C-- Grammar

Tokens

INT	→	/* A sequence of digits without spaces[1] */
FLOAT	→	/* A real number consisting of digits and one decimal point. The decimal point must be surrounded by at least one digit[2] */
ID	→	/* A character string consisting of 52 upper and lower case alphabetic characters, the 10 digits and the underscore. In addition, identifiers must not start with a digit[3] */
SEMI	→	;
COMMA	→	,
ASSIGNOP	→	=
RELOP	→	> | < | >= | <= | == | !=
PLUS	→	+
MINUS	→	–
STAR	→	*
DIV	→	/
AND	→	&&
OR	→	||
DOT	→	.
NOT	→	!
TYPE	→	int | float
LP	→	(
RP	→	)
LB	→	[
RB	→	]
LC	→	{
RC	→	}
STRUCT	→	struct
RETURN	→	return
IF	→	if
ELSE	→	else
WHILE	→	while

High-Level Definitions

Program	→	ExtDefList
ExtDefList	→	ExtDef ExtDefList
	|	ε

[1] You have to work out how to represent integers, floating point numbers and identifiers using regular expression by yourself. For convenience, you can always refuse to accept integers that are larger than 32bits.

[2] For convenience, you can always refuse to accept numbers that cannot be converted into a float constant in C.

[3] For convenience, you can always assume that the length of identifiers will never exceed 32.

ExtDef	→	Specifier ExtDecList SEMI
	|	Specifier SEMI
	|	Specifier FunDec CompSt
ExtDecList	→	VarDec
	|	VarDec COMMA ExtDecList

Specifiers

Specifier	→	TYPE
	|	StructSpecifier
StructSpecifier	→	STRUCT OptTag LC DefList RC
	|	STRUCT Tag
OptTag	→	ID
	|	ε
Tag	→	ID

Declarators

VarDec	→	ID
	|	VarDec LB INT RB
FunDec	→	ID LP VarList RP
	|	ID LP RP
VarList	→	ParamDec COMMA VarList
	|	ParamDec
ParamDec	→	Specifier VarDec

Statements

CompSt	→	LC DefList StmtList RC
StmtList	→	Stmt StmtList
	|	ε
Stmt	→	Exp SEMI
	|	CompSt
	|	RETURN Exp SEMI
	|	IF LP Exp RP Stmt
	|	IF LP Exp RP Stmt ELSE Stmt
	|	WHILE LP Exp RP Stmt

Local Definitions

DefList	→	Def DefList
	|	ε
Def	→	Specifier DecList SEMI
DecList	→	Dec

	| Dec COMMA DecList
Dec	→ VarDec
	| VarDec ASSIGNOP Exp

Expressions

Exp	→	Exp ASSIGNOP Exp
	|	Exp AND Exp
	|	Exp OR Exp
	|	Exp RELOP Exp
	|	Exp PLUS Exp
	|	Exp MINUS Exp
	|	Exp STAR Exp
	|	Exp DIV Exp
	|	LP Exp RP
	|	MINUS Exp
	|	NOT Exp
	|	ID LP Args RP
	|	ID LP RP
	|	Exp LB Exp RB
	|	Exp DOT ID
	|	ID
	|	INT
	|	FLOAT
Args	→	Exp COMMA Args
	|	Exp