DEPARTMENT OF : CSE

NAME OF THE LABORATORY : CC LAB

Name K·S·I· SIVANI Roll No. 052 Page No.

11) Implement lexical analyzer in lex:

```
%{
 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|main
{printf("\n\t %s is a keyword", yytext);}
 "%*" {comment=1;}
 "*/" {comment=0;}
{identifier}\{ if(!comment)printf("\n\n FUNCTION\n\t
              %s", yytext);}
 \{ if(!comment) printf("\n Block Begins");}
 \}{ if (!comment) printf("\n\t %s is a NUMBER", yytext);}.
\)(\;)?{ if(!comment) printf("\n\t");
ECHO; printf("\n"); }.
\( ECHO; ={ if (!comment) printf ("\n\t an assignment operator
                                              yytext);
```

DEPARTMENT OF : CSE

NAME OF THE LABORATORY : CCLAB

Name K·S·I·SIVANI Roll No. -052 Page No.

```
\<=|\>=|\<|==|\>{if(!comment)printf("\n\t%s is a
relational operator", yytext);}
\b|\t{if(!comment)printf("\n white space", yytext);}.
\n{;}
%%
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();
  printf("\n\n");
  return0;}
 int yywrap()
 {return0;}.
```

DEPARTMENT OF : CSE

NAME OF THE LABORATORY : CCLAB

Name K·S·I·SIVANI Roll No. -052 Page No.

WEEK-3 : Find first set and follow set:

```
#include <ctype.h>
#include <stdio.h>
# include <string.h>

int count, n=0, m=0, k, e;
char final_first[10][100], final_follow[10][100];
char production[10][10], f[10], first[10];
char ck;
void grammarfollow (char c, intC1, int c2)
{ int k;
   if (!(isupper(c)))
       f[m++] = c;
   else{
    int i=0, j= 1;
    for (i=0; i<count; i++)
     { if (final_first[i][0] == c)
             break; }
    while (final_first[i][j] != '!')
     { if (final_first[i][j] != '#')
       { f[m++] = final_first[i][j]; }
```

```
else{
  if (production[c1][c2]=='\0')
   { follow(production[c1][0]); }

else { grammarfollow(production[c1][c2], c1, c2+1);

 } }
 j++; } } }

void find_first(char c, int q1, int q2)
{ int j;
  if (!(isupper(c)))
 {  first[n++] = c; }
  for (j=0; j<count; j++)
   { if (production[j][0]==c)
      { if (production[j][2] == '#')
        { if (production[q1][q2] == '\0')
                 first[n++] = '#';
          else if (production[q1][q2] != '\0' &&
                      (q1!=0 || q2!=0))
          {   find_first(production[q1][q2], q1, (q2+1)); }
```

DEPARTMENT OF : CSE

NAME OF THE LABORATORY : CC LAB

Name K·S·J·SIVANI Roll No. --052 Page No.

```
    else
      first[n++] = '#'; }
else if (!isupper(production[j][2]))
  { first [n++] = production [j][2]; }
else { find_first(production[j][2], j, 3) ; }
 }}
}
void follow (char c)
{ int i, j:
   if (production[0][0] == c)
    { f[m++] = '$'; }
   for (i=0; i<10; i++)
   { for(j=2; j<10; j++)
      { if (production[i][j] == c)
         { if (production[i][j+1] != '\0')
            { grammar follow (production [i][j+1], i, (j+2)); }
             if (production [i][j+1] == '\0' &&
                        c != production [i][0])
             { follow (production [i][0]); }}
   }}}
```

