#### RA20110260100061

INDEX

Name: Addarsh Joshi Class: Q1 Subject: Compiler Design

S.No.	Date	Topic	Page No.	Signature
1	13/1/23	Implementation of	141:5+	1 6 (3.5)
2	30/1/23	Implementation of Regular Expression to NFA	5/5	* XX
3	1/2/23	Conversion of NAA to DFA	8/10	Jun 12
ч	8 2 123	Left Recursion and	4/5	6n
5	15/2/3:3	Pirst and Follow	6+3:	8 8 16 2
6	श्वीराग्ड	Bredictive Parsing Table	+3	20/2
7	9/3/23	Shiff Reduce Parsing	F+4	Pal3
8	9/3/22	Computation of Lead	5+3	potuo.
5	16/2717	Computation of LR[0]	8	They
lo	21/3/23	Intermediate Code	1+4	d_
11	3/4/23	The mediate Code  Cremeration - Busidingple,	C+5	27/3
		Trypic and Inclined		3/4
12	10/4/22	Implementation of simple Code	54	A STA
12	101.11.5	crementer		101
[3	18/4/2	Implementation of DACE	دل د	WAT OF

# 13/1/83 EXPERIMENT - 1

To write a program to implement a lexical analyser in C+1/c. flythum

#### ALGORIUM:

- 1. Start
- 2. Cret the input program from the fle
- if each word in a line is a keyword, identifier, moth operator, logical operator, number of almerical value or other symbol.
- 4. For each levene read, generate a token as follows:
  - a. If the lexence read is a keyword, then the token is a keyword itself
    - token generated is printed on the consider of identifier.
    - c. In the same way the moth operator, logical operator, numerical value and other symbol one printed on the console
- 6. The stream of token generated are displayed in the console output
- 6. Stop

```
CODE
# include < iostream>
# in clude < estring =
#Include & stalleb h>
#include < ctype. h>
 # include < fstream >
     namespace std;
string ar [] = { "void", "using", "namespace", "int",
       "include", "iostream", "std", "main", "cin",
                 " cout ", " return", "float", " double"
                   " storng", "print(");
 bool iskeyword (string a) t
      for ( inti=0, 1214; 14+)?
              if (ar Li) == a) 2
                       return true;
      return talse;
                                  Street and in 18
Int main() i
      fstream file;
       strings, filename,
     filename = " . pedic");
      file open (filename. c-str());
    while (file>>s) {
        if (s== 1+1 | g == "-" | s== " | | -" | s== "/" | s== "/"
           11 8== "=" || 3 == "== " || 3 == " | 1"
           H 3 = = " /" - 11 s = = " +++" +1 s = = " -- " 11 s = = " +="
           [ ] 3=="-="1 | s=="/=" 1 | s=="="|s==""/.=")
            3
              cout es « " is an operator "";
                8 = " ";
            elseif (is key word (s))
               { cour < < s << 11 is a keyword h";
```

```
ele if (s == "(" 11 s == " 2" 11 s == "[" 11 s == "]" 11 c == "3"
             11 s == " ] " 11 s = = " z " 11 s == " >" " 11 s == " ()"
            11 5== ";" 11 5 == " <<" 11 5 == " >" 11 5 == " >" " 3 => " )"
            11 5 = 2 "#") 2
                 cout << see " is a symbol in";
                     $ = 0 0
      else if ( === " \n" | | === " | ) === ")
       else if ( is digit ( = [0])) {
            int x = 0;
              if (! is digit (s[x++1)) &
                     continue)
              else ?
                 cout es se "is a constant "";
                   S = "";
             4
           else ;
               cour ces « " is an identifier In";
                    8 > "";
                 3
    return o,
RESULT
       The following implementation of lexical analyzer
        C++ tous compiled , executed and verified
     successfully
```

OUTPUT s mountife. #include is an identifier mgiringh i p < stdo. h > 1's an identifier sar July 18 is an identifier my rypk. main is a keyword - INTETH (is a punchuahan " toron " } - [ ) is a punctuation int is a keyword x is an operator Fire Energy ( DADE + H 6 is a constant int is a keyword return is a keyword 0 is constant as lot mide I is puntuation Cold Assessed

The block of the second

Mily & Bridge of Income of

" PHARMALL F BANK

Total Bill Control Branch

#### AIM:

To write a program Regular Expression to NFA.

#### ALCORITHM!

- 1. Start
- 2. Cret the input from the user
- 3. Inhalize separate variables and function for postfix, Da's play and NKA
- 5. By using Switch case Installize different cases for the input
- 6- For '. ' operator initialize different case for the
- 6. For '. 'Operator inchalize a separate method by way various stack functions do the same for the other operators like '\*' and '+'.
- 7. Regular expression is in the form like a to last
- 8. Display the output
- 9. Stop.

Procream:

transition - table = [[0]\*3 for -in range (20)]

re = input ("Enter the regular expression:")

re #="

i=0

j=1

while (ic Ion (re))

if re[i] == 'a':

