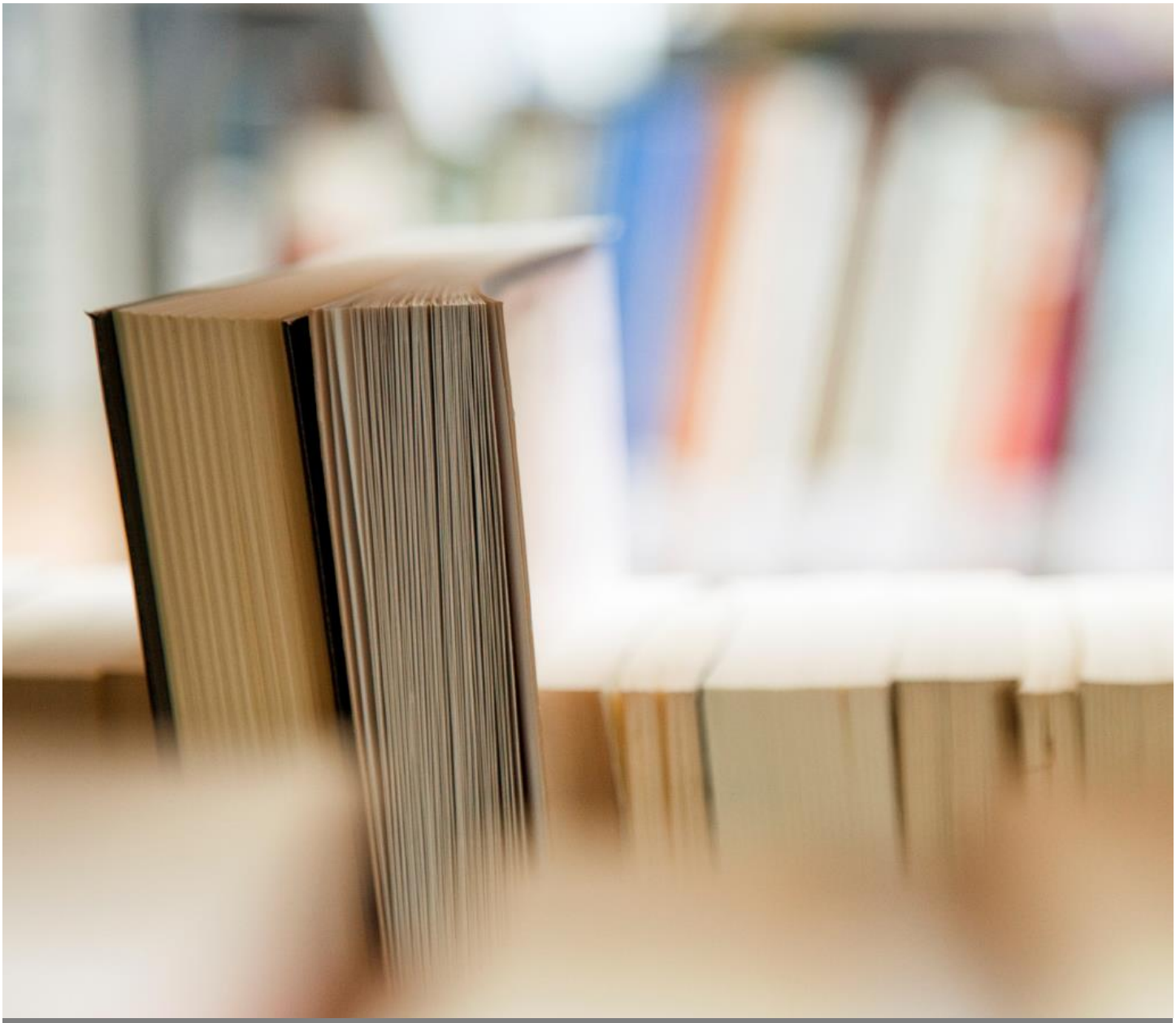


Manual Técnico

Organización de Compiladores

Universidad de San Carlos de Guatemala

Javier Monterroso 201700831 Facultad de Ingeniería

Contents

Análisis léxico usando Flex

La primera fase de la compilación es el análisis léxico: la descomposición de la entrada en tokens.

Un token generalmente se describe mediante un número entero que representa el tipo de token, posiblemente junto con un

atributo, que representa el valor del token. Por ejemplo, en la mayoría de los lenguajes de programación nosotros

Cada palabra reservada o símbolo especial se considera un tipo diferente de token, en la medida en que

Se distinguen por un entero diferente para representar su tipo.

Diferentes identificadores tienen el mismo tipo, y se distinguen por tener un diferente

atributo (tal vez el texto que compone el identificador, o un índice entero en una tabla de identificadores).

