ANALYSE SEMANTIQUE

- **Etape clé** dans le processus de compilation, intervenant après l'analyse syntaxique.
- Cet analyse s'assure que le code est **logiquement cohérent** et signifie ce qu'il est censé signifier.
- Elle garantit que le programme est non seulement syntaxiquement correct, mais aussi logiquement valide selon les règles définies par le langage.
- Parmi les routines sémantiques, on trouve :
  - Double déclaration d'une variable
  - Non déclaration d'une variable
  - Modification de la valeur d'une constante
  - Incompatibilité de type
  - Division par 0 dans le cas d'une constante.

.....

Routine sémantique de double déclaration d'une variable

Comment savoir si une variable est double déclarée ou non ?

Vérifiez le champ **Type** de cette variable dans la table des symboles :

- Si le champ est vide, cela signifie que la variable n'a pas encore été déclarée : Inserer le type.
- Sinon, cela indique que la variable est déjà déclarée : signalez une double déclaration.

```
declaration: type listeparam ';' declaration
                | type listeparam ';'
;
listeparam: listeparam ',' idf  { /*vérification de double déclaration*/}
          | idf { /*vérification de double déclaration*/}
;
type: mc_integer {strcpy(sauvType,$1); }
    | mc_real  {strcpy(sauvType,$1);}
;
```

Exemple:

int x; // Première déclaration

float x; // Erreur : double déclaration

Routine sémantique de variable non déclarée

Comment savoir si une variable est déclarée ou non ?

Vérifier le champs Type de cette variable:

- Si: il est vide alors la variable n'est pas encore déclarée : Signaler erreur de non déclaration.
- Sinon variable est déclarée : utiliser la variable selon sa position

```
instaff: idf '=' idf ';' { /*vérification de la déclaration des variables*/
}
```

Exemple:

Le programme source:

se Sémantique > DoubleDec- Nc

```
Program  L3ISILACAD
PDec
integer x,i,j;
Reel  y,z,a,x;
InBEGIN
 i=2;
 j=i/0;
 h=y+1;
InEnD
```

Lexical.l

```
%{
#include "syn.tab.h"
#include "TS.h"
extern YYSTYPE yylval;
extern int nb_ligne;
extern int col;
%}
lettre [a-zA-Z]
chiffre [0-9]
idf {lettre}({lettre}|{chiffre})*
cst [1-9][0-9]*|0
%%
Program {return mc_program;}
PDec {return mc_pdec;}
"integer" {yylval.str=strdup(yytext);return mc_integer;}
"Reel"  {yylval.str=strdup(yytext);return mc_real;}
InBEGIN {return mc_inbegin;}
InEnD {return mc_inend;}
[,;=/+] {return yytext[0];}
{cst} {yylval.entier=atoi(yytext); return cst;}
{idf} {yylval.str=strdup(yytext); inserer(yytext,"idf"); return idf;}
[ \t]
\n {nb_ligne++;col=1;}
. {printf ("erreur lexical sur l'entite %s a la ligne %d a la colonne %d",yytext,nb_ligne,col);
return err;}
%%
```

