GRAMMAR OUTLINE

➤ ExpL Grammar outline

Annotated ExpL Grammar outline

An outline for the ExpL grammar is given here. Calls to functions that update the symbol table, type table and the abstract syntax tree data structures are indicated as semantic actions at certain places.

- 1. Program :
- TypeDefBlock
- GDeclBlock
- FDefBlock
- MainBlock

```
TypeDefBlock  : TYPE TypeDefList ENDTYPE
              |
              ;

TypeDefList   : TypeDefList TypeDef
              | TypeDef
              ;

TypeDef       : ID '{' FieldDeclList '}'   { TInstall(tname,size,$3); }
              ;

FieldDeclList : FieldDeclList FieldDecl
              | FieldDecl
              ;

FieldDecl    : TypeName ID ';'

TypeName     : INT
             | STR
             | ID       //TypeName for user-defined types
```

```
21              ;
```

Grammar_outline_TypeDef hosted with ❤ by **GitHub** **view raw**

```
1  GDeclBlock : DECL GDeclList ENDDECL
2             |
3             ;
4
5  GDeclList  : GDecList GDecl
6             | GDecl
7             ;
8
9  GDecl      : TypeName Gidlist ';'
10            ;
11
12 Gidlist    : Gidlist ',' Gid
13            |   Gid
14            ;
15
16 Gid        :   ID                      { GInstall(varname,ttableptr, 1,
17            |   ID '(' ParamList ')'      { GInstall(varname,ttableptr, 0
18            |   ID '[' NUM ']'          { GInstall(varname,ttableptr, $3,
19            ;
20
21 ParamList    :  ParamList ',' Param  { AppendParamlist($1,$2);}
22            |  Param
23            |  //There can be functions with no parameters
24            ;
25
26 Param        : TypeName ID { CreateParamlist($1,$2); }
27            ;
28 //NOTE 1: The second argument to the function Ginstall() must be a point
29        2: The functions CreateParamlist() and AppendParamlist() help to
30           of parameters specified in an ExpL function declaration. Desig
```

Grammar_outline_GDeclBlock hosted with ❤ by **GitHub** **view raw**

```
1  FDefList  : FDefBlock
2            | FDefList FDefBlock
3            ;
4  FDefBlock : TypeName ID '(' ParamList ')' '{' LdeclBlock Body '}'  { GUp
5            ;
6
7  Body      : BEGIN Slist Retstmt END
8            ;
9
10 Slist     : Slist Stmt
```

```
11          |
12          ;
13
14  Stmt    : ID ASGN Expr ';'
15          | ....
16          | IF '(' Expr ')' THEN Slist ELSE Slist ENDIF ';'
17          | ...
18          | ID ASGN ALLOC'(' ')' ';'
19          | FIELD ASGN ALLOC'(' ')' ';'
20          | FREE '(' ID ')' ';'
21          | FREE '(' FIELD ')' ';'
22          | READ '(' ID ')' ';'
23          | READ '(' FIELD ')' ';'
24          | WRITE '(' Expr ')' ';'
25          ;
26
27  FIELD   : ID '.' ID
28          | FIELD '.' ID
29          ;
30
31  Expr    : Expr PLUS Expr   { $$ = TreeCreate(TLookup("int"),NODETYPE_P
32          | ....
33          | '(' Expr ')'
34          | NUM
35          | ID
36          | ID '[' Expr ']'
37          | FIELD
38          | ID '(' ArgList ')'  {
39                                  gtemp = GLookup($1->name);
40                                  if(gtemp == NULL){
41                                          yyerror("Yacc : Undefined functi
42                                  }
43                                  $$ = TreeCreate(gtemp->type,NODETYPE_FUN
44                                  $$->Gentry = gtemp;
45                                }
46          ;
```

Grammar_outline_FDefBlock hosted with ❤ by **GitHub** **view raw**

```
1   MainBlock : INT MAIN '(' ')' '{' LdeclBlock Body '}'
2                                   {
3                                       type = TLookup("int");
4                                       gtemp = GInstall("MAIN",type,0,NULL)
5                                       //...Some more work to be done
9                                   }
10          ;
```

Grammar_outline_MainBlock hosted with ❤ by **GitHub** **view raw**

Github

Contributed By : Thallam Sai Sree Datta, N Ruthvik

Home | About