1. YACC program to accept the language L={a^n b^n : n>=0}

%{

#include <stdio.h>

%}

%token A B

%%

s : s1 { printf("valid");}

;

s1 : A s1 B

|

;

%%

main()

{

printf("Expression : ");

yyparse();

}

yyerror()

{

printf("INVALID");

}

LEX – Program

%{

#include "y.tab.h"

%}

%%

a {return A;}

b {return B;}

[ ] {return yytext[0];}

%%

1. YACC program to validate Arithmetic expression

%{

#include <stdio.h>

%}

%token ID D

%left '+' '-'

%left '\*' '/'

%%

s : e {printf("\n valid \n");}

;

e : e '+' e

| e '-' e

| e '\*' e

| e '/' e

| ID

| D

;

%%

main()

{

printf("Enter expression :");

yyparse();

}

yyerror()

{

printf("INVALID");

}

LEX Program

%{

#include "y.tab.h"

extern int yylval;

%}

%%

[0-9]+ {yylval=atoi(yytext);return D;}

[\+\-\\*\/] {return yytext[0];}

%%

1. YACC program to validate IF construct.

%{

#include <stdio.h>

%}

%token IF ELSE ID

%%

s : s1 { printf("valid");}

;

s1 : IF ' ' '(' e ')' ' ' s2 ' ' ELSE ' ' s3

;

s2 : ID '=' ID '+' ID';'

;

s3 : ID '=' ID '-' ID';'

;

e : ID relop ID

;

relop : '>'

| '<'

;

%%

main()

{

printf("Expression : ");

yyparse();

}

yyerror()

{

printf(" Invalid ");

}

LEX - Program

%{

#include "y.tab.h"

%}

%%

if {return IF;}

else {return ELSE;}

[a-zA-Z] { return ID; }

[\<\>\+\-\\*\/\=\;] { return yytext[0]; }

[\(\)] { return yytext[0];}

[ ] {return yytext[0];}

%%

1. YACC program to validate WHILE construct

%{

#include <stdio.h>

%}

%token WHILE ID DO

%%

s : s1 { printf("valid");}

;

s1 : WHILE ' ' '(' e ')' ' ' DO '{' s1 '}'

| ID '=' ID op ID ';'

;

e : ID relop ID

;

op : '+'

| '-'

| '\*'

| '/'

;

relop : '<'

| '>'

;

%%

main()

{

printf(" While statement : ");

yyparse();

}

yyerror()

{

printf(" Invalid ");

}

%{

#include "y.tab.h"

%}

%%

while {return WHILE;}

do {return DO;}

[a-zA-Z] {return ID;}

[\<\>\+\-\\*\-\{\}\=] { return yytext[0]; }

[\(\)\;] { return yytext[0];}

[ ] {return yytext[0];}

1. YACC program to validate and evaluate arithmetic expression

%{

#include <stdio.h>

%}

%token D

%left '+' '-'

%left '\*' '/'

%%

s : e { printf("valid"); printf("\n value is %d",$1);}

;

e : e '+' e {$$ = $1+$3;}

| e '-' e {$$ = $1-$3;}

| e '\*' e {$$ = $1\*$3;}

| e '/' e {$$ = $1/$3;}

| D {$$ = $1;}

;

%%

main()

{

printf("Expression : ");

yyparse();

}

yyerror()

{

printf("INVALID");

}

LEX – Program

%{

#include "y.tab.h"

extern int yylval;

%}

%%

[0-9]+ {yylval=atoi(yytext);return D;}

[\+\-\\*\/] {return yytext[0];}

%%