Akshat Agarwal  
RA1911003010646

**Compiler Design**

**Lab – 9**

**LR and SLR**

**LR**

**Code :**

**#include<iostream>**

**#include<conio.h>**

**#include<string.h>**

**using namespace std;**

**char prod[20][20],listofvar[26]="ABCDEFGHIJKLMNOPQR";**

**int novar=1,i=0,j=0,k=0,n=0,m=0,arr[30];**

**int noitem=0;**

**struct Grammar**

**{**

**char lhs;**

**char rhs[8];**

**}g[20],item[20],clos[20][10];**

**int isvariable(char variable)**

**{**

**for(int i=0;i<novar;i++)**

**if(g[i].lhs==variable)**

**return i+1;**

**return 0;**

**}**

**void findclosure(int z, char a)**

**{**

**int n=0,i=0,j=0,k=0,l=0;**

**for(i=0;i<arr[z];i++)**

**{**

**for(j=0;j<strlen(clos[z][i].rhs);j++)**

**{**

**if(clos[z][i].rhs[j]=='.' && clos[z][i].rhs[j+1]==a)**

**{**

**clos[noitem][n].lhs=clos[z][i].lhs;**

**strcpy(clos[noitem][n].rhs,clos[z][i].rhs);**

**char temp=clos[noitem][n].rhs[j];**

**clos[noitem][n].rhs[j]=clos[noitem][n].rhs[j+1];**

**clos[noitem][n].rhs[j+1]=temp;**

**n=n+1;**

**}**

**}**

**}**

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

**{**

**for(j=0;j<strlen(clos[noitem][i].rhs);j++)**

**{**

**if(clos[noitem][i].rhs[j]=='.' && isvariable(clos[noitem][i].rhs[j+1])>0)**

**{**

**for(k=0;k<novar;k++)**

**{**

