**PROGRAM 5: WAP TO IMPLEMENT A REGULAR GRAMMAR. THE PROGRAM SHOULD READ A R.G. THROUGH A FILE AND SHOULD CHECK WHETHER A STRING GIVEN FROM THE CONSOLE IS ACCEPTABLE BY THE R.G. OR NOT.**

**CODE:**

/\* C program to check given grammar is Regular Grammar or not. \*/

#include<stdio.h>

#include<string.h>

int i,j,k,l,m,n=0,o,p,nv,z=0,t,x=0;

char str[10],temp[10],temp2[10],temp3[10];

struct prod

{

char lhs[10],rhs[10][10];

int n;

}pro[10];

void findter()

{

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

{

if(temp[i]==pro[k].lhs[0])

{

for(t=0;t<pro[k].n;t++)

{

for(x=0;x<10;x++)

temp2[x]='\0';

for(l=i+1;l<strlen(temp);l++)

temp2[l-i-1]=temp[l];

temp[i]='\0';

for(l=0;l<strlen(pro[k].rhs[t]);l++)

temp[i+l]=pro[k].rhs[t][l];

strcat(temp,temp2);

if(str[i]==temp[i])

return;

}

}

}

}

void main()

{

FILE \*f;

for(i=0;i<10;i++)

pro[i].n=0;

f=fopen("tab3.txt","r");

while(!feof(f))

{

fscanf(f,"%s",pro[n].lhs);

if(n>0)

{

if( strcmp(pro[n].lhs,pro[n-1].lhs) == 0 )

{

pro[n].lhs[0]='\0';

fscanf(f,"%s",pro[n-1].rhs[pro[n-1].n]);

pro[n-1].n++;

continue;

}

}

fscanf(f,"%s",pro[n].rhs[pro[n].n]);

pro[n].n++;

n++;

}

n--;

printf("\n\nTHE GRAMMAR IS AS FOLLOWS\n\n");

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

for(j=0;j<pro[i].n;j++)

printf("%s -> %s\n",pro[i].lhs,pro[i].rhs[j]);

o=0;

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

{

for(j=0;j<pro[i].n;j++)

if( pro[i].rhs[j][0]>=65 && pro[i].rhs[j][0]<=90 )

{

o=1;

break;

}

if(o==1)

break;

}

if(i==n)

printf("\n\nTHE GRAMMAR is a REGULAR GRAMMAR !!!");

else

{

printf("\n\nTHE GRAMMAR is NOT a REGULAR GRAMMAR !!!");

exit(1);

}

while(1)

{

for(x=0;x<10;x++)

str[x]='\0';

printf("\n\nENTER ANY STRING ( 0 for EXIT ) : ");

scanf("%s",str);

if(str[0]=='0')

exit(1);

for(j=0;j<pro[0].n;j++)

{

for(x=0;x<10;x++)

temp[x]='\0';

strcpy(temp,pro[0].rhs[j]);

m=0;

for(i=0;i<strlen(str);i++)

{

if(str[i]==temp[i])

m++;

else if(str[i]!=temp[i] && temp[i]>=65 && temp[i]<=90)

{

findter();

if(str[i]==temp[i])

m++;

}

}

for(x=0;x<10;x++)

temp3[x]='\0';

strcpy(temp3,temp);

temp3[strlen(temp)-1]='\0';

//printf("%s",temp);

if(m==strlen(str) && strcmp(temp3,str)==0 && strlen(temp3)!=1)

{

printf("\n\nTHE STRING can be PARSED !!!");

break;

}

if(m==strlen(str) && strlen(str)==strlen(temp))

{

printf("\n\nTHE STRING can be PARSED !!!");

break;

}

}

if(j==pro[0].n)

printf("\n\nTHE STRING can NOT be PARSED !!!");

}

printf("\n\n");

}

**Input File Regular Grammar Program:**

S aS

S aB

B c

A pA

A p

**PROGRAM 6: WAP TO IMPLEMENT A CONTEXT-FREE GRAMMAR. THE PROGRAM SHOULD READ A CGG. THROUGH A FILE AND SHOULD CHECK WHETHER A STRING GIVEN FROM THE CONSOLE IS ACCEPTABLE BY THE CFG. OR NOT.**

**CODE:**

/\* C program to implement CFG(Context Free Grammar). \*/

#include<stdio.h>

#include<string.h>

int i,j,k,l,m,n=0,o,p,nv,z=0,t,x=0;

char str[10],temp[20],temp2[20],temp3[20];

struct prod

{

char lhs[10],rhs[10][10];

int n;

}pro[10];

void findter()

{

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

{

if(temp[i]==pro[k].lhs[0])

{

for(t=0;t<pro[k].n;t++)

