CS 365 — Programming Language Concepts

Methods (Functions)

Feb 22, 2008

FunJay 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	$\longrightarrow$	int | boolean | void
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*) ;
CallStatement	$\longrightarrow$	*Identifier* ([*Arguments*]);
Arguments	$\longrightarrow$	*Expression* { , *Expression* }*
ReturnStatement	$\longrightarrow$	return *Expression* ;

FunJay Concrete Syntax, continued

Expression	$\longrightarrow$	*Conjunction* { `||` *Conjunction* }*
Conjunction	$\longrightarrow$	*Relation* { `&&` *Relation* }*
Relation	$\longrightarrow$	*Addition* { [| < | <= | > | >= | == | ! =] *Addition* }$_{opt}$
Addition	$\longrightarrow$	*Term* { [+ | -] *Term* }*
Term	$\longrightarrow$	*Negation* { ['*' | /] *Negation* }*
Negation	$\longrightarrow$	{ ! }$_{opt}$ *Factor*
Factor	$\longrightarrow$	*Identifier* | *Literal* | (*Expression*) | *Call*
Call	$\longrightarrow$	*Identifier* ([*Arguments*])

FunJay Abstract Syntax

Program	$\longrightarrow$	*Declaration* Method* Block*
Method	$\longrightarrow$	**Type Identifier** *Parameter* Block*
Parameter	$\longrightarrow$	**Type Identifier**
Declaration	$\longrightarrow$	**Type Identifier***
Statement	$\longrightarrow$	*Skip* | *Block* | *Assignment* | *Conditional* | *Loop* | *Print* | *CallStmt* | *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*

FunJay Abstract Syntax, continued

Expression	$\longrightarrow$	*Variable* | *IntLitExpr* | *BoolLitExpr* | *BinaryExpr* | *UnaryExpr* | *Call*
Variable	$\longrightarrow$	**Identifier**
IntLitExpr	$\longrightarrow$	**IntLiteral**
BoolLitExpr	$\longrightarrow$	**BoolLiteral**
BinaryExpr	$\longrightarrow$	*Expression* **Operator** *Expression*
UnaryExpr	$\longrightarrow$	**Operator** *Expression*
Call	$\longrightarrow$	**Identifier** *Expression*$*$