**ABHIJIT DASH**

**RA1811031010112**

**ELIMINATION OF LEFT RECURSSION**

**AIM:-**

To write a program to eliminate the left recursion from a Grammar.

**CODE:-**

#include<iostream>

#include<string>

using namespace std;

int main()

{

string ip,op1,op2,temp;

int sizes[10] = {};

char c;

int n,j,l;

cout<<"Enter the Parent Non-Terminal : ";

cin>>c;

ip.push\_back(c);

op1 += ip + "\'->";

ip += "->";

op2+=ip;

cout<<"Enter the number of productions : ";

cin>>n;

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

{

cout<<"Enter Production "<<i+1<<" : ";

cin>>temp;

sizes[i] = temp.size();

ip+=temp;

if(i!=n-1)

ip += "|";

}

cout<<"Production Rule : "<<ip<<endl;

for(int i=0,k=3;i<n;i++)

{

if(ip[0] == ip[k])

{

cout<<"Production "<<i+1<<" has left recursion."<<endl;

if(ip[k] != '#')

{

for(l=k+1;l<k+sizes[i];l++)

op1.push\_back(ip[l]);

k=l+1;

op1.push\_back(ip[0]);

op1 += "\'|";

}

}

else

{

cout<<"Production "<<i+1<<" does not have left recursion."<<endl;

if(ip[k] != '#')

{

for(j=k;j<k+sizes[i];j++)

op2.push\_back(ip[j]);

k=j+1;

op2.push\_back(ip[0]);

op2 += "\'|";

}

else

{

op2.push\_back(ip[0]);

op2 += "\'";

}

}

}

op1 += "#";

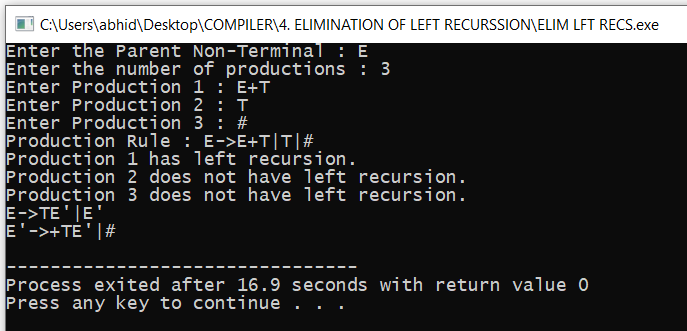
cout<<op2<<endl;

cout<<op1<<endl;

return 0;

}

**OUTPUT SCREENSHOTS:-**



**RESULT:-**

The code has been run and the output is successfully been produced.