{

for(l=0;l<20;l++)

temp2[l]='\0';

for(l=i+1;l<strlen(temp);l++)

temp2[l-i-1]=temp[l];

for(l=i;l<20;l++)

temp[l]='\0';

for(l=0;l<strlen(pro[k].rhs[t]);l++)

temp[i+l]=pro[k].rhs[t][l];

strcat(temp,temp2);

if(str[i]==temp[i])

return;

else if(str[i]!=temp[i] && temp[i]>=65 && temp[i]<=90)

break;

}

break;

}

}

if(temp[i]>=65 && temp[i]<=90)

findter();

}

void main()

{

FILE \*f;

clrscr();

for(i=0;i<10;i++)

pro[i].n=0;

f=fopen("input.txt","r");

while(!feof(f))

{

fscanf(f,"%s",pro[n].lhs);

if(n>0)

{

if( strcmp(pro[n].lhs,pro[n-1].lhs) == 0 )

{

pro[n].lhs[0]='\0';

fscanf(f,"%s",pro[n-1].rhs[pro[n-1].n]);

pro[n-1].n++;

continue;

}

}

fscanf(f,"%s",pro[n].rhs[pro[n].n]);

pro[n].n++;

n++;

}

n--;

printf("\n\nTHE GRAMMAR IS AS FOLLOWS\n\n");

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

for(j=0;j<pro[i].n;j++)

printf("%s -> %s\n",pro[i].lhs,pro[i].rhs[j]);

while(1)

{

for(l=0;l<10;l++)

str[0]=NULL;

printf("\n\nENTER ANY STRING ( 0 for EXIT ) : ");

scanf("%s",str);

if(str[0]=='0')

exit(1);

for(j=0;j<pro[0].n;j++)

{

for(l=0;l<20;l++)

temp[l]=NULL;

strcpy(temp,pro[0].rhs[j]);

m=0;

for(i=0;i<strlen(str);i++)

{

if(str[i]==temp[i])

m++;

else if(str[i]!=temp[i] && temp[i]>=65 && temp[i]<=90)

{

findter();

if(str[i]==temp[i])

m++;

}

else if( str[i]!=temp[i] && (temp[i]<65 || temp[i]>90) )

break;

}

if(m==strlen(str) && strlen(str)==strlen(temp))

{

printf("\n\nTHE STRING can be PARSED !!!");

break;

}

}

if(j==pro[0].n)

printf("\n\nTHE STRING can NOT be PARSED !!!");

}

getch();

}

**Input File For CFG Program**:

S aBaA

S AB

A Bc

B c

**PROGRAM 7: WRITE A PROGRAM TO FIND THE FIRST AND FOLLOW VALUES FOR A GIVEN CONTEXT-FREE GRAMMAR. THE PROGRAM SHOULD READ THE CFG FROM THE FILE.**

**CODE:**

/\* C program to find First and Follow in a given Grammar. \*/

#include<stdio.h>

#include<string.h>

int i,j,l,m,n=0,o,p,nv,z=0,x=0;

char str[10],temp,temp2[10],temp3[20],\*ptr;

struct prod

{

char lhs[10],rhs[10][10],ft[10],fol[10];

int n;

}pro[10];

void findter()

{

int k,t;

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

{

if(temp==pro[k].lhs[0])

{

for(t=0;t<pro[k].n;t++)

{

if( pro[k].rhs[t][0]<65 || pro[k].rhs[t][0]>90 )

pro[i].ft[strlen(pro[i].ft)]=pro[k].rhs[t][0];

else if( pro[k].rhs[t][0]>=65 && pro[k].rhs[t][0]<=90 )

{

temp=pro[k].rhs[t][0];

if(temp=='S')

pro[i].ft[strlen(pro[i].ft)]='#';

findter();

}

}

break;

}

}

}

void findfol()

{

int k,t,p1,o1,chk;

char \*ptr1;

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

{

chk=0;

for(t=0;t<pro[k].n;t++)

{

ptr1=strchr(pro[k].rhs[t],temp);

if( ptr1 )

{

p1=ptr1-pro[k].rhs[t];

if(pro[k].rhs[t][p1+1]>=65 && pro[k].rhs[t][p1+1]<=90)

{

for(o1=0;o1<n;o1++)

if(pro[o1].lhs[0]==pro[k].rhs[t][p1+1])

{

strcat(pro[i].fol,pro[o1].ft);

chk++;

}

}

else if(pro[k].rhs[t][p1+1]=='\0')

{

temp=pro[k].lhs[0];

if(pro[l].rhs[j][p]==temp)

continue;

if(temp=='S')

strcat(pro[i].fol,"$");

findfol();

chk++;

}

else

{

pro[i].fol[strlen(pro[i].fol)]=pro[k].rhs[t][p1+1];

chk++;

}

}

}

if(chk>0)

break;

}

}

void main()

