**Practical 8**

**[[1]](#footnote-0)**

**Aim - Desk calculator with error recovery.**

**Code-**

**Lex Code (calc.l):\***

%{

#include "y.tab.h"

#include <stdlib.h>

%}

%%

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

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

. { return yytext[0]; }

%%

int yywrap() {

return 1;

}

**\*Yacc Code (calc.y):\***

%{

#include <stdio.h>

#include <stdlib.h>

int yylex();

int yyerror(char \*s);

%}

%token NUMBER

%left '+' '-'

%left '\*' '/'

%%

input : /\* empty \*/

| input line

;

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

| error '\n' { printf("Syntax Error! Please re-enter.\n"); yyerrok; }

;

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

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

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

| expr '/' expr { if ($3 == 0) { printf("Error: Division by zero!\n"); $$ = 0; } else $$ = $1 / $3; }

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

| NUMBER

;

%%

int main() {

printf("Desk Calculator: Enter expressions (Ctrl+C to exit)\n");

yyparse();

return 0;

}

int yyerror(char \*s) {

return 0; // Error message already printed in grammar

}

**Compilation Commands:**

bash

lex calc.l

yacc -d calc.y

gcc lex.yy.c y.tab.c -o calc

./calc

**Sample Input & Output:**

Input: 5+3\*2

Output: Result = 11

Input: 10/(2+3)

Output: Result = 2

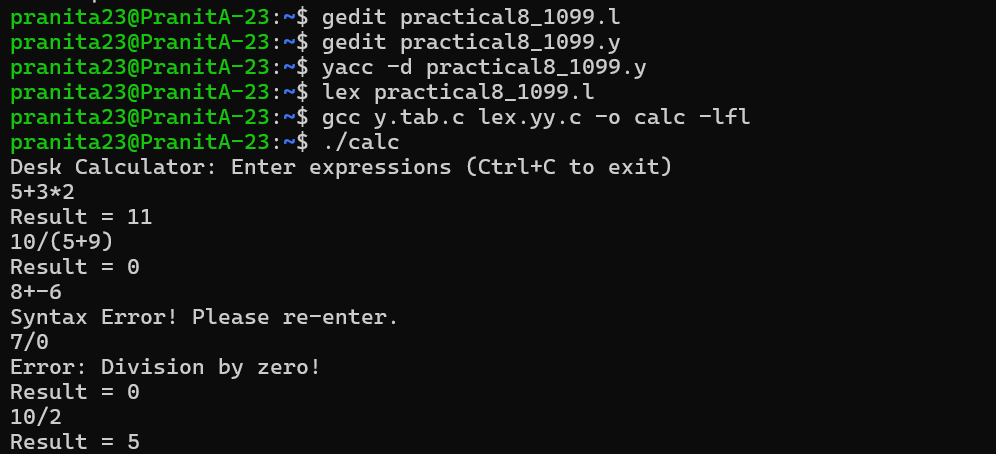
Input: 8+\*2

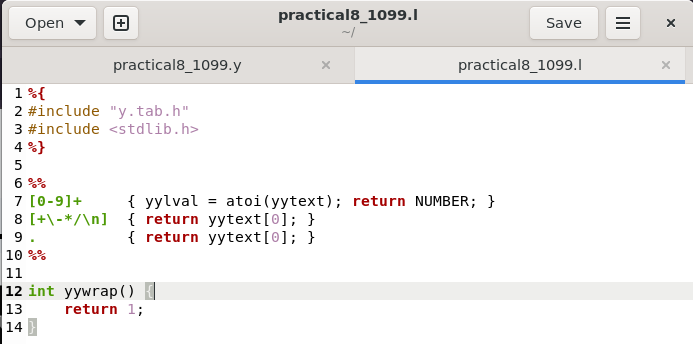
Output: Syntax Error! Please re-enter.

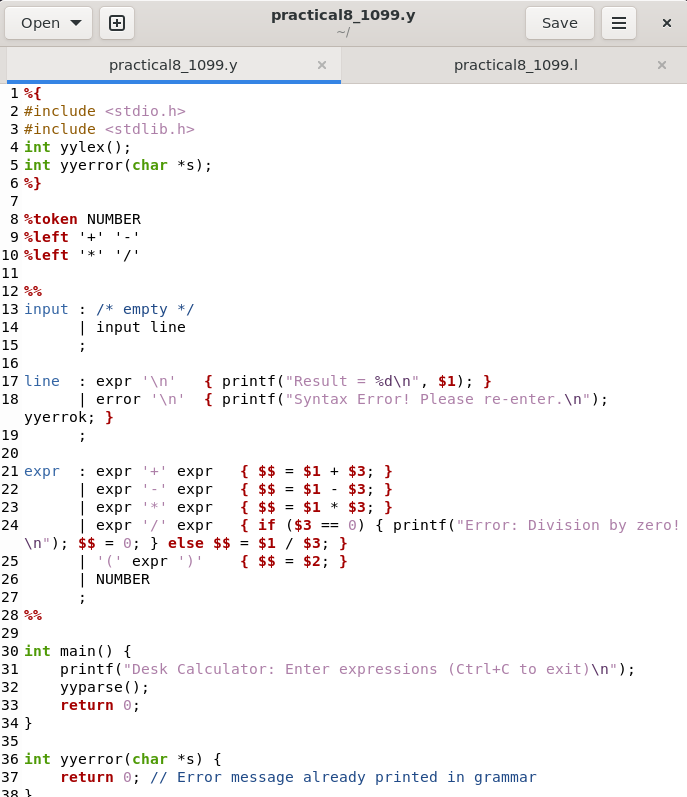
Input: 7/0

Output: Error: Division by zero!

**CODE**



****

****

1. **Name**-Pranita Dadhe; **PRN**-22070521099 [↑](#footnote-ref-0)