Project Report on

Compiler for Calculator Program

Developed by

Rajan Patel – IT110 – 20ITUOS098

Shail Patel – IT111 – 20ITUOS114

Vraj Patel – IT117 – 20ITUOS071

Karan Patwa – IT122 – 20ITUON123

Guided By

Prof. Nilamba Vala

Dept. Of Information Technology

Department of Information Technology (FOT)

Dharmsinh Desai University

College Road, Nadiad-387001 2022-2023

CERTIFICATE

- This is to certify that the project entitled "Compiler for Calculator Program" is a bona fide report of the work carried out by:

➢ Mr. Rajan Patel, Student ID No: 20ITUOS098
➢ Mr. Shail Patel, Student ID No: 20ITUOS114
➢ Mr. Vraj Patel, Student ID No: 20ITUOS071
➢ Mr. Karan Patwa, Student ID No: 20ITUON123

- Of the Department of Information Technology, semester VI, under the guidance and supervision for the award of the degree of Bachelor of Technology at Dharmsinh Desai University, Nadiad (Gujarat). They were involved in the Project in the subject of "Compiler for Calculator Program" during the academic year 2022-2023.

- This certificate is awarded to acknowledge their hard work, dedication, and commitment to excellence in their academic pursuits.

- **Signed And Awarded By:**
  Prof. Nilamba Vala (Lab In Charge)
  Department Of Information Technology, Faculty Of Technology,
  Dharmsinh Desai University, Nadiad

- *Date: 16/03/2023*

Prof. (Dr.) V.K. Dabhi,
Head, Department of Information Technology, Faculty of Technology,
Dharmsinh Desai University, Nadiad

Introduction

Project Details

Grammar Name: Senti Grammar

Valid Sentences In Language:

1. 4 + 5 = 9
2. 2 - 19 = -17
3. 7 / 2 = 3.5
4. 7 * 9 = 63

Keywords: CE

Operators: +, - , * , /

Digit:
[0-9]+   int
[0-9]+(.[0-9]+) float

Punctuation Marks:

Punctuation Mark's Name:	**Punctuation Marks**	**Punctuation Mark's character**
New Line	[\n]	n
Eos	.	e
Separator	,	s
White Space	[/t]	w

First & Follow

Grammar:

```
S -> x = E
E -> TE'
E'->+TE'| -TE'| ε
T -> FT'
T'->*FT'| /FT'| ε
F ->(E)| num | -num
num ->0|1|2|3|4|5|6|7|8|9
```