**if(clos[noitem][i].rhs[j+1]==clos[0][k].lhs)**

**{**

**for(l=0;l<n;l++)**

**if(clos[noitem][l].lhs==clos[0][k].lhs && strcmp(clos[noitem][l].rhs,clos[0][k].rhs)==0)**

**break;**

**if(l==n)**

**{**

**clos[noitem][n].lhs=clos[0][k].lhs;**

**strcpy(clos[noitem][n].rhs,clos[0][k].rhs);**

**n=n+1;**

**}**

**}**

**}**

**}**

**}**

**}**

**arr[noitem]=n;**

**int flag=0;**

**for(i=0;i<noitem;i++)**

**{**

**if(arr[i]==n)**

**{**

**for(j=0;j<arr[i];j++)**

**{**

**int c=0;**

**for(k=0;k<arr[i];k++)**

**if(clos[noitem][k].lhs==clos[i][k].lhs && strcmp(clos[noitem][k].rhs,clos[i][k].rhs)==0)**

**c=c+1;**

**if(c==arr[i])**

**{**

**flag=1;**

**goto exit;**

**}**

**}**

**}**

**}**

**exit:;**

**if(flag==0)**

**arr[noitem++]=n;**

**}**

**int main()**

**{**

**cout<<"ENTER THE PRODUCTIONS OF THE GRAMMAR(0 TO END) :\n";**

**do**

**{**

**cin>>prod[i++];**

**}while(strcmp(prod[i-1],"0")!=0);**

**for(n=0;n<i-1;n++)**

**{**

**m=0;**

**j=novar;**

**g[novar++].lhs=prod[n][0];**

**for(k=3;k<strlen(prod[n]);k++)**

**{**

**if(prod[n][k] != '|')**

**g[j].rhs[m++]=prod[n][k];**

**if(prod[n][k]=='|')**

**{**

**g[j].rhs[m]='\0';**

**m=0;**

**j=novar;**

**g[novar++].lhs=prod[n][0];**

**}**

**}**

**}**

**for(i=0;i<26;i++)**

**if(!isvariable(listofvar[i]))**

