Compiler Design  
Lab Exercise 4

short line

1. **Aim: Implementation of Left Recursion**

**Algorithm:**

1. Start
2. Get the input from the user.
3. Initialise string vectors for Non-Terminals and Left Recursion equation.
4. Feed in the terminals and the production of that terminal.
5. Display the output
6. Compare with manual output

**Code:(Language : C++)**

#include <iostream>

#include <vector>

#include <string>

using namespace std;

int main()

{

    int n;

    cout<<"\nEnter number of non terminals: ";

    cin>>n;

    cout<<"\nEnter non terminals one by one: ";

    int i;

    vector<string> nonter(n);

    vector<int> leftrecr(n,0);

    for(i=0;i<n;++i) {

            cout<<"\nNon terminal "<<i+1<<" : ";

        cin>>nonter[i];

    }

    vector<vector<string> > prod;

    cout<<"\nEnter 'esp' for null";

    for(i=0;i<n;++i) {

        cout<<"\nNumber of "<<nonter[i]<<" productions: ";

        int k;

        cin>>k;

        int j;

        cout<<"\nOne by one enter all "<<nonter[i]<<" productions";

        vector<string> temp(k);

        for(j=0;j<k;++j) {

            cout<<"\nRHS of production "<<j+1<<": ";

            string abc;

            cin>>abc;

            temp[j]=abc;

            if(nonter[i].length()<=abc.length()&&nonter[i].compare(abc.substr(0,nonter[i].length()))==0)

                leftrecr[i]=1;

        }

        prod.push\_back(temp);

    }

    for(i=0;i<n;++i) {

        cout<<leftrecr[i];

    }

    for(i=0;i<n;++i) {

        if(leftrecr[i]==0)

            continue;

        int j;

        nonter.push\_back(nonter[i]+"'");

        vector<string> temp;

        for(j=0;j<prod[i].size();++j) {

            if(nonter[i].length()<=prod[i][j].length()&&nonter[i].compare(prod[i][j].substr(0,nonter[i].length()))==0) {

                string abc=prod[i][j].substr(nonter[i].length(),prod[i][j].length()-nonter[i].length())+nonter[i]+"'";

                temp.push\_back(abc);

                prod[i].erase(prod[i].begin()+j);

                --j;

            }

            else {

                prod[i][j]+=nonter[i]+"'";

            }

        }

        temp.push\_back("esp");

        prod.push\_back(temp);

    }

    cout<<"\n\n";

    cout<<"\nNew set of non-terminals: ";

    for(i=0;i<nonter.size();++i)

        cout<<nonter[i]<<" ";

    cout<<"\n\nNew set of productions: ";

    for(i=0;i<nonter.size();++i) {

        int j;

        for(j=0;j<prod[i].size();++j) {

            cout<<"\n"<<nonter[i]<<" -> "<<prod[i][j];