Non-terminals:

```
 E | E’ | T | T’ | F | num
```

Terminals:

```
0|1|2|3|4|5|6|7|8|9| ε | + | - | / | * | x | =
```

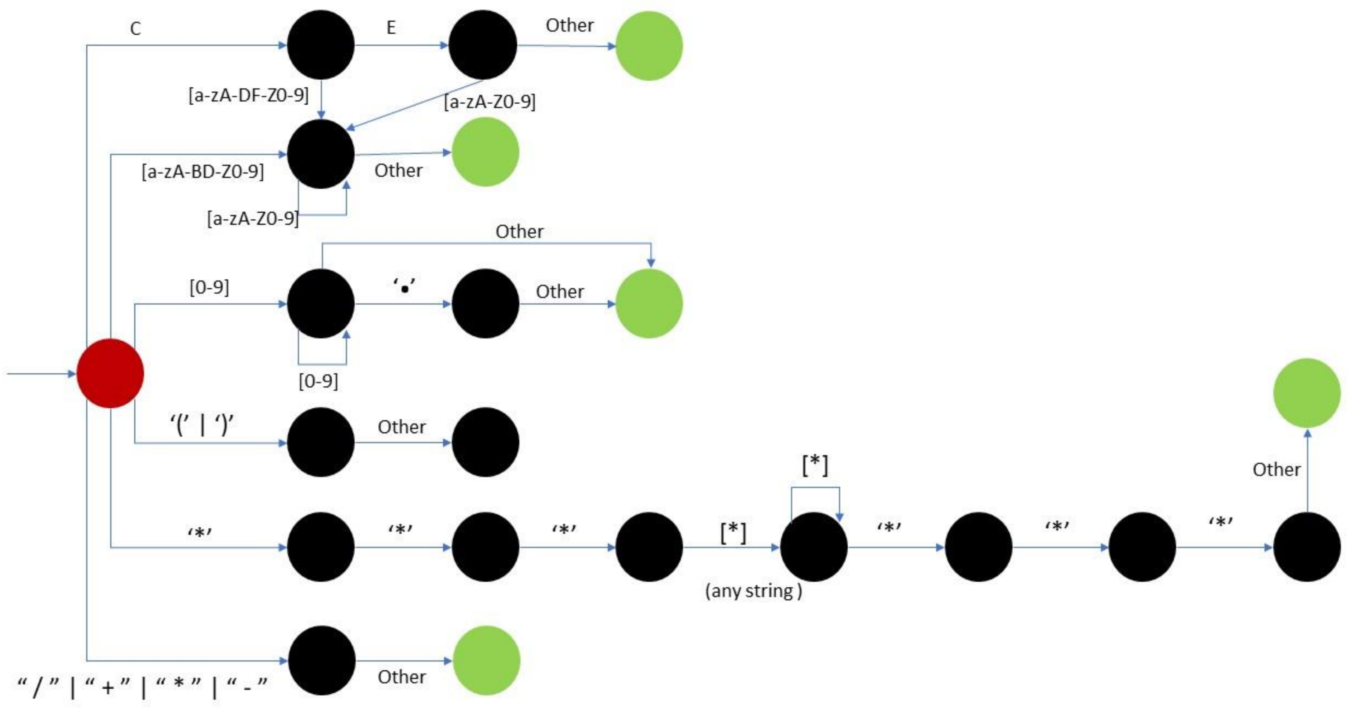
First:

```
FIRST[S] = { x }
FIRST[E]={ (, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
FIRST[E’] = { +, -, ε }
FIRST[T]={ (, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
FIRST[T’] = { *, /, ε }
FIRST[F] ={ -. (, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
FIRST[num]={ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
```

Follow:

```
FOLLOW[S] ={$}
FOLLOW[E] ={ ),x }
FOLLOW[E’] = { )}
FOLLOW[T]={ )}
FOLLOW[T’] = { )}
FOLLOW[F] ={ )}
FOLLOW[num]={ -, (, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
```

