

Compiler Design Lab

NAME: Priyanka Srinivas

REGISTER NUMBER: RA1911026010014

SECTION: K1

Exercise 6: Predictive Parsing Table

AIM: Evaluate predictive parsing for all the productions

INTRODUCTION:

A predictive parser is a recursive descent parser with no backtracking or backup. It is a top-down parser that does not require backtracking. At each step, the choice of the rule to be expanded is made upon the next terminal symbol.

Consider - $A \rightarrow A1 \mid A2 \mid \dots \mid A_n$

If the non-terminal is to be further expanded to 'A', the rule is selected based on the current input symbol 'a' only.

ALGORITHM:

1. Input the required number of productions.
2. Specify first of each production.
3. Enter terminals and non-terminals.
4. Enter follow for the specified terminals.
5. And so enter the string to be parsed.
6. Consider a transition diagram (DFA/NFA) for every rule of grammar.
7. Optimize the DFA by reducing the number of states, yielding the final transition diagram.
8. Simulate the string on the transition diagram to parse a string.
9. If the transition diagram reaches an accepted state the input is consumed, it is parsed.

PROGRAM:

```
#include <bits/stdc++.h>
using namespace std;

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;

cout << ("enter the no. of nonterminals\n");
scanf("%d",&n);
cout << ("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;
        l=0;
        a=0;
        l1:if(!((st[i][j]>64)&&(st[i][j]<91)))
        {
            for(m=0;m<l;m++)
            {
                if(ft[i][m]==st[i][j])
                    goto s1;
            }
            ft[i][l]=st[i][j];
            l=l+1;
            s1:j=j+1;
        }
        else
        {
            if(s>0)
            {
                while(st[i][j]!=st[a][0])
                {
                    a++;
                }
                b=0;
                while(ft[a][b]!='\0')
                {

```

```

        for(m=0;m<l;m++)
        {
            if(ft[i][m]==ft[a][b])
                goto s2;
        }
        ft[i][l]=ft[a][b];
        l=l+1;
        s2:b=b+1;
    }
}
while(st[i][j]!='\0')
{
    if(st[i][j]=='|')
    {
        j=j+1;
        goto l1;
    }
    j=j+1;
}

ft[i][l]='\0';
}
}
cout << ("first \n");
for(i=0;i<n;i++)
    cout << ("FIRS[%c]=%s\n",st[i][0],ft[i]);
fol[0][0]='$';
for(i=0;i<n;i++)
{
    k=0;
    j=3;
    if(i==0)
        l=1;
    else
        l=0;
    k1:while((st[i][0]!=st[k][j])&&(k<n))
    {
        if(st[k][j]=='\0')

```

```

        {
            k++;
            j=2;
        }
        j++;
    }

    j=j+1;
    if(st[i][0]==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<1;m++)
                {
                    if(fol[i][m]==st[k][j])
                        goto q3;
                }
                fol[i][1]=st[k][j];
                l++;
                q3:;
            }
            else
            {
                while(st[k][j]!=st[a][0])
                {
                    a++;
                }
                p=0;
                while(ft[a][p]!='\0')
                {
                    if(ft[a][p]!='@')
                    {
                        for(m=0;m<1;m++)
                        {
                            if(fol[i][m]==ft[a][p])
                                goto q2;

```

```

        }
        fol[i][l]=ft[a][p];
        l=l+1;
    }
    else
        e=1;
        q2:p++;
    }
    if(e==1)
    {
        e=0;
        goto a1;
    }
}
else
{
    a1: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 q1;
        }
        fol[i][l]=fol[a][c];
        l++;
        q1:c++;
    }
}
goto k1;
}
fol[i][l]='\0';
}

```

```

cout << ("follow \n");
for(i=0;i<n;i++)
    cout << ("FOLLOW[%c]=%s\n",st[i][0],fol[i]);
cout << ("\n");
s=0;
for(i=0;i<n;i++)
{
    j=3;
    while(st[i][j]!='\0')
    {
        if((st[i][j-1]=='|')||(j==3))
        {
            for(p=0;p<=2;p++)
            {
                fin[s][p]=st[i][p];
            }
            t=j;
            for(p=3;((st[i][j]!='|')&&(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][0]!=st[i][0])
                {
                    a++;
                }
                while(fol[a][b]!='\0')
                {
                    cout <<
("M[%c,%c]=%s\n",st[i][0],fol[a][b],fin[s]);
                    b++;
                }
            }
            else if(!((st[i][t]>64)&&(st[i][t]<91)))

```

```

        cout <<
("M[%c,%c]=%s\n",st[i][0],st[i][t],fn[s]);
    else
    {
        b=0;
        a=0;
        while(st[a][0]!=st[i][3])
        {
            a++;
        }
        while(ft[a][b]!='\0')
        {
            cout <<
("M[%c,%c]=%s\n",st[i][0],ft[a][b],fn[s]);
            b++;
        }
        s++;
    }
    if(st[i][j]=='|')
    j++;
}
}
}

```

INPUT:

E->E+T
 E->T
 T->T*F
 T->F
 F->(E)
 F->i
 S->CC
 C->eC
 C->d

OUTPUT:

```
Command Prompt
C:\Users\priya\Desktop\Coding\compiler-design-lab\lab-6>python3 pp.py
Python was not found; run without arguments to install from the Microsoft Store, or disable
this shortcut from Settings > Manage App Execution Aliases.

C:\Users\priya\Desktop\Coding\compiler-design-lab\lab-6>python pp.py
      |*      |(      |)      |+      |i      |d
-----
E      |      |E->TE'  |      |      |E->TE'  |
-----
T      |      |T->FT'  |      |      |T->FT'  |
-----
F      |      |F->(E)   |      |      |F->i     |
-----
E'     |      |      |E'->e   |E'->TE'  |      |
-----
T'     |T'->FT'  |      |T'->e   |T'->e   |      |
-----
C:\Users\priya\Desktop\Coding\compiler-design-lab\lab-6>
```

RESULT:

Predictive parsing table is constructed.