CS 365 — Programming Language Concepts

Pointers and Arrays

Feb 29, 2008

PoiJay Concrete Syntax (abreviated)

$$
\begin{array}{rcl}
\mathit{Type} & \rightarrow & \mathsf{int} \mid \mathsf{boolean} \mid \text{`}\ast\text{'}\ \mathit{Type} \\
\mathit{Factor} & \rightarrow & \mathit{Identifier} \mid \mathit{Literal} \mid (\ \mathit{Expression}\ ) \mid \\
 & & \&\ \mathit{Identifier} \mid \text{`}\ast\text{'}\ \mathit{Expression} \\
\mathit{Assignment} & \rightarrow & \text{`}\ast\text{'}\ \ast\ \mathit{Identifier} = \mathit{Expression}
\end{array}
$$

'RayJay Concrete Syntax

Program	$\longrightarrow$	*Declarations* { *Method* }* void main () '{' *Declarations Statements* '}'
Method	$\longrightarrow$	*Type Identifier* ([*Parameters*]) '{' *Declarations Statements* '}'
Parameters	$\longrightarrow$	*Parameter* { , *Parameter* }
Parameter	$\longrightarrow$	*Type Identifier*
Declarations	$\longrightarrow$	{ *Declaration* }*
Declaration	$\longrightarrow$	*Type Identifiers;*
Type	$\longrightarrow$	void | *PrimitiveType* | *ArrayType*
PrimitiveType	$\longrightarrow$	int | boolean
ArrayType	$\longrightarrow$	*PrimitiveType* { [] } +
Identifiers	$\longrightarrow$	*Identifer* { , *Identifier* }*
Statements	$\longrightarrow$	{ *Statement* }*
Statement	$\longrightarrow$	; | *Block* | *Assignment* | *IfStatement* | *WhileStatement* | *PrintStatement* | *CallStatement* | *ReturnStatement*
Block	$\longrightarrow$	'{' *Declarations Statements* '}'
Assignment	$\longrightarrow$	*Identifier* = *Expression* ;
IfStatement	$\longrightarrow$	if (*Expression*) *Statement* { else *Statement* }$_{opt}$
WhileStatement	$\longrightarrow$	while (*Experssion*) *Statement*
PrintStatement	$\longrightarrow$	System.out.println (*Expression*) ;

'RayJay Concrete Syntax, continued

$$
\begin{array}{rcl}
\textit{CallStatement} & \rightarrow & \textit{Identifier}\ (\ [\ \textit{Arguments}\ ]\ );\\
\textit{Arguments} & \rightarrow & \textit{Expression}\ \{\ ,\ \textit{Expression}\ \}*\\
\textit{ReturnStatement} & \rightarrow & \texttt{return}\ \textit{Expression}\ ;\\
\textit{Expression} & \rightarrow & \textit{Conjunction}\ \{\ \texttt{||}\ \textit{Conjunction}\ \}*\\
\textit{Conjunction} & \rightarrow & \textit{Relation}\ \{\ \&\&\ \textit{Relation}\ \}*\\
\textit{Relation} & \rightarrow & \textit{Addition}\ \{\ [\ |\ <\ |\ <=\ |\ >\ |\ >=\ |\ ==\ |\ !\ =]\ \textit{Addition}\ \}_{opt}\\
\textit{Addition} & \rightarrow & \textit{Term}\ \{\ [\ +\ |\ \text{-}]\ \textit{Term}\ \}*\\
\textit{Term} & \rightarrow & \textit{Negation}\ \{\ [\ \text{'*'}\ |\ /]\ \textit{Negation}\ \}*\\
\textit{Negation} & \rightarrow & \{\ !\ \}_{opt}\ \textit{Factor}\\
\textit{Factor} & \rightarrow & \textit{Identifier}\ |\ \textit{Literal}\ |\ (\ \textit{Expression}\ )\ |\ \textit{Call}\ |\\
 & & \underline{\textit{IndexedVariable}}\ |\ \underline{\textit{Creation}}\\
\textit{Call} & \rightarrow & \textit{Identifier}\ (\ [\ \textit{Arguments}\ ]\ )\\
\underline{\textit{IndexedVariable}} & \rightarrow & \underline{\textit{Identifier}\ \{\ [\ \textit{Expression}]\ \}\ \texttt{+}}\\
\underline{\textit{Creation}} & \rightarrow & \underline{\texttt{new}\ \textit{PrimitiveType}\ \{\ [\ \textit{Expression}\ ]\ \}\ \texttt{+}}
\end{array}
$$

'RayJay Abstract Syntax

Program	$\longrightarrow$	*Declaration*$*$ *Method*$*$ *Block*
Method	$\longrightarrow$	**Type Identifier** *Parameter*$*$ *Block*
Parameter	$\longrightarrow$	**Type Identifier**
Declaration	$\longrightarrow$	**Type Identifier**$*$
Type	$\longrightarrow$	*PrimitiveType* $\mid$ *ArrayType*
PrimitiveType	$\longrightarrow$	**PrimType**
ArrayType	$\longrightarrow$	**PrimType Dimensions**
Statement	$\longrightarrow$	*Skip* $\mid$ *Block* $\mid$ *Assignment* $\mid$ *Conditional* $\mid$ *Loop* $\mid$ *Print* $\mid$ *CallStmt* $\mid$ *Return*
Skip	$\longrightarrow$	
Block	$\longrightarrow$	*Declaration*$*$ *Statement*$*$
Assignment	$\longrightarrow$	*Identifier Expression*
Conditional	$\longrightarrow$	*Expression Statement Statement*
Loop	$\longrightarrow$	*Expression Statement*
Print	$\longrightarrow$	*Expression*
CallStmt	$\longrightarrow$	**Identifier** *Expression*$*$
Return	$\longrightarrow$	*Expression*

'RayJay Abstract Syntax, continued

Expression	→	*Variable* | *IntLitExpr* | *BoolLitExpr* | *BinaryExpr* | *UnaryExpr* | *Call* | *IndexedVariable* | *Creation*
Variable	→	**Identifier**
IntLitExpr	→	**IntLiteral**
BoolLitExpr	→	**BoolLiteral**
BinaryExpr	→	*Expression* **Operator** *Expression*
UnaryExpr	→	**Operator** *Expression*
Call	→	**Identifier** *Expression*∗
IndexedVariable	→	**Identifier** *Expression*∗
Creation	→	**PrimType** *Expression*∗