try

if re[ii] == 'a':

```
tohile (relentre)):
     if re[.] = 10;
           if re[++1] = "1" and re[++1] = "+"
              transitum_table [][0] = 1+1
               1+=1
              -brancotion -table [] [0] -
             elif re[1+1] == 11 and re[i+2] == 16:
                transition = table [][2] = ((]+1)*10)+[]+3)
                Je = 1
                 tromothor table (g)(0] = 1+1
                 tremotion table [][2]=1+3
                1+=1
                 transition table [][1]=j+1
                transition table [7][1]=JAI
                 franshim table [][2] = ((1)*10) + (1+1)
          except:
              transitions table [][0]=1+1
       elif relij -- 'b'
         try!

if re[i+1] |= 111 and re[i+1] |= 1*1
                transition = table (g) [1] = q11
           elif re[i+1]=='1' and re[i+2]== 'a',
                 fransition_table [] ][2] = ((]+1)*10)+(]+3)
                 1+=1
                dransition_ table [1] [1] = 1+1
                74=1
               transition. the [p[e] =1+3
               1+=1
```

```
transityon_table [y] [1] =1+1
     transition_table [j] [1] = 1-3
      11=1
       transfirm-table 4][2]=1+1
        1-1-12
   except:
         transition-table & JEIJ = J+1
 elif re[i]=> 'e' and re[i+1] ]= '| ' and re[i+1]!= '+ ':
        transition -torbic (y)[2] = 1+1
         J -= 1
    edif re[1]==1)' and re[11]==21x1:
          transition = table (0][2]= ((j1) *10)+1
          transtan -table [] [2] = ((g+1) * 70) +1
          1-1=1
        print (" Transition function! ")
       for invange (g):
         if I transition take [i][0] != 0):
               print ("g[10) -> 113".
                                  format (i, transition_table [1] [0])
          if (transition_table [1] [2] 1=0):
                 if fransitm = table [i][2][=0)
                       pnnt("q (603, e] --> {15";
                          format (i, transition - toole[i][2D)
                 pant ("q[103,e]-> 113.8.615
                          format li rinkl tran 11 hm -table [1][2] (0),
                      transhu-total (17 [2] 1/-10))
RESULT
      The program to convert regular
                                      expression to NFA
  was implemented successfully.
```

INPUT: " Illie I ge box 1" - 1 fit id as (a 1,6) \* abb 11 = 10) 1 = 100 OUTPUT ( Salar Salar ) - [ -1] at 1 7 CoJetes --> 7 8 7 q [2] [a] -> 3 - [ ii) [ ] - 1 + ... 9 [3/e] ->6 9 [4,6] ->5 [3] ->5 [3] ->6 9 [6 e] -- 76 - [1][[] side to we to note 9[1,a] -79 9[1,a] -79 9[8,b] -79 9[9,b] -79 to 1 - [o] [o] - [o] - [o] - [o] - [o]

state = input (" state riame.

nfa [state] = 13

for 1 in range(t):

path = input (" path: ")

EXPERIMENTA

INPUT

the man-in is specific No of State 13

No of itrousition: 2

state name: A

path : 0 1931) 341- 11/2 + + 119/11 2015 Exter end state from A travelling tunigh patho

RIMITARY A 10"

12 1 T

The small to 55 been to read

" he that the variety some Path: 9

Enter and state from state A travelling through.

MAR photon and a ton state name; 18 , 1116 11 11

path 0

Enter and state from state B travelling through partie production of the state of state and it

Milly DIND 5 MADILA path 1

LAMP ARIA WITH MILLS Enter end state from state 13 travelling through path 1: Time or other deals

C

state name : c

pour 0 Enter end state from state C travelling through (Carrieries to all I proper to path 0

rath 1

End state from a stravelling through path?

In other I have a other

mapping and cope

```
print ("Enter end state from 13 travelling
                 through poth Ey " . format (state, path)
      reaching state [in tern in imput () spolitu]
      nfa [state][path] = reaching - state
print ("In NIFA :- In")
 print ("In Printing NFA tolle:-")
print (nfa)
 nfa-table = pd. DataFrame (nfa)
 print ( mfa-table . transpose ()
. Print ("Enter Amal State of NFA: ")
refer-final-state = Ix for x in imputu splitus]
new-state-list = []
olfa = 4. 11/20/ 0 - 4A' Call. 11 A' 0] A
Key - list = list (
        list (nfa. keys (1) 10])
  path-list = list (nfa [ kays- list [0]]. key ()
offa [key-list Lo]] = 23
for y in range (+) "
    var = " Jan (nfa [key- list [ol] [ path-list [4]])
    dfa [ key - lio [0]] [path 11st [4]], [ var
                                              380
 if var not in Keys - list 1 113 (var) is whole land
       keys list . append (var)
while In Lnew-State-list) 1 =0:
      dfa [ new states - list 60] = 19
     for I'm range (lan (now-state-list [0])):
           for it invange (Icn ( peth-1ist)):
temp=17
```

```
NEAS: " Evel and of made what her wheel " Hory
Y 'A' : 20': [A'] , 'I' : [A' , 'B'] 9, ....
  18' : 2'0' : ['2'],'1' ['2'] 5, 'C' : 10: [],'1' ([13]
                  prendance ( 1445-176-detel 125-detel
Printing NFA toble
                             ( or the state of ) token
                                         (april) burg
              [A,B]
       [A]
               La Ton- which with and and all all a times
              nfor totals : put technicome (total 1)
       state of NKA
   En ter final
                Propertience from State of Mr A. ")
       white they were a for a law interesting
OUTPUT
                                    - test white warn
 DFA:
(A: 2'0': 'A'; 11: 'AD'3, 'AB' (10': 'AC', 11', 'ABC')
 'AC': 10', A, 11- 'AB' 4, + ABC' 3 2 D' : AC' , 11 ABC' 53
         poster this which keeps has left required
Printing DRA toble !-
                               (3 = [13 | 161] = [2]
          A
                                  itt spines on b soft
              ABC [[0] 20 - hay ] what well the
  AB AC
        A cont Abent all strong [ [ ] and _ part] . the
 ABC
       state of the DRA are: ['AC', 'ABC']
 Final
                      I berge sell a aloca and
                      Ned Tills - til bernet from J.
                   ort filt - de A - wind wit
                Classe Land of the Y was sale with
           Maria - 1 - 10 - 10 - 10
```

```
for 1 in range (len (now- states_ Hist[0]));
            tempt = nfa [ new - state - (ist [0] [ ] ] [ [auth_list [ i]]
     S=" "
     s = s.jan (Hamp)
     if s not in keys-list:
          new_states_list.appends(s)
        Key - list- append (s)
    offa [new-states-list [0]] [path_list[i]]=s
new: states list: remove (new -states - list [0])
print ("InDEA:- In")
print (difa)
print (" printing DRA table :- ")
dfa-table = pd-Ruta Frame (dfa)
print (dfa - totale transpose (1)
  dfa_state_list = list (clfa-keysu)
  dfa-final - States = []
   for n in dfa -state-list:
       der i in no
        if ir in nota-timal-state:
          dfa - final-states append (20)
  print ("In Final slate of the DFA are ; ", dipor final-skiles)
PESULT
                         was converted to a DFA wang
    The given NEA
  python suggestully.
```

AIM:

To implement a program for elemination of left factoring and left recursion

ALGORITM FOR LEFT RECURSION:

- 1. Start the program
- 2. Insualize the array for taking input from the user
- Prompt the user to injust no of non-terminal naving left recursion and no of production for these non-terminals
- 4. Prompt the user to input the production. ["[" , "[+]"] . '] for non-terminals
- 5. Elinunate left recursion using the following [47" (3)"] . "[ rules.

