VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

COMPILER DESIGN

Submitted by

JEEVANTHI KASHYAP(1BM21CS080)

Under the Guidance of
Prof. Sunayana S
Assistant Professor, BMSCE

in partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING

(Autonomous Institution under VTU)

BENGALURU-560019

November 2023-February 2024

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum) Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "**Compiler Design**" carried out by **JEEVANTHI KASHYAP(1BM21CS080)** , who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfilment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2023-24.

The Lab report has been approved as it satisfies the academic requirements in respect of **Compiler Design- (22CS5PCCPD)** work prescribed for the said degree.

Prof. Sunayana S	Dr. Jyothi Nayak
Assistant professor	Professor and Head
Department of CSE	Department of CSE
BMSCE, Bengaluru	BMSCE, Bengaluru

B. M. S. COLLEGE OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

I, Jeevanthi Kashyap (1BM21CS080), student of 5th Semester, B.E, Department of Computer Science and Engineering, B. M. S. College of Engineering, Bangalore, here by declare that, this lab report entitled " **Compiler Design**" has been carried out by me under the guidance of Prof. Sunayana S, Assistant Professor, Department of CSE, B. M. S. College of Engineering, Bangalore during the academic semester November-2023-February-2024.

I also declare that to the best of my knowledge and belief, the development reported here is not from part of any other report by any other students.

TABLE OF CONTENTS

CD- Observation Book.

INDEX

Name: Jeevanthi Kashyap Std: Sec:

Roll No.: Subject: ~~Logic~~ Compiler Design School/College:

School/College Tel. No.: Parents Tel. No.:

Sl. No.	Date	Title	Page No.	Tea Si Rem
9.	11/1/24	Program to demonstrate String Parser		
10.	29/1/24	Program to show syntax tree generator		
11.	29/1/24	Program to show infix to postfix [YACC]		
12.	29/1/24	Three-address Code generator using YACC		

PROGRAM-1

Lex Program to identify ~~alphab~~ each character as vowel or consonants in a given sentence.

```
%{ #include<stdio.h>
%}
[aeiouAEIOU]    { printf("%c is a vowel",
                        yytext[0]);}
[a-zA-Z]        { printf("%c is a consonant",
                        yytext[0]);}

%%
int main (){
    yylex();
    return 0;
}
```

Output :-

```
Hi World
H is a Consonant
I   "  "  vowel
W   "  "  consonant
O   "  "  vowel
R   "  "  consonant
L   "  "  "
D   "  "  "
```

```
%option noyywrap
%{
    #include <stdio.h>
int vowel= 0;
int cons= 0;
%}
%%
[aeiouAEIOU]* {vowel++;} {printf("vowels = %s", vowel);}
[a-zA-Z]* {cons ++;} {printf("Consonants = %s", cons);}

%%
int main ()
{
    yylex();
    return 0;
}
```

Output:-

Trees
vowel = 2
consonants = 3

OUTPUT SCREENSHOT:

```
Enter a sentence:
Compiler design
 No of vowels and consonants are 5 and 9
This is a book
   No of vowels and consonants are 5 and 6
^C
```

PROGRAM-2

```
%{
  #include<stdio.h>
%}%
%%
[0-9]+   { printf ("Integer: %s", yytext);}
[a-zA-Z]+  { printf ("variable: %s", yytext);}
[\t\n]     {printf ("Whitespace
void main ()
{ printf ("Enter the file name: ");
  Scanf ("%s", fname);
  yylen = fopen (fname, "r");
  yylex();
  fclose (yylen);
}
Output:
keywords: float
number: 0978
Character: abc

To print output on output file
%%
int |float |char { printf (yyout, "keyword: %s",
                                  yytext);}
  [0-9]*  {printf (yyout, "number: %s
                                  yytext);}
[a-zA-Z]* {printf (yyout, "character: %s"
                                  yytext);}
%%
void main ()
{ printf ("Enter input file name:");}
  Scanf ("%s", fname);
  printf ("Enter output file name:");
      scanf ("%s", fname);
```

Count the no of words

```
%{
  #include <stdio.h>
 int c = 0;
%}
%%
 [a-zA-Z 0-9]+ { c++; }
\n { printf ("Count is %d", c);}
%%
 int yywrap()
 {
   }
 int main ()
{
 printf ("Enter the sentence:");
  yylex();
  return0;
}
```

Output :

Enter the sentence: My name is Jeevanthi
Count = 4.

