```c
/*************************************************************************************
                Othello Game (Text Version)
*************************************************************************************/

#include <stdio.h>

#include "othello.h"



/* 전역변수와 타입 선언 */
char turn='0';          /* 무슨 돌이 둘 차례인지 저장 */
char board[BOARD_SIZE+1][BOARD_SIZE+2];

STONE stone_no;

/*************************************************************************************/
int main() {
    POSITION current_pos;

    /* 보드 초기화후 출력*/
    init_board();
    draw_board();

    /* 게임 메인 루프 */
    while (1) {
        if (!check_pass_condition()) {
            change_turn();
            draw_board();
            if (!check_pass_condition()) { /* 양 쪽 모두 pass 상황이면 게임 종료 */
                output_game_result();
                return 0;
            }
        }
        current_pos = get_new_pos();
        if (!update_board(current_pos)) {
            printf("돌을 둘수 없는 곳!! 재입력:");
            continue;
        }
        count_stone();
        change_turn();
        draw_board();
        if (check_stop_condition()) break;
    }; /* while */
    output_game_result();
    return 0;
} /* main */
```

```
48
49  /*************************************************************************
    **/
50  int check_stop_condition() {
51      if (stone_no.n == 0 || stone_no.o ==0 || stone_no.x ==0 ) return 1; /* 게임 종
        료 상황 */
52      return 0;
53  }
54
55  /*************************************************************************
    **/
56  void change_turn() {
57      if (turn == 'O') turn = 'X';
58      else turn = 'O';
59  }
60
61  /*************************************************************************
    **/
62  void count_stone() {
63      int row, col;
64
65      stone_no.o = stone_no.x = 0;
66      for (row=1; row<=BOARD_SIZE; row++) {
67          for (col=1; col<=BOARD_SIZE; col++) {
68              switch(board[row][col]) {
69                  case 'O' : stone_no.o += 1; break;
70                  case 'X' : stone_no.x += 1; break;
71                  case '.' : stone_no.n += 1; break;
72              } /* switch */
73          } /* for col */
74      } /* for row */
75
76  } /* count_stone() */
77
78  /*************************************************************************
    **/
79  POSITION get_new_pos() {
80      int pos;
81      POSITION new_pos;
82
83      int max_pos = BOARD_SIZE*10 + BOARD_SIZE;
84      for(;;) {
85          scanf("%d",&pos);
86          new_pos.row = pos / 10; /* 세로 위치값 */
87          new_pos.col = pos % 10; /* 가로 위치값 */
88          if (new_pos.row <= 0 || new_pos.row > BOARD_SIZE ||
89              new_pos.col <= 0 || new_pos.col > BOARD_SIZE) {
90              printf("좌표값이 틀렸습니다. 다시 입력하세요 :");
91              continue;
```

```
92          }
93          if (board[new_pos.row][new_pos.col] != '.') {
94              printf("이미 돌이 놓인 자리입니다. 다시 입력하세요 :");
95              continue;
96          }
97          break;
98      } /* for */
99      return new_pos;
100 }
101
102
103 /*****************************************************************************
    **/
104 int update_board(POSITION pos) {
105     int y, x, opponent_no, i, j;
106     char stone_changed='N', opponent;
107
108     if (turn == 'O') opponent = 'X';
109     else opponent = 'O';
110
111     /* 정북 방향으로 검사 */
112     y = pos.row-1;
113     opponent_no=0;
114     while (y>1 && board[y][pos.col]==opponent) {
115         y--;        /*상대방 돌을 발견하는 동안 계속 진행 */
116         opponent_no++;
117     }
118             /* board경계를 벗어나지 않았고, 내돌을 만나면 ok */
119     if (opponent_no>0 && y>=1 && board[y][pos.col]==turn) {
120         /*상대방 돌들을 내돌로 변경(뒤집기) */
121         for(i=pos.row-1; i>y; i--) board[i][pos.col]=turn;
122         stone_changed='Y';
123     }
124
125     /* 정남 방향으로 검사 */
126     y = pos.row + 1;
127     opponent_no=0;
128     while (y<BOARD_SIZE && board[y][pos.col]==opponent) {
129         y++;        /*상대방 돌을 발견하는 동안 계속 진행 */
130         opponent_no++;
131     }
132             /* board경계를 벗어나지 않았고, 내돌을 만나면 ok */
133     if (opponent_no>0 && y<=BOARD_SIZE && board[y][pos.col]==turn) {
134         /*상대방 돌들을 내돌로 변경(뒤집기) */
135         for(i=pos.row+1; i<y; i++) board[i][pos.col]=turn;
136         stone_changed='Y';
137     }
138
139     /* 정동 방향으로 검사 */
140     x = pos.col + 1;
141     opponent_no=0;
142     while (x<BOARD_SIZE && board[pos.row][x]==opponent) {
```

```
143         x++;        /*상대방 돌을 발견하는 동안 계속 진행 */
144         opponent_no++;
