```
 1: /*
 2:  * CSV
 3:  * accroding to RFC4180
 4:  */
 5: #include <stdio.h>
 6: #include <stdlib.h>
 7: #include <unistd.h>
 8: #include <string.h>
 9:
10: #define ISCOMMA     (ch==0x2c)
11: #define ISCR        (ch==0x0d)
12: #define ISDQUOTE    (ch==0x22)
13: #define ISLF        (ch==0x0A)
14: #define ISTEXT      ( (ch>=0x20&&ch<=0x21)|| \
15:     (ch>=0x23&&ch<=0x2b)|| (ch>=0x2d&&ch<=0x7e) )
16:
17: int  hasheader=0;
18: int  ch=-1;
19: int  vused;
20: int  vlen;
21: char value[BUFSIZ];
22: int  rno=0;
23: int  fno=0;
24:
25: int
26: readch()
27: {
28:     ch = fgetc(stdin);
29:     return 0;
30: }
31:
32: int
33: clearvalue()
34: {
35:     vused = 0;
36:     vlen  = BUFSIZ;
37:     value[0] = '\0';
38:     return 0;
39: }
40:
41: int
42: addch()
43: {
44:     char *p;
45:     int   c;
46:
47:     p = value;
48:     c = 0;
49:     while(*p) {
50:         p++;
51:         c++;
52:     }
53:     if(c+1<vlen && (ch>=0 && ch<=0x7f)) {
54:         *p++ = ch%255;
55:         *p = '\0';
56:         c++;
57:     }
58:     else {
59:         fprintf(stderr, "ignore charator\n");
60:     }
61:     return 0;
62: }
```

```
 63:
 64: int
 65: readescaped()
 66: {
 67:     readch();
 68:     while(ISTEXT || ISCOMMA || ISCR || ISLF || ISDQUOTE ) {
 69:         if(ISDQUOTE) {
 70:             readch();
 71:             if(ISDQUOTE) {
 72:                 /* 2DQUOTE */
 73:             }
 74:             else {
 75:                 goto shiftS3;
 76:             }
 77:         }
 78:         addch();
 79:         readch();
 80:     }
 81:     if(ISDQUOTE) {
 82:         readch();
 83:     }
 84:     else {
 85:         fprintf(stderr, "strange\n");
 86:     }
 87: shiftS3:
 88:     return 0;
 89: }
 90:
 91: int
 92: readnonescaped()
 93: {
 94:     while(ISTEXT) {
 95:         addch();
 96:         readch();
 97:     }
 98:     return 0;
 99: }
100:
101: int
102: readfield()
103: {
104:     clearvalue();
105:     readch();
106:     if(ch<0){
107:         fprintf(stderr, "EOF\n");
108:         return -1;
109:     }
110:
111:     if(ISDQUOTE){
112:         readescaped();
113:     }
114:     else {
115:         readnonescaped();
116:     }
117:     fprintf(stderr, "%d-%d field '%s'\n", rno, fno, value);
118:     return 0;
119: }
120:
121: int
122: readname()
123: {
124:     return readfield();
125: }
```

```
  126:
  127: int
  128: readrecord()
  129: {
  130:     int ck;
  131:
  132:     fno = 0;
  133:     do {
  134:         ck = readfield();
  135:         if(ck<0) {
  136:             return -1;
  137:         }
  138:         if(ISCOMMA) {
  139:             /* OK */
  140:         }
  141:         else {
  142:             break;
  143:         }
  144:         fno++;
  145:     } while(1);
  146:     rno++;
  147:
  148:     return 0;
  149: }
  150:
  151: int
  152: readheader()
  153: {
  154:     int ck;
  155:
  156:     fno = 0;
  157:     do {
  158:         ck = readname();
  159:         if(ck<0) {
  160:             return -1;
  161:         }
  162:         if(ISCOMMA) {
  163:             /* OK */
  164:         }
  165:         else {
  166:             break;
  167:         }
  168:         fno++;
  169:     } while(1);
  170:     rno++;
  171:
  172:     return 0;
  173: }
```

```
174:
175: int
176: readfile()
177: {
178:     int ck;
179:
180:     if(hasheader) {
181:         readheader();
182:     }
183:     do {
184:         ck = readrecord();
185:         if(ck<0) {
186:             return 0;
187:         }
188:         if(ISCR) {
189:             readch();
190:         }
191:         if(ISLF) {
192:             /* OK */
193:         }
194:         else {
195:             fprintf(stderr, "no LF (record separator)\n");
196:             break;
197:         }
198:     } while(1);
199:
200:     return 0;
201: }
202:
203: int
204: main()
205: {
206:     readfile();
207:     exit(0);
208: }
```