Techno India NJR Institute of Technology

Lab Manual
Compiler Design Lab (5CS4-22)

Prof P Chakrabarti
Department of CSE

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

RAJASTHAN TECHNICAL UNIVERSITY, KOTA

Syllabus

III Year-V Semester: B.Tech. Computer Science and Engineering

5CS4-22: Compiler Design Lab

Credit: 1 | **Max. Marks:50 (IA:30, ETE:20)**

0L+0T+2P | **End Term Exam: 2 Hours**

SN	List of Experiments
1	Introduction: Objective, scope and outcome of the course.
2	To identify whether given string is keyword or not.
3	Count total no. of keywords in a file. [Taking file from user]
4	Count total no of operators in a file. [Taking file from user]
5	Count total occurrence of each character in a given file. [Taking file from user]
6	Write a C program to insert, delete and display the entries in Symbol Table.
7	Write a LEX program to identify following: 1. Valid mobile number 2. Valid url 3. Valid identifier 4. Valid date (dd/mm/yyyy) 5. Valid time (hh:mm:ss)
8	Write a lex program to count blank spaces,words,lines in a given file.
9	Write a lex program to count the no. of vowels and consonants in a C file.
10	Write a YACC program to recognize strings aaab,abbb using a^nb^n, where b>=0.
11	Write a YACC program to evaluate an arithmetic expression involving operators +,-,* and /.
12	Write a YACC program to check validity of a strings abcd,aabbcd using grammar a^nb^nc^md^m, where n , m>0
13	Write a C program to find first of any grammar.

For Techno India NJR Institute of Technology

पंकज पोरवाल

Dr. Pankaj Kumar Porwal
(Principal)

Office of Dean Academic Affairs
Rajasthan Technical University, Kota

Course Overview: The course has certain outcomes by virtue of which the students will get an idea of the subject Compiler Design.

CO No	**Cognitive Level**	**Course Outcome (LAB)**
1	Comprehension	Ability to get fundamentals of files and strings
2	Application	Ability to count keywords, operators.
3	Application	Ability to apply LEX programs in context to various strings and alphabets, grammar
4	Application	Ability to recognize strings and evaluate arithmetic expression involving operators +,-,* and / using YACC program.
5	Application	Design program to check validity of strings and also to find first of any grammar

Course Outcome Mapping with Program Outcome (LAB):

Course Outcome	**Program Outcome (LAB)**														
CO No.	**Domain Specific**					**Domain Independent**							**PSO**		
	PO1	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**	**PSO 1**	**PSO 2**	**PSO 3**
CO 1	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0
CO 2	2	1	1	2	0	0	0	0	0	0	0	0	0	0	0
CO 3	2	2	1	2	0	0	0	0	0	0	0	0	0	0	0
CO 4	2	0	1	2	0	0	0	0	0	0	0	0	0	0	0
CO 5	2	0	1	2		0	0	0	0	0	0	0	0	0	0
1: Slight (Low), 2: Moderate (Medium), 3: Substantial (high)															

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

For Techno India NJR Institute of Technology

पंकज पोरवाल

Dr. Pankaj Kumar Porwal

(Principal)

COMPILER DESIGN LAB SYLLABUS

Sl. No.	List of Experiments	Page No.
1	**Design a lexical analyzer for given language and the lexical analyzer should ignore redundant spaces, tabs and new lines. It should also ignore comments. Although the syntax specification states that identifiers can be arbitrarily long, you may restrict the length to some reasonable value. Simulate the same in C language.**	1
2	*** Write a C program to identify whether a given line is a comment or not.**	4
3	***Write a C program to recognize strings under 'a', 'a*b+', 'abb'.**	5
4	***Write a C program to test whether a given identifier is valid or not.**	8
5	***Write a C program to simulate lexical analyzer for validating operators.**	9
6	Implement the lexical analyzer using JLex, flex or other lexical analyzer generating tools.	11
7	Write a C program for implementing the functionalities of predictive parser for the mini language specified in Note 1.	13
8	a) *Write a C program for constructing of LL (1) parsing. b) *Write a C program for constructing recursive descent parsing.	17
9	Write a C program to implement LALR parsing.	24
10	a) *Write a C program to implement operator precedence parsing. b) *Write a C program to implement Program semantic rules to calculate the expression that takes an expression with digits, + and * and computes the value.	32
11	Convert the BNF rules into Yacc form and write code to generate abstract syntax tree for the mini language specified in Note 1.	39
12	Write a C program to generate machine code from abstract syntax tree generated by the parser. The instruction set specified in Note 2 may be considered as the target code.	44

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

Note 1:
A simple language written in this language is

```
{int a[3],t1,t2;
T1=2;
A[0]=1;a[1]=2;a[t]=3;
T2=-( a[2]+t1*6)/(a[2]-
t1); If t2>5then
Print(t2)
Else{
Int t3;
T3=99;
T2=25;
Print(-t1+t2*t3);/*this is a comment on 2
lines*/ }endif
}
```

Comments(zero or more characters enclosed between the standard C/JAVA Style comment brackets/*…*/)can be inserted .The language has rudimentary support for1-dimenstional array,the declaration int a[3] declares an array of three elements,referenced as a[0],a[1] and a[2].Note also you should worry about the scopping of names.

Note 2:

Consider the following mini language, a simple procedural high –level language, only operating on integer data, with a syntax looking vaguely like a simple C crossed with pascal. The syntax of the language is defined by the following grammar.

```
<program>::=<block>
<block>::={<variable
definition><slist>} |{<slist>}
<variabledefinition>::=int <vardeflist>
<vardec>::=<identifier>|<identifier>[<constant>]
<slist>::=<statement>|<statement>;<slist>
<statement>::=<assignment>|<ifstament>|<whilestatement>
|<block>|<printstament>|<empty>
<assignment>::=<identifier>=<expression>
|<identifier>[<expression>]=<expression>
<if statement>::=if<bexpression>then<slist>else<slist>endif
|if<bexpression>then<slisi>endif
<whilestatement>::=while<bexpreession>do<slisi>enddo
<printstatement>:;=print(<expression>)
<expression>::=<expression>::=<expression><addingop><term>|<term>|<addingop>
<term>
<bexprssion>::=<expression><relop><expression>
<relop>::=<|<=|==|>=|>|!= <addingop>::=+|-
<term>::=<term><multop><factor>|<factor>
<Multop>::=*|/
<factor>::=<constant>|<identifier>|<identifier>[<expression>]
|(<expression>)
<constant>::=<digit>|<digit><constant>
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

<identifier>::=<identifier><letter or digit>|<letter>
<letter or digit>::=<letter>|<digit>
<letter>:;=a|b|c|d|e|f|g|h|I|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z
<digit>::=0|1|2|3|4|5|^|7|8|9
<empty>::=has the obvious meaning

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

COMPILER DESIGN LABORATORY

OBJECTIVE:

This laboratory course is intended to make the students experiment on the basic techniques of compiler construction and tools that can used to perform syntax-directed translation of a high-level programming language into an executable code. Students will design and implement language processors in C by using tools to automate parts of the implementation process. This will provide deeper insights into the more advanced semantics aspects of programming languages, code generation, machine independent optimizations, dynamic memory allocation, and object orientation.

OUTCOMES:

Upon the completion of Compiler Design practical course, the student will be able to:

1. **Understand** the working of lex and yacc compiler for debugging of programs.
2. **Understand** and define the role of lexical analyzer, use of regular expression and transition diagrams.
3. **Understand** and use Context free grammar, and parse tree construction.
4. **Learn** & use the new tools and technologies used for designing a compiler.
5. **Develop** program for solving parser problems.
6. **Learn** how to write programs that execute faster.

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

EXPERIMENT- 1

1.1 OBJECTIVE:

Design a lexical analyzer for given language and the lexical analyzer should ignore redundant spaces, tabs and new lines. It should also ignore comments. Although the syntax specification states that identifiers can be arbitrarily long, you may restrict the length to some reasonable value. Simulate the same in C language.

1.2 RESOURCE:

Turbo C ++

1.3 PROGRAM LOGIC:

1. Read the input Expression
2. Check whether input is alphabet or digits then store it as identifier
3. If the input is is operator store it as symbol
4. Check the input for keywords

1.4 PROCEDURE:

Go to debug -> run or press CTRL + F9 to run the program

1.5 PROGRAM:

