18CSC304J/ Compiler Design

Submitted By:- ANANNYA P. NEOG (RA1911003010367)

Exp-4: Elimination of Ambiguity, Left Recursion and Left Factoring

Aim:- To write code for elimination of Ambiguity, Left Recursion and Left Factoring

Codes:-

- **Elimination of Left Recursion:**

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
   int n, j, l, i, k;
   int length[10] = {};
   string d, a, b, flag;
   char c;
   cout<<"Enter Parent Non-Terminal: ";
   cin >> c;
   d.push_back(c);
   a += d + "\'->";
   d += "->";
   b += d;
   cout<<"Enter productions: ";
   cin >> n;
   for (int i = 0; i < n; i++)
   {
      cout<<"Enter Production ";
      cout<<i + 1<<" :";
      cin >> flag;
      length[i] = flag.size();
      d += flag;
      if (i != n - 1)
      {
         d += "|";
      }
   }
   cout<<"The Production Rule is: ";
   cout<<d<<endl;
   for (i = 0, k = 3; i < n; i++)
   {
      if (d[0] != d[k])
      {
         cout<<"Production: "<< i + 1;
         cout<<" does not have left recursion.";
         cout<<endl;
         if (d[k] == '#')
         {
            b.push_back(d[0]);
            b += "\'";
         }
```

```
            else
            {
               for (j = k; j < k + length[i]; j++)
               {
               b.push_back(d[j]);
               }
               k = j + 1;
               b.push_back(d[0]);
               b += "\'|";
            }
         }
         else
         {
            cout<<"Production: "<< i + 1 ;
            cout<< " has left recursion";
            cout<< endl;
            if (d[k] != '#')
            {
               for (l = k + 1; l < k + length[i]; l++)
               {
                  a.push_back(d[l]);
               }
               k = l + 1;
               a.push_back(d[0]);
               a += "\'|";
            }
         }
      }
      a += "#";
      cout << b << endl;
      cout << a << endl;
      return 0;
   }
```

Run | Debug | Stop | Share | Save | { } Beautify | Language C++

main.cpp

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
    int n, j, l, i, k;
    int length[10] = {};
    string d, a, b, flag;
    char c;
    cout<<"Enter Parent Non-Terminal: ";
    cin >> c;
    d.push_back(c);
    a += d + "\'->";
    d += "->";
    b += d;
    cout<<"Enter productions: ";
    cin >> n;
    for (int i = 0; i < n; i++)
    {
        cout<<"Enter Production ";
        cout<<i + 1<<" :";
        cin >> flag;
        length[i] = flag.size();
        d += flag;
        if (i != n - 1)
        {
            d += "|";
        }
    }
    cout<<"The Production Rule is: ";
    cout<<d<<endl;
    for (i = 0, k = 3; i < n; i++)
    {
```

```
    {
        if (d[0] != d[k])
        {
            cout<<"Production: "<< i + 1;
            cout<<" does not have left recursion.";
            cout<<endl;
            if (d[k] == '#')
            {
                b.push_back(d[0]);
                b += "\'";
            }
            else
            {
                for (j = k; j < k + length[i]; j++)
                {
                b.push_back(d[j]);
                }
                k = j + 1;
                b.push_back(d[0]);
                b += "\'|";
            }
        }
        else
        {
            cout<<"Production: "<< i + 1 ;
            cout<< " has left recursion";
            cout<< endl;
            if (d[k] != '#')
            {
                for (l = k + 1; l < k + length[i]; l++)
                {
                    a.push_back(d[l]);
                }
                k = l + 1;
                a.push_back(d[0]);
                a += "\'|";
            }
        }
    }
    a += "#";
    cout << b << endl;
    cout << a << endl;
    return 0;
}
```

Output:-

```
input
Enter Parent Non-Terminal: A
Enter productions: 2
Enter Production 1 :A+B
Enter Production 2 :B
The Production Rule is: A->A+B|B
Production: 1 has left recursion
Production: 2 does not have left recursion.
A->BA'|
A'->+BA'|#


...Program finished with exit code 0
Press ENTER to exit console.
```

- ## Left Factoring:

```
#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 << "\nAfter Left Factoring : " << endl;
    cout << a << endl;
    cout << b1 << endl;
    cout << b2 << endl;
    return 0;
}
```

Run | Debug | Stop | Share | Save | { } Beautify | Language C++

main.cpp

```cpp
#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++] != '|');
        }
```

Run | Debug | Stop | Share | Save | { } Beautify | Language C++

main.cpp

```
        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 << "\nAfter Left Factoring : " << endl;
    cout << a << endl;
    cout << b1 << endl;
    cout << b2 << endl;
    return 0;
}
```

Output:-

input

```
Enter the Parent Non-Terminal : a
Enter total number of productions : 4
Enter the Production 1 : a
Enter the Production 2 : aX
Enter the Production 3 : (X)
Enter the Production 4 : aX+X
The Production Rule is : a->a|aX|(X)|aX+X

After Left Factoring :
a->aa'|(X)
a'->#|Xa''
a''->#|+X


...Program finished with exit code 0
Press ENTER to exit console.
```