A -> Aal | Aa21 - --- | Aam

A x p1/1 321 -- 1 pm

Then replace 1+by

A -> Bi A'2 = 1,2,3 [ ... da' 11] -

A'-ay A'y = 1,2,3 - - [7.11]

- 6 After eliminating the left recursion by applying these rules i-display the problem production without left, recursion
- 7 Stop.

ALGORITHM FOR LEFT PACTORING L/Start 2. Ask the user to enter the set of production 3 - Check for common symbol in the given set of production by comparing with: A - aB1 | aB2 4. If found replace the particular production with: A -> aA' A1 -> B1 | B2 | E 5. Display the output Exit. CORE FOR LEFT RECURSION: gram = 2 "E": [ "E+T", "T"], "T": ["T\*F", "F"], "F" : [" (E)', \$" i"] def remove byrectal Rigram A, A): tramp & grama [A] -tenysor = :[] Fempiner = [] for i in temp: if 1 [0] == A: tempIncr. append ( i[i]+[A+" "]) temper.append (it [A ":"]) tempIncr :append ([te"]) gram a Ca +" '"] = tempInc return gram A

About the same of the a INPUT hong of boggs of war sale of production "E": ["E+T", "T"] stage of the man of Committed in "TAK", I'A " I'A " TONG THE HEAR STORMED IN ^pr , ["(E)", "i"] MEAL ... I SEA ! LEST .... A A property par OU TPUT has a mereld of many

111

prop. 19

eg The City

souther hand for the south of party representations

T→ [["r", "I,"]] ..... F> [['(', 'E'; ')']] + E' -> [["+", T]; 'E'"], ['e']] 7" → [[" \*", " \*"], ['e']]

```
def checkfor moveer (gromm; " ail;
      if ai not m gram A:
           return false
     if a ==ai:
           return True
      if i [o] in grama:
           return Check For Inclined (gramma, a, i[o])
   . return False
def rep (gram A, A)
     temp = grama[A]
     new Temp=[]
      for i in temp:
if check For Inchreer (gram A, A, 1:601):
          for KingramA[i[0]:
             2+=C[1:]
                newtemp, append (4)
              new Temp. expend (i)
      grama[A] = naoTemp
     return grama
def . rem (gram):
      conv = 2
      grama = 14
      revenu - Ey
     for 1 m gran:

convigi = " A" + str CU
            grant = ["A" + sr(c)] = ()
```

```
for i in grum!
    for I in gram [i]!
        for Km 1:
      if k in com:
               tromp. append (conv [x])
           esce
               temp. append (K)
          grand [conv[i]]. append (temp)
 for i'm range (1, c)
      ai = "A" + str (i)
      for 1 in grama [ai]
        if ai = 1 los;
               gram A = remove Pireet (gram A, ai)
op= 15.
for i'm granuA:
   por g in conv: 3
        a = a replace (con (y) )
   perconv[i] = a
for i ingrama:
   1= []
   for j in gramA[i];

K = []
       form inj:
            if m in revenus:
               k-append (m. replace.lm, revconv[m]))
            else
                 k-append (m)
         1. append Lies
      op L'revous:1) =1
```

```
rent - ren (grum)
dr in result.
     print (f'(i) -> result (i)))
WAS AUR LEFT PALTORING
from iterhood Ampost takewhile
 def group by (15)!
      d= 14
     6 - [4 [0] firy in rule]
     inhal + list (set (1))
      for y in initial:
         for i in Yule.
            if i starta oth Ly):
               ify not ind:
                  d:[4]=1]
              dily] appoind (i)
       meturn ol
      neturn/lenliser(m)) == 1
Starting :
 relle / [
 alphobersel = ["A", "B", "c", "D", "E", "E", "G", "H'
ermmon, []
               "M", "N", "0", "P"", "R", 'S", "T";"U"
                 'V", "W", "X", "Y", "Z'"]
      "Stifts | iEtsesla"
while (True):
      rules = []
        common = []
        split = 1 split("->")
        starting - up lut (0)
```

restriction of the second tal forther lange I grant I hamper . [[i] war I have to INPUT A Long about the "S-7 iEts | iEtses | a" Tio James Loil foll sin !! TO MINE PORCHER COLLE 21143 S → 1,E.f3) verso me o w. 1, -> e 1,62(1) will) mind die 10 = 10 2 . (1) 1000 A 16 1 4-14-146 41. 1 (( N VYU) 1-05 1 50

```
for i'm splat [1]. splat ('1"):
        ralles . append (i)
for kilin' groupby (nules) . items ():
      r = [1[0] for I in takewhile ( prefix, zip (+1))]
        common appoind (' ' Join (8))
 for i in common:
      newalphaber = alphabetser.popU
      print (starting +"->" fi'+ newalphabet)
      moder = []
       for k in rules:
if (k. startswith (i)):
                  indea.appendlk)
        print (newalphabet + "->", end="")
        for g in index [: -1]:
             stringto print = graplace(11,"", 1), "1"
             if string to print == "1":
                  print (" \u0885", "", end="")
            else
                  print ("1403B5";" |", end="")
                print (j. replace (i "", 1) + "1", and = "")
        stringdoprime = index {-1] . replace (i,",1) +"1"
          if string form => "1":
                 print ("\u0805","." end = "")
          else
               print (index[-1] . replace (1',"",1) + "", end = "")
          pnw-("")
                       programs are successfully
          The given
      executed.
```

write a program to perform first and follows using any language. I have not sell all the