```
#include<string.h>
#include<ctype.h>
#include<stdio.h>
void keyword(char str[10])
{
 if(strcmp("for",str)==0||strcmp("while",str)==0||strcmp("do",str)==0||strcmp("int",str)==0||str
cmp("float",str)==0||strcmp("char",str)==0||strcmp("double",str)==0||strcmp("static",str)==0||strcmp("switch",str
)==0||strcmp("case",str)==0) printf("\n%s
          is a keyword",str);
  else
          printf("\n%s is an identifier",str);
}
main()
{
          FILE *f1,*f2,*f3;
          char c,str[10],st1[10];
          int num[100],lineno=0,tokenvalue=0,i=0,j=0,k=0;
          printf("\nEnter the c program");/*gets(st1);*/
          f1=fopen("input","w"); while((c=getchar())!=EOF)

                  putc(c,f1);
          fclose(f1);
          f1=fopen("input","r");
          f2=fopen("identifier","w");
           f3=fopen("specialchar","w");
           while((c=getc(f1))!=EOF)  {
                  if(isdigit(c))
                  {
                                  tokenvalue=c-'0';
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwa
(Principal)

```
                                        c=getc(f1);
                                        while(isdigit(c)) {
                                          tokenvalue*=10+c-'0';
                                          c=getc(f1);
                                          }
                                    num[i++]=tokenvalue;
                                    ungetc(c,f1);
                        }
                        else
                                  if(isalpha(c))
                                    {
                                        putc(c,f2);
                                        c=getc(f1);
                                        while(isdigit(c)||isalpha(c)||c=='_'||c=='$')
                                          {
                                                  putc(c,f2);
                                                  c=getc(f1);
                                          }
                                        putc(' ',f2);
                                        ungetc(c,f1);
                                    }
                                    else
                                      if(c==' '||c=='\t')
                                        printf(" ");
                                     else
                                        if(c=='\n')
                                                  lineno++;
                                        else
                                                  putc(c,f3);
              }
fclose(f2);
fclose(f3);
fclose(f1);
printf("\nThe no's in the program are");
for(j=0;j<i;j++)
printf("%d",num[j]);
printf("\n");
f2=fopen("identifier","r");
k=0;
printf("The keywords and identifiersare:");
while((c=getc(f2))!=EOF) {
          if(c!=' ')
          str[k++]=c;
          else
                    {
                    str[k]='\0';
                    keyword(str);
                    k=0;              }
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
                }
                        fclose(f2);
        f3=fopen("specialchar","r");
        printf("\nSpecial characters are");
        while((c=getc(f3))!=EOF)
        printf("%c",c);
        printf("\n");
        fclose(f3);
        printf("Total no. of lines are:%d",lineno);
}
```

1.6 PRE LAB QUESTIONS

1. What is token?
2. What is lexeme?
3. What is the difference between token and lexeme?
4. Define phase and pass?
5. What is the difference between phase and pass?
6. What is the difference between compiler and interpreter?

1.7 LAB ASSIGNMENT

1. Write a program to recognize identifiers.
2. Write a program to recognize constants.
3. Write a program to recognize keywords and identifiers.
4. Write a program to ignore the comments in the given input source program.

1.8 POST LAB QUESTIONS

1. What is lexical analyzer?
2. Which compiler is used for lexical analyzer?
3. What is the output of Lexical analyzer?
4. What is LEX source Program?

1.9 INPUT & OUTPUT:

Input:

```
Enter Program $ for termination:
 {
  int a[3],t1,t2;
  t1=2; a[0]=1; a[1]=2; a[t1]=3;
  t2=-(a[2]+t1*6)/(a[2]-t1);
  if t2>5 then
  print(t2);
  else {
     int   t3;
     t3=99;
     t2=-25;
     print(-t1+t2*t3); /* this is a comment on 2 lines */ }
     endif
     }
     $
```

Output:

```
 Variables : a[3] t1 t2 t3
Operator : - + * / >
Constants :  2 1 3 6 5 99 -25
Keywords : int if then else endif
Special Symbols : , ; ( ) { }
Comments :  this is a comment on 2 lines
```

For Techno India NJR Institute of Technology
[illegible]
Dr. Pankaj Kumar Porwal
(Principal)

EXPERIMENT-2

2.1 OBJECTIVE:

* Write a C program to identify whether a given line is a comment or not.

2.2 RESOURCE:

Turbo C++

2.3 PROGRAM LOGIC:

Read the input string.

Check whether the string is starting with ‘/’ and check next character is ‘/’ or’*’.

If condition satisfies print comment.
Else not a comment.

2.4 PROCEDURE:

Go to debug -> run or press CTRL + F9 to run the program.

2.5 PROGRAM:

```
#include<stdio.h>
 #include<conio.h>
 void main()      {
   char com[30];
   int i=2,a=0;
   clrscr();
   printf("\n Enter
   comment:"); gets(com);
   if(com[0]=='/')  {
                    if(com[1]=='/')
                      printf("\n It is a comment");
                    else if(com[1]=='*') {
                                    for(i=2;i<=30;i++)
                                    {
                                          if(com[i]=='*'&&com[i+1]=='/')
                                         {
                                               printf("\n It is a comment");
                                                a=1;
                                                break;   }
                                         else
                                         continue; }
                                    if(a==0)
                                   printf("\n It is not a comment");
                              }
                    else
                     printf("\n It is not a comment");
              }
   else
           printf("\n It is not a comment");
   getch(); }
```

2.6 INPUT & OUTPUT:

Input: Enter comment: //hello
Output: It is a comment
Input: Enter comment: hello
Output: It is not a comment

EXPERIMENT-3

3.1 OBJECTIVE:

Write a C program to recognize strings under 'a', 'a*b+', 'abb'.

3.2 RESOURCE:

Turbo C++

3.3 PROGRAM LOGIC:

By using transition diagram we verify input of the state.

If the state recognize the given pattern rule.

Then print string is accepted under a*/ a*b+/ abb.

Else print string not accepted.

3.4 PROCEDURE:

Go to debug -> run or press CTRL + F9 to run the program.

3.5 PROGRAM:

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<stdlib.h>
void main()
{
        char s[20],c; int
        state=0,i=0;
        clrscr();
         printf("\n Enter a
        string:"); gets(s);
         while(s[i]!='\0')
                {
                        switch(state)
                                {
                                        case 0: c=s[i++];
                                                if(c=='a')
                                                        state=1;
                                                else if(c=='b')
                                                        state=2;
                                                else
                                                  state=6;
                                         break;
                                case 1: c=s[i++];
                                        if(c=='a')
                                                state=3;
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
                else if(c=='b')
                        state=4;
                else
                        state=6;
                break;
        case 2: c=s[i++];
                if(c=='a')
                        state=6;
                else if(c=='b')
                        state=2;
                else
                        state=6;
                break;
        case 3: c=s[i++];
                if(c=='a')
                        state=3;
                else if(c=='b')
                        state=2;
                else
                        state=6;
                break;
        case 4: c=s[i++];
                if(c=='a')
                        state=6;


                else if(c=='b')
                        state=5;
                else
                        state=6;
                break;
        case 5: c=s[i++];
                if(c=='a')
                        state=6;
                else if(c=='b')
                        state=2;
                else
                        state=6;
                        break;
        case 6: printf(""\n %s is not recognised.",s);
                exit(0);
        }
}
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
    I          f(state==1)
                         printf("\n %s is accepted under rule
               'a'",s); else if((state==2)||(state==4))
                         printf("\n %s is accepted under rule
               'a*b+'",s); else if(state==5)
                        printf("\n %s is accepted under rule 'abb'",s);
               getch();
}
```

3.6 INPUT & OUTPUT:

Input :

