```c

#include <stdio.h>
#include "othello.h"

/* 전역변수와 타입 선언 */
extern char turn;           /* 무슨 돌이 둘 차례인지 저장 */
extern char board[BOARD_SIZE+1][BOARD_SIZE+2];
extern STONE stone_no;


/
  ********************************************************************************
  **/
int check_if_possible_position(int row, int col) {
    int y, x, opponent_no;
    char place_exist_to_put='N', opponent;
    POSITION pos;

    pos.row=row, pos.col=col;

    if (turn == '0') opponent = 'X';
    else opponent = '0';

    /* 정북 방향으로 검사 */
    y = pos.row-1;
    opponent_no=0;
    while (y>1 && board[y][pos.col]==opponent) {
        y--;        /*상대방 돌을 발견하는 동안 계속 진행 */
        opponent_no++;
    }
            /* board경계를 벗어나지 않았고, 내돌을 만나면 ok */
    if (opponent_no>0 && y>=1 && board[y][pos.col]==turn) {
        place_exist_to_put='Y';
    }

    /* 정남 방향으로 검사 */
    y = pos.row + 1;
    opponent_no=0;
    while (y<BOARD_SIZE && board[y][pos.col]==opponent) {
        y++;        /*상대방 돌을 발견하는 동안 계속 진행 */
        opponent_no++;
    }
            /* board경계를 벗어나지 않았고, 내돌을 만나면 ok */
    if (opponent_no>0 && y<=BOARD_SIZE && board[y][pos.col]==turn) {
        place_exist_to_put='Y';
    }

    /* 정동 방향으로 검사 */
    x = pos.col + 1;
    opponent_no=0;
    while (x<BOARD_SIZE && board[pos.row][x]==opponent) {
        x++;        /*상대방 돌을 발견하는 동안 계속 진행 */
        opponent_no++;
```

```
52      }
53              /* board경계를 벗어나지 않았고, 내돌을 만나면 ok */
54      if (opponent_no>0 && x<=BOARD_SIZE && board[pos.row][x]==turn) {
55          place_exist_to_put='Y';
56      }
57
58      /* 정서 방향으로 검사 */
59      x = pos.col - 1;
60      opponent_no=0;
61      while (x>1 && board[pos.row][x]==opponent) {
62          x--;        /*상대방 돌을 발견하는 동안 계속 진행 */
63          opponent_no++;
64      }
65              /* board경계를 벗어나지 않았고, 내돌을 만나면 ok */
66      if (opponent_no>0 && x>=1 && board[pos.row][x]==turn) {
67          place_exist_to_put='Y';
68      }
69
70
71      /* 북동 방향으로 검사 */
72      y = pos.row-1;
73      x = pos.col+1;
74      opponent_no=0;
75      while (y>1 && x< BOARD_SIZE && board[y][x]==opponent) {
76          y--;        /*상대방 돌을 발견하는 동안 계속 진행 */
77          x++;
78          opponent_no++;
79      }
80              /* board경계를 벗어나지 않았고, 내돌을 만나면 ok */
81      if (opponent_no>0 && y>1 && x<BOARD_SIZE && board[y][x]==turn) {
82          place_exist_to_put='Y';
83      }
84
85
86      /* 북서 방향으로 검사 */
87      y = pos.row-1;
88      x = pos.col-1;
89      opponent_no=0;
90      while (y>1 && x>1 && board[y][x]==opponent) {
91          y--;        /*상대방 돌을 발견하는 동안 계속 진행 */
92          x--;
93          opponent_no++;
94      }
95              /* board경계를 벗어나지 않았고, 내돌을 만나면 ok */
96      if (opponent_no>0 && y>1 && x>1 && board[y][x]==turn) {
97          place_exist_to_put='Y';
98      }
99
100     /* 남동 방향으로 검사 */
101     y = pos.row+1;
102     x = pos.col+1;
103     opponent_no=0;
104     while (y< BOARD_SIZE && x< BOARD_SIZE && board[y][x]==opponent) {
```

```
105         y++;        /*상대방 돌을 발견하는 동안 계속 진행 */
106         x++;
107         opponent_no++;
108     }
109             /* board경계를 벗어나지 않았고, 내돌을 만나면 ok */
110     if (opponent_no>0 && y< BOARD_SIZE && x<BOARD_SIZE && board[y][x]==turn) {
111         place_exist_to_put='Y';
112     }
113
114
115     /* 남서 방향으로 검사 */
116     y = pos.row+1;
117     x = pos.col-1;
118     opponent_no=0;
119     while (y< BOARD_SIZE && x>1 && board[y][x]==opponent) {
120         y++;        /*상대방 돌을 발견하는 동안 계속 진행 */
121         x--;
122         opponent_no++;
123     }
124             /* board경계를 벗어나지 않았고, 내돌을 만나면 ok */
125     if (opponent_no>0 && y< BOARD_SIZE && x>1 && board[y][x]==turn) {
126         place_exist_to_put='Y';
127     }
128
129     if (place_exist_to_put == 'N') return 0;
130     else {
131         return 1;
132     }
133 }
134
135 void output_game_result(){
136     draw_board();
137
138 }
139
140 /*****************************************************************************
    **/
141 /* 모든 비어 있는 점에 대하여 검사한 후 비어 있는 곳이 있으면 1을 리턴
142    없으면 0을 리턴
143 */
144 int check_pass_condition() {
145     int row, col;
146
147     for (row=1; row<=BOARD_SIZE; row++) {
148         for (col=1; col<=BOARD_SIZE; col++) {
149             if (board[row][col]=='.') {
150                 if (check_if_possible_position(row,col)) return 1;
151             }
152         }
153     }
154     return 0;
155 }
```

156
157
158
159
160