20/11/23

continued:

```
yyin = fopen(fname, "r");
yyout = fopen (foname, "w");
yylex();
fclose(yyin);
fclose(yyout);
}
```

Output:
Enter input file name
p.txt
enter output file name
pout.txt

OUTPUT SCREENSHOT:

```
enter the input file name
input.txt
enter the output file name
output.txt
```

```
int a,b;
```

```
int Keywords a Identifiers, Seperatorb Identifiers; Seperator
```

PROGRAM-3:

Lab-3:-

① Write a program in LEX to recognize Floating Point Numbers. Check for all the following input cases.

```
%{
 #include <stdio.h>
%}
%%
^[+-]?[0-9]*[.][0-9]+ {printf("Floating Point
 Number"); }
^[+-]?[0-9]* {printf("Not a valid floating
 point Number"); }
%%
int yywrap()
{
}
int main()
{
 yylex()
 return 0
}
```

Output:-

```
55.
Not a valid floating point Number
33.69
Floating point Number
+12.
Not a floating point number
-625.0
Floating point number
-.3
Floating point Number
```

OUTPUT SCREENSHOT:

```
Enter a number:
23
Not a floating point number!

0.5
Floating point number!

.8
Floating point number!

-.9
Floating point number!

+56
Not a floating point number!
```

PROGRAM-4

Lab-4:-

① Write a LEX program that copies a file, replacing each nonempty sequence of white space by a single blank.

```
%{
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
char str1[200];
%}
%%
[\n]  {fprintf(yyout,"%s\n", str1); str1[0]='\0';}
[ ]*|[\t]  {fprintf(yyout, "%s", str1);
    str1[0]='\0'; fprintf(yyout, "%s", " ");}
.  strcat(str1, yytext);
<<EOF>>  {fprintf(yyout, "%s", str1);
        return 0; }
%%
int main()
{
extern FILE *yyin, *yyout;
char filename[100];
printf("Enter the name of the file to copy:\t");
scanf("%s", filename);
yyin = fopen(filename, "r");
if (yyin == NULL)
{ exit(0);
}
printf("Enter the name of the file to write:\t");
scanf("%s", filename);
yyout = fopen(filename, "w");
if (yyout == NULL)
{ exit(1);
}
yylex();
}
int yywrap()
{
}
```

Output:-

Enter the name of the file to copy
input.txt
Enter the name of the file to write
Output.txt.

OUTPUT SCREENSHOT:

```
Enter the name of the file to copy:     input.txt
Enter the name of the file to write:    output.txt
```

```
re1.l   re2.l   re3.l   re4.l   re5.l   re6.l   re7.l   output.txt   input.txt
python is an    interpreted     programming language.
```

```
re1.l   re2.l   re3.l   re4.l   re5.l   re6.l   re7.l   output.txt   input.txt
python is an interpreted programming language.
```

PROGRAM-5

② Write a LEX program to recognize the following tokens over the alphabets {0, 1, .. 9}

a] Set of all strings ending with 00

```
%%
[0-9]*00 { printf ("String accepted"); }
 [0-9]* { printf ("String rejected"); }
%%
int yywrap()
{
}
int main()
{
 yylex();
 return 0;
}
```

Output :-

1 0100
String accepted
34550
String rejected

b] Set of all strings with 3 consecutive 222's

```
%%
[0-9]* 222 [0-9]* {printf ("string accepted");
[0-9]*            { printf ("string rejected");
```

d)

```
%{
#include <stdio.h>
#include <math.h>
int value =0, i, j=0, flag=0;
%}
%%
1[01]* { for (i= yyleng-1; i>=0; i--)
        {
          value = value+(yytext [i]-48) * pow(2,j);
          j++;
        }
        if (value % 5 == 0)
        { flag = 1;
        }
        }
[\n] return 0;
%%
int yywrap(){}
int main()
{ yylex();
  if (flag == 1)
  { printf("success\n");
  }
  else
  { printf("fail\n");
  }
  return 0;
}
```