```
Enter a String: aaaabbbbb
```

Output:

```
aaaabbbbb is accepted under rule 'a*b+'

Enter a string: cdgs

cdgs is not recognized
```

For Techno India NJR Institute of Technology

पंकज पोरवाल

Dr. Pankaj Kumar Porwal
(Principal)

EXPERIMENT-4

4.1 OBJECTIVE:

*Write a C program to test whether a given identifier is valid or not

4.2 RESOURCE:

Turbo C++

4.3 PROGRAM LOGIC:

Read the given input string.

Check the initial character of the string is numerical or any special character except '_' then print it is not a valid identifier.

Otherwise print it as valid identifier if remaining characters of string doesn't contains any special characters except '_'.

4.4 PROCEDURE:

Go to debug -> run or press CTRL + F9 to run the program.

4.5 PROGRAM:

```
#include<stdio.h>
#include<conio.h>
#include<ctype.h>
void main()
{
        char a[10];
         int flag, i=1;
         clrscr();
         printf("\n Enter an identifier:");
         gets(a);
        if(isalpha(a[0]))
                flag=1;
         else
                printf("\n Not a valid
        identifier"); while(a[i]!='\0')
        {
                 if(!isdigit(a[i])&&!isalpha(a[i]))
                {
                        flag=0;
                         break;
                }
                i++;
         }
         if(flag==1)
        printf("\n Valid
        identifier"); getch();
}
```

4.6 INPUT & OUTPUT:

Input: Enter an identifier: first

Output:

```
Valid identifier
Enter an identifier:1aqw
Not a valid identifier
```

For Techno India NJR Institute of Technology

पंकज पोरवाल

Dr. Pankaj Kumar Porwal

(Principal)

EXPERIMENT-5

5.1 OBJECTIVE:

*Write a C program to simulate lexical analyzer for validating operators.

5.2 RESOURCE:

Turbo C++

5.3 PROGRAM LOGIC :

Read the given input.

If the given input matches with any operator symbol.
Then display in terms of words of the particular
symbol. Else print not a operator.

5.4 PROCEDURE:

Go to debug -> run or press CTRL + F9 to run the program.

5.5 PROGRAM:

```
#include<stdio.h>
#include<conio.h>
void main()
{
            char s[5];
            clrscr();
            printf("\n Enter any
            operator:"); gets(s);
            switch(s[0])
           {
            case'>': if(s[1]=='=')
                        printf("\n Greater than or
                        equal"); else
                        printf("\n Greater
                        than"); break;
            case'<': if(s[1]=='=')
                        printf("\n Less than or
                        equal"); else
                        printf("\nLess
                        than"); break;
           case'=': if(s[1]=='=')
                        printf("\nEqual to"); else
                        printf("\nAssignment");
                        break;

           case'!':   if(s[1]=='=')
                        printf("\nNot
                        Equal"); else
                         printf("\n Bit
                        Not"); break;
           case'&': if(s[1]=='&')
                        printf("\nLogical AND");
                        else
                        printf("\n Bitwise
                        AND"); break;
            case'|':   if(s[1]=='|')
                        printf("\nLogical OR");
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
                              else
                              printf("\nBitwise
                               OR"); break;
                    case'+':  printf("\n Addition");
                              break;
                     case'-': printf("\nSubstraction");
                               break;
                    case'*':   printf("\nMultiplication");
                              break;
                    case'/': printf("\nDivision");
                               break;
                    case'%':  printf("Modulus");
                              break;
                    default:  printf("\n Not a operator");
                    }
                    getch();
           }
```

5.6 INPUT & OUTPUT:

Input

Enter any operator: *

Output

Multiplication

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

EXPERIMENT-6

6.1 OBJECTIVE:

Implement the lexical analyzer using JLex, flex or other lexical analyzer generating tools.

6.2 RESOURCE:

Linux using Putty

6.3 PROGRAM LOGIC:

Read the input string.

Check whether the string is identifier/ keyword /symbol by using the rules of identifier and keywords using LEX Tool

6.4 PROCEDURE:

Go to terminal .Open vi editor ,Lex lex.l , cc lex.yy.c , ./a.out

6.5 PROGRAM:

```
/* program name is lexp.l */
%{
/* program to recognize a c program
*/ int COMMENT=0;
%}
identifier [a-zA-Z][a-zA-Z0-9]*
%%
#.* { printf("\n%s is a PREPROCESSOR DIRECTIVE",yytext);}
int |float |char |double |while |for |do |if |break |continue |void |switch |case |long |struct |const |typedef |return
|else |goto {printf("\n\t%s is a KEYWORD",yytext);}
"/*" {COMMENT = 1;}
/*{printf("\n\n\t%s is a COMMENT\n",yytext);}*/
"*/" {COMMENT = 0;}
/* printf("\n\n\t%s is a COMMENT\n",yytext);}*/
{identifier}\( {if(!COMMENT)printf("\n\nFUNCTION\n\t%s",yytext);}
{ {if(!COMMENT) printf("\n BLOCK BEGINS");}
} {if(!COMMENT) printf("\n BLOCK ENDS");}
{identifier}(\[[0-9]*\])? {if(!COMMENT) printf("\n %s IDENTIFIER",yytext);}
".*\" {if(!COMMENT) printf("\n\t%s is a STRING",yytext);}
[0-9]+ {if(!COMMENT) printf("\n\t%s is a NUMBER",yytext);}
 {if(!COMMENT) printf("\n\t");ECHO;printf("\n");}
( ECHO;
{if(!COMMENT)printf("\n\t%s is an ASSIGNMENT OPERATOR",yytext);}
<= |>= |< |== |> {if(!COMMENT) printf("\n\t%s is a RELATIONAL OPERATOR",yytext);}
%%
int main(int argc,char **argv)
{
if (argc > 1)
{
FILE *file;
file = fopen(argv[1],"r");
if(!file)
{
printf("could not open %s
\n",argv[1]); exit(0);
}
yyin = file;
}
yylex();
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
printf("\n\n");
return 0;
} int yywrap()
{
return 0;
}
```

6.6 PRE LAB QUESTIONS:

1. List the different sections available in LEX compiler?
2. What is an auxiliary definition?
3. How can we define the translation rules?
4. What is regular expression?
5. What is finite automaton?

6.7 LAB ASSIGNMENT:

1. Write a program that defines auxiliary definitions and translation rules of Pascal tokens?
2. Write a program that defines auxiliary definitions and translation rules of C tokens?
3. Write a program that defines auxiliary definitions and translation rules of JAVA tokens

6.8 POST LAB QUESTIONS:

1. What is Jlex?
2. What is Flex?
3. What is lexical analyzer generator?
4. What is the input for LEX Compiler?
5. What is the output of LEX compiler?

6.6 INPUT & OUTPUT:

Input

```
$vi var.c
#include<stdio.h>
main()
{
int a,b;
}
```

Output $lex

```
lex.l $cc
lex.yy.c
$./a.out var.c
#include<stdio.h> is a PREPROCESSOR
DIRECTIVE FUNCTION
main (
)
BLOCK BEGINS
int is a KEYWORD
a IDENTIFIER
b IDENTIFIER
BLOCK ENDS
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

EXPERIMENT-7

7.1 OBJECTIVE:

Write a C program for implementing the functionalities of predictive parser for the mini language specified in Note 1.

7.2 RESOURCE:

Turbo C++

7.3 PROGRAM LOGIC:

Read the input string.

By using the FIRST AND FOLLOW values.

Verify the FIRST of non terminal and insert the production in the FIRST value

If we have any @ terms in FIRST then insert the productions in FOLLOW values
Constructing the predictive parser table

7.4 PROCEDURE:

Go to debug -> run or press CTRL + F9 to run the program.

