Trabajo Práctico N.º 5

En base a lo hecho en TP4 agregaremos las rutinas semánticas para tener el compilador completo (en realidad un front end si consideramos que nuestro seudo ensamblador es código intermedio)
A las consideraciones del TP4 agregaremos.

Consideraciones:

- Agruparemos las rutinas que manejan el diccionario en symbol.c, no se requiere la rutina más eficiente, nos basta con una que sea clara y correcta
- Agruparemos las rutinas semánticas en sematic.c, hay varias funciones del compilador micro que pueden reutilizarse, eventualmente con los ajustes pertinentes, también pueden necesitar nuevas funciones.
- Al final del programa hay que controlar los 3 tipos de errores e informar la cantidad de cada uno de ellos, es decir, a lo que ya informábamos en TP4 agregamos los errores semánticos.
- Para las variables temporales usaremos Temp#i donde i es el número de temporal.
- Cuando haya errores semánticos usaremos YYERROR para dejar de analizar sintácticamente la sentencia o definición donde ocurrió el error.
- NO usaremos yyerrorok.

Códigos del seudo ensamblador:

- Declare, para declarar las variables como enteras (ver más abajo tipo de dato) la primera vez
- Load, para la carga de la biblioteca en tiempo de ejecución, debe quedar: `Load rtlib,,`
- ADD, para la suma
- SUBS, para la resta
- MULT, para la multiplicación
- DIV, para la división
- INV, para el menos unario. OJO hay usar precedencia de contexto. Generaremos algo como: `INV Var,,VarInv` es decir, el segundo operando lo dejamos en blanco y guardamos (consistentemente con el resto de las operaciones en el tercer operando)
- Store, para la asignación
- Read, para la lectura de un identificador
- Write, para la escritura de una expresión
- Stop, para frenar el programa

Para Tipo de dato

- **Integer**

Ejemplos de ejecución:

Para el archivo entradaok.txt

```
programa
      variables
            definir total. //comentario
            definir var1.
            definir var2.
      codigo
            leer (var1, var2).
            total <- var1 + 32 * -var2.
            var2 <- 2 * (512 + 4) .
            escribir (total/2).
fin
```

Salida

```
Load rtlib,
Declare total,Integer
Declare var1,Integer
Declare var2,Integer
Read var1,Integer
Read var2,Integer
Declare Temp#1,Integer
INV var2,,Temp#1
Declare Temp#2,Integer
MULT 32,Temp#1,Temp#2
Declare Temp#3,Integer
ADD var1,Temp#2,Temp#3
Store Temp#3,total
Declare Temp#4,Integer
ADD 512,4,Temp#4
Declare Temp#5,Integer
MULT 2,Temp#4,Temp#5
Store Temp#5,var2
Declare Temp#6,Integer
DIV total,2,Temp#6
Write Temp#6,Integer
Stop ,
Compilación terminada con éxito
Errors sintácticos: 0 - Errores léxicos: 0 - Errores Semánticos: 0
```

Para el archivo entradaerr.txt

```
programa
      variables
            definir total. //comentario
            definir var1.
            definir var2.
            definir var2.
            definir ##@.
      codigo
            leer (var1, var2).
            total <- var1 + 32z * -var5.
            var2#2@1 <- 2 * (512 + 4z2m) .
            escribir (total/2).
fin
```

Salida

```
Load rtlib,
Declare total,Integer
Declare var1,Integer
Declare var2,Integer
línea #6: Error semántico: identificador var2 ya declarado
línea #7: Error léxico: cadena desconocida: ##@
Read var1,Integer
Read var2,Integer
línea #10: Error léxico: constante con sufijo inválido: 32z
línea #10: syntax error, unexpected '*', expecting ID or CONSTANTE or '-' or
'('
línea #11: Error léxico: identificador inválido: var2#2@1
línea #11: Error léxico: constante con sufijo inválido: 4z2m
Declare Temp#1,Integer
DIV total,2,Temp#1
Write Temp#1,Integer
Stop ,
Errores de compilación
Errors sintácticos: 1 - Errores léxicos: 4 - Errores Semánticos: 1
```

Para el archivo entradaerr2.txt

```
programa
      variables
            definir total. //comentario
            definir var1.
            definir var2.
            definir var2.
            definir ##@.
      codigo
            leer (var1, var2).
            total <- var1 + 32z 23 * -var5.
            var2#2@1 <- 2 * (512 + 4) .
            escritura (total/2).
fin
```

Salida

```
Load rtlib,
Declare total,Integer
Declare var1,Integer
Declare var2,Integer
línea #6: Error semántico: identificador var2 ya declarado
línea #7: Error léxico: cadena desconocida: ##@
Read var1,Integer
Read var2,Integer
línea #10: Error léxico: constante con sufijo inválido: 32z
línea #10: Error semántico: identificador var5 NO declarado
línea #11: Error léxico: identificador inválido: var2#2@1
línea #12: Error semántico: identificador escritura NO declarado
Stop ,
Errores de compilación
Errors sintácticos: 0 - Errores léxicos: 3 - Errores Semánticos: 3
```

Entrega: Directorio compactado con los archivos: scanner.l, main.c, parser.y, semantic.c, semantic.h, symbol.c y symbol.h En main.c como comentario al principio pongan los datos del grupo y sus integrantes, o sea, la carátula del TP.
Si lo consideran necesario pueden entregar archivos adicionales, por ejemplo más fuentes o encabezados

Fechas

Primera fecha: es la fecha en que deben haber entregado al menos una vez el trabajo completo, aunque eventualmente con errores

K2053 (miércoles): 14/11
K2004 (jueves): 15/11
K2055 (viernes): 16/11

Segunda fecha: es la fecha en que deben haber entregado y correcto.

K2053 (miércoles): 05/12
K2004 (jueves): 06/12
K2055 (viernes): 07/12