Statement\_list -> Statement statement\_list

|épsilon

Statement -> **assign**

|**printf**();

|**scanf**();

|**if** (expression) Compound\_statement **else** Compound\_statement

|**while** (expression) Compound\_statement

|**do** Compund\_statement **while**(expresion);

|**for (**expression opt; expression opt2; expression op3**)** optional\_statements  
 | **return**;  
 | **break**;

|**switch** (expression){[case CHAR|INT : optional\_statements;]\* default: optional\_statements;

Compound\_statement -> **{**optional\_statements **}**

Compund\_Struct -> {[assign;]\*}

Optional\_statements -> statement\_list

|epsilon

Assign -> tipo id lvalue

lvalue -> ;  
 | [expression] arreglos  
 | = expression

arreglos -> = expression;  
 | ;

Statement\_p -> **(**expression\_list**)**

|**[**expression**]**

|epsilon

expression\_list -> expression **,** expression\_list

|epsilon

expression -> andexp expression\_prime

expression\_prime -> **or** expression\_prime

|epsilon

andexp -> relexp andexp\_prime

andexp\_prime -> **and** andexp\_prime

|epsilon

relexp -> addexp relexp\_prime

relexp\_prime -> relop addexp

|epsilon

relop -> **equals**

**|not\_equals**

**|greater\_than**

**|lesser\_than**

**|lesser\_or\_equal**

**|greater\_or\_equal**

addexp -> multexp addexp\_prime

addexp\_prime -> **+** addexp\_prime

|**-** addexp\_prime

|epsilon

multexp -> parexp multexp\_prime

multexp\_prime -> **\*** multexp\_prime

|**/** multexp\_prime

|**%** multexp\_prime

|epsilon

parexp -> (expression  
 | tipo

Tipo -> char String  
 | int  
 |doublé  
 |float

String -> \*   
 | epsilon