Actividad 06. Calculadora (2)

El objetivo de esta actividad es ampliar la calculadora diseñada en la actividad 5 para que acepte variables (solo de tipo numérico: entero o real) dentro de las expresiones aritméticas.

Será necesario utilizar una estructura de datos, la tabla de símbolos, que permita almacenar y controlar el tipo y el valor de las variables utilizadas.

Las operaciones para gestionar la tabla de símbolos deben estar en una librería independiente de los analizadores léxico y sintáctico. Se pueden usar clases de la biblioteca estándar (STL).

Además, la calculadora dejará de ser un intérprete y se convertirá en un traductor que generará un fichero de salida con la información almacenada en la tabla de símbolos.

Todos los ficheros fuente (analizador léxico, sintáctico y librería que manipula la tabla de símbolos) deben entregarse acompañados de un fichero **makefile** que permitirá generar el programa ejecutable, llamado "**calculadora**".

Al hacer el trabajo se deben tener en cuenta los siguientes aspectos:

- El traductor tendrá dos parámetros que son los nombres de los ficheros de entrada y salida respectivamente.

  Para usar los ficheros **yyin** e **yyout** como se hizo en la actividad 3, hay que incluir ambos de la siguiente forma en **calculadora.y**:

  ```
  extern FILE* yyin;
  extern FILE* yyout;
  ```

- No es necesario usar **yyout**; se puede definir otro tipo de fichero de salida.
- En el fichero de entrada puede haber comentarios. Estos comienzan con los símbolos **//** y acaban con un salto de línea.
- Las variables que aparezcan dentro de las expresiones (en la parte derecha de las asignaciones) solo serán de tipo numérico, para evitar ambigüedades con la gramática. No se admitirán variables de tipo lógico dentro de las expresiones.
- Podemos suponer que el número total de identificadores definidos será inferior a 100.
- No hay instrucciones específicas para la declaración de variables. Por eso, cuando a una variable se le asigna un valor por primera vez se considera que ha sido definida con el mismo tipo de la expresión que se le asigna. Este tipo no puede cambiar a lo largo del proceso.
- Como el programa ya no es un intérprete y analiza un fichero de entrada, no tiene sentido utilizar el token SALIR ni tampoco la función prompt().
- Sólo se mostrarán en la ventana del terminal los mensajes de error sintácticos o semánticos.
- Además, hay que tener en cuenta todo lo indicado en la actividad 5.

Errores semánticos que deben ser detectados:

1. Utilizar, en una expresión, una variable numérica que no ha sido definida previamente.

2. Cambiar el tipo de una variable ya definida. (Por tanto, no sería posible asignar un valor entero a una variable que se había definido antes como real.)

3. Cualquier operación que suponga dividir por 0 (divisiones y módulo).

4. Utilizar una operación con una variable cuyo tipo no es el adecuado. Por ejemplo, no se pueden hacer operaciones lógicas (and, or, not) con variables numéricas; con variables de tipo real no se puede utilizar el operador de división entera (div) ni el de módulo (%).

Algunos de estos errores se pueden detectar como error semántico o sintáctico, dependiendo de cómo se hayan definido las expresiones en la gramática.

Cuando hay un error sintáctico o semántico en una instrucción, ésta no se ejecuta y, por tanto, no tiene ningún efecto sobre el valor de las variables afectadas.

No importa si, en el número de línea indicado por un error, hay una diferencia de una unidad con respecto a la línea real donde está el error.

Ejemplo de funcionamiento

Al ejecutar el siguiente comando con el contenido del fichero que se indica a continuación (**entrada.txt**) se deberían obtener los mensajes de error que se indican en la ventana del terminal y el fichero de salida (**salida.txt**) que se muestra al final.

```
> ./calculadora entrada.txt salida.txt
```

Fichero de entrada: entrada.txt (los comentarios aquí aparecen en varias líneas, pero en el fichero original solo hay salto de línea al final)

```
A := 2*3+1
Bb := (3+A)*2.54
Cc := A < Bb
logico_ := (B == 3) and (Bb==10)     //error, B no existe
A := 1 div 2
A_2 := 12 % Bb    //error semántico, Bb es de tipo real
A := 12.2*Bb      //error semántico, A es de tipo entero y recibe un valor de tipo real. A debe seguir valiendo 0
A := A+1
X := true + 2   //error sintáctico o semántico, no se pueden hacer operaciones aritméticas con valores lógicos
salir       //error sintáctico
aa := A + A
bandera := false
aux := bandera // error sintáctico o semántico, no admitimos variables lógicas en la parte derecha
aux := not bandera // error sintáctico o semántico, no admitimos variables lógicas en la parte derecha
```

Mensajes en la ventana del terminal:

```
Error semántico en la instrucción 4: la variable B no está
definida
Error semántico en la instrucción 6: se usa la operación %
con operandos reales
Error semántico en la instrucción 7: la variable A es de
tipo entero y no se le puede asignar un valor real
Error sintáctico en la instrucción 9
Error sintáctico en la instrucción 10
Error semántico en la instrucción 13: variables de tipo
lógico no pueden aparecer en la parte derecha de una
asignación
Error sintáctico en la instrucción 14
```

Los errores de las instrucciones 9,13 y 14 podrían ser sintácticos o semánticos, dependiendo de cómo se haya definido la gramática.

Los mensajes de error no tienen que ser idénticos a los de los ejemplos, pero deben incluir información lo más exacta posible sobre el tipo de error.

Fichero de salida: salida.txt

```
Tabla de símbolos:
A     entero     1
Bb    real       25.400000
Cc    lógico     true
aa    entero     2
bandera    lógico     false
```

Las variables no tienen que aparecer en el fichero de salida en el mismo orden en el que se fueron creando.

Detalles del envío y de la evaluación de la calculadora (2)

Esta actividad debe entregarse antes del **28 de abril** (a las 23:55).

Puede realizarse **individualmente** o **en pareja**, igual que la actividad 5. Si se hace en pareja, ambos miembros deberán subir la misma tarea al aula virtual. El nombre de las dos personas deberá aparecer en un comentario dentro del fichero **makefile** y de los ficheros fuente

Se subirá a la tarea un fichero comprimido con el nombre y apellidos de las personas que hayan hecho la actividad.

Ese fichero comprimido deberá contener los ficheros fuente correspondientes al analizador léxico y sintáctico, la librería que gestiona la tabla de símbolos y un fichero makefile que permita generar el ejecutable.

En el aula virtual estarán disponibles algunos ficheros de ejemplo similares a los que se utilizarán para probar el funcionamiento de la calculadora.

El fichero *makefile* debe estar diseñado para que se compile todo y ejecute el programa calculadora con los argumentos "entrada.txt" y "salida.txt" al escribir la orden "*make*".