#### ALCORITHM

- · For computing the first
- 1. If X is a formunal other FIRST (X) = LX3 Example F-I | id 12 miles where the set of We can write it as FIRST (F) > { (, id)
  - 2. If × is non terminal tike E-> [ than Fix57 substitute T with other production until you get a terminal as the first symbol

white and

34 1 3

- If X -> E then adol & to PIRSTERS
- · FOR company the follow:
  - 1. Always check the right side of the production for a non terminal whose FULLOW set is being found
  - 2. (a) If that non-terminal (SIATB-.) is followed by any terminal (a, b -.. , \* 1+ 1(,1,1) then add that terminal into the Follow ser
    - b) If that non-terminal it followed by any other am-terminal the add FIRST of the other non terminal into the FOLLOW . So

```
CORE
import sys
sys , set recuramlinut (60)
def first (string):
     first = set U
     if string in non-terminals:
           alternative = procluction - diet [string
         for alternature in alternature:
             first - 2 = fire (alternative)
               firs = first - Ifinst2
       elie string in terminal
           fira_ = f. ( smug )
       elif string =: ' or string == '@':
             first = 1 10'3
           first = = first ( string [0])
       else:
           if @' in first ?
               while 10' in first 2:
                    fre - = fre 1 (in 2 (6'3)
                    if string [i] in terminals:
                        fire = = first = 1 string[i:]3
                           break
                    elif strag[i:]= "!
                            first = first 1 f. L'e'y
                    first _ 2 = first (storing [1:])
                    forst - = 100 = 1/1052-2- {10'3
```

7-1-1

LNPUT! free of many top in which Enter no of femunals: 3 strong mo Inter the Horminal 1 9 trips with purity give 6 Enter the no of non terminals: 31 Enter the non terminals: bif I - 7 :41 CON WITH DO FIRSTLED SILED A MANY CONTRACT THE E - T MAY OF YOU Enter the starting symbol : A Enter no of production: 5. Forter the moduchan A: >nA B → PB company the files. way the right sate band fram B -> K . I want out and and and harman and the second of the s to the second se the water of the state

```
first -= first - lifera-2
follow (nT):
follow - = set U
prods = production_did. Hems 4
if nT == starting - symbol:
      follow - = follow - 1 2 4'4
for nt, the in prods;
    for all in this
            fir char in alt!
                 of char = : nTi
                    following str=alt[aft-index (um )+1.7
                 if fortlowing -str -= "1
                   if nt= 2 nT :
                   else:
                      fellers == fellow_ 1 fellow (n+)
            else
               Collow - 2 = first (following - Str)
               if '@' in follow-21
                  folloro_ = fillow - 1 fillow - 2 - (10)
                   follow == "follow - 1 follow (INT)
               Elm
                  follow - = follow - 1 follow - 2
 return follow-
```

```
no of term mas = int (input ("Enter ino of terminaler, "))
terminal = []
 print ( "Enter the mod terminal: "))
 Alor _ in range (no of terminals):
      terminal, append (Input (1):
no - of - nun-ferminal = int linput ( Enter of no. of
                                        non terminal)
Most prinon - terminal = D
  print (" "Enter the non tarminals: ")
   for - in range ( no - of non-terminal):
          non-terminal appared ( inpures)
 production-dut = 14
for not in non + terminal:
      production -det (n7] = []
for production in productions:
     nuntermenta prod = production.splet ("->")
    alternative := non tornito-prod [1]. splet("/")
    for alternative in alternative
           production _det · [non term_to-prod lo]].
                                      append (alternative)
FIRST = LY
```

FOLLOW = 4

for non-terminal in non-terminals: · YIKS "[non- Fermi not] set () nm-terminal in non-terminal: KULLOW (non- ferminal) = set 0 om terminal in nm-terminals: KIRST Phon = terminal] = KIRST [ non-ferminal] I frost Enm tomine FULLOW (Shorting\_ eymbol) = FOLLOW (clarting - Sy rubol] 1 \$3 print (" {: ^20 {: ^20 } {: ^20 } 2 - ^20": firmat (Non terminals, ' Kirst ( 'Falow')] for non-terminal in non terminal. print (" {! ^20" 2: ^20 " {: ^20": firmat (non farmino str ( KIRST ( Non -terminoul), RESULT str ( KOLLOW [nm.terminal]) The first and FOLLOW set of non-terminal of a grammer were found and succentully using python language

### EXPERIMENT 6

#### PREDICTIVE PARSING TABLE

AIMI

To write a program for Predictive Parsing Table

#### ALGORITHM!

For the production A -> a of Creammer G - For each terminal, a is ARST (a) add A -> a to M[AIN]

- If 6 is m ARST (a) and b in FOLLOW (A) then add A -> a to M[A,b]
- If E is in FIRST (as) and & isin FOLLOW (A) then add A -> a to M[A, \$] - All remaining entires in Table Mane emors A STATE OF THE PARTY OF

3 4 7

#### PROURAM:

#include < stelio-hz

#Include < string. h>

char pod [7][10] = ["8", "A", "A", "B", "B", "c", "c", "c"; char pror [7] [10] = {" ," (36", "Cd", "aB", " @", "Ce", "@" cher prod [7] [10] = 2 S -> A", "A -> 'Bb", "A -> CD"; "B-> QB", "B→@", "c→·cc", "c →@"\$;