{

FILE \*f;

clrscr();

for(i=0;i<10;i++)

pro[i].n=0;

f=fopen("tab5.txt","r");

while(!feof(f))

{

fscanf(f,"%s",pro[n].lhs);

if(n>0)

{

if( strcmp(pro[n].lhs,pro[n-1].lhs) == 0 )

{

pro[n].lhs[0]='\0';

fscanf(f,"%s",pro[n-1].rhs[pro[n-1].n]);

pro[n-1].n++;

continue;

}

}

fscanf(f,"%s",pro[n].rhs[pro[n].n]);

pro[n].n++;

n++;

}

printf("\n\nTHE GRAMMAR IS AS FOLLOWS\n\n");

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

for(j=0;j<pro[i].n;j++)

printf("%s -> %s\n",pro[i].lhs,pro[i].rhs[j]);

pro[0].ft[0]='#';

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

{

for(j=0;j<pro[i].n;j++)

{

if( pro[i].rhs[j][0]<65 || pro[i].rhs[j][0]>90 )

{

pro[i].ft[strlen(pro[i].ft)]=pro[i].rhs[j][0];

}

else if( pro[i].rhs[j][0]>=65 && pro[i].rhs[j][0]<=90 )

{

temp=pro[i].rhs[j][0];

if(temp=='S')

pro[i].ft[strlen(pro[i].ft)]='#';

findter();

}

}

}

printf("\n\nFIRST\n");

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

{

printf("\n%s -> ",pro[i].lhs);

for(j=0;j<strlen(pro[i].ft);j++)

{

for(l=j-1;l>=0;l--)

if(pro[i].ft[l]==pro[i].ft[j])

break;

if(l==-1)

printf("%c",pro[i].ft[j]);

}

}

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

temp2[i]=pro[i].lhs[0];

pro[0].fol[0]='$';

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

{

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

{

for(j=0;j<pro[i].n;j++)

{

ptr=strchr(pro[l].rhs[j],temp2[i]);

if( ptr )

{

p=ptr-pro[l].rhs[j];

if(pro[l].rhs[j][p+1]>=65 && pro[l].rhs[j][p+1]<=90)

{

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

if(pro[o].lhs[0]==pro[l].rhs[j][p+1])

strcat(pro[i].fol,pro[o].ft);

}

else if(pro[l].rhs[j][p+1]=='\0')

{

temp=pro[l].lhs[0];

if(pro[l].rhs[j][p]==temp)

continue;

if(temp=='S')

strcat(pro[i].fol,"$");

findfol();

}

else

pro[i].fol[strlen(pro[i].fol)]=pro[l].rhs[j][p+1];

}

}

}

}

printf("\n\nFOLLOW\n");

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

{

printf("\n%s -> ",pro[i].lhs);

for(j=0;j<strlen(pro[i].fol);j++)

{

for(l=j-1;l>=0;l--)

if(pro[i].fol[l]==pro[i].fol[j])

break;

if(l==-1)

printf("%c",pro[i].fol[j]);

}

}

getch();

}

**Input File For First and Follow Program:**

S ABE

S a

A p

A t

B Aq

S f

A w

**PROGRAM 8: WRITE A PROGRAM THAT VERIFIES WHETHER A GIVEN CFG IS SUITABLE FOR LL(1) PARSING OR NOT. IF NOT THEN THE PROGRAM SHOULD MAKE CONVERT THE GIVEN CFG TO A FORM WHICH IS SUITABLE FOR LL PARSING.**

**CODE:**

/\* C program to implement Simple LR Parser. \*/

#include<stdio.h>

#include<string.h>

int i,j,k,m,n=0,o,p,ns=0,tn=0,rr=0,ch=0;

char read[15][10],gl[15],gr[15][10],temp,templ[15],tempr[15][10],\*ptr,temp2[5],dfa[15][15];

struct states

{

char lhs[15],rhs[15][10];

int n;

}I[15];

int compstruct(struct states s1,struct states s2)

{

int t;

if(s1.n!=s2.n)

return 0;

if( strcmp(s1.lhs,s2.lhs)!=0 )

return 0;

for(t=0;t<s1.n;t++)

if( strcmp(s1.rhs[t],s2.rhs[t])!=0 )

return 0;

return 1;

}

void moreprod()

{

int r,s,t,l1=0,rr1=0;

char \*ptr1,read1[15][10];

for(r=0;r<I[ns].n;r++)

{

ptr1=strchr(I[ns].rhs[l1],'.');

t=ptr1-I[ns].rhs[l1];

if( t+1==strlen(I[ns].rhs[l1]) )

{

l1++;

continue;

}

temp=I[ns].rhs[l1][t+1];

l1++;

for(s=0;s<rr1;s++)

if( temp==read1[s][0] )

break;

if(s==rr1)

{

read1[rr1][0]=temp;

rr1++;