Se utilizó esta librería para generar el Scanner o analizador léxico del proyecto utilizando el siguiente código para generar la clase

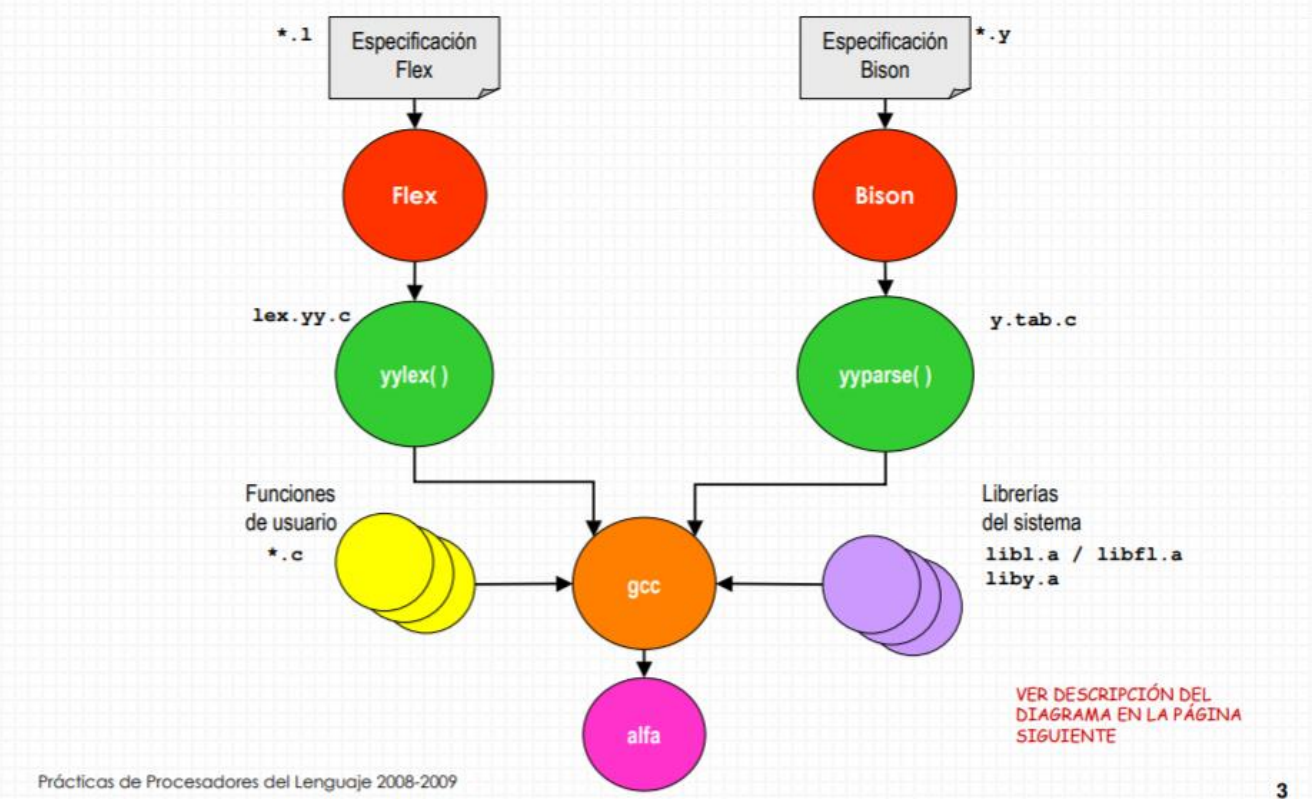
La función yylex() lee el fichero de entrada para identificar los tokens descritos en el fichero de especificación correspondiente. Cada vez que la función yylex() devuelve un token al analizador sintáctico, si el token tiene un valor asociado, yylex() guarda ese valor en la variable yylval antes de terminar. Por ejemplo, un identificador de una variable, además de tener un número de token que lo identifica y distingue de otros tipos de tokens, también tiene asociado como valor o atributo, el lexema del identificador. Sin embargo, un paréntesis no tiene asociado ningún valor o atributo. La función yyparse() utiliza el valor de la variable yylval en las acciones de las reglas de la gramática.