Char first[7][10] = { "abd", "ab", "cd", "a@", "@", "c@", "e char follo [7] [10] > { "4", 9", "\$", "Q\$", "bf", "c\$", "c\\$","c\\$", char table [5][6][6];

```
Int numr (charc) &
    switch (c) {
        case 's':
           return 0;
        case 'A';
           return 1;
        case 'B';
          return 2;
         case 'st':
            return 3;
        case 'a':
         return 0;
        case b:
          refum 1;
          case cl:
           netum 2;
          case 'd'
             neturn 3;
           case's' return 4;
   neturn (2);
  for (1=0) 125; 14+)
     $ (1=0 ; 1 × 6; 1+1),
        stropy (table [i] [j];"");
printf ("The following grammer is used for Parsing Table: ");
  for ( =0; (<7; (++)
         printf(" y.s hi", prod [i']);
```

```
INPUT - TOTAL - MALE PARTY PARTY
  enter the no of non-terminals
                                          300
      36
     Enter the production in the grammar murias
      5->EITT S-rabol was brigged and
                 Byce or position of and will
                  Cable
    THE POLICE OF HOLDERS (D)
                  E ragalom of nor a laber mod
     DUTPUT F & f (6) TEMP NO 1 3 11
   first wo M states in Table M and total
                            FOLLOWES]=$
     FIRST[B] = a
                            FOLLOW [13] -ges 36
      FIRST[B] = CC
                             FOLLANG CJ : ge$
      FIRST [C] - be
     FIRST [D] = ge

FOLLOW[F] = h

FOLLOW[F] = h

FOLLOW[F] = h

FOLLOW[F] = h

FOLLOW[F] = h
       1 m-5" . "0 -8"
th so "i - [ii] [r] but
 grant to the "test, to be the test of the test of
                                [a] [4][2] retoF
```

```
printf ("Predictive Parsing Table");
    fflush (stdin);
      for (1:00; 147; 14+) {
           K = Strlen (first[1])
           for (j=0; g < 10; j++)
                  if first[[[]]!= '@")
                  Stropy (table [norm (prol [i][0]+1]
                                    [ numr(firs [i][j]+1], pod (i]);
       for (1=0; 1=7,1++) ¿
          if (pror [i][o] == '@") {
               K = strlen (follow [1]);
               for (1=0; jek, j++)
                   stropy (table [numr (pro) [i][0]) 71]
                           [numr (folloo [i]]] + 1], prod [i]);
 stropy (table [0][0], "");
 smpy (table[0][1], "a");
Strepy (table [O][2],"b");
styppy (table [0] [3], "c");
Stropy (tuble [0] [4], "d");
stropy (table[0][5],"$")
strapy (table[1][0] ,"s");
shry (table [e][o] ("A");
stropy (table [3] [0], " B");
strypy ( take (4) [0], "C");
```

1 2 day will M[sm] = s - aBon 1707 M[B,c] = B +CC PER SELLY 2 A 15 SQ M[C,e] = C > e on the M[Dig]: D > EF (H) (U.S.) M[Dre] = D > EF . 15 5/10 MLEGJ= E -9 Same MICIOT-E-E 0 1000 M (kit) = f-it D WINTER M[r,e]=F+e 2 1 1 1 1 1 1 1 1 1 SHARDH BACC b DHEF 1 For lipto 1 mi (riv(92,1/0-1) the state of the state of the sale of the Chester 1 Land Took of the William ( St. ) I low reg

for li=0; 125; 14H) for y=0; 1 < 6; 1++) { printf ("1.-10s", table (1)(7)) if (1==5) printf("In - - - - - - - - \n"); RESULT

The implementation and creation of predictive parse toble using c was rexecuted successfully

Shiff Reduce Parsing

AIM:

To write a program to implement lexical Analysis using a

#### ALCORITHM:

- · Shift reduce parsing is to process of reducing a string to the start symbol of a grammer
- · Shift reduce parsing uses a stack to hold the grammer and an imput tape to hold the string
- · sift reduce parsing performs the five actions: whiff and reduce · That is why it is known as shift reduce parsing
- · At the shift action, the current symbol in the input siring pushed to a stack.
- by the non-terminals.

  The symbol is the right side of the production and post party is the left side of the production

#### PROGRAM:

#Include < string h>
#Include < string h>
In k-0, z=0, v=0; 1=0, c=0;
ohara [16], ac[20], ptt.[15], act[10];
yord checky;

## INPUT:

"E": ["E\*E" "E +E" " "i"] and digital of the mind = 100 E's grant stand = 100 E's grant

(Fig. ) and

and the first the description

#### OUTPUT:

Stack.	1 Input Butter	1 Parsing Achan
\$i'	1 4 1 21	suf
\$ E	+ 1 *1	Reduce S→1
\$64		Shift
\$ EM	1 + 1	shift
\$E+E	*1,	Reduce s-1
\$ E*	t.	Reduce 5- 8-48
\$ E*	I t	y shift
\$ 52	li	1 Regerred

RESULTI-

The implementation of shift reduce parsing was executed and verified successfully

17 Just /

## AIM:

To write a program to compute of lead and Trail

#### ALGORITHM:

- 1. For leading, check for the first non-terminal
- 2. If found, print it.
  - 3. Look for next production for the same nonterminal
- 4. If not found incoursively call the procedure for the single non-terminal present before the comma or End of production string.
- 5. Include it's results in the result of this non-deminal.
- 6. For trailing, we compute same as leading but we start from the end of the proclusion to the beginning

## 7 Stop

#### PROURAM:

#Include < ibstream >

#Include < ahrng. hz

# include < stolly h7

wang hanyelpace stel;

int vars, terms, i',j, k, m, rep, count, temp=-1; char var[10], term [10], [ead [10] [10], trail [w] [10];

```
struct grammer &
     unt prodno;
     char lh; rh [20][20];
   y gram [50];
 voidget U t
       COUL CO " IN LEADING MAD TRAILING ";
       cout < " In Enter the no of variables: ";
                     Exit of a lot of the day
        CIMYY VOURS)
       cource "In Enter the vanable: ")
   for (120) 1' ( Wars; 14) 2
         an 77 gram [i] . ths; 1 - [3] . 10
        var [1] - gram [1] . thy;
    course" In Enter the no of terminals:";
    anzz tems;
    cout cc" in Enter the terminals: ";
   for ( =0; Jeterms; J++)
         cin 77 term [] ];
      eout ce" In PRODUCTION DETALTS IN";
world reading U L
        for (i=0; i' < vars; i++) {
          for 11 =0; 1. gram [i] - prodno; 1+1) {
                ter (1000; Keterme , K++) {
                   if (gram [1]. rhs (4:)[0] == term [K])
                         read CIJCKJ=1;
                    else ¿
                      if (gram [1]. rancy][1] = = term [x])
                              lead [1][ x] = 1;
                       ž
```

