**Predictive Parser Table Construction**

RA1811030010023

SAYAN PRAMANICK

**CODE**

#include<stdio.h>

#include<conio.h>

#include<string.h>

int main() {

char fin[10][20],st[10][20],ft[20][20],fol[20][20];

int a=0,e,i,t,b,c,n,k,l=0,j,s,m,p;

printf("enter the no. of productions\n");

scanf("%d",&n);

printf("enter the productions in a grammar\n");

for(i=0;i<n;i++) scanf("%s",st[i]);

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

fol[i][0]='\0';

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

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

j=3; |=0; a=0;

ll:if(!((st[i][j]>64)&&(st[ i][j]<91))){

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

if(ft[i][m]==st[i][j])

goto 11;}

ft[i][l]=st[i][j];

l=l+l;

j=j+i;

}

else {

if(s>0) {

while(st[i][j] !=st[a][0])

{

a++;

}

b=0;

while(ft[a][b]l='\0'){

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

if(ft[i][m]==ft[a][b]) goto s2;

} ft[i][l]=ft[a][b];

1=1+1;

s2:b=b+l;} } }

while(st[i][j]!='\O') {

if(st[i][j]=='|') {

j=j+l; goto 11;

}

j=j+i; }

ft[i][l]='\O'; } }

printf("first pos\n"); for(i=0;i<n;i++) printf("FIRST[%c]=%s \n", st[i][O],ft[i]); fol[0][0]='$';

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

k=0; j=3;

if(i==0)

1=1;

else

1=0;

kl:while((st[i][0] !=st[k][j ])&&(k<n)) { if(st[k][j]=='\O'){

k++;j=2; }

j++; }

j=j+i;

if(st[i][O]==st[k][j-1]) {

if((st[k][j]!='|')&&(st[k][j] !='\0')){ a=0;

if(!((st[k][j]>64)&&(st[k][j]<91))){ for(m=0;m<l;m++) {

if(fol[i][m]==st[k][j]) goto q3; } fol[i][l]=st[k][j]; I++;

q3:;}

else {

while(st[k][j]!=st[a][O]){

a++;}

p=0;

while(ft[a][p]!='\0'){ if(ft[a][p]!='@'){

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

if(fol[i][m]==ft[a][p]) goto q2;

} fol[i][l]=ft[a][p];

1=1+1;

}

else e=l;

q2:p++;

}

if(e==l) {

e=0; goto al; } } }

else {

al:c=0;

a=0;

while(st[k][0]!=st[a][0]){

a++; }

while((fol[a][c]!='\0')&&( st[a][0] !=st[i][0]))

{

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

if(fol[i] [m]==fol[a][c]) goto ql;

}

fol[i][l]=fol[a][c]; I++; ql:c++;

} }

goto kl;

}fol[i][l]='\0'; }

printf("follow pos\n"); for(i=0;i<n;i++)

printf("FOLLOW[%c]=% s \n",st[i][0],fol[i]); printf("\n"); s=0;

for(i=0;i<n;i++) { j=3;

while(st[i][j] !='\0') {

if((st[i][j-l]=='|')l l(j==3)) {

for(p=0;p<=2;p++) {

fin[s][p]=st[i][p];}

t=j;

for(p=3;((st[i][j]!=T)&&( st[i][j]!='\0'));p++) {

fin[s][p]=st[i][j];j++; }

fin[s][p]='\0';

if(st[i][k]=='@') {

b=0; a=0;

while(st[a][O]!=st[i][O]){

a++; }

while(fol[a][b]!='\0'){

printf("M[%c,%c]=%s \n", st[i][O],fol[a][b],fin[s]); b++;}} else if(!((st[i][t]>64)&&(st[i][t ]<91)))

printf("M[%c,%c]=%s \n", st[i][O],st[i][t],fin[s]); else { b=0; a=0; while(st[a][0] !=st[i] [3])

{a++; }

while(ft[a][b]!='\0'){

printf("M[%c,%c]=%s \n", st[i][0],ft[a][b],fin[s]); b++;}} s++;}

if(st[i][j]==T)

j++;} }

return 0;}

**OUTPUT:**

C C:\Users\srini\Desktop\CD programs\predictive parser.exe

enter the no. of productions

r

enter the productions in a grammar

S->CC

C->a€ d

first pos

FIRST[S]=ad

FIRST[C]=ad

follow pos

FOLLOW[S]=$

FOLLOW[C]=ad$

M[S,a]=S->€€

M[S,d]=S->CC

M[CJa]=€->aC

M[CJd]=C->d

Process exited after 9.684 seconds with return value 0 Press any key to continue . . .