Análisis Sintáctico con Bison

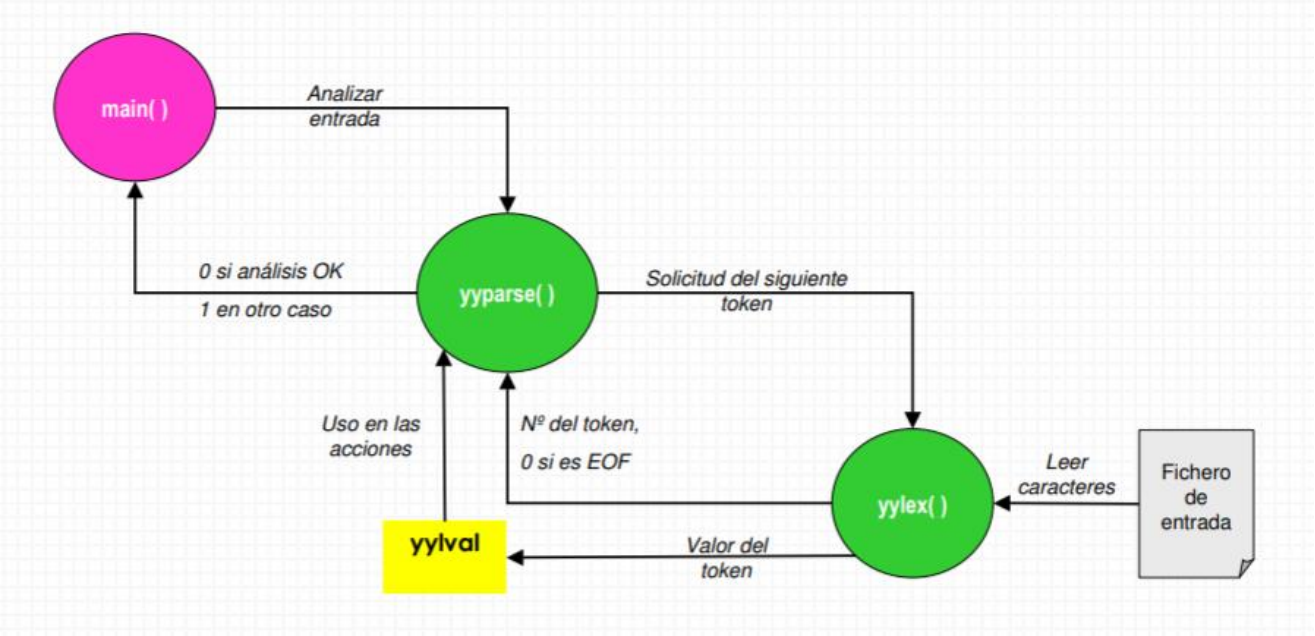
GNU bison es un programa generador de analizadores sintácticos LALR(1) de propósito general perteneciente al proyecto GNU disponible para prácticamente todos los sistemas operativos, se usa normalmente acompañado de flex aunque los analizadores léxicos se pueden también obtener de otras formas.

La función yyparse() solicita a la función yylex() los tokens de la entrada y comprueba si forman una construcción válida de acuerdo a las reglas de la gramática descritas en el fichero de especificación correspondiente. Cuando detecta un error sintáctico invoca a la función yyerror() y termina el proceso de análisis devolviendo un 1.

Estructura principal del interprete



Para obtener el programa objetivo alfa, se compilan y enlazan los ficheros descritos anteriormente.



Gramática del interprete:

```
START: START2


START2: START2 BODY
      | BODY


BODY: DECLARATION
      | ASSIGNATION
      | UPDATE
      | PRINT
      | SHOW
      | IF
      | FOR
      | WHILE


ASSIGNATION: ID ASSIGN2 semicolon

ASSIGN2: equal E
       | openB E closeB ASSIGN2

DECLARATION: DATATYPE DECLARATION2

DECLARATION2: OBJECTS semicolon
```

```
             | tarreglo ID ARRAY semicolon

DATATYPE: tint
        | tbool
        | tstring
        | tdouble
        | tchar

OBJECTS: OBJECTS comma ID ASSIGN
      | ID ASSIGN

ASSIGN: equal E
      |

ARRAY: openB E closeB ARRAY2

ARRAY2: openB E closeB ARRAY
      | equal ARRAYASIGN
      |


ARRAYASIGN: openCB ARRAYASIGN2 closeCB

ARRAYASIGN2: ARRAYASIGN3
           | ARRAYLIST

ARRAYASIGN3: ARRAYASIGN
           | ARRAYASIGN3 comma ARRAYASIGN

ARRAYLIST: ARRAYLIST comma E
        | E

NATIVE: integer
    | caracter
    | String
    | boolean
    | number

PRINT: timprimir openPar E closePar semicolon

SHOW: tshow openPar E comma E closePar semicolon

IF: tsi openPar E closePar openCB START2 closeCB ELSE

ELSE: tsino IF
```

```
    | tsino openCB START2 closeCB
    |

FOR: tpara openPar VARMANAGMENT E semicolon UPDATE closePar openCB START2

WHILE: trepetir openPar E closePar openCB START2 closeCB

VARMANAGMENT: DECLARATION
            | ASSIGNATION

UPDATE: ESINGLE increase
      |ESINGLE decrease

ESINGLE: NATIVE
  |ID INDEX

INDEX: openB E closeB
       |


ID: iden

E: E plus E
  |E minus E
  |E by E
  |E slash E
  |E power E
  |E doubleEqual E
  |E different E
  |E lessThan E
  |E greaterThan E
  |E lessThanEqual E
  |E greaterThanEqual E
  |E tor E
  |E tand E
  |tnot E
  |ESINGLE
  |openPar E closePar
  |minus E
```