11th February 2022

Experiment 4 : Elimination of left Recursion and left Factoring.

4a. Aim : To eliminate left recursion from given set of grammar rules.

Algorithm :

1. Start
2. Get input grammar rule from user in the form of "A = A b | a".
3. Print first character "A"
4. Print all characters from input till we find '|'.
5. Print first character with apostrophe "A'" This prints rule 2.
6. For rule 1, print first character and all characters after '|'.
7. Stop. no left recursion found, print no left recursion
8. Stop.

MANUAL WORKING

Elimination of Left Recursion

Left recursion is of the form :

$$A \longrightarrow Aa \mid b$$

Elimination rules :

RULE 1 : $A \longrightarrow bA'$

RULE 2 : $A' \longrightarrow aA' \mid \epsilon$

Sashrika Surya
RA1911027010092
Section: N2
Lab Batch: 1
Date: 11th February 2022

Compiler Design Lab
Experiment 4a: Elimination of Left Recursion

Code:

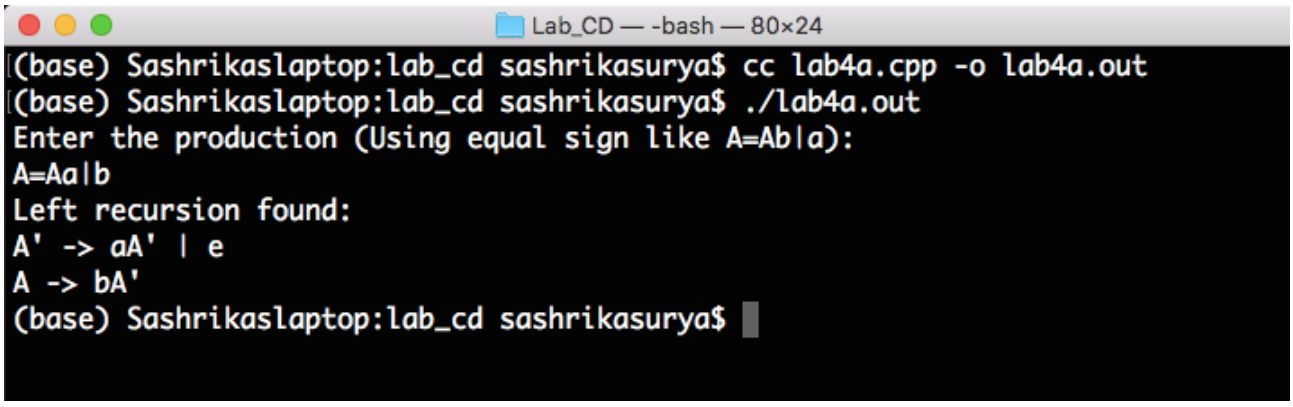
Lab 4a: Elimination of Left Recursion

```
// Elimination of Left recursion

// Left recursion: A -> Aa|b
// Elimination rules:
// Rule 1: A -> bA'
// Rule 2: A' -> aA' | e

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

int main()
{
   int i=0,k=0,n=0;
   char c,ch;
   char a[10];
   printf("Enter the production (Using equal sign like A=Ab|a): \n");
   scanf("%s%c",a,&ch);

   c=a[0];
   if(a[2] == c)
   {
      printf("Left recursion found: \n");
      // Rule 2
      printf("%c' -> ",c);
      // Search all characters till we find |
      for(k=3;k<strlen(a) && a[k] != '|';k++)
      {
         printf("%c",a[k]);
      }
      printf("%c' | e",c);
```

```
        // Rule 1
        n=k; i=0;
        printf("\n%c -> ",c);
        for(k=n+1; k<strlen(a) && a[k]!='\0'; k++)
        {
           printf("%c",a[k]);
        }
        printf("%c'\n",c);
    }
    else
    {
        printf("No left recursion\n");
    }
    return 0;
}
```

Output:

Lab 4a:



Result:
Hence, Left Recursion was successfully found and eliminated from productions by the program.