and to strepared to wrong it is

INPUT

Enter the no of Production: 6

Enter the production one by one

E > E + E

truck the constraint for the property of the truck

F-) ( of mynnporg to brid to

THE CIEBLES IN THE DECKING ALLE

P -i

the conditions of the production to

SU.

```
dir 1 repzo , rep (vars , rep ++) }
         for (120; ic vars; it) L
             for you; J c gram [1] . prod no; J++) &
                  fr (m=1; m cvars; m +H) {
                        if (gram [1] . or ho (y) [0] == var[m]) }
                 out:
               for (K=0) K = terms; K++/2
                       if · Clead [temp] [12] == 1)
                                lead Cilliks = 1;
         y
y
world trailing U {
       for (1=0; i < Vare; i++) 2
            tor (1=0;1< gram [1]. prod no : 1++) &
                  ound = 0;
                 while (gram [1] . This [] I [wunt] 1 = "1x0")
                         COURT ++)
                  for (K=0; K < terms; K++) {
                      if Gram [17. rhs [] [ count = 2]== term[1
                            frout filte =1)
       <sup>و</sup> ي ب
```

subject the planting I all all many In Jung OUTPUT 1 th ng THE TENENT WHAT TENENTS IN LEADING LEJ : 2+ , \* ( C, 14 ) LEADING [4] = 2\*, (, i) (2004 COME) LEADING CFT 1 ( 19 1 (150) 1 CAGAR ) 1911) T TRAIL MU [E] = 2+ (\*1): 13] mass 500 TRALLING [T] = (+1); [11/1000] - [1] and TRAILING ET 2 2 9: 24 WH WARD OF SOLD faction of 26 13= 47 24 (1) m +35 -4 41) rout as the presentations extreme he is i Upychosi I (THE CARRY S' TO SOL) WITE of the Champand - City of City - Champan City i (ma ; motors ; en) in the same of the plant of the same of O- ENGINEER (Interest to profit to

Int main U i

get U;

leading U;

trailing U;

chisplay U;

y

RESULT:

The program to find lead and trail
was successfully compiled and oun

AIM! Computation of LR (0) item

To write a program to implement Lk(0) items

#### ALGORITHM :-

- 1. Start
- 2. create structure for production with RHS and LHS
- z open file and read imputifile
- 4. Build State 0 from butra grammer Law S'→S \$

  that is all start symbol of grammer and one dot(1)

  be fire & symbol
- 5. If not symbol is before a non-terminal, add grammar law that this nun-terminal is left hand side of that have and set Dot in before of first part of RHS.
- 6 If state exist , used that incread
- 7. Now find set of torminal and non-torminal in which pot exist in before
- e. If step 7 set is non empty goto 5, else go to 10
- 3. For each terminal | non-terminal in set step 7 create new state by using all grammer law tent Dot position is before of that terminal I non-terminal

THE MODEL NO. 700

1000

- w. Go to step 5
- 11. End of state building
- 12. Display the output
- 13 · End

PROURAM!

Hindladec 10s Hamz

#Included conto. h?

#Include & string to

using numespace stdy

char [prod [20] [20], listofur [20] = "ABCDEFUH IJKL MNOPOR";

mt novar =1 ) 1'=0, K=0, n=0, m=0, arr[30];

int nother = 0;

6 AND ALBOR 3 with course who is a 10) of the property of the managery of the of the INPUT: ENTER THE PRODUCTIONS OF THE CHAMMER (0 TO END) ENGINE THE HERE MADELLED WE SHIPLE 14 05'15 alf depend from Ent. 11 th regard - fle. E-T 1 2 Total warming orders much or whose bleed in that it all there symbol of grammer and one dollar. be for a specket F -> (E) The confirmed is begin a non-distance of the garme low that the number of hours of the that the how and set their in before of their part of the hardons tods toms, dairs state to 1 OUTPUT ist mad been leavened by the best wolf Which Dot must in Enforce If stop I see is no empty fits, else it to to E - EtT to lower of love / loverer to the of CORDAY THE STEEL OF USING ALL GROWNER JAST C-F when the street of their street (3) commend of 2 gate of 010 of F-7 i rubband about to beet YTEMS ARE the company B. End To A->·E PROUNTAIN E TIETT Hindual City branes E->·T Hinduide Samonit ナナ・ナナ This or shows ! that [mest [20] [20] . Lievapan [20] = " Fire coefficien ) - E. . F->illostro, was pro pro son son in province the Donath to

1/2/11

A HARRIST AND A STREET I3 TA We and F - (.E) €-> F.+T Iz TAP SON the world of a rever in E-> T. Manual - - VIII ( ) ) T→ T.\*F they maked For Charge O. Is Te (Teloure (m) months of (T) T\*·P

F→·(E) E-→E4.T T+> THE - Cent ( (4-0) ) ( 11 (4 ) ( 12 ) ( 12 ) ( 12 ) ( 12 ) ( 12 ) ( 0 - 1 [ 1 - 1] 1 - 1 - 1 ] 1 - 1 ] 1 - 1 ] 2 - 1 ] 2 - 1 ] 28 . . = = [ [] net . [ ] [ [ ] ] ] ) ]. E→ E+T.4).[v](v)(v) TTT TKY [i] [i] [i] also plant [i] [i] [i] TTTT Eliter [ ] [motored and a great work [ without the last of the freshord in all I and the start of I was 1 400 201 the state of the smarteness of the state of the termination of the state of the state of the state of Contract Contract of (6) when

```
dr (k=3; Kc Ehrlen (prod (m)); K+1+) L
     if ( hua [w] [k])=1,)
          g [g]. Ins (m+1) = prod [n][1);
         if (prod [h][k] == 111) L
                 ay ]. rhs [m] = '10';
                 1 = novar;
                 g [novar+1]. ths = | mod [n][0];
    y y y
for (1=0; 1=26; 2+4)
         of ( ) is variable ( list of var [17))
  good the list of vertil;
   char temp[2] = {g[i].lhs, 10'8;
  Streat (g Co]. +hs, temp);
   cour 12" augumented grammer";
 for (1:0; i c novar; 1:4)
      clus (no item) [i] · lhs = g[i]. rhs;
      strupy (clos [nothon] [i]. who , g[i]. rhs);
      If (shamp Lelos [no item][i].rhs; "G") ==0)
               strepy (clos [noitom] (i). rhs ".");
for ( int 270; z choitem; z+1) {
     char list [10];
      Int 1=0;
     for yeo; Jean(2); 1++) {
             for (K=0; kcsrren ((los[z] 97. rh)-1;
                  if (who (2) 47.7 ho (12) = = 1.1)
```

list[1+1] + clos [2][7]. This [k+1];

y

cout ce "The set or ITEM e are";

for (mb 2 = 0; Z = no item; z ++) {

cout ce "T" cez xe "\n";

for (g = 0; g ear [1]; g ++)

cast ce clos [2][7]. In re "->" ee cas [1][]+h

a """;

RESOUT:

Are program for computation of LK[0] was

stressfully compiled and run

Intermediate adde Generation - Postfix, Profix

MI

A program to implement Intermediate abde grantendestroix, Breglix. C. 100

#### GORITHM

Declare set of operations

the state of the state Inchalize an empty stack

To convert INPIX to POSTPIX following Steps

4. Scan the infix expression from left to right 5. Scan theinfix expression from le

If the scanned character is an operand, orthwest.

- 6. Else, If the precedence of the scanned operator is greater luon the precedence of the operator m the stack
  - Else, Bp all the operators from the stack. which are greator than or equal to in precedence than that of the scanned operator
    - & If the scanned character is an '(', push it to the stack,
    - If the seamed character ")", pop the stack and it untu a'l' is encountered and dis and both the porathesis
      - 10. Pop and output from the stack until it is not
      - To convert INFIX to PROTEX follow the steps
      - scan the expression from left to right
      - 14. Whenever the operands arrive , print them
    - 15. Repeat step 6 to 9 until the stack is empty

```
PRO GRAM
  OPERATION = set ( [+' , '-', '*', '/', '(', ')'])
       infix- to- postfix (formula):-
      stack : []
     output > " "
   for the in formula:
      if th not in OPERATORSI
           pulput #= ch
      ewf ch == "(':
           stack. append ("L")
      eluf ch == 1)1:
          whele Stack and stack [-1] != '(':
               output += suck popu
      else 1
         while said and stack [-1] != '(' and
                           PRI [ch] <= PRI [stack[-1]]:
             output += stack popu
        stack append (ch)
    coule stack!
         output + = stack, popu
    print of POSTAX: Eoutauty)
   beturn output
def infix-to-prefix (formula):
    Op- Sauk = []
    exp stack = 17
     for in formula:
         if not the IND PERATION:
             exp_stack.append(ch)
        elif ch == '( ':
             op - Stack. append (Uh)
         ely ch2 21)
```

INPUT:

No. 1 Try

INPUT THE EXPRESSION: (a+b) \* (c+d) /8

to the application of the second seco

PRESENT 14-+ab+/coly

affective securical abouted William with the series with the wife

# OUTPUT:

PREFIX: 1 \* + ab+ col x

700

POSTAX: abit cd + \*x/ to some it was a

Parket Bally Indiana was a second

Many destruction of the same

while op- Stack [-1] = 1(1) of + = sop - Stack. pop 0 a = exp - stack . Popu b= exp- slack. popls exp-stack, append (op+b+a) op\_ stack . popU print (f' PREAX : { exp - stack (+) 5') return tep-stack [-1] intix- do - pretix (express) infor - to- postox 1 express was successfully compiled and RESULT

Intermediate Code Creneration
Obtainable, Triple Indirect Triple

AIM! To implement code generation - acad raple. Triple indirect triple

### ALLORITHM!

The algorithm take a sequence of three address statement as input.

For each three address statements as input

For each three address statement of the form.

a:26 or perform the various action

- 1. Invoke a function get reg to find out the location L where the result of computation b. op a should be stored
- 2. Consult the address description for y to defermine y'.

  If the value of y currently in memory and register both then prefer the register y'.

If the value of y is not already in L.

then generate the instruction Mov y', L to place a

copy of y in L.

If the current value of y are have no next uses or not live on eart from the block or in register than after the register descriptor to the descriptor that after execution of the seguister will no 21=4 of 2 those register will no longer contain 4 or 2

NPUT

- Enter the expression: CI) a= 5+ c +d + 100
  - 2) Enter the expression: a = b x - c + b x - c

Name of Table of the State of t

war and the second of the second

Margar 1 10 to Section of the Sec.

The party

The first of the first of the first on a first on a first

OUTPUT: del mone 10 illen elle mone 11 malano (1) formale of the The intermediate code Councily 62 2 5+H a = tz

it is a freedom for an angeline and the 2) The intermediate code is 乜=以tis = cate to plan and to the control of the

th = 1-61-14 map lay 100 t5222 43 Mayor all 1819 ...... a = 15 10 material with the state of

```
PROGOURE:
   # include < std10.hz
                                                                                                            with a significant over the re-
# include < stellib.h>
  # indude < string. ho
        word smalle;
      word dove (int 17);
      int p[5] = {0,1,2,2,49,0 = {11, led, m, ipi;
      char sw[5] = { = ",'-1, '+1, 1/", '413, 1
                                                                                           1 [20], a[5], b[5], ch[2],
         void main () L
                     printf (" Enter the expression");
              Scanf ( "ys" ,9);
                                                                                                                                                                 and one
              print fl" It The intermediate coolers: In');
                                                                            meany as and the state of the state of
                 small ();
                                   resolved to the later of the la
                void dove Cintill
                     a loj = bloj = 10;
                  if [ ! is diget (][i+2] 8181 is diger (][i-2])) }
                              a fol = 560 1= 10)
                                  alo1= 1 [1-1]; /
                                      66J= 9(1+U)
                     if (isolight ( [1+2)) {
                                a [0] - 1[1-1];
                                6601= 't';
                            bu) - 1 [i+2];
```

XPRESSION \ az b +-c THE SHARE STREET PREFIX : - - bcbc will be to be the same POSTFIX: Q =6 \*c-b\*+c-THEREIS APPRESA COPE EXENCERATION 42 = 41-6 +3 = +2 × b tu - a + + 3 E= anay mores 15 = tu-c Time entire admi The quadruple for the expression 11012-00 · AKAZ RESULT +0 +(2) (9/ +760 - 1 N 3 Real To agency Charle to the Gird . Tr. ALTER END

if (111) == sw [m]) if cpuc = mrcm3) i pr= plms; "f (1==.1) dove (k); else exited) RESULT successfully compiled and 10 4/23 EXPERIMENT-12

Am !

