Programming Assignment-3 - Implementation of Left Recursion Elimination

AIM:

To write a program to find whether the given grammar is left recursive or not and eliminate the left recursion .

PROGRAM CODE:

```
#include<stdio.h>

#include<string.h>


int main()

{

        int n,i,x=0,j=0,y=0,l=0,k,s,flag;

        char ch;

        char str[10],new_str[20];

        char alpha[10][10];

        char beta[10][10];

printf("Enter the no of productions: ");

scanf("%d",&n);


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

{       x=y=j=l=0;

        flag=0;

        getchar();
```

```
printf("\nEnter LHS of production %d: ",i+1);
scanf("%c",&ch);
printf("\nEnter RHS of production %d: ",i+1);
scanf("%s",str);
k=0;
while(str[k]!='\0')
{
        if(str[k]==ch)
        {       //printf("yes"); exit(1);
                flag=1; //there's left recursion
                k++;
                strcpy(alpha[x],"\0");
                //printf("AL%sal",alpha[x]);
                j=0;
                while(str[k]!='|' && str[k]!='\0')
                {
                        alpha[x][j]=str[k];
                        //printf("\n%c",alpha[x][j]);
                        j++;
                        k++;
                }
                alpha[x][j]='\0';
                //printf("\nalpha %d: %s",x+1,alpha[x]);
```

```
            x++;
            if(str[k]!='\0')
                k++;
        }
        else
        {
            strcpy(beta[y],"\0");
            //printf("\nhi"); exit(1);
            l=0;
            while(str[k]!='|' && str[k]!='\0')
            {
                beta[y][l]=str[k];
                //printf("\n%c",beta[y][l]); // exit(1);
                l++;
                k++;
            }
            beta[y][l]='\0';
            //printf("\nbeta %d: %s",y+1,beta[y]);
            y++;
            if(str[k]!='\0')
                k++;
        }
}
```

```
/*printf("\nno of alpha: %d",x);
for(s=0;s<x;s++)
        printf("\nalpha %d: %s",s+1,alpha[s]);

printf("\nno of beta: %d",y);
for(s=0;s<y;s++)
        printf("\nbeta %d: %s",s+1,beta[s]);*/



if(flag==1)
{
        printf("\nThe new productions are: \n");
        for(s=0;s<y;s++)  //y--no of beta
        {
                strcpy(new_str,"\0");
                strncat(new_str,&ch,1);
                strcat(new_str,"-->");
                strcat(new_str,beta[s]);
                strncat(new_str,&ch,1);
                strcat(new_str,"'");
                printf("\n%s\n",new_str);
```

```
        }
        for(s=0;s<x;s++)                //x--no of alpha
        {
                strcpy(new_str,"\0");
                strncat(new_str,&ch,1);
                strcat(new_str,"'");
                strcat(new_str,"-->");
                strcat(new_str,alpha[s]);
                strncat(new_str,&ch,1);
                strcat(new_str,"'");
                printf("\n%s\n",new_str);
        }
        if(x>0)
        {
                strcpy(new_str,"\0");
                strncat(new_str,&ch,1);
                strcat(new_str,"'");
                strcat(new_str,"-->");
                strcat(new_str,"ε");
                printf("\n%s\n",new_str);
        }
}
else if (flag==0) //no left recursion
```

```
        {
                printf("\nNo left recursion\n");
        }
}
}
```

OUTPUT: