

**Compiler Design Lab**

**SUBMITTED BY: SUBMITTED TO:**

Name: Harsh Agrawal Dr. Ajay Prasad

SAP ID: 500107136

Enrollment No: R2142220326

Course- B.Tech. (CSE)

Batch: 4 (AI/ML) (Non- Hons)

Assignment 4

1. WAP to test the validity of a simple expression involving operators- +, -, \*, and /.

Sol-

**lexer.l-**

%{

#include <stdlib.h>

#include <stdio.h>

void yyerror(char \*);

#include "y.tab.h"

extern int yylval;

%}

%%

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

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

\( { return '('; }

\) { return ')'; }

[ \t\n] { /\* Ignore whitespaces \*/ }

. { printf("Invalid character: %s\n", yytext); }

%%

int yywrap() {

return 1;

}

**parser.y-**

%{

#include <stdio.h>

#include <stdlib.h>

void yyerror(const char \*s);

int yylex();

%}

%token NUMBER

%left '+' '-'

%left '\*' '/'

%right UMINUS

%%

expr: expr '+' expr { printf("Valid Expression\n"); }

| expr '-' expr { printf("Valid Expression\n"); }

| expr '\*' expr { printf("Valid Expression\n"); }

| expr '/' expr {

if ($3 == 0) {

printf("Error: Division by zero!\n");

YYABORT;

}

$$ = $1 / $3;

printf("Valid Expression\n"); }

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

| '-' expr %prec UMINUS { $$ = -$2; }

| NUMBER { $$ = $1; }

;

%%

void yyerror(const char \*s) {

printf("Syntax Error: %s\n", s);

}

int main() {

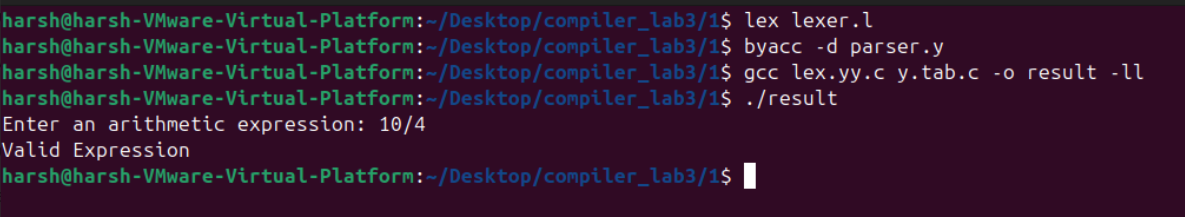
printf("Enter an arithmetic expression: ");

yyparse();

return 0;

}

**Output-**



1. WAP to evaluate an arithmetic expression involving operators- +, -, \*, and /.

Sol-

**lexer.l-**

%{

#include <stdlib.h>

#include <stdio.h>

void yyerror(char \*);

#include "y.tab.h"

extern int yylval;

%}

%%

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

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

\( { return '('; }

\) { return ')'; }

[ \t\n] { /\* Ignore whitespaces \*/ }

. { printf("Invalid character: %s\n", yytext); }

%%

int yywrap() {

return 1;

}

**parser.y-**

%{

#include <stdio.h>

#include <stdlib.h>

void yyerror(const char \*s);

int yylex();

%}

%token NUMBER

%left '+' '-'

%left '\*' '/'

%right UMINUS

%%

expr: expr '+' expr { $$ = $1 + $3; printf("Result = %d\n", $$); }

| expr '-' expr { $$ = $1 - $3; printf("Result = %d\n", $$); }

| expr '\*' expr { $$ = $1 \* $3; printf("Result = %d\n", $$); }

| expr '/' expr {

if ($3 == 0) {

printf("Error: Division by zero!\n");

YYABORT;

}

$$ = $1 / $3;

printf("Result = %d\n", $$);

}

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

| '-' expr %prec UMINUS { $$ = -$2; }

| NUMBER { $$ = $1; }

;

%%

void yyerror(const char \*s) {

printf("Syntax Error: %s\n", s);

}

int main() {

printf("Enter an arithmetic expression: ");

yyparse();

return 0;

}

**Output-**