DEPARTMENT OF : CSE

NAME OF THE LABORATORY : CCLAB

Name K.S.I.SIVANI Roll No. -052 Page No.

```
int main(int argc, char **argv)
{ int jm=0,km=0, i, choice;
  char c, ch;
   count = 3;
  strcpy(production[0], "E=AB");
  strcpy(production[1], "A=ilove");
  strcpy(production[2], "B=jtptutorials");
   int ff;
   char done[count];
   int ptr=-1;
   for(k=0;k<count ;k++)
   { for(ff=0;ff<100;ff++)
   { final_first[k][ff]='!';}}
     int point1 =0, point2, xxx;

    for(k=0;k<count ;k++)
    {  c=production[k][0];
        point2 =0;
        xxx =0;

       for (ff=0;ff<=ptr;ff++)
       {  if(c==done[ff])
               xxx=1;
       if(xxx==1) continue;
~~feed~~ find_first(c,0,0);
      ptr+=1;
     done[ptr]=c;
    printf("\n First (%c)={", c);
  final_first[point1][point2++]=c;
  for(i=0+jm;i<n;i++)
  { int fs=0, chk=0;
     for(fs=0;fs<point2;fs++)
     { if(first[i]==final_first[point1][fs])
       { chk=1; break;}}
 if(chk==0){ printf("%c", first[i]);
final_first[point1][point2++]=first[i];}} printf("}\n");
 jm=n; point1++; } printf("\n");
printf("==========""\n\n");
char donee[count]; ptr=-1;
for(k=0;k<count;k++)
 { for(ff=0;ff<100;ff++)
   { final_follow[k][ff]='!';}}
   point1=0; int land=0;
   for(e=0;e<count;e++)
```

```
    ck = production[e][0];
    point2 =0;
    xxx =0;
  for(ff=0; ff<= ptr; ff++)
       if (ck== donee[ff]
              xxx=1;
  if (xxx===1)
       continue;
   land +=1;
  follow (ck); ptr+=1; donee[ptr]=ck;
  printf("Follow(%c) ={", ck);
 final_follow[point1][point2++] =ck;
    for(i=0+km; i<m; i++)
    { int fs =0, chk =0;
      for(fs=0; fs<point2; fs++)
        { if (f[i] == final_follow[point1][fs]) chk=1;
               break; }}
    if (chk==0) {printf("%c, ",f[i]);
    { final_follow[point[1][point2++] = f[i]; }}
         printf("}\n\n"); km =m; point1++;}
```

OUTPUT:

```
First(E) = {i, }
First (A) = {i, }
First (B) = {j, }
= = = = = = =   = = = = =
Follow(E) = {$, }
Follow(A) = {j, }
Follow (B) = {$, }.
```

DEPARTMENT OF : CSE

NAME OF THE LABORATORY : CCLAB

Name K·S·I·SIVANI Roll No. - 052 Page No.

Recursive Descent Parser:

```
#include <stdio.h> #include <string.h>
#define SUCCESS 1
# define FAILED 0
 int E(), T(), F(), X(char);
  const char *c; char s[64];
 int main()
 { puts("Enter the string");
   sscanf("i+(i+i)*i", "%s", s);
   c=s; puts("");
   ~~p~~puts("Input Action");
  if (E() && *c=='\0')
   { puts("String is successfully parsed");
     return 0;}
  else { puts("Error"); return 1;}}

 int E()
  { printf("%-16s E→TX\n", c);
    if (T())
    { if (X('+')) return SUCCESS;
   else return FAILED; }
  return FAILED; }
int x(char ch)
 { if (*c==ch)
  { printf("%-16sX→ %cTX\n", c, ch);
    c++;
   if (T()) {
    if (x(ch)) return SUCCESS;
   else return FAILED; }
   return FAILED; }
 else { printf("%-16sX→$\n", c);
  return SUCCESS; }}
int T(){ printf("%-16sT→FY\n", c);
 if (F())
 { if (x('*')) return SUCCESS ;
  else return FAILED; }
 return FAILED; }
int F()
{ if (*c=='(') { printf("%-16sF→(E)\n", c);
     c++;
  if (E()) {
```

```
{ if(*c=='')')
 {c++; return SUCCESS;}
 return FAILED;}
return FAILED;}
else if (*c=='i')
  {c++; printf("%-16s F→i\n",c);
   return SUCCESS;}
   return FAILED;}
```

OUTPUT:

```
Input              Action
i+(i+i)*i          E→TX
i+(i+i)*i          T→FY
+(i+i)*i           F→i
+(i+i)*i           X→$
+(i+i)*i           X→+TX
(i+i)*i            T→FY
(i+i)*i            F→(E)
i+i)*i             E→TX
i+i)*i             T→FY
+i)*i              F→i
+i)*i              X→$
i)*i               X→+TX
)*i                T→FY
```

```
)*i     F→i
)*i     X→$
*i      X→$
i        X→*TX
         T→FY
         F→i
         X→$
         X→$
         X→$
```