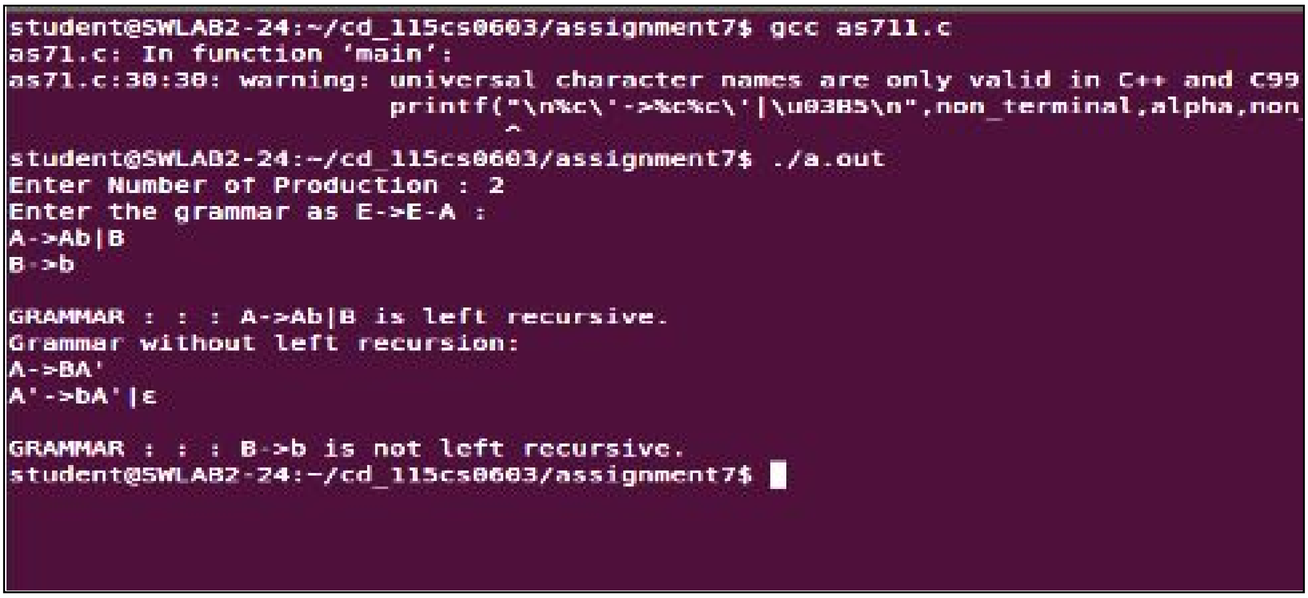
Assignment 7

Q1. Write a C program to eliminate left recursionin a given grammar.

PROGRAM:

```
#include<stdio.h>
#include<string.h>
#define SIZE 10
  int main () {
    char non_terminal;
    char beta,alpha;
    int num;
    int i;
    char production[10][SIZE];
    int index=3; /* starting of the string following "->" */
    printf("Enter Number of Production : ");
    scanf("%d",&num);
    printf("Enter the grammar as E->E-A :\n");
    for(i=0;i<num;i++){
          scanf("%s",production[i]);
    }
    for(i=0;i<num;i++){
          printf("\nGRAMMAR : : : %s",production[i]);
          non_terminal=production[i][0];
          if(non_terminal==production[i][index]) {
                alpha=production[i][index+1];
                printf(" is left recursive.\n");
                while(production[i][index]!=0 && production[i][index]!='|')  {
          index++;  }
                if(production[i][index]!=0)  {
                      beta=production[i][index+1];
                      printf("Grammar without left recursion:\n");
                      printf("%c->%c%c\'",non_terminal,beta,non_terminal);
                      printf("\n%c\'->%c%c\'|^\n",non_terminal,alpha,non_terminal);
                      }
                else
                      printf(" can't be reduced\n");
          }
          else
                printf(" is not left recursive.\n");
          index=3;
    }
    }
```



```
student@SWLAB2-24:~/cd_115cs0603/assignment7$ gcc as711.c
as71.c: In function ‘main’:
as71.c:30:30: warning: universal character names are only valid in C++ and C99
                           printf("\n%c\'->%c%c\'|\u03B5\n",non_terminal,alpha,non_
                                  ^
student@SWLAB2-24:~/cd_115cs0603/assignment7$ ./a.out
Enter Number of Production : 2
Enter the grammar as E->E-A :
A->Ab|B
B->b

GRAMMAR : : : A->Ab|B is left recursive.
Grammar without left recursion:
A->BA'
A'->bA'|ε

GRAMMAR : : : B->b is not left recursive.
student@SWLAB2-24:~/cd_115cs0603/assignment7$ 
```

Q2. Write a C program to eliminate left factoring in a given grammar.

