type id (, Type id)*

ArgumentList ::= Expression (, Expression)*

Reference ::= (id | this) (. id)*

Statement ::=

{ Statement* }

| Type id = Expression ;

| Reference (= Expression ;

| [Expression] = Expression ;

| (ArgumentList?) ;)

| return Expression? ;

| if (Expression) Statement (else Statement)?

| while (Expression) Statement

Expression ::=

Reference (ε | [Expression] | (ArgumentList?))

|

| unop Expression

| Expression binop Expression

| (Expression)

| num | true | false

| new (id () | int [Expression] | id [Expression])