

BNF for minijava.jj

NON-TERMINALS

Goal ::= [MainClass](#) ([TypeDeclaration](#)) * <EOF>

MainClass ::= "class" [Identifier](#) "{" "public" "static" "void" "main" "(" "String" "[" "]" [Identifier](#) ")" "{" ([VarDeclaration](#)) * ([Statement](#)) * "}" "}"

TypeDeclaration ::= [ClassDeclaration](#)
| [ClassExtendsDeclaration](#)

ClassDeclaration ::= "class" [Identifier](#) "{" ([VarDeclaration](#)) * ([MethodDeclaration](#)) * "}"

ClassExtendsDeclaration ::= "class" [Identifier](#) "extends" [Identifier](#) "{" ([VarDeclaration](#)) * ([MethodDeclaration](#)) * "}"

VarDeclaration ::= [Type](#) [Identifier](#) ";;"

MethodDeclaration ::= "public" [Type](#) [Identifier](#) "(" ([FormalParameterList](#)) ? ")" "{" ([VarDeclaration](#)) * ([Statement](#)) * "return" [Expression](#) ";;" "}"

FormalParameterList ::= [FormalParameter](#) [FormalParameterTail](#)

FormalParameter ::= [Type](#) [Identifier](#)

FormalParameterTail ::= ([FormalParameterTerm](#)) *

FormalParameterTerm ::= ";;" [FormalParameter](#)

Type ::= [ArrayType](#)
| [BooleanType](#)
| [IntegerType](#)
| [Identifier](#)

ArrayType ::= [BooleanArrayType](#)
| [IntegerArrayType](#)

BooleanArrayType ::= "boolean" "[" "]"

IntegerArrayType ::= "int" "[" "]"

BooleanType ::= "boolean"

IntegerType ::= "int"

Statement ::= [Block](#)
| [AssignmentStatement](#)
| [ArrayAssignmentStatement](#)
| [IfStatement](#)
| [WhileStatement](#)
| [PrintStatement](#)

Block ::= "{" ([Statement](#)) * "}"

AssignmentStatement ::= [Identifier](#) "=" [Expression](#) ";;"

ArrayAssignmentStatement ::= [Identifier](#) "[" [Expression](#) "]" "=" [Expression](#) ";;"

IfStatement ::= "if" "(" [Expression](#) ")" [Statement](#) "else" [Statement](#)

WhileStatement ::= "while" "(" [Expression](#) ")" [Statement](#)

PrintStatement ::= "System.out.println" "(" [Expression](#) ")" ";;"

Expression ::= [AndExpression](#)
| [CompareExpression](#)
| [PlusExpression](#)
| [MinusExpression](#)
| [TimesExpression](#)

| [ArrayLookup](#)
| [ArrayLength](#)
| [MessageSend](#)
| [Clause](#)

AndExpression ::= [Clause](#) "&" [Clause](#)

CompareExpression ::= [PrimaryExpression](#) "<" [PrimaryExpression](#)

PlusExpression ::= [PrimaryExpression](#) "+" [PrimaryExpression](#)

MinusExpression ::= [PrimaryExpression](#) "-" [PrimaryExpression](#)

TimesExpression ::= [PrimaryExpression](#) "*" [PrimaryExpression](#)

ArrayLookup ::= [PrimaryExpression](#) "[" [PrimaryExpression](#) "]"

ArrayLength ::= [PrimaryExpression](#) "." "length"
 MessageSend ::= [PrimaryExpression](#) "." [Identifier](#) "(" ([ExpressionList](#))? ")"
 ExpressionList ::= [Expression](#) [ExpressionTail](#)
 ExpressionTail ::= ([ExpressionTerm](#))
 ExpressionTerm ::= "," [Expression](#)
 Clause ::= [NotExpression](#)
 | [PrimaryExpression](#)
 PrimaryExpression ::= [IntegerLiteral](#)
 | [TrueLiteral](#)
 | [FalseLiteral](#)
 | [Identifier](#)
 | [ThisExpression](#)
 | [ArrayAllocationExpression](#)
 | [AllocationExpression](#)
 | [BracketExpression](#)
 IntegerLiteral ::= <INTEGER_LITERAL>
 TrueLiteral ::= "true"
 FalseLiteral ::= "false"
 Identifier ::= <IDENTIFIER>
 ThisExpression ::= "this"
 ArrayAllocationExpression ::= [BooleanArrayAllocationExpression](#)
 | [IntegerArrayAllocationExpression](#)
 BooleanArrayAllocationExpression ::= "new" "boolean" "[" [Expression](#) "]"
 IntegerArrayAllocationExpression ::= "new" "int" "[" [Expression](#) "]"
 AllocationExpression ::= "new" [Identifier](#) "(" " "
 NotExpression ::= "!" [Clause](#)
 BracketExpression ::= "(" [Expression](#) ")"