9 BISON

Monday, September 30, 2024 12:14 PM

- SHIFT-REDUCE PARSER

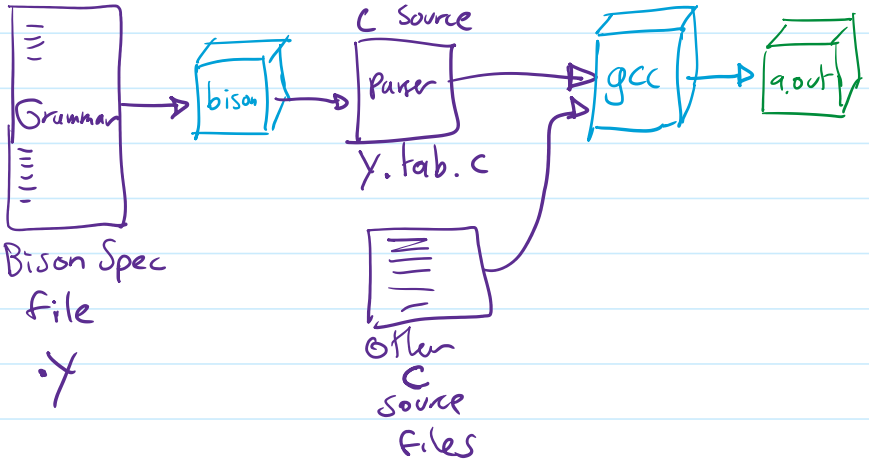
BISON - an implementation.
input - Grammar specification
output - source code for a shift-reduce parser
the corresponding
BISON is designed to work with "flex"

- HISTORY

'70 S. Johnson, YACC B, BCPL, C
'80 GNU R. Corbett Bison

- BISON

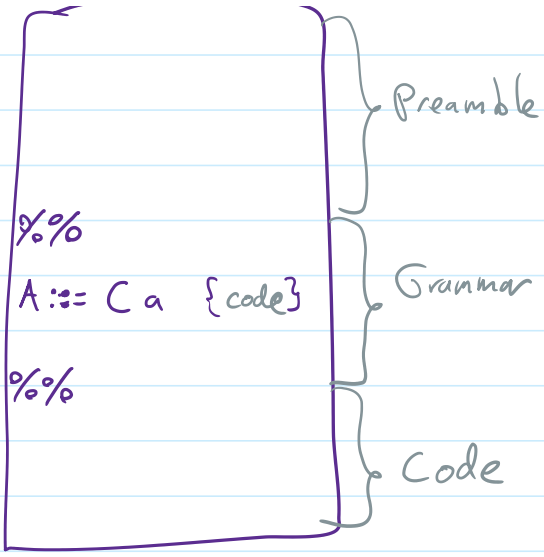
Bison is a "parser Generator"



- Bison expects a function called yylex() to be available to read terminal symbols from the input.

terminal symbols → lexemes.

- THE BISON SPECIFICATION FILE



Bison flags & options
Definitions
Code. %{ ≡ %}

Grammar + Action Rules.

Code copy pasted verbatim to the output file.

- THE CODE GENERATED BY BISON

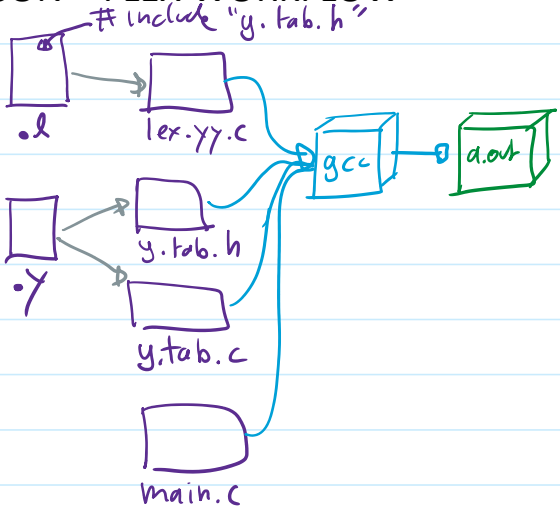
Functions :

yyparse() - implements the shift-reduce. paser
- calls yylex() to get terminal symbols.
returns 0 if parsing succesful
1 otherwise

yyerror(char *) - called by yyparse() when an error is encountered.

- you need to provide yylex() and yyerror().

- BISON + FLEX WORKFLOW




```
$ bison -d <.y file>  → y.tab.h
$ flex <.l file>      → lex.yy.c
$ bison <.y file>     → y.tab.c
$ g++ -lfl *.c *.cpp  → a.out
```

- BISON AND CONFLICTS
  - Bison will issue warnings when conflicts are detected in the grammar.

    How Bison "handles" conflicts:
    - Reduce-Reduce Conflict

      Bison resolves using first rule in file.
    - Shift-Reduce Conflict

      Bison resolves in favor of shift.

- LINK

  https://www.gnu.org/software/bison/manual/

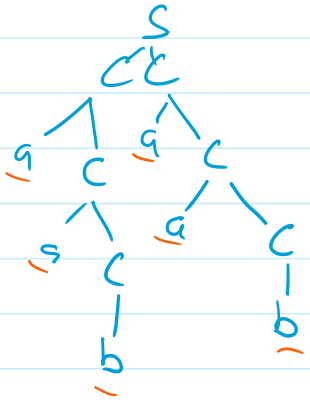
- BISON DEMO

  Example #1

  $S \to CC$
  $C \to b$
  $C \to aC$

  terminals = { a, b }

  aabaab



  Example #2

  flex + bison

  $S \to \lambda \mid E$

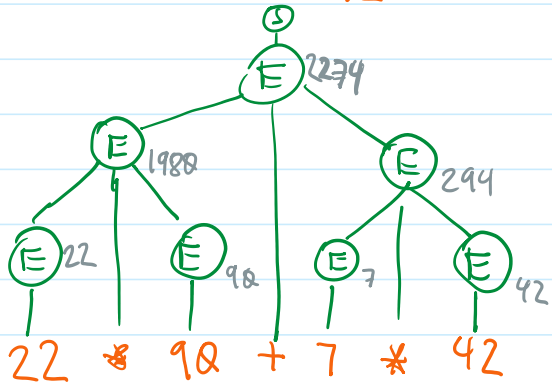
  - flex .l

$S \to \lambda \mid E$

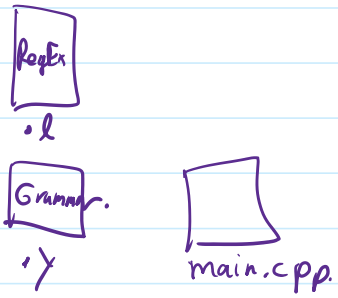
$E \to \underline{int} \mid E + E \mid E * E$

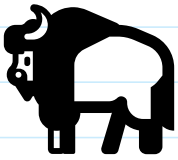
- flex .l
  - recognize terminals
    - int + *

22 * 90 + 7 * 42



- bison .y
  - = parse the grammar
  - - Compute a value for S





—o— EOF