Parte II: Analizador semántico

Objetivo:

Incorporar acciones semánticas en Bison para realizar comprobaciones semánticas.

Fecha límite entrega Parte II:

11/12/2022

Modo de entrega:

- Grupos de 3 alumnos
- A través de una tarea PoliformaT

Bison II:
Atributos y acciones semánticas

Lenguajes de Programación y
Procesadores de Lenguajes

Objetivos

1. Definir los tipos de los atributos
2. Asociar atributos sintetizados a los símbolos
3. Incorporar acciones semánticas.

1. Definir tipos de atributos

Se definen con %union

```
%union {
    tipoC1  nombre_tipo1 ;
    tipoC2  nombre_tipo2 ;
    …
}
```

¿ Y si un símbolo necesita más de un atributo ?

Definir un tipo estructura con tantos campos como atributos necesite

Ejemplo:

```
%union  {
    int     cent ;
    char    *ident ;
}
```

2. Asociar tipos a símbolos

- Cada símbolo solo tiene un tipo de atributo
- Atributos de terminales:

  %token <nombre_tipo> <símbolo terminal>

  Devolver desde Flex en la variable yylval
- Atributos de no-terminales:

  %type <nombre_tipo> <símbolo no-terminal>

Ejemplo:

```
%token PARA_ PARC_ MAS_ MENOS_ POR_ DIV_
%token <cent> CTE_
%type <cent> exp term fac
```

3. Acciones semánticas

- Sentencias C entre llaves
- Aparecen entre símbolos del lado derecho de regla
- $n Atributo asociado al símbolo n-ésimo del lado derecho
- $$ en la última acción: Atributo del no-terminal del lado izquierdo
- Acción por defecto: {$$ = $1 ;}

Ejemplo:

```
   $$        $1                $3          descripcion en c de lo que tiene que hacer lo de la izquierda
exp : exp  MAS_  term    { $$ = $1 + $3}
```

Acciones semánticas a mitad de regla

- La acción semántica cuenta como símbolo
- $$ a mitad de regla asigna valor a la propia acción semántica
- Tipo de la acción semántica indicado explícitamente al usarlo

Ejemplo:

esto no hace referencia a f porque esta por en medio de la regla y no al final.

```
f : exp          { $<cent>$ = $1 ;}
    MAS_ term    { $$ = $<cent>2 + $4 ;}
```

realmente esto es : f:exp MAS_ term

cuando insertamos por mitad la accion semantica que hemos añadido tambien cuenta a la hora de contar el numero de elementos en la parte derecha

Ejemplo (1/2)

```
%{
#include <stdio.h>
#include "header.h"
%}

%union {
   int cent ;  /* Para el terminal "cte" */
}

%token PARA_ PARC_ MAS_ MENOS_ POR_ DIV_
%token <cent> CTE_
%type  <cent> exp term fac

%%
```

ponemos todos los que cuando en codigo c tengamos que asignar del tiron o trabajar con ellos

Ejemplo (2/2)

```
expMat : exp  { printf("\nValor = %d\n",$1); }
       ;
exp : exp MAS_ term     { $$ = $1 + $3; }
    | exp MENOS_ term   { $$ = $1 - $3; }
    | term              { $$ = $1;      }
    ;
term : term POR_ fac     { $$ = $1 * $3; }
     | term DIV_ fac     { $$ = $1 / $3; }
     | fac               { $$ = $1;      }
     ;
fac : PARA_ exp PARC_   { $$ = $2;      }
    | CTE_              { $$ = $1;      }
    ;
%%
```

Desde Flex debe devolverse el valor semántico de la CTE_

```
{entero} { yylval.cent= atoi(yytext);    return(CTE_); }
```