7.5 PROGRAM:

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
char prol[7][10]={"S","A","A","B","B","C","C"};
char pror[7][10]={"A","Bb","Cd","aB","@","Cc","@"};
char prod[7][10]={"S->A","A->Bb","A->Cd","B->aB","B->@","C->Cc","C->@"};
char first[7][10]={"abcd","ab","cd","a@","@","c@","@"};
char    follow[7][10]={"$","$","$","a$","b$","c$","d$"};
char table[5][6][10];
numr(char c)
 {
          switch(c)
          {
                 case 'S': return 0;
                case 'A': return 1;
                case 'B': return 2;
                case 'C': return 3;
                case 'a': return 0;
                case 'b': return 1;
                case 'c': return 2;
                 case 'd': return 3;
                case '$': return 4;
          }
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
                return(2);
        }
        void main()
        {
                int i,j,k;
                clrscr();
                for(i=0;i<5;i++)
                for(j=0;j<6;j++)
                        strcpy(table[i][j]," ");
                printf("\nThe following is the predictive parsing table for the following
                grammar:\n"); for(i=0;i<7;i++)
                        printf("%s\n",prod[i]);
                printf("\nPredictive parsing table is\n");
                fflush(stdin);
                for(i=0;i<7;i++)
                {
                        k=strlen(first[i]);
                        for(j=0;j<10;j++)
                        if(first[i][j]!='@')
                        strcpy(table[numr(prol[i][0])+1][numr(first[i][j])+1],prod[i]);
                }
                for(i=0;i<7;i++)
                {
                        if(strlen(pror[i])==1)
                        {
                        if(pror[i][0]=='@')
                        {
                                k=strlen(follow[i]);
                                for(j=0;j<k;j++)
                                strcpy(table[numr(prol[i][0])+1][numr(follow[i][j])+1],prod[i]);
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
                                }

                      }

            }

           strcpy(table[0][0]," ");

           strcpy(table[0][1],"a");

           strcpy(table[0][2],"b");

           strcpy(table[0][3],"c");

           strcpy(table[0][4],"d");

           strcpy(table[0][5],"$");

           strcpy(table[1][0],"S");

           strcpy(table[2][0],"A");

           strcpy(table[3][0],"B");

           strcpy(table[4][0],"C");

           printf("\n--------------------------------------------------------\n");

           for(i=0;i<5;i++)

            for(j=0;j<6;j++)

                      {

                     printf("%-10s",table[i][j]);

                      if(j==5)

                      printf("\n--------------------------------------------------------\n");

                      }

           getch();

          }
```

7.6 PRE LAB QUESTIONS:

1. What is top-down parsing?
2. What are the disadvantages of brute force method?
3. What is context free grammar?
4. What is parse tree?
5. What is ambiguous grammar?
6. What are the derivation methods to generate a string for the given grammar?
7. What is the output of parse tree?

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

7.7 LAB ASSIGNMENT:

1. Write a program to compute FIRST for the following grammar?

   E □ TE'
   E' □ +TE'/î
   T □ FT'
   T' □ *FT'/î
   F □ (E)/i

2. Write a program to compute FIRST for the following grammar?

   S □ iCtSS'
   S' □ eS/ î

3. Write a program to construct predictive parsing table for the following grammar?

   S □ iCtSS'
   S' □ eS/ î

7.8 POST LAB QUESTIONS

1. What is Predictive parser?
2. How many types of analysis can we do using Parser?
3. What is Recursive Decent Parser?
4. How many types of Parsers are there?
5. What is LR Parser?

7.9 INPUT & OUTPUT:

The following is the predictive parsing table for the following grammar:

```
S->A
A->Bb
A->Cd
B->aB
B->@
C->Cc
C->@
Predictive parsing table is
------------------------------------------------------------------
          A         b            c            d           $

------------------------------------------------------------------

SS->AS->A                 S->A       S->A

------------------------------------------------------------------

A   A->Bb   A->Bb        A->Cd     A->Cd

------------------------------------------------------------------

B   B->aB   B->@         B->@                  B->@

------------------------------------------------------------------

C     C->@      C->@   C->@

------------------------------------------------------------------
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

EXPERIMENT-8(a)

8.1 OBJECTIVE:

*Write a C program for constructing of LL (1) parsing.

8.2 RESOURCE:

Turbo C++

8.3 PROGRAM LOGIC:

Read the input string.

Using predictive parsing table parse the given input using stack .

If stack [i] matches with token input string pop the token else shift it repeat the process until it reaches to $.

8.4 PROCEDURE:

Go to debug -> run or press CTRL + F9 to run the program.

8.5 PROGRAM

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
char s[20],stack[20];
void main()
 {
        char m[5][6][3]={"tb"," "," ","tb"," "," "," ","+tb"," "," ","n","n","fc"," "," ","fc"," "," "," ","n","*fc","
                                        a ","n","n","i"," "," ","(e)"," "," "};
         int size[5][6]={2,0,0,2,0,0,0,3,0,0,1,1,2,0,0,2,0,0,0,1,3,0,1,1,1,0,0,3,0,0}; int
        i,j,k,n,str1,str2;
        clrscr();
        printf("\n Enter the input string:
        "); scanf("%s",s);
         strcat(s,"$");
        n=strlen(s);
        stack[0]='$';
        stack[1]='e';
        i=1;
        j=0;
         printf("\nStack Input\n");
        printf("__________________\n");
        while((stack[i]!='$')&&(s[j]!='$'))
        {
                if(stack[i]==s[j])
                {
                        i--;
                        j++;
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
                }
               switch(stack[i])
                {
                        case 'e': str1=0;
                         break; case 'b':
                          str1=1; break;

                        case 't': str1=2;
                         break; case 'c':
                          str1=3; break;

                        case 'f': str1=4;
                                break;
               }
          switch(s[j])
               {
                        case 'i': str2=0;
                        break; case '+':
                         str2=1; break;

                        case '*': str2=2;
                                break;
                        case '(': str2=3;
                         break; case ')':
                         str2=4; break;

                        case '$': str2=5;
                                break;
                }
               if(m[str1][str2][0]=='\0')
                {
                        printf("\nERROR");
                        exit(0);
                 }
                else
                if(m[str1][str2][0]=='n') i--;
               else if(m[str1][str2][0]=='i')
```

For Techno India NJR Institute of Technology
पँकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
                stack[i]='i';
                else
                {
                for(k=size[str1][str2]-1;k>=0;k--)
                 {
                        stack[i]=m[str1][str2][k];
                        i++;
                }
                 i--;
        }
        for(k=0;k<=i;k++)
        printf(" %c",stack[k]);
        printf(" ");
        for(k=j;k<=n;k++)
        printf("%c",s[k]);
        printf(" \n ");
 }
        printf("\n SUCCESS");
        getch(); }
```

8.6 INPUT & OUTPUT:

```
Enter the input string:i*i+i
Stack                    INPUT
$bt                       i*i+i$
$bcf                      i*i+i$
$bci                      i*i+i$
$bc                        *i+i$
$bcf*                      *i+i$
$bcf                         i+i$
$bci                         i+i$
$bc                           +i$
$b                            +i$
$bt+                          +i$
$bt                             i$
$bcf                            i$
$ bci                           i$
$bc                              $
$b                              $
$                               $
success
```

For Techno India NJR Institute of Technology

पंकज पोरवाल

Dr. Pankaj Kumar Porwal

(Principal)

EXPERIMENT-8(b)

8.1 OBJECTIVE:

Construction of recursive descent parsing for the following grammar E->TE'

E'->+TE/@ "@ represents null character" T->FT'

T`->*FT'/@
F->(E)/ID

8.2 RESOURCE:

Turbo C++

8.3 PROGRAM LOGIC:

Read the input string.

Write procedures for the non terminals

Verify the next token equals to non terminals if it satisfies match the non terminal.
If the input string does not match print error.

8.4 PROCEDURE:

Go to debug -> run or press CTRL + F9 to run the program.