PROGRAM:

```
#include<iostream>
#include<stdio.h>
#include<string.h>
using namespace std;
struct production
{
         char lf;
         char rt[10];
         int prod_rear;
         int fl;
};
struct production prodn[20],prodn_new[20];   //Creation of object
int b=-1,d,f,q,n,m=0,c=0;
char terminal[20],nonterm[20],alpha[10],extra[10];
char epsilon='^';
int main()
{
   cout<<"\nEnter the number of Special characters(except non-terminals): ";
   cin>>q;
   cout<<"Enter the special characters for your production: ";
   for(int cnt=0;cnt<q;cnt++)
   {
    cin>>alpha[cnt];
   }

   cout<<"\nEnter the number of productions: ";
   cin>>n;
   for(cnt=0;cnt<=n-1;cnt++)
   {
    cout<<"Enter the "<< cnt+1<<" production: ";
    cin>>prodn[cnt].lf;
    cout<<"->";
    cin>>prodn[cnt].rt;
    prodn[cnt].prod_rear=strlen(prodn[cnt].rt);
    prodn[cnt].fl=0;
   }
 for(int cnt1=0;cnt1<n;cnt1++)
   {
    for(int cnt2=cnt1+1;cnt2<n;cnt2++)
    {
         if(prodn[cnt1].lf==prodn[cnt2].lf)
         {
               cnt=0;
               int p=-1;
               while((prodn[cnt1].rt[cnt]!='\0')&&(prodn[cnt2].rt[cnt]!='\0'))
               {
               if(prodn[cnt1].rt[cnt]==prodn[cnt2].rt[cnt])
               {
               extra[++p]=prodn[cnt1].rt[cnt];
               prodn[cnt1].fl=1;
               prodn[cnt2].fl=1;
               }
               else
               {
               if(p==-1)
                           break;
               else
               {
```