**break;**

**g[0].lhs=listofvar[i];**

**char temp[2]={g[1].lhs,'\0'};**

**strcat(g[0].rhs,temp);**

**cout<<"\n\n augumented grammar \n";**

**for(i=0;i<novar;i++)**

**cout<<endl<<g[i].lhs<<"->"<<g[i].rhs<<" ";**

**for(i=0;i<novar;i++)**

**{**

**clos[noitem][i].lhs=g[i].lhs;**

**strcpy(clos[noitem][i].rhs,g[i].rhs);**

**if(strcmp(clos[noitem][i].rhs,"ε")==0)**

**strcpy(clos[noitem][i].rhs,".");**

**else**

**{**

**for(int j=strlen(clos[noitem][i].rhs)+1;j>=0;j--)**

**clos[noitem][i].rhs[j]=clos[noitem][i].rhs[j-1];**

**clos[noitem][i].rhs[0]='.';**

**}**

**}**

**arr[noitem++]=novar;**

**for(int z=0;z<noitem;z++)**

**{**

**char list[10];**

**int l=0;**

**for(j=0;j<arr[z];j++)**

**{**

**for(k=0;k<strlen(clos[z][j].rhs)-1;k++)**

**{**

**if(clos[z][j].rhs[k]=='.')**

**{**

**for(m=0;m<l;m++)**

**if(list[m]==clos[z][j].rhs[k+1])**

**break;**

**if(m==l)**

**list[l++]=clos[z][j].rhs[k+1];**

**}**

**}**

**}**

**for(int x=0;x<l;x++)**

**findclosure(z,list[x]);**

**}**

**cout<<"\n THE SET OF ITEMS ARE \n\n";**

**for(int z=0; z<noitem; z++)**

**{**

**cout<<"\n I"<<z<<"\n\n";**

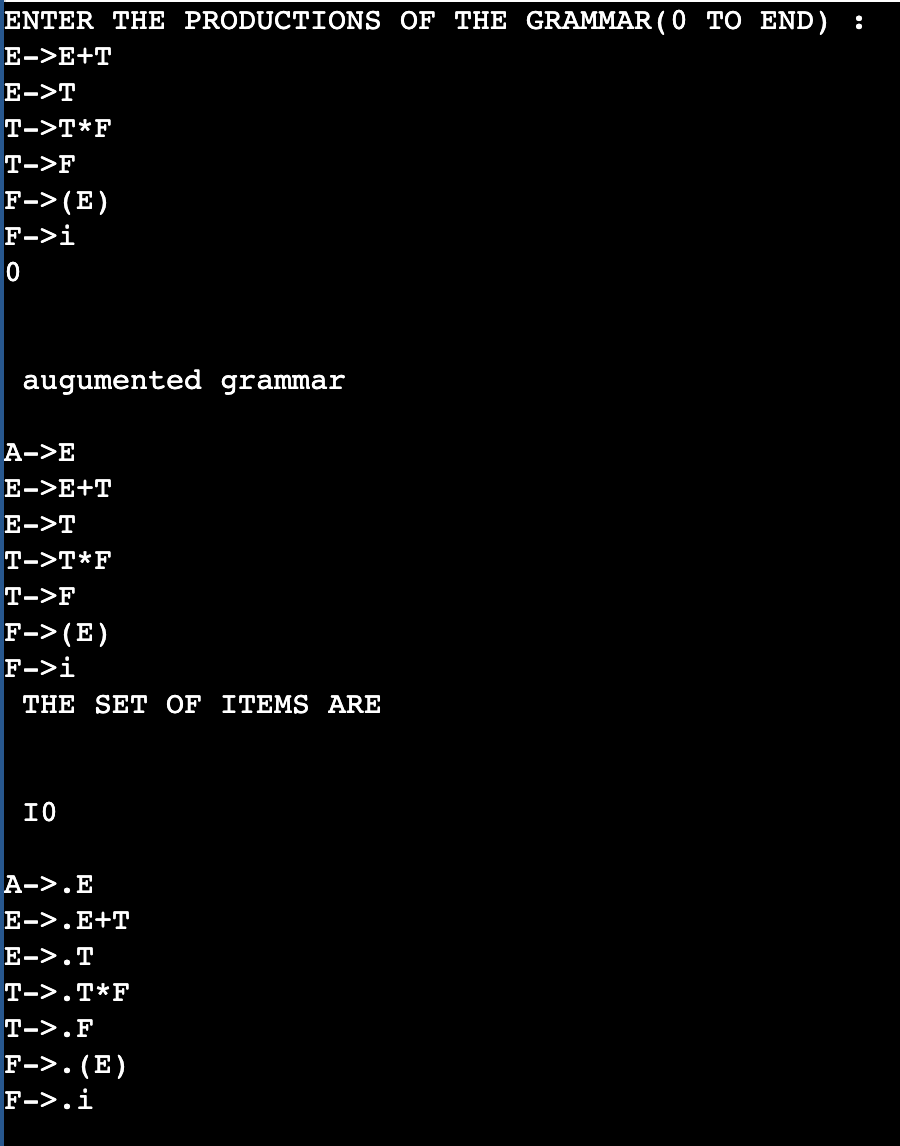
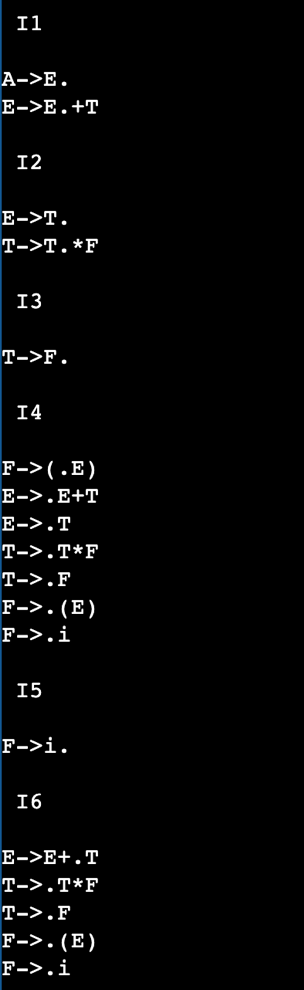
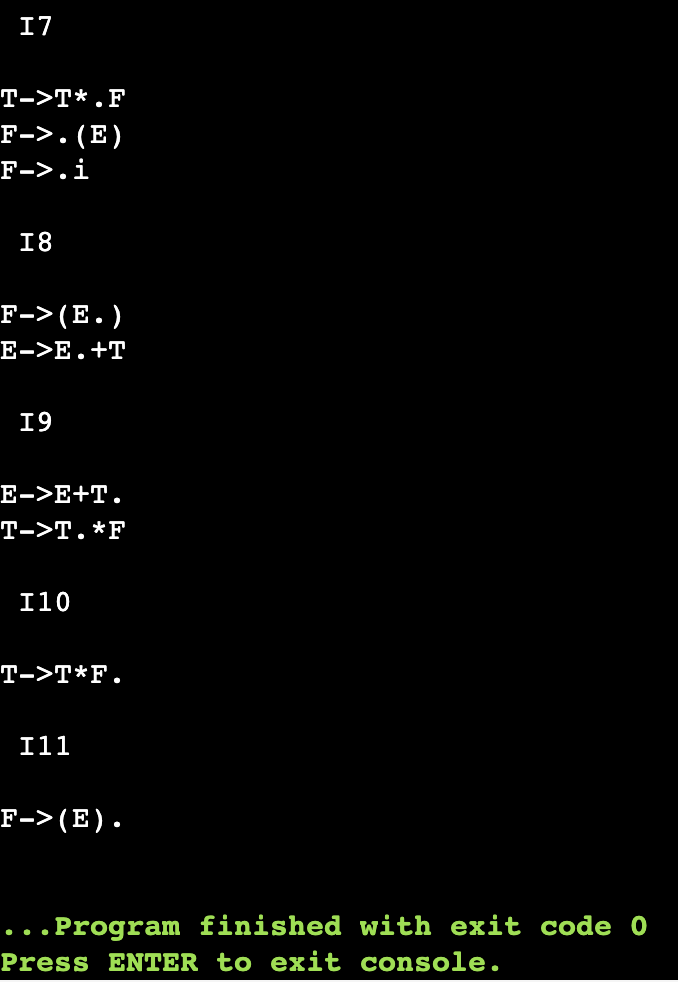
**for(j=0;j<arr[z];j++)**