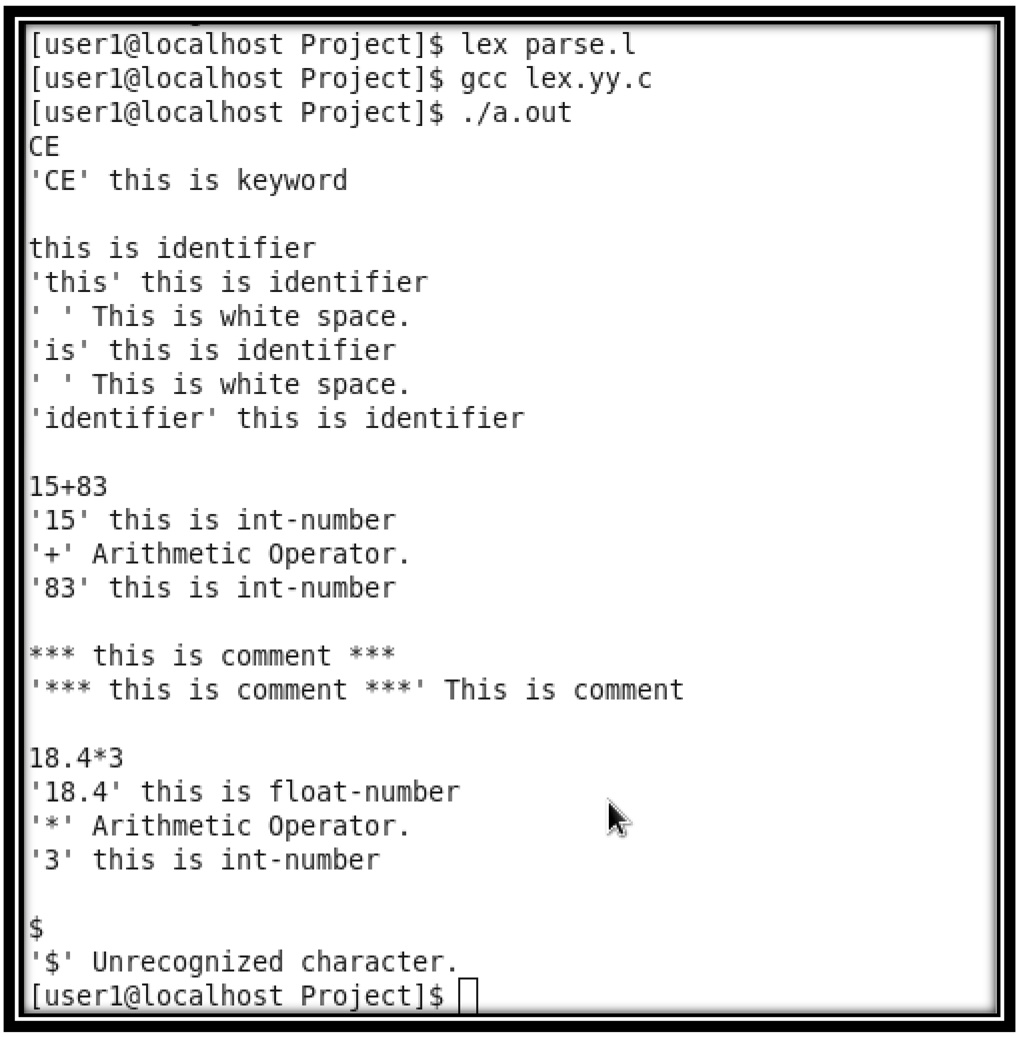
DFA


C
E
Other
[a-zA-DF-Z0-9]
[a-zA-Z0-9]
[a-zA-BD-Z0-9]
Other
[a-zA-Z0-9]
Other
[0-9]
'•'
Other
[0-9]
'(' | ')'
Other
[*]
Other
'*'
'*'
'*'
[*]
'*'
'*'
'*'
(any string)
" / " | " + " | " * " | " - "
Other

Scanner Phase in Lex

```

%%

CE {printf("'%s' this is keyword\n",yytext);}

[a-zA-Z][a-zA-Z0-9]* {printf("'%s' this is identifier\n",yytext);}

[0-9]+"."[0-9]+ {printf("'%s' this is float-number\n",yytext);}

[0-9]+ {printf("'%s' this is int-number\n",yytext);}

"("|")" {printf("'%s' this is Pancuation-mark\n",yytext);}

"+"|"-"|"/"|"*" {printf("'%s' Arithmetic Operator. \n",yytext);}

"*"{3}.*"*"{3} {printf("'%s' This is comment\n",yytext);}

" " {printf("'%s' This is white space. \n",yytext);}

"	" {printf("'%s' This is tab space. \n",yytext);}

. {printf("'%s' Unrecognized character. \n",yytext);return 0;}
%%


int yywrap(){}
int main(){

    yylex();

    return 0;
}
```

Output:



```
[user1@localhost Project]$ lex parse.l
[user1@localhost Project]$ gcc lex.yy.c
[user1@localhost Project]$ ./a.out
CE
'CE' this is keyword

this is identifier
'this' this is identifier
' ' This is white space.
'is' this is identifier
' ' This is white space.
'identifier' this is identifier

15+83
'15' this is int-number
'+' Arithmetic Operator.
'83' this is int-number

*** this is comment ***
'*** this is comment ***' This is comment

18.4*3
'18.4' this is float-number
'*' Arithmetic Operator.
'3' this is int-number

$
'$' Unrecognized character.
[user1@localhost Project]$ 
```

<u>Yacc useful in our Language</u>

Code:

