COMPILADORES
PROYECTO-SEGUNDA PARTE
CONSTRUCCIÓN DE UN ANALIZADOR SINTÁCTICO

I. Objetivo general

Aplicar las técnicas de construcción de analizadores sintácticos descendentes dirigidos por algoritmo y sin retroceso.

II. Objetivos específicos

• Construir un analizador sintáctico para un lenguaje de programación, que tome en cuenta los aspectos básicos pero fundamentales involucrados en esta fase de compilación.

• Que el estudiante enfoque el problema de la construcción de compiladores y de los ambientes en que estos trabajan, usando principios de la programación de sistemas y de la ingeniería de software.

• Que el estudiante encuentre un sitio natural para la aplicación del conocimiento adquirido en otras áreas de las ciencias de la computación, tales como: programación, estructura de datos, sistemas operativos y teoría matemática de la computación.

III. Descripción

Se construirá un analizador sintáctico para una variante (superset) del lenguaje de programación PL0, así como un sencillo pero funcional ambiente de desarrollo para el programador. El proyecto incluye la implementación de la fase de análisis semántico.

IV. Requerimientos

Una vez asignado el superset del lenguaje pl0, los estudiantes procederán a construir un analizador sintáctico descendente dirigido por algoritmo y sin retroceso, usando las técnicas vistas en clase.

Todos los grupos deberán implementarse las siguientes características:

1. Agregar los comentarios: *{comentario}*.

2. Extender la estructura de datos del lenguaje para manejar los tipos de datos ENTERO y REAL.

3. Incorporar las instrucciones de lectura y escritura:

read(variable)
readln(variable)
write([“cadena de caracteres”,] variable)
writeln(“cadena de caracteres”)
writeln([“cadena de caracteres”,] variable)
writeln(“cadena de caracteres”)

Cuándo la instrucción termina con el sufijo "**ln**", se salta a la siguiente línea en el dispositivo de entrada o salida luego de efectuar la operación.

4. Cambiar en el compilador la estructura tipo array de la TDS, por una estructura dinámica de datos (lista autorreferenciada), modificando en consecuencia los mecanismos de almacenamiento y búsqueda que sean necesarios.

5. Asegurarse que no se puedan redefinir "objetos" en las declaraciones de un mismo procedure.

6. Modificar la interfaz gráfica para permitir el uso del scanner y el uso de la combinación scanner-parser.

7. Los resultados del análisis sintáctico se mostrarán en una "ventana de resultados" en la interfaz. En el caso de que el análisis sea exitoso, se mostrará un mensaje análogo al del parser visto en clase. En caso de error, el análisis se detiene, la línea en el texto donde se ubica el único error **será resaltada con color amarillo** y en la ventana de resultados **aparecerá el mensaje de error**. Al dar doble clic sobre el mensaje de error el cursor se posicionará en la línea donde se ha encontrado el error señalado.

8. La opción de "ayuda" en la interfaz de usuario, mostrará ahora también los diagramas de sintaxis **actualizados** y correspondientes a todas las instrucciones que el lenguaje puede manejar.

9. Se requiere que el analizador sintáctico sea escrito en Visual C++. **Ningún otro lenguaje es elegible**.

10. Se requiere que la interfaz con el usuario sea escrita en Visual C++, Visual Basic, C# o Java. **Ningún otro lenguaje es elegible**.

11. Además de los puntos anteriores, a cada grupo le será asignada una de 5 variantes de PL0. Cada grupo deberá implementar las especificaciones propias de la variante asignada.

V. Entregables

Un CD con una portada PEGADA EN LA CARA EXTERIOR DE LA FUNDA DEL CD especificando el nombre del grupo, el nombre completo de los miembros del grupo y la variante de PL0 asignada. NO LO ENTREGE EN BOLSA DE PAPEL MANILA NI ENTREGE DUPLICADOS O COPIAS DEL CD.

El CD contendrá:

1. En la raíz:
    a. Un archivo ejecutable llamado "IDE.EXE" que arranca la interfaz con el usuario.
    b. Dos programas ejemplo en el lenguaje asignado llamados: ejemplo1.txt y ejemplo2.txt.