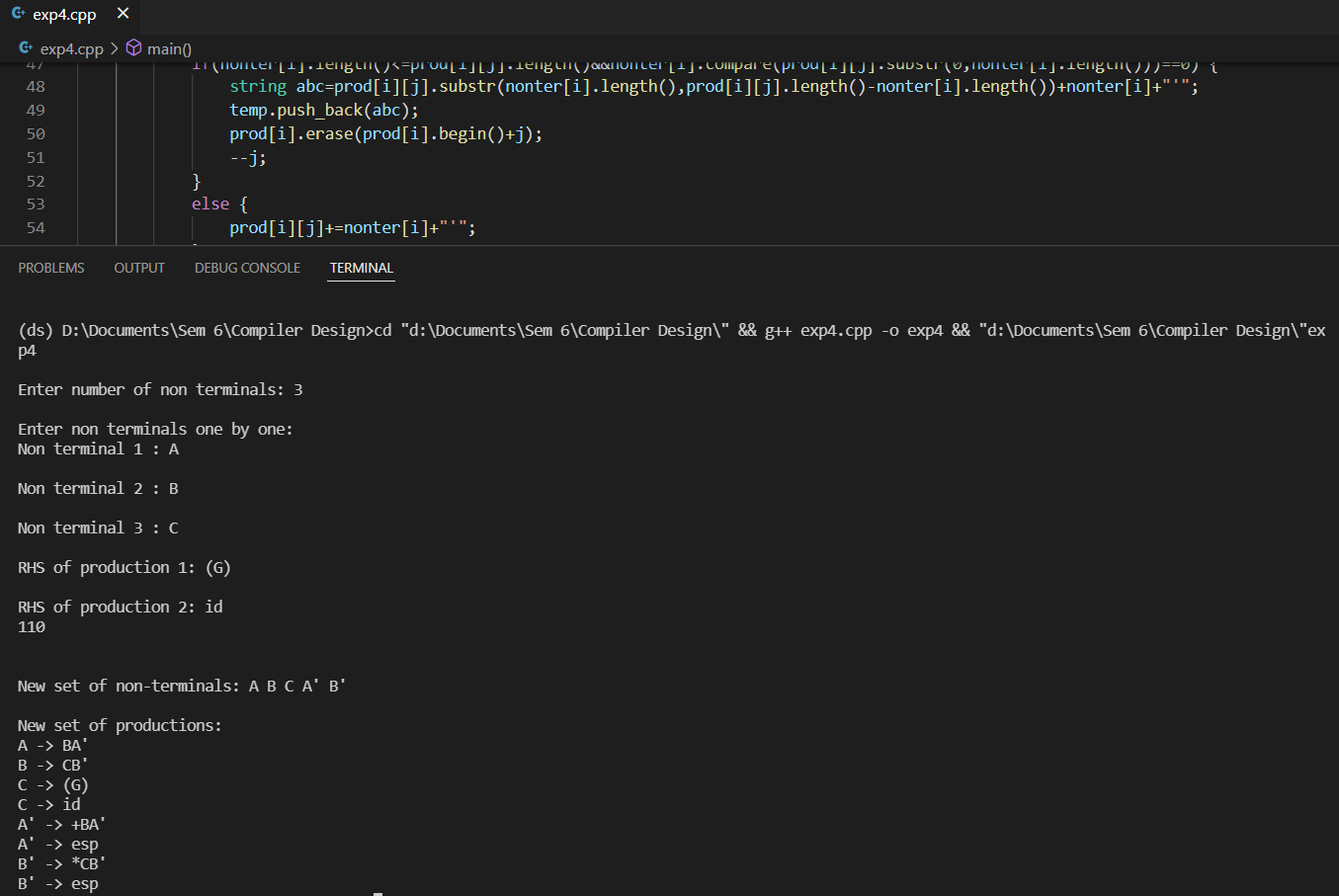
        }

    }

    return 0;