}

else

continue;

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

{

if(gl[s]==temp)

{

I[ns].rhs[I[ns].n][0]='.';

I[ns].rhs[I[ns].n][1]=NULL;

strcat(I[ns].rhs[I[ns].n],gr[s]);

I[ns].lhs[I[ns].n]=gl[s];

I[ns].lhs[I[ns].n+1]=NULL;

I[ns].n++;

}

}

}

}

void canonical(int l)

{

int t1;

char read1[15][10],rr1=0,\*ptr1;

for(i=0;i<I[l].n;i++)

{

temp2[0]='.';

ptr1=strchr(I[l].rhs[i],'.');

t1=ptr1-I[l].rhs[i];

if( t1+1==strlen(I[l].rhs[i]) )

continue;

temp2[1]=I[l].rhs[i][t1+1];

temp2[2]=NULL;

for(j=0;j<rr1;j++)

if( strcmp(temp2,read1[j])==0 )

break;

if(j==rr1)

{

strcpy(read1[rr1],temp2);

read1[rr1][2]=NULL;

rr1++;

}

else

continue;

for(j=0;j<I[0].n;j++)

{

ptr=strstr(I[l].rhs[j],temp2);

if( ptr )

{

templ[tn]=I[l].lhs[j];

templ[tn+1]=NULL;

strcpy(tempr[tn],I[l].rhs[j]);

tn++;

}

}

for(j=0;j<tn;j++)

{

ptr=strchr(tempr[j],'.');

p=ptr-tempr[j];

tempr[j][p]=tempr[j][p+1];

tempr[j][p+1]='.';

I[ns].lhs[I[ns].n]=templ[j];

I[ns].lhs[I[ns].n+1]=NULL;

strcpy(I[ns].rhs[I[ns].n],tempr[j]);

I[ns].n++;

}

moreprod();

for(j=0;j<ns;j++)

{

//if ( memcmp(&I[ns],&I[j],sizeof(struct states))==1 )

if( compstruct(I[ns],I[j])==1 )

{

I[ns].lhs[0]=NULL;

for(k=0;k<I[ns].n;k++)

I[ns].rhs[k][0]=NULL;

I[ns].n=0;

dfa[l][j]=temp2[1];

break;

}

}

if(j<ns)

{

tn=0;

for(j=0;j<15;j++)

{

templ[j]=NULL;

tempr[j][0]=NULL;

}

continue;

}

dfa[l][j]=temp2[1];

printf("\n\nI%d :",ns);

for(j=0;j<I[ns].n;j++)

printf("\n\t%c -> %s",I[ns].lhs[j],I[ns].rhs[j]);

getch();

ns++;

tn=0;

for(j=0;j<15;j++)

{

templ[j]=NULL;

tempr[j][0]=NULL;

}

}

}

void main()

{

FILE \*f;

int l;

clrscr();

for(i=0;i<15;i++)

{

I[i].n=0;

I[i].lhs[0]=NULL;

I[i].rhs[0][0]=NULL;

dfa[i][0]=NULL;

}

f=fopen("tab6.txt","r");

while(!feof(f))

{

fscanf(f,"%c",&gl[n]);

fscanf(f,"%s\n",gr[n]);

n++;

}

printf("THE GRAMMAR IS AS FOLLOWS\n");

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

printf("\t\t\t\t%c -> %s\n",gl[i],gr[i]);

I[0].lhs[0]='Z';

strcpy(I[0].rhs[0],".S");

I[0].n++;

l=0;

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

{

temp=I[0].rhs[l][1];

l++;

for(j=0;j<rr;j++)

if( temp==read[j][0] )

break;

if(j==rr)

{

read[rr][0]=temp;

rr++;

}

else

continue;

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

{

if(gl[j]==temp)

{

I[0].rhs[I[0].n][0]='.';

strcat(I[0].rhs[I[0].n],gr[j]);

I[0].lhs[I[0].n]=gl[j];

I[0].n++;

}

}

}

ns++;

printf("\nI%d :\n",ns-1);

for(i=0;i<I[0].n;i++)

printf("\t%c -> %s\n",I[0].lhs[i],I[0].rhs[i]);

for(l=0;l<ns;l++)

canonical(l);

printf("\n\n\t\tPRESS ANY KEY FOR DFA TABLE");

getch();

clrscr();

printf("\t\t\tDFA TABLE IS AS FOLLOWS\n\n\n");

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

{

printf("I%d : ",i);

for(j=0;j<ns;j++)

if(dfa[i][j]!='\0')

printf("'%c'->I%d | ",dfa[i][j],j);

printf("\n\n\n");

}

printf("\n\n\n\t\tPRESS ANY KEY TO EXIT");

getch();

}

**Input File For SLR Parser:**

S S+T

S T

T T\*F

T F

F (S)

F t