Program	::=	VarDecl* ProcedureDecl*
VarDecl	::=	‘int’ ID ‘;’
ProcedureDecl	::=	‘int’ ID ‘(’ $\overline{\text{FormalParam}}$ ‘)’ $\overline{\text{Prepost}}$
		‘{’ Stmt* ‘return’ Expr ‘;’ ‘}’
FormalParam	::=	‘int’ ID
Prepost	::=	Requires | Ensures | CandidateRequires | CandidateEnsures
Requires	::=	‘requires’ Expr
Ensures	::=	‘ensures’ Expr
Stmt	::=	VarDecl | AssignStmt | AssertStmt | AssumeStmt | HavocStmt |
		CallStmt | IfStmt | WhileStmt | BlockStmt
AssignStmt	::=	ID ‘=’ Expr ‘;’
AssertStmt	::=	‘assert’ Expr ‘;’
AssumeStmt	::=	‘assume’ Expr ‘;’
HavocStmt	::=	‘havoc’ ID ‘;’
IfStmt	::=	‘if’ ‘(’ Expr ‘)’ BlockStmt (‘else’ BlockStmt)$^?$
BlockStmt	::=	‘{’ Stmt* ‘}’
Expr	::=	Expr ‘?’ Expr ‘:’ Expr | Expr BinaryOp Expr | UnaryOp Expr |
		non-negative decimal integer | ID | ‘(’ Expr ‘)’ |
		‘\result’ | ‘\old’ ‘(’ ID ‘)’
BinaryOp	::=	‘||’ | ‘&&’ | ‘|’ | ‘^’ | ‘&’ | ‘==’ | ‘!=’ |
		‘<’ | ‘<=’ | ‘>’ | ‘>=’ | ‘<<’ | ‘>>’ |
		‘+’ | ‘-’ | ‘*’ | ‘/’ | ‘%’
UnaryOp	::=	‘+’ | ‘-’ | ‘!’ | ‘~’
ID	::=	*any legal C identifier*
CandidateRequires	::=	‘candidate_requires’ Expr
CandidateEnsures	::=	‘candidate_ensures’ Expr
CallStmt	::=	ID ‘=’ ID ‘(’ $\overline{\text{Expr}}$ ‘)’ ‘;’
WhileStmt	::=	‘while’ ‘(’ Expr ‘)’ $\overline{\text{LoopInvariant}}$ BlockStmt
LoopInvariant	::=	Invariant | CandidateInvariant
Invariant	::=	‘invariant’ Expr
CandidateInvariant	::=	‘candidate_invariant’ Expr