2. En una carpeta denominada "parser", la solución completa creada en Visual Studio, incluyendo programas fuente y ejecutables correspondientes al parser.

3. En una carpeta denominada "ide", los programas fuente y objeto correspondientes a la interface del usuario.

POR FAVOR: No incluya nada más en el medio. Revise que su medio no esté defectuoso y que esté libre de virus.

VI. Notas importantes

• La única plataforma que se puede presuponer disponible es Microsoft Windows 7 y 8.

• El producto será probado por el profesor desde la interfaz con el usuario. NO SUPONGA QUE LOS ARCHIVOS CON LOS QUE SE PROBARÁ EL PROGRAMA ESTÁN EN ALGÚN SITIO PREDETERMINADO. NO SUPONGA QUE EL PROGRAMA EJECUTABLE DEBE ESTAR EN ALGÚN SITIO PREDETERMINADO.

• Lo que usualmente se hace es copiar el contenido del CD a disco duro y probarlo desde ese lugar.

• Fecha y hora de entrega: 6 de noviembre, 6:30pm a más tardar. **Por ninguna razón se recibirá una segunda entrega del CD, ni se recibirá el CD más allá de la fecha y hora asignada. No se aceptará ningún otro medio de entrega que no sea CD.**

ASIGNACIONES DEL LENGUAJE DE PROGRAMACION PARA LOS GRUPOS

PL-1

1. Extender la estructura de datos de PL0 para manejar arrays unidimensionales.

2. Extender la sintaxis de PL-0 para referenciar a una celda en un array unidimensional mediante un índice.

3. Añadir la instrucción FOR al lenguaje

4. Añadir el encabezado de programa PROGRAM de manera optativa

5. Aumentar las funciones predefinidas: LOG10, LOGE, SQRT,EXP (e_x)

PL-2

1. Extender la estructura de datos de PL-0 para manejar el tipo caracter.

2. Extender la sintaxis de PL-0 para permitir ELSE en el IF.

3. Eliminar la necesidad de la palabra CALL para llamar a los procedimientos.

4. Aumentar la instrucción CASE.

5. Aumentar las funciones predefinidas: MOD, DIV, CHR (conversión de entero a caracter), ORD (caracter a entero).

PL-3

1. Extender sintaxis de PL-0 para permitir paso de parámetros entre los procedures (tanto por valor como por referencia).

2. Aumentar la instrucción REPEAT...UNTIL.

3. Extender la sintaxis de PL-0 para manejar la declaración de FUNCIONES, así como su uso en el programa.

4. Aumentar la instrucción CASE

5. Aumentar las funciones predefinidas: ABS, TRUNC, ROUND, PRIMO.

PL-4.

1. Extender la estructura de datos de PL-0 para manejar el tipo BOOLEAN y por supuesto, su uso en un programa.

2. Aumentar el encabezado de programa PROGRAM de manera optativa

3. Aumentar las instrucción FOR

4. Aumentar la posibilidad de escribir código-p 'inmerso' en el código fuente de PL-0 usando al instrucción :

INLINE (i1/i2/i3/.../in)

donde 'i_n' es una instrucción de código-p. (Preguntar)

5. Aumentar las funciones predefinidas: RND,CLRSCR,HALT,PITAG (x,y,z) (que devuelve true si x,y,z son números pitagóricos y false en caso contrario).

PL-5

1. Extender la sintaxis de PL-0 como usted crea conveniente (instrucción, función, etc.) para poder indicar una URL que debe accesarse desde el programa.

2. Extender la sintaxis de PL-0 para poder definir, accesar, abrir y cerrar un archivo de texto.

3. Extender la sintaxis de PL-0 para poder mostrar (volcado) el contenido de un archivo de texto.

4. Aumentar la instrucción CASE

5. Aumentar las funciones predefinidas: SIN,COS, TAN y sus respectivas inversas

NOTA: Las modificaciones que se introducirán en cada versión del compilador, deben ser congruentes con las sintaxis del lenguaje PASCAL, excepto en aquellos lugares donde esto no sea posible debido a que PASCAL no soporta la modificación por introducir.