BNF for FJ.jj

NON-TERMINALS

```
                         Goal ::= TypeDeclaration <EOF>
              TypeDeclaration ::= "class" Identifier "extends" ExtendedType "{" ( VarDeclaration )*
                                  ClassConstructor ( MethodDeclaration )* "}"
                 ExtendedType ::= Identifier
                                | ObjectIdentifier
               VarDeclaration ::= Type Identifier ";"
             ClassConstructor ::= Type "(" ( FormalParameterList )? ")" "{" "super"
                                  "(" ( ExpressionList )? ")" ";" ( FieldAssign )* "}"
                  FieldAssign ::= ThisIdentifier "." Identifier "=" Identifier ";"
            MethodDeclaration ::= BinaryOpOverloadDeclaration
                                | DefaultMethodDeclaration
     DefaultMethodDeclaration ::= Type Identifier "(" ( FormalParameterList )? ")" "{" "return"
                                  Expression ";" "}"
  BinaryOpOverloadDeclaration ::= "static" Type "operator " BinaryOperator "(" Type Identifier ","
                                  Type Identifier ")" "{" "return" Expression ";" "}"
               BinaryOperator ::= "+"
                                | "-"
                                | "*"
                                | "/"
          FormalParameterList ::= FormalParameter ( FormalParameterRest )*
              FormalParameter ::= Type Identifier
          FormalParameterRest ::= "," FormalParameter
                         Type ::= IntegerType
                                | Identifier
                                | ObjectIdentifier
                  IntegerType ::= "int"
                   Expression ::= Term ( PlusExpressionRest | MinusExpressionRest )*
           PlusExpressionRest ::= "+" Term
```

```
 MinusExpressionRest ::= "-" Term
                Term ::= PrimaryExpression ( TimesExpressionRest | DivideExpressionRest )
                         *
 TimesExpressionRest ::= "*" PrimaryExpression
DivideExpressionRest ::= "/" PrimaryExpression
   PrimaryExpression ::= IntegerLiteral
                       | MethodInvoke
                       | FieldInvoke
                       | Identifier
                       | AllocationExpression
                       | CastExpression
                       | NestedExpression
        MethodInvoke ::= ( AllocationExpression | NestedExpression | Identifier ) "." Identifier
                         "(" ( ExpressionList )? ")"
         FieldInvoke ::= ( AllocationExpression | NestedExpression | Identifier ) "." Identifier
AllocationExpression ::= "new" Identifier "(" ( ExpressionList )? ")"
      CastExpression ::= "(" Type ")" PrimaryExpression
    NestedExpression ::= "(" Expression ")"
      ExpressionList ::= Expression ( ExpressionRest )*
      ExpressionRest ::= "," Expression
      IntegerLiteral ::= <INTEGER_LITERAL>
          Identifier ::= <IDENTIFIER>
      ThisIdentifier ::= "this"
    ObjectIdentifier ::= "Object"
```