Josepher a code der simple cede

### ALCOKITHM.

1. Take user inputs for the name of the class and the number of field.

" week

- 2 Installer an empty list called field.
  - Take user input for the name and type of seen field
    - Append a typic containing the field name and type to the field List
  - 4. Print generated code to amsole

CODE

class-name = input ("Enter the name of your class:")

num-field = intlinguabl "Enter the number of field for

your class"))

fields = []

for i in range (num. fields):

field-name = input (f" Enter name for field 21+13: ")
field-type = input (f" Enter type for field 21+13: ")
field-type = (field-name, field-type)

INPUT:

Enter the number of fields for

market a larger to the law on the court

hoder - Mile - a"

A Brown I'm Brown

OUTPUT

SUB RI, RZ

MOV 4, 82

MOV ZIR3

MUL RZ, R3

```
if op= = +1
      machine- institution. appered (f" Appliang 15, larges
                                               (nesult)
 else op >= 'w',
       machine - Instruct. appared Lt " MULSargis, 2 augzs, losus
elast op = 1/1:
      machine-instruct - grand (f' DIV lange), langes, (nows
elif op 2= '= ':
         machine_ instruction. append (f" mov larg 19, Cresults")
alef red-three-addres-coole ()
    three -addres - code = []
     white har !
         instruction: in put:
          if instruction = = clime:
             break
         part = insmutan split
         of = part(i)
        anger parts (e)
         arg 22 parily
         rout = part[4]
        three -address-orde-append (op , ang 1, ange, result))
```

nachine -instructions = generate-machine instruction (three adding code)

INPUT x + 3 = Y 2 \* 4 = Z becare in the to some out of degree was John J. 15 Va OUTPUT Crenerated mainine instruction POTAL DAMANCE OF TO MONEY THE MESON APP 2,3,X ADP MI 31 4 6)21 MUL 21412 a pulle confirmed in help have and to the state time percented control & console and the contract of the world of the contract of

= MIRING ALTERNA TO MODERN OF FILE A

print ("trenerated machine insmuchini)

for insmuchin in machine: insmuchini

print (insmuchini)

print Resour

The following code was implemented

for a simple code generator.

18/4/23

EXPERIMENT 13

AIM :

A program to Implementation of DAL-

Algorithm: -

1. The leaves of a graph are labeled by a unique i'clentifue can be vanable name or constant

- 2. Interior nodes of the graph are labeled by an operator symbol.
  - 5. Nocle areaso given a sequence of identifier to labels to store the computed value
- 4. If y operand is undefined then create nodely)
- 5. If z operand is undefined then for case (i) create noode (z)
- 6. For case(i) create noole cop) whose right child is noolely)
- 7. For case ay cheek whether there is node (OP) which one child nodely)
- 8. For case (u), node n will be nodely)
- identifies. Append on to attached identifiers. Inst for the mode in found in step2. Finally set nucle Cry to n

Code:

Hindude < i'ostream?

# includes string>

#include cunordered\_map>

using namespace sta;

Class DAR {

publici

char label; Char down;

DAG+ left; PAG+ orgent;

INPUT

A= x+4

B=A\* Z

C= B/x

the limitary had some most mading in

OUTPUT

LABEL : PAY left Phy night phy

B \* A 2

C / B \*

g traine - and la

al orate

-1100

150

(1113.25

```
DAG (charx) 1
        label = '-'; data = n;
        left = NULL; ngw = NULL;
  DAG ( Char lb, char >, DAG # 1+, DAG + ++) ¿
         label 216; dale = x;
         left = lt; ngw = rt;
3;
    main () E
       10 27
     shing stling)
        sl los = "A = x +y";
        SH11] = "B=A *z";
        st[2]: "C= B/x";
      under ordered imap < that, DACT > label DACT Node;
     for (int 1'=0; 123; 14+){
          string streng = stirl)
          for (m) 1=0; 1 < 5; 1++) i
               char temp Label = stemp [0];
               char templeft = st Temp[2];
               char temp Data = St Temp[3],
               char temp Right 2 St Temp [4],
               DAY * Left Phr;
              back + right Ptr;
              "f (label DALTNOCK Gount (tempteff) == 0) {
                       left Ptr = losser DAG (damplet);
              else L
                1094 Ptr = label DAYNOUL [templett],
            if ( land Drunboe · count (temp Right) = >0) {
                   right PAr 2 new DAG (temp Right);
```

```
else L
         nguiptr 2 label DALINOCLE [tempregni];
 DACI + MN = New DACI (templabel, templata, lett Pr, right Pr)
   label DAGNock. Unsert ( make-pair (remplaisel, nn));
 3
                                           to our T
cout ce " label - Phr left Phr . right Par " ecendl;
 for ( int 1:0) 12n31++1
    DAG *x = label DAGNode [ALLI] TO]
     if (x -> left -Nahel == '-')
    Course m -> nght -> class;
     cout con neft storely
    cetilize " " " " ( " IUM" . " " " " ) a bearing
    17. Ca -> nghi -> 1000el = 0'-1)
                   contect to right solutary
      contecenolly
return 0;
RESULT
       The program was successfully compriled
 and run
                                  ST. F VON
                                  1.1 - 13 - JUM
```