关于您给出的 BNF 中的错误, 我们在您的 BNF 中标出, 并且在下面给出例子

program → external_decls
external_decls → declaration external_decls ■ function_def
declaration → modifier type_name varlist; //这里 type_name 可以是 void,而 varlist 中的项目可以是一个 ident.所以可以支持这样的声明:void a. 这样的声明在 gcc 中是不能通过的.
modifier → extern ■ register ■ epsilon
type_name → void ■ int ■ float ■ char
var_list → varlist , var_item ■ var_item
var_item → array_var ■ scalar_var ■ * scalar_var
array_var → ident [iconstant]
scalar_var→ ident ■ ident (parm_type_list)
function_def → functionhdr { function_body)
function_hdr → type_name ident (parm_type_list)
| type_name * ident (parm_type_list)
| ident (parm_type_list)
parm_type_list → void ■ parmlist
parm_list → parmlist , parm_decl ■ parm_decl
parm_decl → type_name ident ■ type_name * ident
function_body → internal_decls statement_list
internal_decls → declaration internal_decls ■ epsilon
statement_list → statement statement_list ■ epsilon
statement→ compoundstmt ■ nullstmt ■ expression_stmt ■ ifstmt
| forstmt ■ whilestmt ■ return_stmt
compoundstmt → { statement_list }
nullstmt → ;
expression_stmt → expression ;
ifstmt → if (expression) statement
| if (expression) statement else statement
for_st mt → for (expression ; expression ; expression) statement
while_stmt → while (expression) statement
return_stmt → return expression ; ■ return ;
expression → assignment_expr
assignment_expr→ unary_expr = expression ■ binary_expr
binary_expr → binary_expr binary_op unary_expr ■ unary_expr
binary_op → boolean_op ■ rel_op ■ arith_op
boolean_op → && ■ ||
rel_op → == ■ != ■ > ■ < ■ >= ■ <=
arith_op → + ■ - ■ * ■ /
unary_expr → unary_op unary_expr ■ postfix_expr
unary_op → ! ■ + ■ - ■ ++ ■ -- ■ & ■ *
postfix_expr → postfix_expr (expression)
| ident (argumentlist)
| ident ( )
| postfix_expr ++
| postfix_expr --
primary_expr
//在前面个,unary_expr 是可以作为表达式左值的,但是这里的 unary_expr 的形式有很多是不能作为赋值表达式左值的. 比如！A 这样的表达式,是不能被赋值的. 另外,unary_expr 的产生式中,可以有单元操作符在一个 postfix_expr 前面嵌套定义,但是++这个符号,是不能嵌套定义的,比如 ++ ++ ++a 这种表达式在 C 中是错误的, 因为++a 实际上相当于表达式 a = a + 1. 这个表达式是不能在

执行＋＋的.
在比如,postfix_expr 表带式中,很多都不能作为左值,比如 ident()表示一个函数返回值,就不能作为左值.但是如果将它从这个表达式去掉,它却可以做右值.那么在很多表达式中它是无法表示的.
primary_expr → ident ■ constant ■ (expression)
constant → iconstant ■ fconstant ■ char_constant ■ string_constant
argument_list → argument_list , expression ■ expression

以上就是两个最严重的错误,一个是 void a 这种 void 类型的示例的声明,另一个是左值和右值没有非清楚.我们的 bnf 中改正了后一种错误,前一种错误希望您给我们关于类型检查的要求.