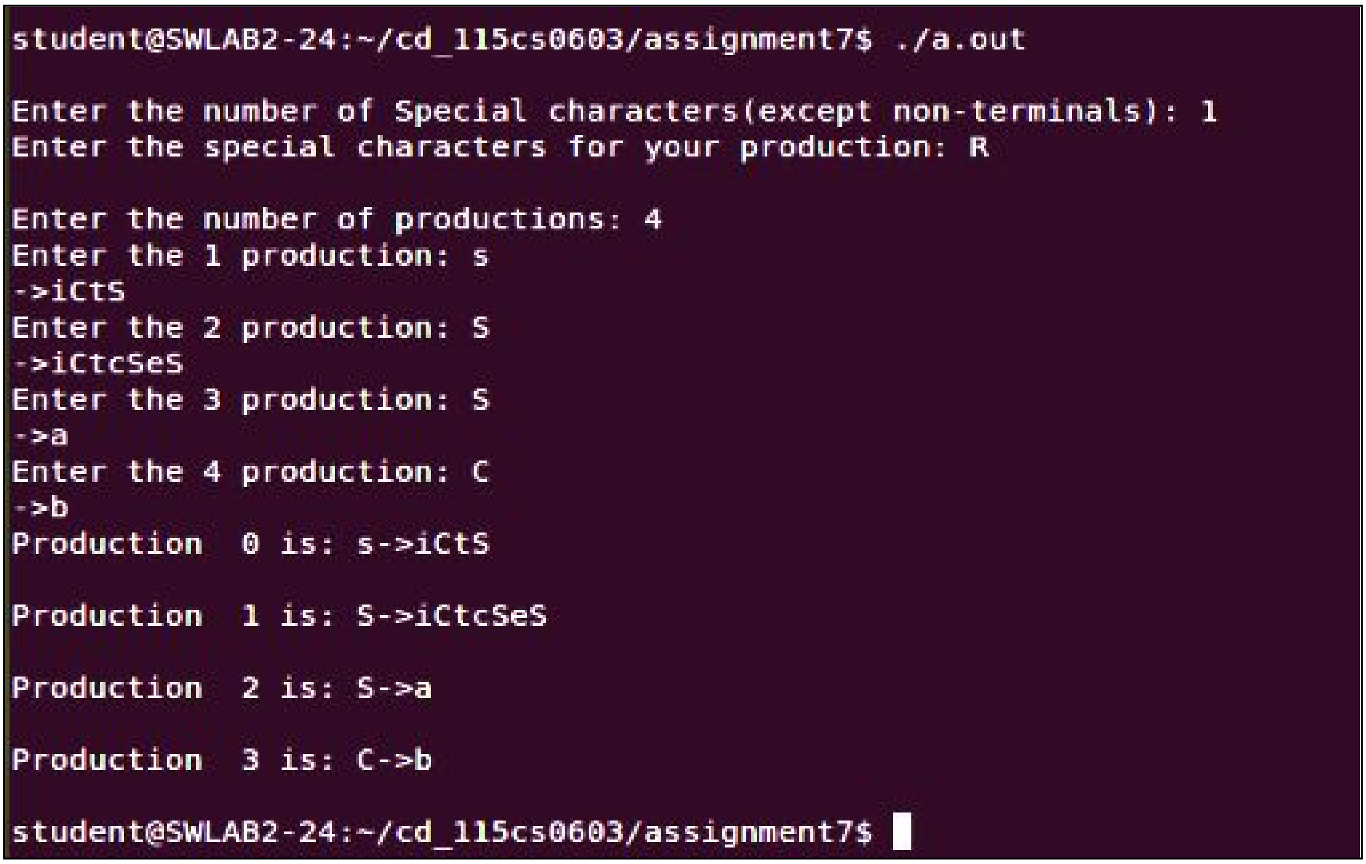
```
                        int h=0,u=0;
                        prodn_new[++b].lf=prodn[cnt1].lf;
                        strcpy(prodn_new[b].rt,extra);
                        prodn_new[b].rt[p+1]=alpha[c];
                        prodn_new[++b].lf=alpha[c];
                        for(int g=cnt;g<prodn[cnt2].prod_rear;g++)
                        prodn_new[b].rt[h++]=prodn[cnt2].rt[g];
                        prodn_new[++b].lf=alpha[c];
                        for(g=cnt;g<=prodn[cnt1].prod_rear;g++)
                        prodn_new[b].rt[u++]=prodn[cnt1].rt[g];
                        m=1;
                        break;
            }
            }
            cnt++;
            }
            if((prodn[cnt1].rt[cnt]==0)&&(m==0))
            {
                  int h=0;
                  prodn_new[++b].lf=prodn[cnt1].lf;
                  strcpy(prodn_new[b].rt,extra);
                  prodn_new[b].rt[p+1]=alpha[c];
                  prodn_new[++b].lf=alpha[c];
                  prodn_new[b].rt[0]=epsilon;
                  prodn_new[++b].lf=alpha[c];
                  for(int g=cnt;g<prodn[cnt2].prod_rear;g++)
                  prodn_new[b].rt[h++]=prodn[cnt2].rt[g];
            }
            if((prodn[cnt2].rt[cnt]==0)&&(m==0))
            {
            int h=0;
            prodn_new[++b].lf=prodn[cnt1].lf;
            strcpy(prodn_new[b].rt,extra);
            prodn_new[b].rt[p+1]=alpha[c];
            prodn_new[++b].lf=alpha[c];
            prodn_new[b].rt[0]=epsilon;
            prodn_new[++b].lf=alpha[c];
            for(int g=cnt;g<prodn[cnt1].prod_rear;g++)
            prodn_new[b].rt[h++]=prodn[cnt1].rt[g];
            }
            c++;
            m=0;
      }
}
}
for(int cnt3=0;cnt3<=b;cnt3++)
      {
                  cout<<"Production "<<cnt3+1<<" is: ";
                  cout<<prodn_new[cnt3].lf;
                  cout<<"->";
                  cout<<prodn_new[cnt3].rt;
                  cout<<endl<<endl;
      }

for(int cnt4=0;cnt4<n;cnt4++)
{
if(prodn[cnt4].fl==0)
{
cout<<"Production  "<<cnt3++<<" is: ";
cout<<prodn[cnt4].lf;
cout<<"->";
cout<<prodn[cnt4].rt;
cout<<endl<<endl;
```

```
      }
    }

}
```

OUTPUT:



```
student@SWLAB2-24:~/cd_115cs0603/assignment7$ ./a.out

Enter the number of Special characters(except non-terminals): 1
Enter the special characters for your production: R

Enter the number of productions: 4
Enter the 1 production: s
->iCtS
Enter the 2 production: S
->iCtcSeS
Enter the 3 production: S
->a
Enter the 4 production: C
->b
Production  0 is: s->iCtS

Production  1 is: S->iCtcSeS

Production  2 is: S->a

Production  3 is: C->b

student@SWLAB2-24:~/cd_115cs0603/assignment7$ 
```