Saket Kumar Baranwal

RA1911003010414 G1 **LEFT FACTORIAL**

Left factoring is a process by which the grammar with common prefixes is transformed

to make it useful for Top down parsers.

CODE:-

#include <iostream>

#include <string>

using namespace std;

int main()

{

int n,j,l,i,m;

int len[10] = {};

string a, b1, b2, flag;

char c;

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

cin >> c;

a.push\_back(c);

b1 += a + "\'->";

b2 += a + "\'\'->";;

a += "->";

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

cin >> n;

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

{

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

cin >> flag;

len[i] = flag.size();

a += flag;

if (i != n - 1)

{

a += "|";

}

}

cout << "The Production Rule is : " << a << endl;

char x = a[3];

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

{

if (x != a[m])

{

while (a[m++] != '|');

}

else

{

if (a[m + 1] != '|')

{

b1 += "|" + a.substr(m + 1, len[i] - 1);

a.erase(m - 1, len[i] + 1);

}

else

{

b1 += "#";

a.insert(m + 1, 1, a[0]);

a.insert(m + 2, 1, '\'');

m += 4;

}

}

}

char y = b1[6];

for (i = 0, m = 6; i < n - 1; i++)

{

if (y == b1[m])

{

if (b1[m + 1] != '|')

{

flag.clear();

for (int s = m + 1; s < b1.length(); s++)

{

flag.push\_back(b1[s]);

}

b2 += "|" + flag;

b1.erase(m - 1, flag.length() + 2);

}

else

{

b1.insert(m + 1, 1, b1[0]);

b1.insert(m + 2, 2, '\'');

b2 += "#";

m += 5;

}

}

}

b2.erase(b2.size() - 1);

cout << "After Left Factoring : " << endl;

cout << a << endl;

cout << b1 << endl;

cout << b2 << endl;

return 0;

}

OUTPUT:-

**Enter the Parent Non-Terminal : L**

**Enter the number of productions : 4**

**Enter Production 1 : i**

**Enter Production 2 : iL**

**Enter Production 3 : (L)**

**Enter Production 4 : iL+L**

**Production Rule : L->i|iL|(L)|iL+L**

**After Left Factoring :**

**L->iL'|(L)**

**L'->#|LL''**