ISC-314

Programación III

Teoría y Práctica del Proceso de Compilación

Especificación de Proyecto Parcial II

Generación del Analizador Sintáctico para El Lenguaje MiniP

Objetivos Generales

- Poner en práctica los conocimientos aprendidos hasta ahora en lo que se refiere a:
  - Gramáticas Libres de Contexto
  - Derivación por La Izquierda
  - Derivación por La Derecha
  - Análisis Sintáctico
    - Ascendente (LL(1))
    - Descendente (LALR)
  - Síntesis de Analizadores Sintácticos
    - Yacc
    - Bison
- Sintetizar y complementar (usando la herramienta Yacc) la codificación del módulo de análisis sintáctico que es el segundo componente del compilador

Instrucciones de Programación

1. Deben leer la ***especificación sintáctica de MiniP***. En ella se da la gramática independiente del contexto y normalizada (recursiva por la derecha) del lenguaje

2. Traducir la gramática de MiniP a la notación soportada por Yacc

3. Compilar la especificación Yacc de MiniP y producir el código (en C) del analizador sintáctico

4. Si la compilación de la especificación gramática de MiniP en Yacc es exitosa, se producirá un archivo de código C donde (entre otras muchas cosas) se define la función *yyparse* que es el punto de entrada (*entry point*) a la función que ejecuta el algoritmo de análisis sintáctico. Más abajo se indica elementos de interés generados por Yacc.

    a. `YYMATCHED`: Constante numérica (no tipada) que indica si sí o si no se ha reducido satisfactoriamente una regla de la gramática

    b. `int yyparse(void)`**:** Punto de entrada (entry point) del módulo de análisis sintáctico. Esta función devuelve "1" si el código del programa analizado satisface la gramática. Devuelve un número diferente de "1" si hay al menos un error sintáctico.

    c. `short matched[]`**:** Arreglo de indicadores donde se registra si sí o si no se ha usado la regla de la gramática (representada por el índice del arreglo) en una reducción satisfactoria.

5. Deben importar este archivo de código al proyecto (o a una copia de) donde Uds. implementaron el módulo de *análisis léxico.*

6. Deben asegurarse de incluir una acción semántica simple (código fuente que se ejecute cada vez que se *reduce* una regla de producción) que indique cuál regla de producción ha sido utilizada

7. Asegúrense de que el *nuevo* proyecto compile satisfactoriamente y que no haya ningún error de ligadura (*linker error*).

Caracterización Sintáctica de MiniP

A continuación se da la gramática libre de contexto (GIC) para el lenguaje MiniP. Esta gramática se especifica vía las reglas de producción que figura más abajo. La convención notacional de estas reglas establece que los *no-terminales* se escriben en minúsculas y los *terminales* se escriben en mayúsculas y/o negritas.

```
program   -->  title ; block
title     -->  _PROGRAM  _ID
block     -->  vars _ENDVARS procs _ENDPROCS code
vars      -->  _VARIABLES  varlist ;  | _NOVARIABLES
varlist   -->  varlist ; vardef  |  vardef
```

```
vardef    -->  _ID : _INTEGER | _ID : _REAL | _ID : _INTEGER bnl
               | _ID : STRING
bnl       -->  [ nlist ]
nlist     -->  nlist , _ICONST  |  _ICONST
procs     -->  _PROCEDURES  proclist  |  _NOPROCEDURES
proclist  -->  proclist ; procdef  |  procdef
procdef   -->  ptitle ; block
ptitle    -->  _PROCEDURE _ID ( varlist )  |  _PROCEDURE _ID ( )
code      -->  _BEGIN  para  _END  |  ;
para      -->  para ; stmt  |  stmt
stmt      -->  assign | cond | loop | input | output | code
assign    -->  ids :=  expr
expr      -->  expr + term  |  expr - term  |  term
term      -->  term * fac  |  term / fac  |  fac
fac       -->  val  |  ( expr )
val       -->  ids  |  _ID ( vallist )  |  _ICONST  |  _RCONST
ids       -->  _ID  |  _ID [ vallist ]
vallist   -->  vallist , it  |  it
it        -->  _ID  |  _ICONST
cond      -->  _IF expr bop expr _THEN stmt _ELSE stmt
bop       -->  =  |  <  |  >  |  <=  |  >=  |  <>
loop      -->  _FOR assign _TO expr _DO stmt
input     -->  _READ ( _ID )
output    -->  _WRITE ( _ID )  |  _WRITE ( LITERAL )
```

Esta es la gramática *inicial* de MiniP – se necesitará hacer algunas modificaciones cuando se implementen diferentes aspectos del análisis semántico. Fíjense que los terminales de esta gramática corresponden a las categorías léxicas o *tokens* que fueron definidos para el módulo de análisis léxico. Las definiciones de _LITERAL, _RCONST, _ICONST y _ID ya han sido definidas por medio de expresiones regulares. Los *tokens* simples (p. ej. ":=", ";", "+", etc.) se dan por medio de los lexemas que los representan. Además, esta gramática **no está** en la notación que Yacc acepta. Asegúrense de tomar lo anterior en cuenta.

Entregables

La evaluación de este proyecto requerirá que se sometan (vía la PVA) los siguientes entregables:

1. El código fuente del módulo del analizador sintáctico
2. El proyecto que integre el modulo léxico y el módulo sintáctico
3. Tres casos de prueba los cuales cubran la gramática en, al menos, un 90%
4. Salidas de cada prueba