145     }
146             /* board경계를 벗어나지 않았고, 내돌을 만나면 ok */
147     if (opponent_no>0 && x<=BOARD_SIZE && board[pos.row][x]==turn) {
148         /*상대방 돌들을 내돌로 변경(뒤집기) */
149         for(i=pos.col+1; i<x; i++) board[pos.row][i]=turn;
150         stone_changed='Y';
151     }
152
153     /* 정서 방향으로 검사 */
154     x = pos.col - 1;
155     opponent_no=0;
156     while (x>1 && board[pos.row][x]==opponent) {
157         x--;        /*상대방 돌을 발견하는 동안 계속 진행 */
158         opponent_no++;
159     }
160             /* board경계를 벗어나지 않았고, 내돌을 만나면 ok */
161     if (opponent_no>0 && x>=1 && board[pos.row][x]==turn) {
162         /*상대방 돌들을 내돌로 변경(뒤집기) */
163         for(i=pos.col-1; i>x; i--) board[pos.row][i]=turn;
164         stone_changed='Y';
165     }
166
167
168     /* 북동 방향으로 검사 */
169     y = pos.row-1;
170     x = pos.col+1;
171     opponent_no=0;
172     while (y>1 && x< BOARD_SIZE && board[y][x]==opponent) {
173         y--;        /*상대방 돌을 발견하는 동안 계속 진행 */
174         x++;
175         opponent_no++;
176     }
177             /* board경계를 벗어나지 않았고, 내돌을 만나면 ok */
178     if (opponent_no>0 && y>1 && x<BOARD_SIZE && board[y][x]==turn) {
179         /*상대방 돌들을 내돌로 변경(뒤집기) */
180         for(i=pos.row-1, j=pos.col+1; i>y ; i--, j++) board[i][j]=turn;
181         stone_changed='Y';
182     }
183
184
185     /* 북서 방향으로 검사 */
186     y = pos.row-1;
187     x = pos.col-1;
188     opponent_no=0;
189     while (y>1 && x>1 && board[y][x]==opponent) {
190         y--;        /*상대방 돌을 발견하는 동안 계속 진행 */
191         x--;
192         opponent_no++;
193     }
194             /* board경계를 벗어나지 않았고, 내돌을 만나면 ok */
195     if (opponent_no>0 && y>1 && x>1 && board[y][x]==turn) {
```

```
196         /*상대방 돌들을 내돌로 변경(뒤집기) */
197         for(i=pos.row-1, j=pos.col-1; i>y ; i--, j--) board[i][j]=turn;
198         stone_changed='Y';
199     }
200
201     /* 남동 방향으로 검사 */
202     y = pos.row+1;
203     x = pos.col+1;
204     opponent_no=0;
205     while (y< BOARD_SIZE && x< BOARD_SIZE && board[y][x]==opponent) {
206         y++;        /*상대방 돌을 발견하는 동안 계속 진행 */
207         x++;
208         opponent_no++;
209     }
210             /* board경계를 벗어나지 않았고, 내돌을 만나면 ok */
211     if (opponent_no>0 && y< BOARD_SIZE && x<BOARD_SIZE && board[y][x]==turn) {
212         /*상대방 돌들을 내돌로 변경(뒤집기) */
213         for(i=pos.row+1, j=pos.col+1; i<y ; i++, j++) board[i][j]=turn;
214         stone_changed='Y';
215     }
216
217
218     /* 남서 방향으로 검사 */
219     y = pos.row+1;
220     x = pos.col-1;
221     opponent_no=0;
222     while (y< BOARD_SIZE && x>1 && board[y][x]==opponent) {
223         y++;        /*상대방 돌을 발견하는 동안 계속 진행 */
224         x--;
225         opponent_no++;
226     }
227             /* board경계를 벗어나지 않았고, 내돌을 만나면 ok */
228     if (opponent_no>0 && y< BOARD_SIZE && x>1 && board[y][x]==turn) {
229         /*상대방 돌들을 내돌로 변경(뒤집기) */
230         for(i=pos.row+1, j=pos.col-1; i<y ; i++, j--) board[i][j]=turn;
231         stone_changed='Y';
232     }
233
234     if (stone_changed == 'N') return 0;
235     else {
236         board[pos.row][pos.col]=turn;
237         return 1;
238     }
239
240
241 } /* update_board */
242
243
244 /*****************************************************************************
    **/
245 void draw_board() {
246     int row, col;
```

```
247
248     printf("₩n₩n₩n₩n₩n");
249     printf("==========================₩n");
250     printf("=       OTHELLO [v.1]     =₩n");
251     printf("==========================₩n");
252     printf("    1  2  3  4  5  6  7  8₩n");
253     for (row=1; row<=BOARD_SIZE; row++) {
254         printf("%2d ",row);
255         for (col=1; col<=BOARD_SIZE; col++) {
256             printf(" %c ",board[row][col]);
257         } /* for col */
258         printf("₩n");
259     } /* for row */
260     printf("==========================₩n");
261     count_stone();
262     printf("O :%2d  ",stone_no.o);
263     printf("[%c]가 둘 차례! ",turn);
264     printf("X :%2d₩n",stone_no.x);
265     printf(".........................₩n");
266     printf("위치값 2자리 입력[세로가로] : ");
267 } /* draw_board() */
268
269 /*****************************************************************************
    **/
270 void init_board() {
271     int row, col;
272
273     for (row=1; row<=BOARD_SIZE; row++) {
274         for (col=1; col<=BOARD_SIZE; col++) {
275             board[row][col]='.';
276         } /* for col */
277     } /* for row */
278     board[4][4]='X';
279     board[5][5]='X';
280     board[4][5]='O';
281     board[5][4]='O';
282 } /* draw_board() */
283
284
285
```