**cout<<clos[z][j].lhs<<"->"<<clos[z][j].rhs<<"\n";**

**}**

**}**

**OUTPUT :**

****

**SLR**

**Code :**

#include<string.h>

#include<conio.h>

#include<stdio.h>

int axn[][6][2]={

{{100,5},{-1,-1},{-1,-1},{100,4},{-1,-1},{-1,-1}},

{{-1,-1},{100,6},{-1,-1},{-1,-1},{-1,-1},{102,102}},

{{-1,-1},{101,2},{100,7},{-1,-1},{101,2},{101,2}},

{{-1,-1},{101,4},{101,4},{-1,-1},{101,4},{101,4}},

{{100,5},{-1,-1},{-1,-1},{100,4},{-1,-1},{-1,-1}},

{{100,5},{101,6},{101,6},{-1,-1},{101,6},{101,6}},

{{100,5},{-1,-1},{-1,-1},{-1,-1},{-1,-1},{-1,-1}},

{{100,5},{-1,-1},{-1,-1},{100,4},{-1,-1},{-1,-1}},

{{-1,-1},{100,6},{-1,-1},{-1,-1},{100,11},{-1,-1}},

{{-1,-1},{101,1},{100,7},{-1,-1},{101,1},{101,1}},

{{-1,-1},{101,3},{101,3},{-1,-1},{101,3},{101,3}},

{{-1,-1},{101,5},{101,5},{-1,-1},{101,5},{101,5}}

};

int gotot[12][3]={1,2,3,-1,-1,-1,-1,-1,-1,-1,-1,-1,8,2,3,-1,-1,-1,-1,

9,3,-1,-1,10,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1};

int a[10];

char b[10];

int top=-1,btop=-1,i;

void push(int k)

{

if(top<9)

a[++top]=k;

}

void pushb(char k)

{

if(btop<9)

b[++btop]=k;

}

char TOS()

{

return a[top];

}

void pop()

{

if(top>=0)

top--;

}

void popb()

{

if(btop>=0)

b[btop--]='\0';

}

void display()

{

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

printf("%d%c",a[i],b[i]);

}

void display1(char p[],int m)

{

int l;

printf("\t\t");

for(l=m;p[l]!='\0';l++)

printf("%c",p[l]);

printf("\n");

}

void error()

{

printf("\n\nSyntax Error");

}

void reduce(int p)

{

int len,k,ad;

char src,\*dest;

switch(p)

{

case 1:dest="E+T";

src='E';

break;

case 2:dest="T";

src='E';

break;

case 3:dest="T\*F";

src='T';

break;

case 4:dest="F";

src='T';

break;

case 5:dest="(E)";

src='F';

break;

case 6:dest="i";

src='F';

break;

default:dest="\0";

src='\0';

break;

}

for(k=0;k<strlen(dest);k++)

{

pop();

popb();

}

pushb(src);

switch(src)

{

case 'E': ad=0;

break;

case 'T': ad=1;

break;

case 'F': ad=2;

break;

default: ad=-1;

break;

}

push(gotot[TOS()][ad]);

}

int main()

{

int j,st,ic;

char ip[20]="\0",an;

clrscr();

printf("Enter any String :- ");

gets(ip);

push(0);

display();

printf("\t%s\n",ip);

for(j=0;ip[j]!='\0';)

{

st=TOS();

an=ip[j];

if(an>='a'&an<='z')

ic=0;

else if(an=='+')

ic=1;

else if(an=='\*')

ic=2;

else if(an=='(')

ic=3;

else if(an==')')

ic=4;

else if(an=='$')

ic=5;

else

{

error();

break;

}

if(axn[st][ic][0]==100)

{

pushb(an);

push(axn[st][ic][1]);

display();

j++;

display1(ip,j);

}

if(axn[st][ic][0]==101)

{

reduce(axn[st][ic][1]);

display();

display1(ip,j);

}

if(axn[st][ic][1]==102)

{

printf("Given String is Accepted");

break;

}

}

getch();

return 0;

}

**OUTPUT :**