8.5 PROGRAM:

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
char input[100];
int i,l;
void main()
 {
            clrscr();
            printf("\nRecursive descent parsing for the following grammar\n");
           printf("\nE->TE'\nE'->+TE'/@\nT->FT'\nT'->*FT'/@\nF-
           >(E)/ID\n"); printf("\nEnter the string to be checked:");
           gets(input);
           if(E())
           {
                    if(input[i+1]=='\0')
                             printf("\nString is accepted");
                     else
                             printf("\nString is not accepted");
            }
            else
                    printf("\nString not accepted");
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
                getch();
 }
E()
 {
                if(T())
                {
                                if(EP())
                                return(1);
                 else
                                 return(0);
                 }
                else
                 return(0);
                }
                EP()
               {
                                if(input[i]=='+')
                                {
                                                 i++;
                                                if(T())
                                                 {
                                                 if(EP())
                                                return(1);
                                                else
                                                return(0);
                                }
                                 else
                                return(0);
                 }
                                else
                                 return(1);
 }
T()
 {
                 if(F())
                                {
                                                if(TP())
                                                 return(1);
                                                else
                                                return(0);
                                }
                 else
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
                    return(0);
  }
TP()
 {
          if(input[i]=='*')
          {
                    i++;
                    if(F())
                    {
                    if(TP())
                    return(1);
                    else
                    return(0);
           }
          else
          return(0);
  }
  else
   return(1);
 }
F()
 {
          if(input[i]=='(')
          {
                    i++;
                    if(E())
                    {
                    if(input[i]==')')
                     {
                              i++;
                              return(1);
                     }
                    else
                    return(0);
                    }
                    else
                    return(0);
          }
          else if(input[i]>='a'&&input[i]<='z'||input[i]>='A'&&input[i]<='Z')
                    {
                              i++;
                              return(1);
```

```
                        }
                       else
                        return(0);
   }
```

8.6 INPUT & OUTPUT:

```
Recursive descent parsing for the following
grammar E->TE'
E'->+TE'/@
T->FT' T'-
>*FT'/@ F-
>(E)/ID

Enter the string to be
checked:(a+b)*c String is accepted
Recursive descent parsing for the following
grammar E->TE'
E'->+TE'/@
T->FT' T'-
>*FT'/@ F-
>(E)/ID

Enter the string to be checked:a/c+d
String is not accepted
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

EXPERIMENT-9

9.1 OBJECTIVE:

Write a program to Design LALR Bottom up Parser.

9.2 RESOURCE:

TURBO C++

9.3 PROGRAM LOGIC:

Read the input string.

Push the input symbol with its state symbols in to the stack by referring lookaheads

We perform shift and reduce actions to parse the grammar.

Parsing is completed when we reach $ symbol.

9.4 PROCEDURE:

Go to debug -> run or press CTRL + F9 to run the program.

9.5 PROGRAM:

```
/*LALR PARSER
  E->E+T
              E->T
              T->T*F
              T->F
              F->(E)
              F->i
*/

#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<string.h>
void push(char *,int *,char);
char stacktop(char *);
void isproduct(char,char);
int ister(char);
int  isnter(char);
int isstate(char);
void error();
void isreduce(char,char);
char pop(char *,int *);
void printt(char *,int *,char [],int);
void rep(char [],int);

struct action
{
  char row[6][5];
};
```

```
const struct action A[12]={
                        {"sf","emp","emp","se","emp","emp"},
                        {"emp","sg","emp","emp","emp","acc"},
                        {"emp","rc","sh","emp","rc","rc"},
                        {"emp","re","re","emp","re","re"},
                        {"sf","emp","emp","se","emp","emp"},
                        {"emp","rg","rg","emp","rg","rg"},
                        {"sf","emp","emp","se","emp","emp"},
                        {"sf","emp","emp","se","emp","emp"},
                        {"emp","sg","emp","emp","sl","emp"},
                        {"emp","rb","sh","emp","rb","rb"},
                        {"emp","rb","rd","emp","rd","rd"},
                        {"emp","rf","rf","emp","rf","rf"}
                };
struct gotol
{
 char r[3][4];
};
const struct gotol G[12]={
                        {"b","c","d"},
                        {"emp","emp","emp"},
                        {"emp","emp","emp"},
                        {"emp","emp","emp"},
                        {"i","c","d"},
                        {"emp","emp","emp"},
                        {"emp","j","d"},
                        {"emp","emp","k"},
                        {"emp","emp","emp"},
                        {"emp","emp","emp"},
                  };
char ter[6]={'i','+','*',')','(','$'};
char nter[3]={'E','T','F'};
char states[12]={'a','b','c','d','e','f','g','h','m','j','k','l'};
char stack[100];
int top=-1;
char temp[10];
struct grammar
{
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
  char left; char
  right[5];
};
const struct grammar rl[6]={ {'E',"e+T"},
                             {'E',"T"},
                             {'T',"T*F"},
                             {'T',"F"},
                             {'F',"(E)"},
                             {'F',"i"},

                           };
void main()
{
         char inp[80],x,p,dl[80],y,bl='a';
         int i=0,j,k,l,n,m,c,len;
         clrscr();
         printf(" Enter the input :");
         scanf("%s",inp);
         len=strlen(inp);
         inp[len]='$';
         inp[len+1]='\0';
         push(stack,&top,bl);
         printf("\n stack \t\t\t input");
         printt(stack,&top,inp,i);
         do
         {
         x=inp[i];
         p=stacktop(stack);

         isproduct(x,p);
         if(strcmp(temp,"emp")==0)
                 error();
         if(strcmp(temp,"acc")==0)
                 break;
         else
         {
                 if(temp[0]=='s')
                 {
                         push(stack,&top,inp[i]);
                         push(stack,&top,temp[1]);
                         i++;
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
                }
                else
                {
                        if(temp[0]=='r')
                        {
                                j=isstate(temp[1]);
                                strcpy(temp,rl[j-
                                2].right); dl[0]=rl[j-
                                2].left; dl[1]='\0';
                                n=strlen(temp);
                                for(k=0;k<2*n;k++)
                                        pop(stack,&top);
                                for(m=0;dl[m]!='\0';m++)
                                        push(stack,&top,dl[m]);
                                l=top; y=stack[l-
                                1];
                                isreduce(y,dl[0]);
                                for(m=0;temp[m]!='\0';m++)
                                        push(stack,&top,temp[m]);
                        }
                }
        }
        printt(stack,&top,inp,i);
 }while(inp[i]!='\0');
 if(strcmp(temp,"acc")==0)
        printf(" \n accept the input ");
 else
        printf(" \n do not accept the input ");
 getch();
}
void push(char *s,int *sp,char item)
{
 if(*sp==100)
        printf(" stack is full ");
 else
 {
        *sp=*sp+1;
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
                s[*sp]=item;
  }
}
char stacktop(char *s)
{
  char i;
  i=s[top];
  return i;
}
void isproduct(char x,char p)
{
                int k,l;
                k=ister(x);
                l=isstate(p);
                strcpy(temp,A[l-1].row[k-1]);
}
int ister(char x)
{
                int i;
                for(i=0;i<6;i++)
                        if(x==ter[i])
                                return i+1;
                return 0;
}
int isnter(char x)
{
                int i;
                for(i=0;i<3;i++)
                        if(x==nter[i]) return
                                i+1;
                return 0;
}
int isstate(char p)
{
                int i;
                for(i=0;i<12;i++)
                        if(p==states[i])
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
                return i+1;
        return 0;
}
void error()
{
        printf(" error in the input
        "); exit(0);
}
void isreduce(char x,char p)
{
        int k,l;
        k=isstate(x);
        l=isnter(p);
        strcpy(temp,G[k-1].r[l-1]);
}


char pop(char *s,int *sp)
{
        char item;
        if(*sp==-1)
                printf(" stack is empty ");
        else
        {
                item=s[*sp];
                *sp=*sp-1;
        }
        return item;
}
void printt(char *t,int *p,char inp[],int i)
{
        int r;
        printf("\n");
        for(r=0;r<=*p;r++)
                rep(t,r);
        printf("\t\t\t");
        for(r=i;inp[r]!='\0';r++)
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
                printf("%c",inp[r]);
}
void rep(char t[],int r)
{
        char c;
        c=t[r];
        switch(c)
        {
                case 'a': printf("0");
                        break;
                case 'b': printf("1");
                        break;
                case 'c': printf("2");
                        break;
                case 'd': printf("3");
                        break;
                case 'e': printf("4");
                        break;
                case 'f': printf("5");
                        break;
                case 'g': printf("6");
                        break;
                case 'h': printf("7");
                        break;
                case 'm': printf("8");
                        break;
                case 'j': printf("9");
                        break;
                case 'k': printf("10");
                        break;
                case 'l': printf("11");
                        break;
                default   :printf("%c",t[r]);
                        break;
        }
}
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

9.6 PRE-LAB QUESTIONS

1 Why bottom-up parsing is also called as shift reduce parsing?
2 What are the different types of bottom up parsers?
3 What is mean by LR (0) items?
4 Write the general form of LR(1) item?
5 What is YACC?

9.7 LAB ASSIGNMENT

1 Write a program to compute FOLLOW for the following grammar?

E□ TE'
E'□ +TE'/î
T□FT'
T'□ *FT'/î
F□ (E)/i

2 Write a program to construct LALR parsing table for the following grammar.

S□ iCtSS'
S'□ eS/ î

9.8 POST-LAB QUESTIONS:

1. What is LALR parsing?
2. What is Shift reduced parser?
3. What are the operations of Parser?
4. What is the use of parsing table?
5. What is bottom up parsing?

9.9 INPUT & OUTPUT:

Enter the input: i*i+1

Output

