

Name - Ushu Kaulaniya

Roll - PE29

Sub - SSC

Lab Assignment - 8

Title: Parser for Arithmetic Grammar using YACC

Aim: Write a program using LEX and YACC to ~~parse~~ create Parser for Arithmetic Grammar - Design calculator

Objective:

- To understand Yacc tool
- To study how to use Yacc tool for implementing Parser
- To understand the compilation and execution of *.y file.

Theory:

→ Introduction to YACC

A parser generator is a program that takes an input a specification of a syntax and produces as output a procedure for recognizing that language. Historically they can also called compiler-compilers.

YACC is an LALR parser generator. YACC was originally designed for being complemented by Lex

→ Study of *.y file

Input: A CFG-file.y

Output: A parser y.tab.c (yacc)

- The output file "file.output" contains the parsing tables
- The file "file.tab.h" contains declarations
- The parser called the yyparse().

→ Description of standard inbuilt variables and function

int yyle (void) - call to invoke lexer, returns token

char * yytext - pointer to matched string

yylen - length of matched string.

yyval - value associated with token

int yywrap (void) - wrapup, return 1 if done, 0 not done

FILE * yyout - output file

FILE * yyin - input file

Initial - initial start condition

Begin condition - which start condition

ECHO - write matched string.

→ Compilation and Execution Process.

For compiling YACC program

→ write lex program in a file .l and yacc in a file .y.

→ Open terminal and Navigate to the Directory where you have saved the file

→ type lex file.l

→ type yacc file.y

→ type cc lex.yy.c tab.h -ll

→ type ./a.out

Input: Source specification (*.y) file for arithmetic expression statements

Output: Result of Arithmetic Expression

FAQ's

- 1 Differentiate between top down and bottom-up parsers.

Top - Down

→ Top-down parsing attempts to find the left most derivation for an input str

→ The parsing technique uses Left Most Derivation

→ Its main decision is to select what productⁿ rule to use in order to construct the string

Bottom - up

→ Bottom-up parsing can be defined as an attempts to reduce the input str to start symbol of grammar

→ This parsing technique uses Right Most Derivation

→ Its main decision is to select when to use a production rule to reduce the string to get the starting symbol.

2 Explain working of shift reduce parser

→ shift reduce parsing is a process of reducing a string to the start symbol of a grammar

→ Shift reduce parsing uses ~~to~~ a stack to hold the grammar and an input tape to hold the string.

→ Shift reducing parsing perform the two actions actions: shift and reduce. That's ~~what~~ why it is known as shift reduce parsing.

→ At the shift action, the current symbol in the input string is pushed to a stack.

→ At each reduction, the symbols will replace by the non-terminal.

3 Explain how communication between LEX & YACC is carried out.

→ Lex and YACC often work well together for developing compiler

→ As noted a program uses the lex-generated scanner by repeatedly calling the function `yylex()`. This name is convenient because a yacc-generated parser calls its lexical analyzer with this name.

→ To use lex to create the lexical analyzer for a compiler, and end each lex action with the statement `return token;`, where `token` is a defined token with an integer value.

→ The integer value of the token returned indicates to the parser what the lexical analyzer has found. The parser, called `yyparse()` by `yacc`, then resumes control and makes another call to the lexical analyzer to get another token.

4. How YACC resolves ambiguities within given grammar

→ Shift/reduce conflict in the parsing table is resolved by giving priority to shift move over a reduce move.

If the string is accepted for shift move, then reduce move is removed, otherwise shift move is removed.

→ Reduce/reduce conflict in the parsing table is resolved by giving priority to first reduce move over second reduce move. If the string accepted for first reduce move, then second reduce move is removed, otherwise first reduce move is removed.