Punto 1 :
1) Determinara los conjuntos N , Σ , SimboloInicial

N = conjunto de no terminales
programa, declaraciones , declaracion ,tipo, lista_var , sentencias , sentencia , expresion

Σ = Los terminales
identificador
,
int
float
=
if
(
)
else
while
+
-
*
/
número

SimboloInicial = programa

Punto 2) Voy a eliminar la ambigüedad después de eliminar la recursividad

Punto 3) Eliminación de recursión izquierda

Se elimino la recursion en declaraciones , lista_var , sentencias y expresion

```
P = {

    programa → declaraciones sentencias

    //declaraciones → declaraciones declaracion | declaracion
    declaraciones →  declaracion declaraciones2
    declaraciones2 ->  declaracion declaraciones2 | empty
    declaracion → tipo lista_var ;
    tipo → int | float

    //lista_var → lista_var , identificador | identificador
    lista_var →  identificador lista_var2
    lista_var2 → , identificador lista_var2 | empty

    //sentencias → sentencias sentencia | sentencia
    sentencias → sentencia sentencias2
    sentencias2 → sentencia sentencias2 | empty
    sentencia → identificador = expresion ; | if ( expresion ) sentencias else sentencias | while ( expresión ) sentencias

    //expresion → expresion + expresion | expresion - expresion | expresion * expresion | expresion / expresión | identificador | numero
    expresion → identificador expresion2 | numero expresion2
    expresion2 →  + expresion expresion2 |  - expresion expresion2 |  * expresion expresion2 | / expresión expresion2 | empty
    expresion → ( expresion )
}
```

Punto 4)
No es necesaria hacerla porque entre los no terminales su posibles producciones no tienen cosas en común

```
lista_var2 → , identificador lista_var2 | parteSentencia |empty

parteSentencia → = expresion        // sin coma porque esa ya la tiene la declaracion
```

Quedando en

```
P = {

    programa → declaraciones sentencias
    declaraciones →  declaracion declaraciones2
    declaraciones2 ->  declaracion declaraciones2 | empty
    declaracion → tipo lista_var ;
    tipo → int | float

    lista_var →  identificador lista_var2
    lista_var2 → , identificador lista_var2 | parteSentencia |empty

    parteSentencia → = expresion        // sin coma porque esa ya la tiene la declaracion

    sentencias → sentencia sentencias2
    sentencias2 → sentencia sentencias2 | empty
    sentencia → identificador = expresion ; | if ( expresion ) sentencias else sentencias | while ( expresión ) sentencias

    expresion → identificador expresion2 | numero expresion2 | ( expresion ) expresion2
    expresion2 →  + expresion expresion2 |  - expresion expresion2 |  * expresion expresion2 | / expresión expresion2 | empty

}
```

Punto 2)
Regresando al punto 2 voy a quitar la ambigüedad de la creación de identificadores porque solo dejaba crear identificadores si tenian la siguiente forma int “a , b ,c ”;
Para solucionarlo agrego el siguiente codigo

Punto 5 :

N = conjunto de no terminales
programa, declaraciones ,declaraciones2, declaracion ,tipo, lista_var,lista_var2 , ,parteSentencia , sentencias,sentencias2 , sentencia , expresion,expresion2

Σ = Los terminales
identificador
,
int
float
=
if
(
)
else
while
+

-
*
/
número