```
Stack                          input
0                            i*i+i$
0i5                           *i+i$
0F3                           *i+i$
0T2                           *i+i$
0T2*7                          i+i$
0T2*7i5                         +i$
0T2*7i5F10                      +i$
0T2                             +i$
0E1                             +i$
0E1+6                            i$
0E1+6i5                           $
0E1+6F3                           $
0E1+6T9                           $
0E1                               $
accept the input*/
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

EXPERIMENT-10(a)

10.1 OBJECTIVE:

*Write a C program to implement operator precedence parsing.

10.2 RESOURCE:

Turbo C++

10.3 PROGRAM LOGIC:

Read the arithmetic input string.

Verify the precedence between terminals and symbols

Find the handle enclosed in < . > and reduce it to production symbol.

Repeat the process till we reach the start node.

10.4 PROCEDURE:

Go to debug -> run or press CTRL + F9 to run the program.

10.5 PROGRAM:

```
#include<stdio.h>
char str[50],opstr[75];
int f[2][9]={2,3,4,4,4,0,6,6,0,1,1,3,3,5,5,0,5,0};
int col,col1,col2;
char c;
swt()
{
      switch(c)
      {
                  case'+':col=0;break;
                   case'-':col=1;break;
                  case'*':col=2;break;
                   case'/':col=3;break;
                   case'^':col=4;break;
                  case'(':col=5;break;
                  case')':col=6;break;
                  case'd':col=7;break;
                  case'$':col=8;break;
                   default:printf("\nTERMINAL MISSMATCH\n");
                              exit(1);
```

```
                                break;
    }
   // return 0;
}
main()
{
    int i=0,j=0,col1,cn,k=0;
   int t1=0,foundg=0; char
   temp[20];
    clrscr();
   printf("\nEnter arithmetic expression:");
    scanf("%s",&str);
    while(str[i]!='\0')
                i++;
   str[i]='$';
   str[++i]='\0';
   printf("%s\n",str);
   come:
    i=0;
   opstr[0]='$';
   j=1;
   c='$';
   swt();
   col1=col;
   c=str[i];
   swt();
   col2=col;
    if(f[1][col1]>f[2][col2])
   {
                opstr[j]='>';
                j++;
    }
 else if(f[1][col1]<f[2][col2])
 {
               opstr[j]='<';
               j++;
 }
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
else
{
        opstr[j]='=';j++;
}

while(str[i]!='$')
{
         c=str[i];
        swt();
        col1=col;
       c=str[++i];
        swt();
        col2=col;
        opstr[j]=str[--i];
        j++;
        if(f[0][col1]>f[1][col2])
       {
              opstr[j]='>';
              j++;
         }
         else if(f[0][col1]<f[1][col2])
       {
             opstr[j]='<';
              j++;
         }
       else
       {
       opstr[j]='=';j++;
         }
         i++;
 }
 opstr[j]='$';
 opstr[++j]='\0';
 printf("\nPrecedence
 Input:%s\n",opstr); i=0;
 j=0;
 while(opstr[i]!='\0')
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
{
          foundg=0;
           while(foundg!=1)
           {
                    if(opstr[i]=='\0')goto redone;
                     if(opstr[i]=='>')foundg=1;
                    t1=i;
                     i++;
          }
           if(foundg==1)
          for(i=t1;i>0;i--)
          if(opstr[i]=='<')break;
            if(i==0){printf("\nERROR\n");exit(1);}
          cn=i;
           j=0;
           i=t1+1;
          while(opstr[i]!='\0')
          {
                     temp[j]=opstr[i];
                    j++;i++;
            }
          temp[j]='\0';
          opstr[cn]='E';
           opstr[++cn]='\0';
           strcat(opstr,temp);
           printf("\n%s",opstr);
            i=1;
  }
   redone:k=0;
   while(opstr[k]!='\0')
      {
            k++;
            if(opstr[k]=='<')
            {
                    Printf("\nError");
                    exit(1);
            }
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
}
if((opstr[0]=='$')&&(opstr[2]=='$'))goto
sue; i=1
while(opstr[i]!='\0')
{
      c=opstr[i];
       if(c=='+'||c=='*'||c=='/'||c=='$')
      {
              temp[j]=c;j++;}
              i++;
       }
      temp[j]='\0';
       strcpy(str,temp);
      goto come;
      sue:
      printf("\n
      success"); return 0;
}
```

10.6 INPUT & OUTPUT:

```
Enter the arithmetic expression
(d*d)+d$
```

Output:

```
(d*d)+d$
Precedence input:$<(<d>*<d>)>+<d>$
$<(E*<d>)>+<d>$
$<(E*E)>+<E>$
$E+<E>$ $E+E$

Precedence
input:$<+>$ $E$
success
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

EXPERIMENT-10(b)

10.1 OBJECTIVE:

Program to implement semantic rules to calculate the expression that takes an expression with digits, + and * and computes the value.

10.2 RESOURCE:

Linux using putty

10.3 PROCEDURE:

Reading an input file
Calculate the sum or multiplication of given expression.
Using expression rule print the result of the given values.

10.4 PROGRAM:

```
<parser.l>


%{
#include<stdio.h>
#include "y.tab.h"
%}
%%
[0-9]+ {yylval.dval=atof(yytext);
return DIGIT;
}
\n|. return yytext[0];
%%
<parser.y>
%{
/*This YACC specification file generates the LALR parser for the
program considered in experiment 4.*/
#include<stdio.h>
%}
%union
{
double dval;
}
%token <dval> DIGIT
%type <dval> expr
%type <dval> term
%type <dval> factor
%%
line: expr '\n' {
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
printf("%g\n",$1);
}
;
expr: expr '+' term {$$=$1 + $3
;} | term
;
term: term '*' factor {$$=$1 * $3 ;}
| factor
;
factor: '(' expr ')' {$$=$2
;} | DIGIT
;
%%
int main()
{
yyparse();
}
yyerror(char *s)
{
printf("%s",s);
}
```

10.6 INPUT & OUTPUT:

```
$lex parser.l

$yacc –d parser.y

$cc lex.yy.c y.tab.c –ll –lm

$./a.out

2+3

5.0000
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

EXPERIMENT-11

11.1 OBJECTIVE:

Convert The BNF rules into Yacc form and write code to generate abstract syntax tree.