syn.y

```
%{
int nb_ligne=1;
 int col=1;
 void yyerror(char *msg);
 char SauvType[20];
%}
%start S
%union
{
int entier;
char* str;
}
%token mc_inbegin mc_inend mc_program mc_pdec err
%token <str>idf <str>  mc_integer <str> mc_real
%token <entier> cst
%%
S: Header mc_pdec DECLARATION INSTRUCTION
{ printf ("programme syntaxiquement correcte . \n"); YYACCEPT;}
;
Header: mc_program idf
;
DECLARATION: type listeparams ';' DECLARATION
    | type listeparams ';'
;
listeparams: listeparams ',' idf { /* Vérification de la double declaration */
            if(rechercheType($3)==0) {insererType($3,SauvType);}
            else printf("Erreur Semantique: double declation de %s, a la ligne %d \n", $3, nb_ligne);
            }
 | idf {  /* Vérification de la double declaration */
      if (rechercheType($1)==0) insererType($1,SauvType);
           else printf("Erreur Semantique: double declation de %s, a la ligne %d \n", $1, nb_ligne);
           }
 ;
type: mc_integer {strcpy(SauvType,$1);}
  | mc_real {strcpy(SauvType,$1);}
;
INSTRUCTION: mc_inbegin ListeInstr mc_inend
;
ListeInstr: ListeInstr Instruction
   | Instruction
;
Instruction : instaff
    | instdiv
    | instadd
;
instaff: idf '=' idf ';' {  /* Vérification de la declaration */
        if (rechercheType($1)==0) printf("Erreur semantique: %s non declare a la ligne
%d \n",$1,nb_ligne);
        if (rechercheType($3)==0) printf("Erreur semantique: %s non declare a la ligne
%d \n",$3,nb_ligne);
}
  | idf '=' cst ';'   { /* Vérification de la declaration */
        if (rechercheType($1)==0) printf("Erreur semantique: %s non declare a la ligne
%d \n",$1,nb_ligne);
        }
;
```

syn.y (suite)

```
instdiv: idf '=' idf '/' idf ';' { /* Vérification de la declaration */
      if(rechercheType($1)==0) printf("Erreur semantique: %s non declare a la ligne %d\n",$1,nb_ligne);
      if(rechercheType($3)==0) printf("Erreur semantique: %s non declare a la ligne %d\n",$3,nb_ligne);
     if(rechercheType($5)==0) printf("Erreur semantique: %s non declare a la ligne %d\n",$5,nb_ligne);
      }
  | idf '=' idf '/' cst ';' {  /* Division par zéro le cas d'une constante */
       if ($5==0) printf("Erreur semantique: Division par zero a la ligne %d\n",nb_ligne);
       /* Vérification de la declaration */
       if(rechercheType($1)==0) printf("Erreur semantique: %s non declare a la ligne %d\n",$1,nb_ligne);
       if(rechercheType($3)==0) printf("Erreur semantique: %s non declare a la ligne %d\n",$3,nb_ligne);
       }
  ;
instadd: idf '=' idf '+' idf ';' {
      if(rechercheType($1)==0) printf("Erreur semantique: %s non declare a la ligne %d\n",$1,nb_ligne);
      if(rechercheType($3)==0) printf("Erreur semantique: %s non declare a la ligne %d\n",$3,nb_ligne);
      if(rechercheType($5)==0) printf("Erreur semantique: %s non declare a la ligne %d\n",$5,nb_ligne);
      }
  | idf '=' idf '+' cst ';' {
      if(rechercheType($1)==0) printf("Erreur semantique: %s non declare a la ligne %d\n",$1,nb_ligne);
      if(rechercheType($3)==0) printf("Erreur semantique: %s non declare a la ligne %d\n",$3,nb_ligne);
      }
  ;
%%
void yyerror(char *msg) {
 printf("Erreur syntaxique %s, a la ligne %d\n", msg, nb_ligne);
}
main ()
{
yyparse();
afficher();
}
yywrap(){
}
```

Résultat de l'execution:

Analyse Sémantique > DoubleDec- NonDEC >

```
1   Program  L3ISILACAD
2   PDec
3   integer x,i,j;
4   Reel  y,z,a,x;
5   InBEGIN
6    i=2;
7    j=i/0;
8    h=y+1;
9   InEnD
```

```
Erreur Semantique: double declation de x, a la ligne 4
Erreur semantique: Division par zero a la ligne 7
Erreur semantique: h non declare a la ligne 8
Programme syntaxiquement correcte .

/***************Table des symboles ******************/
-------------------------------------------------
        | NomEntite |  CodeEntite  | TypeEntite
-------------------------------------------------
        |L3ISILACAD |          idf  |
        |         x |          idf  |     integer |
        |         i |          idf  |     integer |
        |         j |          idf  |     integer |
        |         y |          idf  |        Reel |
        |         z |          idf  |        Reel |
        |         a |          idf  |        Reel |
        |         h |          idf  |
```