**Practical E1**

**Aim: Write YACC specification to check syntax of a simple expression involving operators +, -, \* and /. Also convert the arithmetic expression to postfix.**

**Program:**

**2alex.l**

%{

#include "y.tab.h"

#include <string.h>

%}

%%

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

[a-zA-Z] {yylval=strdup(yytext); return ID;}

\n {return NL;}

. {return yytext[0];}

%%

**Postfix.y**

%{

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

int answer=0;

char postfix[100];

int pos = 0;

%}

%token NUMBER ID NL

%left '+' '-'

%left '\*' '/'

%%

stmt : exp NL { printf("\nValid expression & Answer: %d \n",$1);

exit(0);}

|

exp1 NL { printf("\nValid Expression \nBut, Calculation Can Be Performed On Variables \n");

exit(0);}

;

exp : exp '+' exp {$$=$1+$3; printf("+");}

| exp '-' exp {$$=$1-$3; printf("-");}

| exp '\*' exp {$$=$1\*$3; printf("\*");}

| exp '/' exp {$$=$1/$3; printf("/");}

| '(' exp ')' {$$=$2;}

| NUMBER {$$=$1; printf("%d",yylval);}

;

exp1 : exp1 '+' exp1 {printf("+");}

| exp1 '-' exp1 {printf("-");}

| exp1 '\*' exp1 {printf("\*");}

| exp1 '/' exp1 {printf("/");}

| '(' exp1 ')'

| ID {printf("%s",yylval);}

;

%%

int yyerror(char \*msg)

{

printf("Invalid Expression \n");

exit(0);

}

main()

{

printf("Enter the expression : \n");

yyparse();

}

int yywrap(){return 1;}

**Output:**

C:\Users\silvi\Desktop\Try\College\6th\_sem\E1>flex 2alex.l

C:\Users\silvi\Desktop\Try\College\6th\_sem\E1>bison -dy postfix.y

C:\Users\silvi\Desktop\Try\College\6th\_sem\E1>gcc lex.yy.c y.tab.c

2alex.l: In function 'yylex':

2alex.l:6:9: warning: incompatible implicit declaration of built-in function 'strdup'

[a-zA-Z] {yylval=strdup(yytext); return ID;}

^

2alex.l:6:8: warning: assignment makes integer from pointer without a cast

[a-zA-Z] {yylval=strdup(yytext); return ID;}

^

C:\Users\silvi\Desktop\Try\College\6th\_sem\E1>a.exe

Enter the expression :

a+b

ab+

Valid Expression

But, Calculation Can Be Performed On Variables

C:\Users\silvi\Desktop\Try\College\6th\_sem\E1>a.exe

Enter the expression :

2+3

23+

Valid expression & Answer: 5

**Practical E2**

**Aim: Write YACC specification to recognize strings that can be accepted by grammar of the form: a n bn c, n>=1**

**Program:**

**E2lex.l**

%{

/\* Definition section \*/

#include "y.tab.h"

%}

/\* Rule Section \*/

%%

[aA] {return A;}

[bB] {return B;}

[cC] {return C;}

\n {return NL;}

. {return yytext[0];}

%%

int yywrap()

{

return 1;

}

**E2.y**

%{

/\* Definition section \*/

#include<stdio.h>

#include<stdlib.h>

%}

%token A B C NL

/\* Rule Section \*/

%%

stmt: A S B S C S NL { printf("valid string\n");

exit(0); }

;

S: A S B |

;

%%

int yyerror(char \*msg)

{

printf("invalid string\n");

exit(0);

}

//driver code

main()

{

printf("enter the string\n");

yyparse();

}

**Output:**

C:\Users\silvi\Desktop\Try\College\6th\_sem\E1>gcc lex.yy.c y.tab.c

C:\Users\silvi\Desktop\Try\College\6th\_sem\E1>gcc a.exe

enter the string

aabbc

valid string

C:\Users\silvi\Desktop\Try\College\6th\_sem\E1>gcc a.exe

enter the string

aaabbc

invalid string

**Practical E3**

**Aim: To validate syntax of following programing language construct: if else statement**

**Program:**

**E3.l**

%{

#include "y.tab.h"

%}

%%

"if" { return IF; }

"else" { return ELSE; }

"(" { return '('; }

")" { return ')'; }

. { /\* ignore other characters \*/ }

%%

int yywrap() {

return 1;

}

**E3.y**

%{

#include <stdio.h>

%}

%token IF ELSE

%left THEN

%right ELSE

%%

statement: IF '(' expression ')' statement %prec THEN { printf("\nValid expression\n"); exit(0);}

| IF '(' expression ')' statement ELSE statement { printf("\nValid expression\n"); exit(0);}

| /\* other statements \*/

;

expression: /\* define your expression rules here \*/

;

%%

int main() {

yyparse();

return 0;

}

int yyerror(const char \*s) {

fprintf(stderr, "Parse error: %s\n", s);

return 0;

}

**Output:**

C:\Users\silvi\Desktop\Try\College\6th\_sem\E3>flex E3.l

C:\Users\silvi\Desktop\Try\College\6th\_sem\E3>bison -dy E3.y

C:\Users\silvi\Desktop\Try\College\6th\_sem\E3>gcc lex.yy.c y.tab.c

E3.y: In function 'yyparse':

E3.y:11:39: warning: incompatible implicit declaration of built-in function 'exit'

statement: IF '(' expression ')' statement %prec THEN { printf("\nValid expression\n"); exit(0);}

^

E3.y:12:39: warning: incompatible implicit declaration of built-in function 'exit'

| IF '(' expression ')' statement ELSE statement { printf("\nValid expression\n"); exit(0);}

^

C:\Users\silvi\Desktop\Try\College\6th\_sem\E3>a.exe

if (x < 18)

y = x + 2

else

y = x \* 2

Valid expression