11.2 RESOURCE :

linux using putty

11.3 PROGRAM LOGIC:

Reading an input file line by line.

Convert it in to abstract syntax tree using three address code.

Represent three address code in the form of quadruple tabular form.

11.4 PROCEDURE:

Go to terminal .Open vi editor ,Lex lex.l , cc lex.yy.c , ./a.out

11.5 PROGRAM

<int.l>

```
%{
#include"y.tab.h"
#include<stdio.h>
#include<string.h>
int LineNo=1;
%}
identifier [a-zA-Z][_a-zA-Z0-9]*
number [0-9]+|([0-9]*\.[0-9]+)
%%
main\(\) return MAIN;
if return IF;
else return ELSE;
while return
WHILE; int |
char |
float return TYPE;
{identifier}    {strcpy(yylval.var,yytext);
return VAR;}
{number}    {strcpy(yylval.var,yytext);
return NUM;}
< |> |>= |<= |== {strcpy(yylval.var,yytext);
return RELOP;}
[ \t] ;
\n LineNo++;
. return yytext[0];
%%
```

<int.y>

```
%{
#include<string.h>
#include<stdio.h>
struct quad{
        char op[5]; char
        arg1[10]; char
        arg2[10]; char
        result[10];
}QUAD[30];
struct stack{
        int  items[100];
        int top;
}stk;
int Index=0,tIndex=0,StNo,Ind,tInd;
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
extern int LineNo;
%}
%union{
        char var[10];
}
%token <var> NUM VAR RELOP
%token MAIN IF ELSE WHILE TYPE
%type <var> EXPR ASSIGNMENT CONDITION IFST ELSEST
WHILELOOP %left '-' '+'
%left '*' '/'
%%
PROGRAM : MAIN BLOCK
;
BLOCK: '{' CODE '}'
;
CODE: BLOCK
| STATEMENT CODE
| STATEMENT
;
STATEMENT: DESCT
';' | ASSIGNMENT ';'
| CONDST
| WHILEST
;
DESCT: TYPE VARLIST
;
VARLIST: VAR ',' VARLIST
| VAR
;
ASSIGNMENT: VAR '=' EXPR{
strcpy(QUAD[Index].op,"=");
strcpy(QUAD[Index].arg1,$3);
strcpy(QUAD[Index].arg2,"");
strcpy(QUAD[Index].result,$1);
strcpy($$,QUAD[Index++].result);
}
;
EXPR: EXPR '+' EXPR {AddQuadruple("+",$1,$3,$$);}
| EXPR '-' EXPR {AddQuadruple("-",$1,$3,$$);}
| EXPR '*' EXPR {AddQuadruple("*",$1,$3,$$);}
| EXPR '/' EXPR {AddQuadruple("/",$1,$3,$$);}
| '-' EXPR {AddQuadruple("UMIN",$2,"",$$);}
| '(' EXPR ')' {strcpy($$,$2);}
| VAR
| NUM
;
CONDST: IFST{
Ind=pop();
sprintf(QUAD[Ind].result,"%d",Index);
Ind=pop();
sprintf(QUAD[Ind].result,"%d",Index);
}
| IFST ELSEST
;
IFST: IF '(' CONDITION ')' {
strcpy(QUAD[Index].op,"==");
strcpy(QUAD[Index].arg1,$3);
strcpy(QUAD[Index].arg2,"FALSE");
strcpy(QUAD[Index].result,"-1");
push(Index);
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
Index++;
}
BLOCK {
strcpy(QUAD[Index].op,"GOTO");
strcpy(QUAD[Index].arg1,"");
strcpy(QUAD[Index].arg2,"");
strcpy(QUAD[Index].result,"-1");
push(Index);
Index++;
};
ELSEST: ELSE{
tInd=pop();
Ind=pop();
push(tInd);
sprintf(QUAD[Ind].result,"%d",Index);
}
BLOCK{
Ind=pop();
sprintf(QUAD[Ind].result,"%d",Index);
};
CONDITION: VAR RELOP VAR
{AddQuadruple($2,$1,$3,$$); StNo=Index-1;
}
| VAR
| NUM
;
WHILEST: WHILELOOP{ Ind=pop();
sprintf(QUAD[Ind].result,"%d",StNo);
Ind=pop();
sprintf(QUAD[Ind].result,"%d",Index);

}
;
WHILELOOP: WHILE '(' CONDITION ')'
{ strcpy(QUAD[Index].op,"==");
strcpy(QUAD[Index].arg1,$3);
strcpy(QUAD[Index].arg2,"FALSE");
strcpy(QUAD[Index].result,"-1");
push(Index);
Index++;
}
BLOCK {
strcpy(QUAD[Index].op,"GOTO");
strcpy(QUAD[Index].arg1,"");
strcpy(QUAD[Index].arg2,"");
strcpy(QUAD[Index].result,"-1");
push(Index);
Index++;
}
;
%%
extern FILE *yyin;
int main(int argc,char *argv[]) {
FILE *fp;
int i;
if(argc>1){
fp=fopen(argv[1],"r");
if(!fp) {
printf("\n File not found");
exit(0);
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
}
yyin=fp;
}
yyparse();
printf("\n\n\t\t ----------------------------""\n\t\t Pos Operator Arg1 Arg2 Result" "\n\t\t
--------------------");
for(i=0;i<Index;i++)
{
printf("\n\t\t %d\t %s\t %s\t %s\t
%s",i,QUAD[i].op,QUAD[i].arg1,QUAD[i].arg2,QUAD[i].result);
}
printf("\n\t\t -----------------------");
printf("\n\n");
return 0;
}
void push(int
data){ stk.top++;
if(stk.top==100)
{
printf("\n Stack overflow\n");
exit(0);
}
stk.items[stk.top]=data;
}
int pop()
{
int data;
if(stk.top==-1){
printf("\n Stack underflow\n");
exit(0);}
data=stk.items[stk.top--];
return data;
}
void AddQuadruple(char op[5],char arg1[10],char arg2[10],char result[10])
{
strcpy(QUAD[Index].op,op);
strcpy(QUAD[Index].arg1,arg1);
strcpy(QUAD[Index].arg2,arg2);
sprintf(QUAD[Index].result,"t%d",tIndex++);
strcpy(result,QUAD[Index++].result);
}
yyerror()
{
printf("\n Error on line no:%d",LineNo);
}
```

Input:

```
$vi test.c
main()
{
int a,b,c;
if(a<b)
{
a=a+b;
}
while(a<b){
a=a+b;
}
if(a<=b){
c=a-b;
}
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
else
{
c=a+b;
}
}
```

11.7 PRE-LAB QUESTIONS

1 What are the functions we use to construct a syntax tree?
2 What is Meta data?
3 How list of identifiers are represented using BNF rules?
4 What is three address code?
5 What are the record structures we use to represent three address code?

11.8 LAB ASSIGNMENT

1 Write YACC for the desktop calculator?

2 Write BNF rules for the following grammar?

E → E+T/T

T → T*F/F

F → (E)/id

11.9 POST-LAB QUESTIONS:

1. What is Abstract Syntax tree?
2. What are BNF Rules?
3. What is DAG representation?
4. How LALR (1) states are generates?
5. In which condition the user has to supply more information to YACC?

11.10 INPUT & OUTPUT:

```
$lex int.l
$yacc –d int.y
$gcc lex.yy.c y.tab.c –ll –lm$./a.out test.c
```

OUTPUT

Pos	Operator	Arg1	Arg2	Result
0	<	a	b	t0
1	**==**	t0	**FALSE**	**5**
2	**+**	**a**	**b**	**t1**
3	**==**	**t1**		**5**
4	**GOTO**			
5	**<**	**a**	**b**	**t2**
6	**==**	**t2**	**FALSE**	**10**
7	**+**	**a**	**b**	**t3**
8	**=**	**t3**		**a**
9	**GOTO**			**5**
10	**<=**	**a**	**b**	**t4**
11	**==**	**t4**	**FALSE**	**15**
12	**-**	**a**	**b**	**t5**
13	**=**	**t5**		**c**
14	**GOTO**			**17**
15	**+**	**a**	**b**	**t6**
16	**=**	**t6**		**c**

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

EXPERIMENT-12

12.1 OBJECTIVE:

Write a C program to generate machine code from abstract syntax tree generated by the parser. The instruction set specified in Note 2 may be considered as the target code.