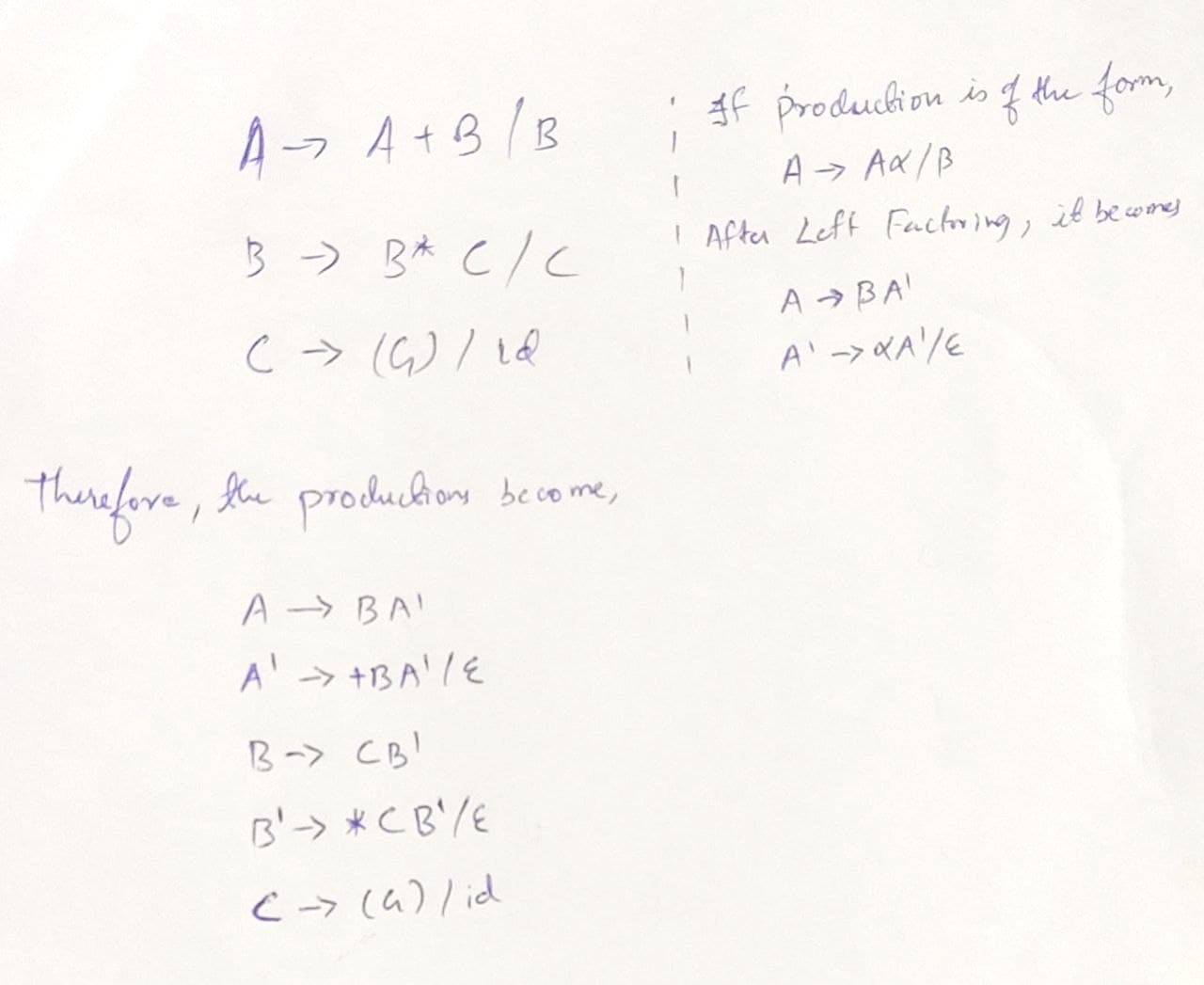
}

**Output:**



**Manual Output:**

We compare the production output from code as well as handwritten and deduce that they are the same for the given production.



**Result:** Implementation of Left Recursion has been completed and verified by Manual Output.

1. **Aim: Implementation of Left Factoring**

**Algorithm:**

1. Start
2. Get the input from the user.
3. Initialise character arrays for input, process and output.
4. Feed in the input production.
5. Display the output
6. Compare with manual output

**Code:(Language : C)**

#include<stdio.h>

#include<string.h>

  int main()

  {

       Char gram[20], part1[20], part2[20], modifiedGram[20], newGram[20], tempGram[20];

       int i,j=0,k=0,l=0,pos;

       printf("Enter Production : A->");

       gets(gram);

       for(i=0;gram[i]!='|';i++,j++)

            part1[j]=gram[i];

       part1[j]='\0';

       for(j=++i,i=0;gram[j]!='\0';j++,i++)

            part2[i]=gram[j];

       part2[i]='\0';

       for(i=0;i<strlen(part1)||i<strlen(part2);i++)

       {

            if(part1[i]==part2[i])

            {

                 modifiedGram[k]=part1[i];

                 k++;

                 pos=i+1;

