SINTAXIS Y SEMÁNTICA DE LOS LENGUAJES

CURSO K2006

GRUPO 9

TP N°4 GRUPAL

|  |  |
| --- | --- |
| INTEGRANTES | LEGAJO |
| Roberto Rodas | 2081301 |
| Alex Fiorenza | 2089865 |
| Oscar Mercado | 2083012 |
| Joaquín Fatur | 2079847 |

Trabajo Práctico N° 4 – Análisis del compilador de micro realizado en C

1. Compilar el Código entregado en clase.
2. Preparar 3 programas en micro: 1 correcto, 1 con error léxico, 1 con error sintáctico y semántico.
3. Analizar cómo funciona el análisis de cada uno de los errores expresando con sus palabras porque se produce el error, no relatando las líneas del código sino con lenguaje natural.

Lineamientos de entrega:

* Carátula con todos los integrantes
* Ejecutable
* Documento con
  1. pantallas mostrando que el programa funciona
  2. análisis de los errores solicitados
* Archivos anexos: se deben incluir los archivos que se utilicen en las pruebas
* Grupo: el mismo de todo el año
* Caso de copia: desaprueban la materia y deberán recuperar en marzo perdiendo por completo la posibilidad de promoción.
* El trabajo para darse por cumplido debe entregarse con todos los ítems desarrollados.
* Se entrega por campus una copia por grupo.

**ANÁLISIS DE LOS PROGRAMAS**

**1)PROGRAMA CORRECTO EN MICRO**

**Texto

Descripción generada automáticamente**

**Texto

Descripción generada automáticamente**

La ejecución del programa micro básicamente lee dos IDs tanto a como b. Y luego se encarga de escribir a partir de la lectura dos expresiones básicas. Una es una suma del id a y 8. Por otro lado, la otra expresión se encarga de sumar los dos identificadores a y b.

**2)PROGRAMA INCORRECTO CON ERROR LÉXICO Y SUCESION DE SINTACTICOS**

Texto

Descripción generada automáticamente con confianza media

Es un programa sencillo de micro que básicamente consiste en una única asignación del numero 5 a la variable de ID “hola%”. Lo que ya podemos decir rápidamente, mediante inspección, es que este programa tiene un error léxico en el nombre del ID ya que este posee un carácter no permitido en micro, el “%”. Analicemos como lo maneja nuestro compilador:

En primer lugar, se inicia el proceso de compilación, llamando a los distintos procedimientos de nuestro compilador (objetivo, programa, etc). Como se trata de una sentencia que empieza con un IDENTIFICADOR, se puede observar en la gramática sintáctica que lo que se espera es después recibir un token de ASIGNACION, una expresión y finalmente un PUNTOYCOMA.

Texto

Descripción generada automáticamente

Viendo la gramática léxica podemos observar que un identificador debe empezar con una letra, y debe ser seguido con una sucesión de letras y/o números.

Texto, Carta

Descripción generada automáticamente

En primer lugar, se hace un match (ID), el parser le pide al scanner el próximo token y verifica si este efectivamente es un ID, lo cual no genera ningún problema. ¿Qué sucede con el “%” ?, analizando detenidamente el resultado del proceso de compilación podemos concluir lo siguiente:

Texto

Descripción generada automáticamente

El % actuó de carácter espurio de hola, entonces se devolvió al flujo, por lo que no hubo error en el match (ID). Sin embargo, cuando se hizo match (ASIGNACION), se esperaba justamente el “:=” (token 10) pero se recibió un “%” (token 14), lo cual, en primer lugar, derivo en un error léxico al ser “%” un carácter no valido para micro y en un error sintáctico al no coincidir el próximo token con lo que se esperaba recibir.

Luego, se hizo un match (PUNTOYCOMA) esperando un “;” y se recibió un “:=” que es lo que esperaba el anterior match(ASIGNACION). Posteriormente se esperaba un FIN y se recibió un token CONSTANTE que es lo que le debía seguir al “:=”. Finalmente se esperaba el token FDT y se recibió el token PUNTOYCOMA.

En conclusión, se produjo lo que se llama un “inconveniente”, errores en cascada producto de haber utilizado un carácter que no puede intervenir en ningún lexema valido en micro, por lo que produce un error léxico en primer lugar y termino produciendo una sucesión de errores de tipo “sintáctico”. El error léxico que se produce proviene del AFD que implementa nuestro scanner, el mismo devuelve un token ERRORLEXICO en caso de producirse una situación así, y el AFD se hace en base a la gramática léxica.

**3)PROGRAMA INCORRECTO CON ERROR SINTACTICO Y SEMANTICO**

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente  
Texto

Descripción generada automáticamente**

No llega a procesar si quiera la variable con el nombre muy largo, el programa se interrumpe abruptamente (no llega a “detiene , ,”), luego del error léxico y sintáctico del “:” al final de leer.

El problema presente aquí es tanto la declaración de la variable “unaVariableDemasiadoLaaaaaaaaargaaaa” como la presencia del “:” al final de la PR “leer”. Por un lado, el largo de la variable excede los 32 caracteres, lo cual definido por micro es un error semántico, ya que se acepta un máximo de 32 caracteres, y por otro lado la presencia de “:” en lugar de un “;” se define como un error sintáctico, ya que cada sentencia en micro debe terminar con un “;” para lograr procesar lo que se quiera hacer. El error sintáctico ocurre cuando el token esperado no coincide con el recibido, ese token esperado surge de la gramática sintáctica.