12.2 RESOURSE:

TURBO C++

12.3 PROGRAM LOGIC:

Read input string

Consider each input string and convert in to machine code instructions

12.4 PROCEDURE:

Go to terminal .Open vi editor ,Lex lex.l , cc lex.yy.c , ./a.out

12.5 PROGRAM:

```
#include<stdio.h>

#include<stdlib.h>

#include<string.h>

int label[20];

int  no=0;

int main()

{

        FILE *fp1,*fp2;

        char fname[10],op[10],ch;

        char operand1[8],operand2[8],result[8];

        int i=0,j=0;

        printf("\n Enter filename of the intermediate code");

        scanf("%s",&fname);
        fp1=fopen(fname,"r");
        fp2=fopen("target.txt","w");
        if(fp1==NULL || fp2==NULL)

        {

                printf("\n Error opening the file");

                exit(0);

        }

        while(!feof(fp1))

        {
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
fprintf(fp2,"\n"); fscanf(fp1,"%s",op);
i++; if(check_label(i))

fprintf(fp2,"\nlabel#%d",i);

if(strcmp(op,"print")==0)

{

        fscanf(fp1,"%s",result);

        fprintf(fp2,"\n\t OUT %s",result);
}
        if(strcmp(op,"goto")==0)
        {
                fscanf(fp1,"%s %s",operand1,operand2); fprintf(fp2,"\n\t
                JMP %s,label#%s",operand1,operand2);
                label[no++]=atoi(operand2);
        }
        if(strcmp(op,"[]=")==0)
        {
                fscanf(fp1,"%s %s %s",operand1,operand2,result);
                fprintf(fp2,"\n\t STORE %s[%s],%s",operand1,operand2,result);
        }
       if(strcmp(op,"uminus")==0)
       {
       fscanf(fp1,"%s %s",operand1,result);
       fprintf(fp2,"\n\t LOAD -%s,R1",operand1);
       fprintf(fp2,"\n\t STORE R1,%s",result);
       }
      switch(op[0])
      {
      case '*':    fscanf(fp1,"%s %s %s",operand1,operand2,result);
                fprintf(fp2,"\n \t
                LOAD",operand1);
                fprintf(fp2,"\n \t LOAD
                %s,R1",operand2);
                fprintf(fp2,"\n \t MUL R1,R0");
                fprintf(fp2,"\n \t STORE
                R0,%s",result); break;

                case '+': fscanf(fp1,"%s %s
                %s",operand1,operand2,result);
                fprintf(fp2,"\n \t LOAD %s,R0",operand1);
                fprintf(fp2,"\n \t LOAD %s,R1",operand2);
                fprintf(fp2,"\n \t ADD R1,R0");

                fprintf(fp2,"\n \t STORE
               R0,%s",result); break;

     case '-': fscanf(fp1,"%s %s
             %s",operand1,operand2,result); fprintf(fp2,"\n
             \t LOAD %s,R0",operand1); fprintf(fp2,"\n \t
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
                                        LOAD %s,R1",operand2); fprintf(fp2,"\n \t
                                        SUB R1,R0");
                                        fprintf(fp2,"\n \t STORE R0,%s",result);
                                        break;

                                   case '/':    fscanf(fp1,"%s %s s",operand1,operand2,result);
                                                 fprintf(fp2,"\n \t LOAD %s,R0",operand1);
                                                 fprintf(fp2,"\n \t LOAD %s,R1",operand2);
                                                 fprintf(fp2,"\n \t DIV R1,R0");
                                                  fprintf(fp2,"\n \t STORE R0,%s",result);
                                                   break;
                                   case '%': fscanf(fp1,"%s %s %s",operand1,operand2,result);
                                                  fprintf(fp2,"\n \t LOAD %s,R0",operand1);
                                                  fprintf(fp2,"\n \t LOAD %s,R1",operand2);
                                                  fprintf(fp2,"\n \t DIV R1,R0");
                                                  fprintf(fp2,"\n \t STORE R0,%s",result);
                                                  break;
                                   case '=': fscanf(fp1,"%s %s",operand1,result); fprintf(fp2,"\n\t
                                                  STORE %s %s",operand1,result); break;

                                   case '>':  j++;
                                                 fscanf(fp1,"%s %s %s",operand1,operand2,result);
                                                 fprintf(fp2,"\n \t LOAD %s,R0",operand1);
                                                 fprintf(fp2,"\n\t JGT %s,label#%s",operand2,result);
                                                 label[no++]=atoi(result);
                                                 break;
                                    case '<': fscanf(fp1,"%s %s %s",operand1,operand2,result);
                                              fprintf(fp2,"\n \t LOAD %s,R0",operand1);
                                              fprintf(fp2,"\n\t JLT %s,label#%d",operand2,result);
                                              label[no++]=atoi(result);
                                               break;
                                     }
                              }
                  fclose(fp2); fclose(fp1);
                  fp2=fopen("target.txt","r");
                   if(fp2==NULL)
                         {
                              printf("Error opening the file\n");
                              exit(0);
                        }
                  do
                     {
                        ch=fgetc(fp2);
                        printf("%c",ch);
                     }while(ch!=EOF);
                     fclose(fp1);
                     return 0;
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

```
}
int check_label(int k)
{
int i;
for(i=0;i<no;i++)
  {
            if(k==label[i])
            return 1;
  }
            return 0;
  }
```

12.6 PRE-LAB QUESTIONS

1 What are the different forms of object code?
2 What is mean by relocatable object code?
3 What is the cost of register to register operation?
4 What is address descriptor?
5 What is register descriptor?

12.7 LAB ASSIGNMENT

1 Write a program to generate the code for the following three address code statements?
   A=B+C
   W=X-Y
2 Write a program to generate the code for the following three address code statements?
   W=(A+B)*C

12.8 POST-LAB QUESTIONS

1. What is target code?
2. What is machine code?
3. What is Cross compiler?
4. Give the example for cross compiler?
5. What is the difference between syntax & Semantics?

12.9 INPUT & OUTPUT:

```
$vi int.txt
=t1 2
[]=a 0 1
[]=a 1 2
[]=a 2 3
*t1 6 t2
    +a[2] t2 t3
-a[2] t1 t2
    /t3 t2 t2
    uminus t2
    t2 print t2
goto t2 t3
    =t3 99
    uminus 25
    t2 *t2 t3 t3
    uminus t1 t1
    +t1 t3 t4
    print t4
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)

Output:

```
Enter filename of the intermediate code:
int.txt STORE t1,2
STORE a[0],1
STORE a[1],2
STORE a[2],3

LOAD t1,R0
LOAD 6,R1
ADD R1,R0
STORE R0,t3

LOAD a[2],R0
LOAD t2,R1
ADD R1,R0
STORE R0,t3

LOAD a[t2],R0
LOAD t1,R1
SUB R1,R0
STORE R0,t2
LOAD t3,R0

LOAD t2,R1
DIV R1,R0
STORE R0,t2

LOAD t2,R1
STORE R1,t2
LOAD t2,R0
JGT 5,label#11
Label#11: OUT t2
 JMP t2,label#13
Label#13: STORE t3,99
LOAD 25,R1
STORE R1,t2

LOAD t2,R0
LOAD t3,R1
MUL R1,R0
STORE R0,t3

LOAD t1,R1
STORE R1,t1

LOAD t1,R0
LOAD t3,R1
ADD R1,R0
STORE R0,t4
OUT t4
```

For Techno India NJR Institute of Technology
पंकज पोरवाल
Dr. Pankaj Kumar Porwal
(Principal)