May 8, 2014

Jani Vihervä̈s
jani.vihervas@cs.helsinki.fi

# Mini-Java Compiler

## 582648 Code Generation

http://www.cs.helsinki.fi/u/vihavain/k12/compiler_project/project/compiler_project_2012.html

# 1 Overview

# 2 Grammar

$$
\begin{array}{rcl}
<program> & \rightarrow & <main\ class><class\ declaration>* \\
<main\ class> & \rightarrow & \textbf{class} <new\ identifier> \textbf{\{public static void main() \{} <statement>* \textbf{\}\}} \\
<class\ declaration> & \rightarrow & \textbf{class} <new\ identifier> [\textbf{extends} <identifier>]\textbf{\{} <declaration>* \textbf{\}} \\
<declaration> & \rightarrow & <variable\ declaration> \mid <method\ declaration> \\
\hline
<method\ declaration> & \rightarrow & \textbf{public} <type><new\ identifier> ([<formals>]) \textbf{\{} <statement>* \textbf{\}} \\
<variable\ declaration> & \rightarrow & <type><new\ identifier><variable\ assignment> \textbf{;} \\
<variable\ assignment> & \rightarrow & \epsilon \mid \textbf{=} <expr> \\
<formals> & \rightarrow & <type><new\ identifier> (\textbf{,} <type><new\ identifier>)* \\
<type> & \rightarrow & <simple\ type><array\ type> \\
<simple\ type> & \rightarrow & \textbf{int} \mid \textbf{boolean} \mid \textbf{void} \mid <type\ identifier> \\
<array\ type> & \rightarrow & \epsilon \mid \textbf{[ ]} \\
<type\ identifier> & \rightarrow & <identifier> \\
\hline
<statement> & \rightarrow & \textbf{assert (} <expr> \textbf{);} \\
 & \mid & <local\ variable\ declaration> \\
 & \mid & \textbf{\{}<\textbf{statement}>\textbf{* \}} \\
 & \mid & \textbf{if (} <expr> \textbf{)} <statement><else> \\
 & \mid & \textbf{while (} <expr> \textbf{)} <statement> \\
 & \mid & \textbf{System.out.println (} <expr> \textbf{);} \\
 & \mid & <identifier> [\textbf{[} <epxr> \textbf{]}] \textbf{=} <expr> \textbf{;} \\
 & \mid & \textbf{return} <expr> \textbf{;} \\
 & \mid & <method\ invocation> \textbf{;} \\
<else> & \rightarrow & \epsilon \mid \textbf{else} <statement> \\
<local\ variable\ declaration> & \rightarrow & <variable\ declaration> \\
<method\ invocation> & \rightarrow & <expr1> \textbf{.} <method\ tail> \\
<method\ tail> & \rightarrow & \textbf{length} \mid <identifier> ([<expr> (\textbf{,} <expr>)*]) \\
\hline
<expr> & \rightarrow & <expr1><expr2> \\
<expr1> & \rightarrow & \textbf{new} <new> \\
 & \mid & \textbf{!} <expr> \\
 & \mid & \textbf{-} <expr> \\
 & \mid & \textbf{(} <expr> \textbf{)} \\
 & \mid & <identifier> \mid <integer\ literal> \\
 & \mid & \textbf{this} \mid \textbf{true} \mid \textbf{false} \\
<expr2> & \rightarrow & \epsilon \mid \textbf{[} <expr> \textbf{]} \mid \textbf{.} <method\ tail> \\
 & \mid & <binary\ operator><expr> \\
<new> & \rightarrow & <simple\ type> \textbf{[} <expr> \textbf{]} \mid <type\ identifier> \textbf{( )} \\
<binary\ operator> & \rightarrow & \&\& \mid \mid\mid \mid < \mid > \mid == \mid + \mid - \mid * \mid / \mid \%
\end{array}
$$