Location(int line):

* ArrayLocation (Expression array, Expression index)
* VariableLocation(int line, String name)

OR VariableLocation(int line, Expression location, String name) [??]

Statement(int line):

* Assignment (Location variable, Expression assignment)
* Break(int line)
* CallStatment(Call call)
* Continue(int line)
* If(Expression condition, Statement operation, Statement elseOperation)
* LocalVariable(Type type, String name)
* Return(int line) OR Return(int line, Expression value)
* StatementBlock(int line, List<Statement> statements) [??]
* While(Expression condition, Statement operation)

Expression(int line):

* BinaryOp(Expression operand1, BinaryOps operator, Expression operand2) [abstract]
* LogicalBinaryOp( … )
* MathBinaryOp(…)
* UnaryOp(UnaryOps operator, Expression operand) [abstract]
* LogicalUnaryOp(…)
* MathUnaryOp(…)
* New(int line) [abstract]
* NewArray(Type type, Expression size)
* NewClass(int line, String name)
* Call(int line, String name, List<Expression> arguments)
* StaticCall(int line, String className, String name, List<Expression> arguments)
* VirtualCall(int line, Expression location, String name, List<Expression> arguemtns) // ,ay pass location as null if called without instance prefix
* ExpressionBlock(Expression exp) // for ( )
* Length(Expression array)
* Literal(int line, LiteralTypes type) // for TRUE, FALSE, NULL

Literal(int line, LiteralTypes type, Object value) // for INTEGER, STRING

* This(int line)

Method (Type type, String name, List<Formal> formals, List<Statement> statements):

* LibraryMethod(Type type, String name, List<Formal> formals) // type includes void
* StaticMethod(Type type, String name, List<Formal> formals, List<Statement> statements)
* VirtualMethod(Type type, String name, List<Formal> formals, List<Statement> statements)

Field(Type type, String name)

Formal(Type type, String name)

ICClass(int line, String name, List<Field> fields, List<Method> methods)

Type(int line) [Abstract] has incrementDimension method

* PrimitiveType(int line, DataTypes type) // possibly void
* UserType(int line, String name) // for Class types

Program(List<ICClass> classes) // super gets 0 as line number

ASTNode (int line) [Root class] :