

```
1  /*
2  Implement one LL(1) parser without error handling
3  capacity.The grammar for the parser is fixed and the Input
4  is the text to be parsed and the output is the sequence of
5  production used.
6
7  Compiled By:
8  Himansu Rath  12/CS/06
9  R Om prakash  12/CS/07
10 */
11
12 #include <string.h>
13 #include <stdio.h>
14 #include <stdlib.h>
15
16 int main()
17 {
18     int i=0,j=0,k=0,m=0,n=0,o=0,ol=0,var=0,l=0,f=0,c=0,f1=0;
19     char str[30],str1[40]="E",temp[20],temp1[20],temp2[20],tt[20],t3[20];
20     strcpy(temp1,"\0");
21     strcpy(temp2,"\0");
22     char t[10];
23     char array[6][5][10] = {
24         "NT", "<id>","+","*",";",
25         "E", "Te","Error","Error","Error",
26         "e", "Error","+Te","Error","\0",
27         "T", "Vt","Error","Error","Error",
28         "t", "Error","\0","*Vt","\0",
29         "V", "<id>","Error","Error","Error"
30     };
31     printf("\n\tLL(1)  PARSER  TABLE \n");
32     for(i=0;i<6;i++)
33     {
34         for(j=0;j<5;j++)
35         {
36             printf("%10s",array[i][j]);
37         }
38         printf("\n");
39     }
40     printf("\n\tENTER THE STRING :");
41     gets(str);
42     if(str[strlen(str)-1] != ';')
43     {
44         printf("END OF STRING MARKER SHOULD BE ';'");
45         exit(1);
46     }
47     printf("\n\tCHECKING VALIDATION OF THE STRING ");
48     printf("\n\t%s",str1);
49     i=0;
50
51     while(i<strlen(str))
52     {
53         again:
54         if(str[i] == ' ' && i<strlen(str))
55         {
56             printf("\n\tSPACES IS NOT ALLOWED IN SOURCE STRING ");
57             exit(1);
```

```
58     }
59     temp[k]=str[i];
60     temp[k+1]='\0';
61     f1=0;
62     again1:
63     if(i>=strlen(str))
64     {
65         exit(1);
66     }
67     for(int l=1;l<=4;l++)
68     {
69         if(strcmp(temp,array[0][l])==0)
70         {
71             f1=1;
72             m=0,o=0,var=0,o1=0;
73             strcpy(temp1,"\0");
74             strcpy(temp2,"\0");
75             int len=strlen(str1);
76             while(m<strlen(str1) && m<strlen(str))
77             {
78                 if(str1[m]==str[m])
79                 {
80                     var=m+1;
81                     temp2[o1]=str1[m];
82                     m++;
83                     o1++;
84                 }
85                 else
86                 {
87                     if((m+1)<strlen(str1))
88                     {
89                         m++;
90                         temp1[o]=str1[m];
91                         o++;
92                     }
93                     else
94                         m++;
95                 }
96             }
97             temp2[o1] = '\0';
98             temp1[o] = '\0';
99             t[0] = str1[var];
100            t[1] = '\0';
101            for(n=1;n<=5;n++)
102            {
103                if(strcmp(array[n][0],t)==0)
104                    break;
105            }
106            strcpy(str1,temp2);
107            strcat(str1,array[n][l]);
108            strcat(str1,temp1);
109            printf("\n\t%s",str1);
110
111            if(strcmp(array[n][l],"\0")==0)
112            {
113                if(i==(strlen(str)-1))
```

```
115         {
116             int len=strlen(str1);
117             str1[len-1]='\0';
118             printf("\n\t%s",str1);
119             printf("\n\n\tENTERED STRING IS VALID");
120             exit(1);
121         }
122         strcpy(temp1,"\0");
123         strcpy(temp2,"\0");
124         strcpy(t,"\0");
125         goto again1;
126     }
127     if(strcmp(array[n][l],"Error")==0)
128     {
129         printf("\n\tERROR IN YOUR SOURCE STRING");
130         exit(1);
131     }
132     strcpy(tt,"\0");
133     strcpy(tt,array[n][l]);
134     strcpy(t3,"\0");
135     f=0;
136     for(c=0;c<strlen(tt);c++)
137     {
138         t3[c]=tt[c];
139         t3[c+1]='\0';
140         if(strcmp(t3,temp)==0)
141         {
142             f=0;
143             break;
144         }
145         else
146             f=1;
147     }
148
149     if(f==0)
150     {
151         strcpy(temp,"\0");
152         strcpy(temp1,"\0");
153         strcpy(temp2,"\0");
154         strcpy(t,"\0");
155         i++;
156         k=0;
157         goto again;
158     }
159     else
160     {
161         strcpy(temp1,"\0");
162         strcpy(temp2,"\0");
163         strcpy(t,"\0");
164         goto again1;
165     }
166 }
167 }
168 i++;
169 k++;
170 }
171 if(f1==0)
```

```

172         printf("\nENTERED STRING IS INVALID");
173     else
174         printf("\n\n\tENTERED STRING IS VALID");
175 }

```

```

182 /*****OUTPUT*****/

```

```

183 *

```

```

184 * LL(1)  PARSER  TABLE

```

```

185
186     NT      <id>      +      *      ;
187     E       Te      Error  Error  Error
188     e       Error    +Te    Error
189     T       Vt      Error   Error  Error
190     t       Error    *Vt
191     V       <id>     Error   Error  Error

```

```

192
193     ENTER THE STRING : <id>+<id>*<id>

```

```

194
195     CHECKING VALIDATION OF THE STRING

```

```

196     E
197     Te
198     Vte
199     <id>te
200     <id>e
201     <id>+Te
202     <id>+Vte
203     <id>+<id>te
204     <id>+<id>*Vte
205     <id>+<id>*<id>te
206     <id>+<id>*<id>e
207     <id>+<id>*<id>

```

```

208
209     ENTERED STRING IS VALID

```

```

210
211
212 *****/

```

```

213

```

```

214

```

```

215

```

```

216

```

```

217

```

```

218

```

```

219

```

```

220

```

```

221

```

```

222

```

```

223

```

```

224

```

```

225

```

```

226

```