Output:-

1111

Success

e]
```
%{
#include <stdio.h>
%}
digits [0-9]
%%
{digits}*1{digits}{digits}{digits}{digits}{digits}{digits}
        {digits}{digits}{digits}{digits}
    {printf("%s 10th symbol from right end is
            1", yytext);}
{digits}* {printf("%s not 1", yytext);}
%%
int yywrap(){}
int main()
{
  yylex();
 return 0;
}
```

Output:-

10428169.73

10 th symbol from right end is 1

f]
```
%{
#include <stdio.h>
#include <math.h>
int value=0, i, j=0, flag=0;
%}
digits [0-9]
%%
{digits}{digits}{digits}{digits}{digits} {for (i=yyleng-1)
                                          i>=0; i--)
{
  value+=(yytext[i]-48);
```

```
    if (value == 9)
    { flag = 1;
    }
  }
[\n] return 0;
%%
int yywrap() { }
int main()
{ yylex();
  if (flag == 1)
  { printf ("success\n");
  }
  else
  { printf ("fail\n");
  }
  return 0;
}
g] %{
#include <stdio.h>
#include <math.h>
int value = 0, i, j=0, flag = 1;
%}
digits [0-9]
%%
{digits}{digits}{digits}{digits}{digits} {for (i=0; i < yyleng-1; i++)
    { if (yytext [i] > yytext [i+1])
        { flag=0;
        }
    }
  }
[\n] return 0;
%%
```

```
int yywrap() { }
int main()
{ yylex();
  if (flag == 1);
  { printf ("success \n");
  }
  else
  { printf ("fail \n");
  }
  return 0;
}
```

11/12/2023

OUTPUT SCREENSHOT:

```
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
1111
successbmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
11
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex re5.l
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
1023002245
1023002245 10th symbol from right end id 1
^Z
[1]+  Stopped                 ./a.out
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex re6.l
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
9000
success
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
4005
success
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
123
123fail
```

```
fail
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex blank.l
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
Enter the name of the file to copy:     input.txt
Enter the name of the file to write:    output.txt
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex re1.l
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
24900
24900 string ends with 00
2352
2352 string does not end with 00
^Z
[2]+  Stopped                 ./a.out
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex re2.l
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
12142
12142 string does not have 222
24322245
24322245 string has 222
```

```
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex re4.l
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
usr/bin/ld: /tmp/ccNpRHPT.o: in function `yylex':
ex.yy.c:(.text+0x33f): undefined reference to `pow'
ollect2: error: ld returned 1 exit status
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c -lm
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
01
uccessbmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c -lm
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
111
uccessbmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
1
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex re5.l
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
023002245
023002245 10th symbol from right end id 1
Z
1]+  Stopped                 ./a.out
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex re6.l
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ cc lex.yy.c
mscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
```

```
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ lex re7.l
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ gcc lex.yy.c
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
45612
2fail
bmscecse@bmscecse-OptiPlex-5070:~/Documents/1BM21CS083$ ./a.out
1234
success
```

