1) write a c program to find first of any grammar

```
# include <stdio.h>
# include <ctype.h>
void FIRST (char[], char);
void addResult (char[], char);
int numofproductions;
char productionset [20][20];
int main()
{ int i;
char choice, c, result[10];
printf ("enter Total no of productions:");
scanf ("%d", & numofproductions++)
{ printf ("productions Number %d:", i++);
scanf ("%g", productionset [i]); }
do
{ printf ("\n Find first of:");
scanf ("%c", &c);
FIRST (result, c);
printf ("\n FIRST (%c) = { ", c);
for (i=0; result [i] != '\0'; i++)
printf ("%c" result [i]);
printf ("}\n");
printf ("press n to continue:");
scanf ("%c", &choice);
}
while (choice == 'n');
```

3

```
void FIRST (char* Result, char c)
{  int i, j, k;
   char subResult [10];
   int foundEpsilon;
   SubResult [0] = '\0';
   Result [0] = '\0';
   if (!(isupper (c)))
   {
        addResult (Result, c);
          return;
   }
      for (i=0; i<num of productions; i++)
      {
         if (productionset [i][0] == c)
         {
      if (productionset [i][2] = '$')   addResult (Result, '$');
      else
          {  j=2;
             while (productionset [i][j] != '\0')
             {  foundEpsilon =0;
                FIRST (subResult, productionset [i][j]);
                for (k=0; subResult[k] != '\0'; k++)
                   addResult (Result, subResult[k]);
                for (k=0; subResult [k] != '\0'; k++)
                   if (subResult [k] == '$')
                   {
                      foundepsilon = 1;
                        break;
```

Teacher's Signature : --------------

```
        if (! foundEpsilon)
            break;
        j++;
        }
        }
    }
    }

    return;

void addResult (char result[], char val)
{
    int k;
    for (k=0; result[k] != '\0'; k++)
        if (result[k] == val)
            return;
    result[k] = val;
    result[k+1] = '\0';
}
```

enter Total no. of productions: 3
productions Number 1: E=AD
productions Number 2: D=+rD
productions Number 3: D =+

find first of : D

FIRST (D) = { * + }
press n to continue : n

find first of : E

FIRST (E) = { }