

```

1 /* /**
2 * ASSIGNMENT: OPERATOR PRECEDENCE WITH RECOVERY
3 * AUTHORS: BIJAY KHATRI & DEEPAK VERMA
4 * ROLL: 12/CS/45 AND 12/CS/46
5 */ **
6 #include<stdio.h>
7 #include<ctype.h>
8 #include<string.h>
9
10 struct stru1
11 {
12     char non_ter[1],pro[25];
13 }cfg[25];
14 int n,st=-1,j,i,t=-1,m,in=0;
15 int v,c,p=1,s=0;
16 char str[20],stack[20],ch,tmp[10],input[20],symbol[2];
17 int main()
18 {
19     char temp[20];
20     char prec[6][6]={{' ','-','+','*','/','i'},{' ','=',<'<','<','<','<'},{' '+'>','=',<'<','<'},{' '*','>','>','=',<'<'},{' '/'>','>','>','=',<'<'},{' i '>','>','>','='>'}}};
21     printf("\n\n");
22     for(i=0;i<6;i++)
23     {
24         for(j=0;j<6;j++)
25             printf(" %c |",prec[i][j]);
26         printf("\n-----\n");
27     }
28     printf("Enter the number of productions:\n\r");
29     scanf("%d",&n);
30     printf("\n\r");
31     printf("Enter the productions:\n\r");
32     for(i=0;i<n;i++)
33     {
34         scanf("%s",cfg[i].non_ter);
35         //printf("\n\r");
36         scanf("%s",symbol);
37         scanf("%s",cfg[i].pro);
38         printf("\n\r");
39     }
40     printf("Enter the input string:\n\r");
41     scanf("%s",str);
42     printf("\n\r");
43     int i,r;
44     printf("ACTION\t\tSTACK\t\tINPUT\n");
45     printf("-\t-\t-\t\t%s$\n",str);
46     label1: stack[++st]=str[s++];
47     int low=0;
48     for(j=s;j<strlen(str);j++)
49     {
50         input[low++]=str[j];
51         m++;
52         input[low++]='$';
53         input[low]='\0';
54     label3: if(str[s-1]=='\0')
55     {
56
57         if(stack[st-1]==')')
58         {
59             printf("unbalanced right paranthesis \n Recovering error : removed right paranthesis\n");
60             if(st==2)
61             {
62                 stack[st-1]='\0';
63                 goto label3;
64             }
65             st--;
66             printf("removed ')' \t%s\t%s\n",stack,input);
67             goto label;
68         }
69         else if(st==1)
70             printf("\n\n\tString accepted\n");
71         else
72             printf("\n\n\tNot accepted by above grammar\n");
73         return 0;
74     }
75     printf("shift %c\t\t%s\t\t%s\n",stack[st],stack,input);
76     if((stack[st]=='+'||stack[st]=='-'||stack[st]=='*'||stack[st]=='/' )&&(input[0]!='+'||input[0]!='-'||input[0]!='*'||input[0]!='/' ))
77         printf("\n---MISSING OPERAND---\n");

```

```

78 // scanf("%d",&t);
79 label: for (i=0;i<=st;i++)
80 {
81     int l=0;
82     for (j=i;j<=st;j++)
83     {
84         temp[l++]=stack[j];
85     }
86     temp[l]='\0';
87     for (r=0;r<n;r++){
88         if(strcmp(temp,cfg[r].pro)==0)
89         {
90             int len = strlen(temp);
91             while(len--)
92             {
93                 stack[st-]=NULL;
94             }
95             stack[++st]=cfg[r].non_ter[0];
96             printf ("Reduced %c->%s\t%s\t\t%s\n",cfg[r].non_ter[0],cfg[r].pro,stack,input);
97
98             if((stack[st-1]==+'||stack[st-1]==-'||stack[st-1]=='+')&&input[0]=='/')
99                 goto label1;
100             else if((stack[st-1]==+'||stack[st-1]==-'&&input[0]=='+')
101                 goto label1;
102             else if((stack[st-1]==-'&&input[0]=='+')
103                 goto label1;
104
105             if((isalnum(stack[st]))&&isalnum(stack[st-1]))
106                 printf("\n---MISSING OPERATOR---\n");
107             goto label;
108         }
109     }
110 }
111 goto label1;
112 return 0;
113 }
114
115 /*
116 | - | + | * | / | i |
117 -----
118 - | = | < | < | < |
119 -----
120 + | > | = | < | < | < |
121 -----
122 * | > | > | = | < | < |
123 -----
124 / | > | > | > | = | < |
125 -----
126 i | > | > | > | > | = |
127 -----
128 Enter the number of productions:
129 4
130 Enter the productions:
131 E
132 ->
133 a
134
135 E
136 ->
137 E+E
138
139 E
140 ->
141 E*E
142
143 E
144 ->
145 (E)
146
147 Enter the input string:
148 (a+a*a)*a+a
149
150 ACTION      STACK      INPUT
151 - - - (a+a*a)*a+a$
152 shift (      ( a+a*a)*a+a$
153 shift a      (a +a*a)*a+a$
154 Reduced E->a (E +a*a)*a+a$

```

```

155 shift +      (E+  a*a)*a+a$
156 shift a      (E+a   *a)*a+a$
157 Reduced E->a (E+E   *a)*a+a$
158 shift *      (E+E*  a)*a+a$
159 shift a      (E+E*a  )*a+a$
160 Reduced E->a (E+E*E   )*a+a$
161 Reduced E->E*E (E+E   )*a+a$
162 Reduced E->E+E (E   )*a+a$
163 shift )      (E)   *a+a$
164 Reduced E->(E) E   *a+a$
165 shift *      E*   a+a$
166 shift a      E*a   +a$
167 Reduced E->a E*E   +a$
168 Reduced E->E*E E   +a$
169 shift +      E+   a$
170 shift a      E+a   $
171 Reduced E->a E+E   $
172 Reduced E->E+E E   $
173
174 string accepted
175
176 */
177

```