PROGRAM-6

Lab-5 :- 18-12-23

Write a program to design Lexical Analyzer in C/C++/Java/Python Language (to recognise any fine keywords, identifiers, nos, operators

```
#include<stdio.h>
#include <string.h>
#include <ctype.h>
void lexAnalyzer (char input[]) {
char *keyword[] = {"if", "else", "while", "for",
                   "return"};
char *operator[] = {"+", "-", "*", "/", "=", "==",
                    "<", ">", "<=", ">="};
char *punctuation[] = {",", ";", "(", ")", "{",
                       "}"};
char *token[] = strtok(input, "\t\n");
while (token != NULL) {
  if (isdigit(token[0])) {
    printf("Number: %s\n", token);
  } else if (isalpha(token[0]) || token[0] == '_') {
    int iskeyword = 0;
    for (int i = 0; i < sizeof(keywords)/
        sizeof(keywords[0]); i++) {
      if (strcmp(token, keyword[i]) == 0) {
        printf("Keyword: %s\n", token);
        iskeyword = 1;
        break;
      }
    }
    if (!keyword) {
      printf("Identifier: %s\n", token);
    }
```

```
{ else if (strchr("+-* /=<>; (), ", token[0]) != NULL){
  printf("Punctuation/Operator: %s\n", token);
 }
 token = strtok(NULL, "\t\n");
}
}
int main(){
char input[] = "if(x != 0) { return 1; }";
lexAnalyzer(input);
return 0;
}
```

Output:-

```
if(x != 0){
return 1; }
```

~~Keyword: if~~
~~Punctuation/Operator: (~~
~~Identifier: x~~
~~Punctuation/Operator: !~~
~~Number~~

Keyword: if
Punctuation/Operator: (
Identifier: x
Punctuation/Operator: =
Number: 0
Punctuation/Operator:)
Keyword: return
Number: 1
Punctuation/Operator: ;

10/10

18/12/2023

OUTPUT SCREENSHOT:

```
enter c code
int a = 1234 ;
Keyword: int
Identifier: a
Punctuation/Operator: =
Number: 1234
Punctuation/Operator: ;
```

PROGRAM-7

Week-6 — Recursive Descent Program 8-01-22

```
#include <stdio.h>
int index=0;
char input_string[100];
void parse_S();
void parse_A();
void parse_S() {
  if (input_string[index]=='c'){
    index++;
    parse_A();
  if (input_string[index]=='d') {
    index++;
  } else {
    printf("Not parsed\n");
      return;
  }
  else {
    printf("Not parsed\n");
      return;
  } }
void parse_A() {
  if (input_string[index]=='a') {
    index++;
    if (input_string[index]=='b') {
      index++;
  } }
} }
int main(){
printf("Enter a string to parse:");
scanf("%s", input_string);
parse_S();
if (input_string[index]=='\0') {
    printf("String can be parsed\n");
} else {
```

```
printf("string cannot be parsed \n");
}
return 0;
}
```

O/P

① Enter a string to parse: cad
String can be parsed

② Enter a string to parse: cabd
string can be parsed

③ Enter a string to parse: caaad
String cannot be parsed.

OUTPUT SCREENSHOT:

```
nter the input string:
ad
ello
arsing failed. Extra characters found.
mscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ ./recursive_descent
nter the input string:
aaad
ello
arsing failed. Extra characters found.
mscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ ./recursive_descent
nter the input string:
ab$
ello
arsing successful.
mscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ ./recursive_descent
nter the input string:
aad$
ello
arsing failed. Extra characters found.
mscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ ./recursive_descent
nter the input string:
abd$
ello
arsing successful.
mscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~/Documents$ ./recursive_descent
nter the input string:
aaad$
ello
arsing failed. Extra characters found.
```