<u>impl.l</u>

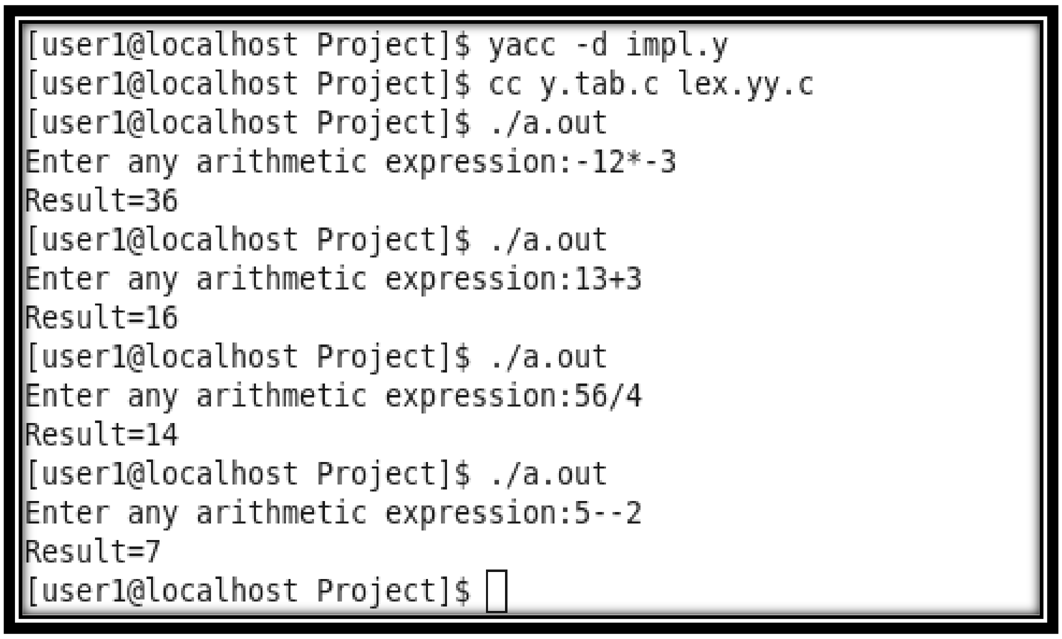
```
%{

%{
#include<stdio.h>
#include "y.tab.h"
extern int yylval;
%}

%%
[0-9]+ {yylval=atoi(yytext); return Num;}
[\n] return 0;
. return yytext[0];
%%

int yywrap(){return 1;}
```

impl.y

```
%{
#include<stdio.h>
%}

%token Num
%left '+' '-'
%left '/' '*'
%left '(' ')'
%left uminus
%%

Arit: E{
    printf("Result=%d\n", $$);
    return 0;
};

E:E'+'E {$$=$1+$3;}
|E'-'E {$$=$1-$3;}

 |E'*'E {$$=$1*$3;}

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

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

 |'-'E {$$=-$2;}

|Num{$$=$1;}
;

%%

void yyerror(){

}
```

```c
void main(){
    printf("Enter any arithmetic expression:");
    yyparse();
}
```

Output:



```
[user1@localhost Project]$ yacc -d impl.y
[user1@localhost Project]$ cc y.tab.c lex.yy.c
[user1@localhost Project]$ ./a.out
Enter any arithmetic expression:-12*-3
Result=36
[user1@localhost Project]$ ./a.out
Enter any arithmetic expression:13+3
Result=16
[user1@localhost Project]$ ./a.out
Enter any arithmetic expression:56/4
Result=14
[user1@localhost Project]$ ./a.out
Enter any arithmetic expression:5--2
Result=7
[user1@localhost Project]$ 
```

Conclusion

- In conclusion, the project report on the creation of a calculator in D-lang using flex-tools, scanner code in C++, and Yacc tool code represents a comprehensive demonstration of the use of these powerful tools in building complex software applications.

- The project showcases the ability of Lexical Analyzer and Parser Generator tools to recognize patterns and handle complex grammars in the input stream, and the use of compiler tools to generate optimized code for executing the program.
- The report demonstrates the effective use of C++ and D-lang in building an interactive calculator with a user-friendly interface, along with the ability to handle basic arithmetic operations.

- Overall, the project report represents a valuable contribution to the field of computer programming and software development, showcasing the power and flexibility of these tools in building complex software applications.

- It can be a useful resource for students, researchers, and professionals interested in building software applications that require parsing and processing of input data.