            }

       }

       for(i=pos,j=0;part1[i]!='\0';i++,j++){

            newGram[j]=part1[i];

       }

       newGram[j++]='|';

       for(i=pos;part2[i]!='\0';i++,j++){

            newGram[j]=part2[i];

       }

       modifiedGram[k]='X';

       modifiedGram[++k]='\0';

       newGram[j]='\0';

       printf("\n A->%s",modifiedGram);

       printf("\n X->%s\n",newGram);

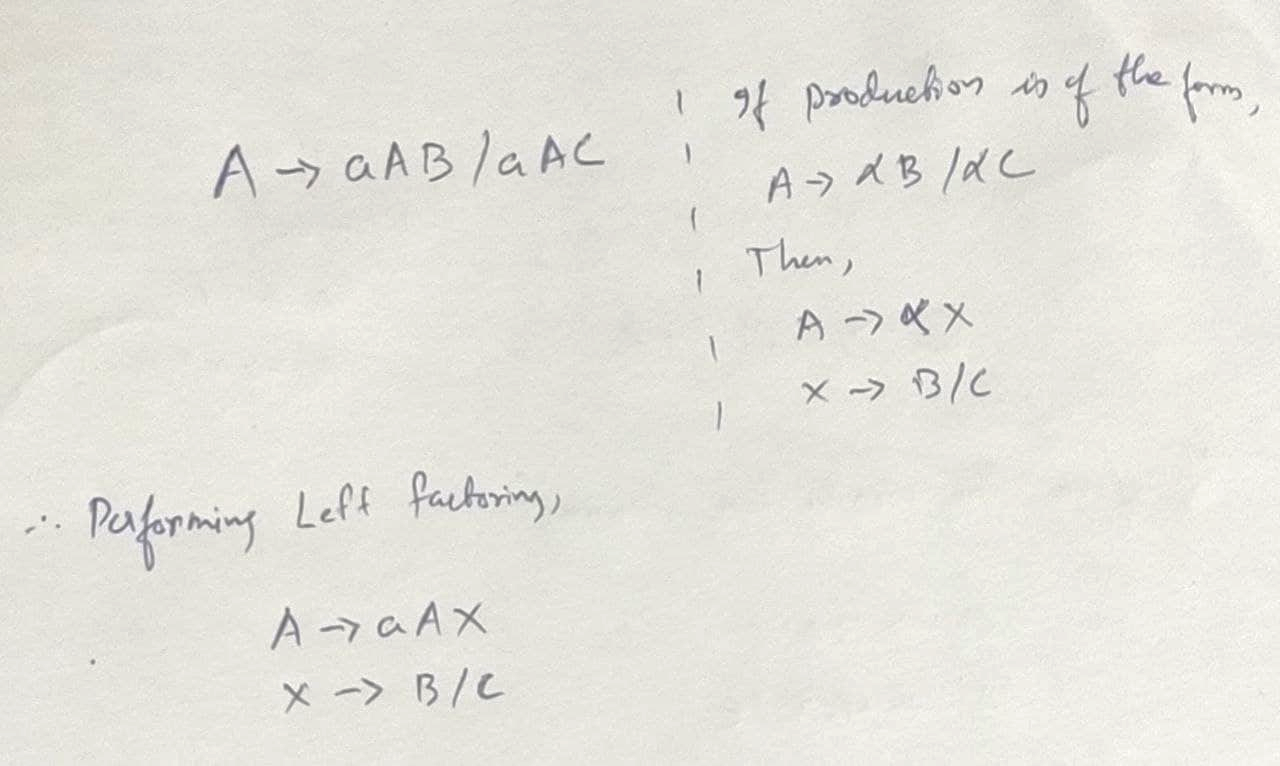
}

**Output:**

****

**Manual Output:**

We compare the production output from code as well as handwritten and deduce that they are the same for the given production.



**Result:** Implementation of Left Factoring has been completed and verified by Manual Output.