PROGRAM-8

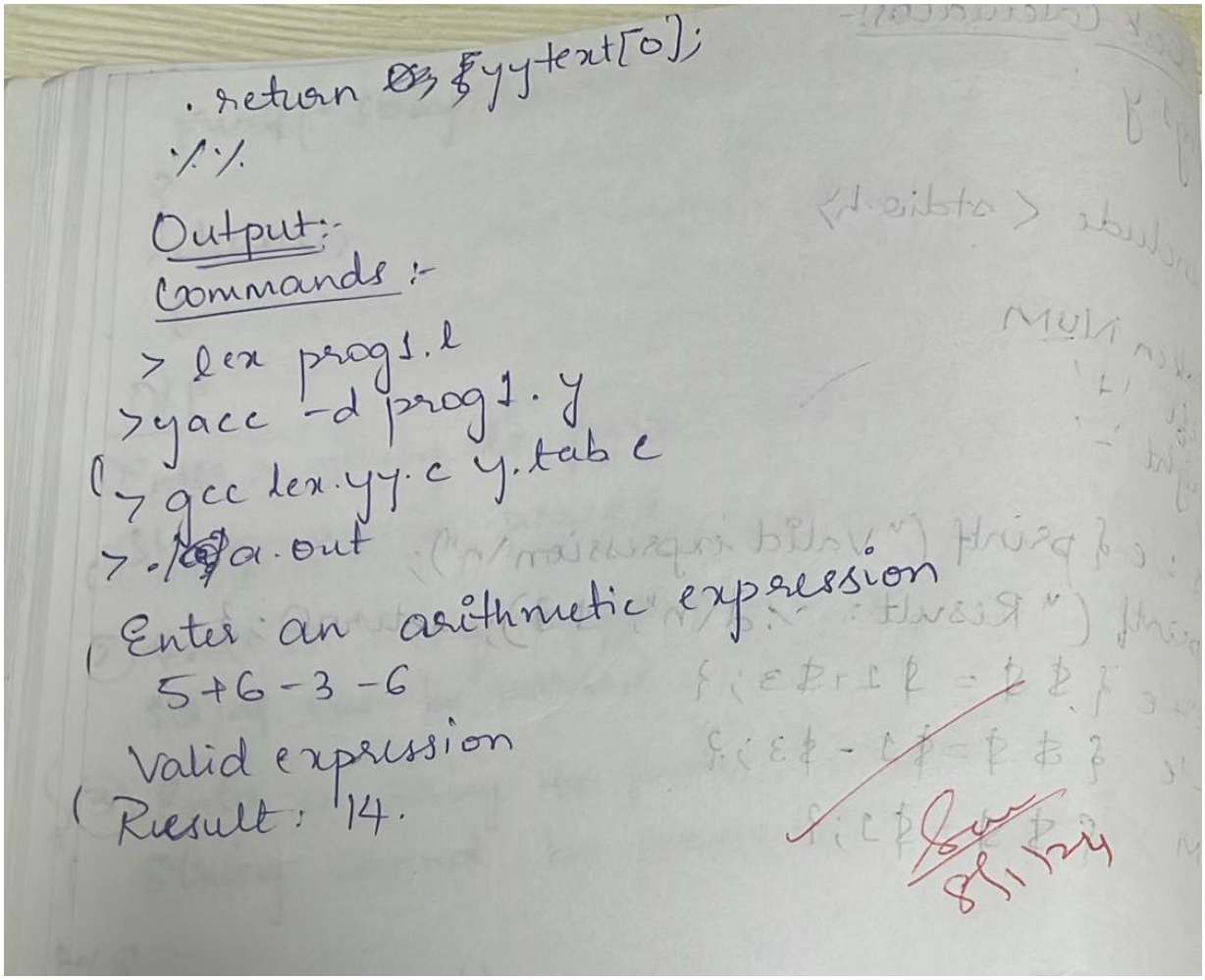
② Desk Calculator:-

prog1.y :-

```
%{
#include <stdio.h>
%}
%token NUM
%left '+'
%right '-'
%%
expr : e {printf ("Valid expression\n");
   printf ("Result: %d\n", $$); return0; }
e: e'+'e { $$ = $1+$3; }
|e'-'e  { $$ = $1 - $3; }
|NUM    { $$ = $1; }
;
%%
int main()
{printf ("\n Enter an arithmetic expression\n");
 yyparse();
 return0;
}
int yyerror()
{
printf ("\n Invalid expression\n");
return 0;
}
```

prog1.l :-

```
%option noyywrap
%{
#include"y.tab.h"
%}
%%
[0-9]+ {yylval=atoi(yytext); return NUM;}
[\t] ;
\n return0;
```



. return yytext[0];

%%

Output:-

Commands :-

> lex prog1.l

> yacc -d prog1.y

> gcc lex.yy.c y.tab.c

> ./a.out

Enter an arithmetic expression

5+6-3-6

Valid expression

Result: 14.

OUTPUT SCREENSHOT:

```
Enter an arithmetic expression:
2+3*4
Valid expression!
Result:14
```

```
Enter an arithmetic expression:
2++3-
Invalid expression!
```

PROGRAM-9

String Match Program:-

P.l :-

```
%{
#include <stdio.h>
#include <stdlib.h>
#include "y.tab.h"
extern int yylval;
%}
%%
[aA] {yyval= yytext[0]; return A;}
[bB] {yyval= yytext[0]; return B;}
\n {return NL;}
. { return yytext[0];}
%%
int yywrap()
{
return 1;
}
```

P.y

```
%{
#include <stdio.h>
#include <stdlib.h>
int yyerror (char * s);
int yylex (void);
%}
%token A
%token B
%token NL
%%
smtr : A A A A A S B NL {printf ("Parsed using
        the rule (a^n)b, n>=5. \n Valid Strin
        \n");}
```

```
;
S: S A
!
;
%%
void main()
{printf("Enter a string!\n");
 yyparse();
}
int yyerror(char *s)
{printf("Invalid String!\n");
 return 0;
}
```

Output:-

Enter a string:
aaaaaaab
Parsed using the rule (a^n)b, n>=5
Valid String!
ab
Invalid String

OUTPUT SCREENSHOT:

```
Enter a string!
aaaaaaab
Parsed using the rule (a^n)b, n>=5.
Valid String!
ab
Invalid String!
```

```
Enter a string!
abc
Invalid String!
```

PROGRAM-10

Write a YACC program to generate syntax tree for the given arithmetic expression.

p.l

```
%{
#include "y.tab.h"
 extern int yylval;
%}
%%
[0-9]+   { yyval = atoi(yytext); return digit ;}
 [\t "] ;
 [\n] return 0;
  return yytext[0];
%%
int yywrap()
{
}
```

p.y

```
%{
#include<math.h>
#include<ctype.h>
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
struct tree node
{
  char val[10];
   int lc;
   int rc;
};
```

```
int ind;
struct treenode syn_tree [100];
void my_print_tree (int curr_ind);
int mknode (int lc, int rc, char val[]
%}
% token digit
%%
S: E { my_print tree ($1);}
;
E: E'+'T { $$= mknode ($1, $3, "+");}
|T { $$= $1;}
;
T: T'*'F { $$=mknode ($1, $3, "*");}
|F { $$= $1; }
;
F: '('E')' {$$= $2;}
|digit { char buf [10]; sprintf (buf "%d",
                                 yyval),
    $$= mknode (-1, -1, buf);}
%%
int main()
{ ind = 0;
printf (" Enter an expression\n");
yyparse();
return 0;
}
int yyerror()
{ printf ("NITW Error\n");
}
```

```
int mknode (int lc, int rc, char val[10])
{ strcpy (syn_tree [ind].val, val);
  syn_tree [ind].lc = lc;
  syn_tree [ind].rc = rc;
  ind++; return ind-1;
}
void my_print_tree (int cur_ind)
{ if (cur_ind == -1)
    return;
  if (syn_tree[cur_ind].lc == -1 && syn_tree
    [cur_ind].rc == -1)
    printf ("Digit Node -> Index: %d", value:
                %s \n",
        cur_ind, syn_tree [cur_ind].val);
  else
  printf("Operator Node -> Index: %d, Value:
    %s, Left Child Index: %d, Right Child
    Index: %d \n", cur_ind, syn_tree [curind].
    val, syn_tree [cur_ind].lc, syn_tree[cur_
                ind].rc);
  my_print_tree (syn_tree [cur_ind].lc);
  my_print_tree (syn_tree [cur_ind].rc);
}
```

Output:-

Enter an expression:
4 + 6 * 9

Operator Node -> Index : 4, Value '+', left child index : 0, Right Child Index : 3

Digit Node -> Index : 0, Value : 4

Operator Node -> Index : 3, Value : *, left child Index : 1, Right Child Index : 2

Digit Node -> Index : 1, Value : 6

Digit Node -> Index = 2, Value = 9.

OUTPUT SCREENSHOT:

```
Enter an expression:
2*3+5*4
Operator Node -> Index : 6, Value : +, Left Child Index : 2,Right Child Index : 5
Operator Node -> Index : 2, Value : *, Left Child Index : 0,Right Child Index : 1
Digit Node -> Index : 0, Value : 2
Digit Node -> Index : 1, Value : 3
Operator Node -> Index : 5, Value : *, Left Child Index : 3,Right Child Index : 4
Digit Node -> Index : 3, Value : 5
Digit Node -> Index : 4, Value : 4
```

PROGRAM-11

Infix to Postfix Program :-

p.l :-

```
%{
#include <stdio.h>
#include <stdlib.h>
#include "y.tab.h"
extern int yylval;
%}
%%
[0-9]+ {yylval= atoi(yytext); return num;}
[\t] ;
\n { return 0;}
. { return yytext[0];}
%%
int yywrap()
{
}
```

p.y :

```
%{
#include <stdio.h>
#include <stdlib.h>
int yyerror (const char *s);
int yylex (void);
%}
%token num
%left '+' '-'
%left '*' '/'
%left ')'
%left '('
%right '^'
%%
s: e { printf ("\n");}
;
```

```
e : e '+' t { printf ("+"); }
| e '-' t { printf ("-"); }
| t
;
t : t '*' h { printf ("*"); }
| t '/' h { printf ("/"); }
| h
;
h : f '^' h { printf ("^"); }
| f
;
f : '(' e ')'
| num { printf ("%d", $1); }
;
%%
void main ()
{
printf ("Enter an infix expression :\n");
yyparse ();
}
int yyerror (const char * s)
{
printf ("Invalid Infix Expression !\n");
return 0;
}
```

Output:-

Enter an infix expression:

2+3*8/4^3-3

238*43^/+3-

OUTPUT SCREENSHOT:

```
Enter an infix expression:
2+3*8/4^3-3
238*43^/+3-
```

PROGRAM-12

Address Code - Program

p.l

```
%{
#include <stdio.h>
# include <stdlib.h>
# include "y.tab.h"
extern int yylval;
extern char idn[20];
%}
d [0-9]+
a [a-z A-Z]+
%%
{d} { yylval = atoi (yytext); return digit;
{a} { strcpy (idn, yytext); yylval = 1;
                                return id;}

[\t] {;}
\n return 0;
. return yytext[0];
%%
int yywrap()
{
return 1
}
```

p.y

```
%{
#include <math.h>
#include <ctype.h>
#include <stdio.h>
int yyerror (char *s);
int yylex (void);
int var_cnt = 0;
char idn[20];
%}
```

```
%token id
%token digit
%%
S: id '=' E {printf("%s = t%d\n", iden, var_cnt-1);}
E: E '+' T {$$= var_cnt; var_cnt++; printf("t%d=
                t%d + t%d; \n", $$, $1, $3);}
| E '-' T { $$= var_cnt; var_cnt++; printf
                ("t%d = t%d + t%d; \n",
                $$, $1, $3);}
| T = {$$ = $1;}
;
T: T '*' F { $$= var_cnt; var_cnt++;
            printf("t%d = t%d * t%d; \n", $$,
                $1, $3);}
| T '/' F { $$= var_cnt; var_cnt++; printf("t%d=
            t%d / t%d; \n", $$, $1, $3);}
| F. {$$ = $1;}
;
F: P '^' F {$$= var_cnt; var_cnt++; printf("t%d=
            t%d ^ t%d; \n", $$, $1, $3);}
| P. {$$ = $1;}
;
P: '(' E ')' {$$ = $2;}
| digit {$$=var_cnt; var_cnt++; printf("t%d=%d; \
                n", $$, $1);}
;
%%
int main()
{ var_cnt = 0;
printf("Enter an Expression: \n");
  yyparse();
  return 0;
}
```

OUTPUT SCREENSHOT:

```
Enter an expression:
a=2*3/6-4
t0 = 2;
t1 = 3;
t2 = t0 * t1;
t3 = 6;
t4 = t2 / t3;
t5 = 4;
t6 = t4 - t5;
a=t6
```