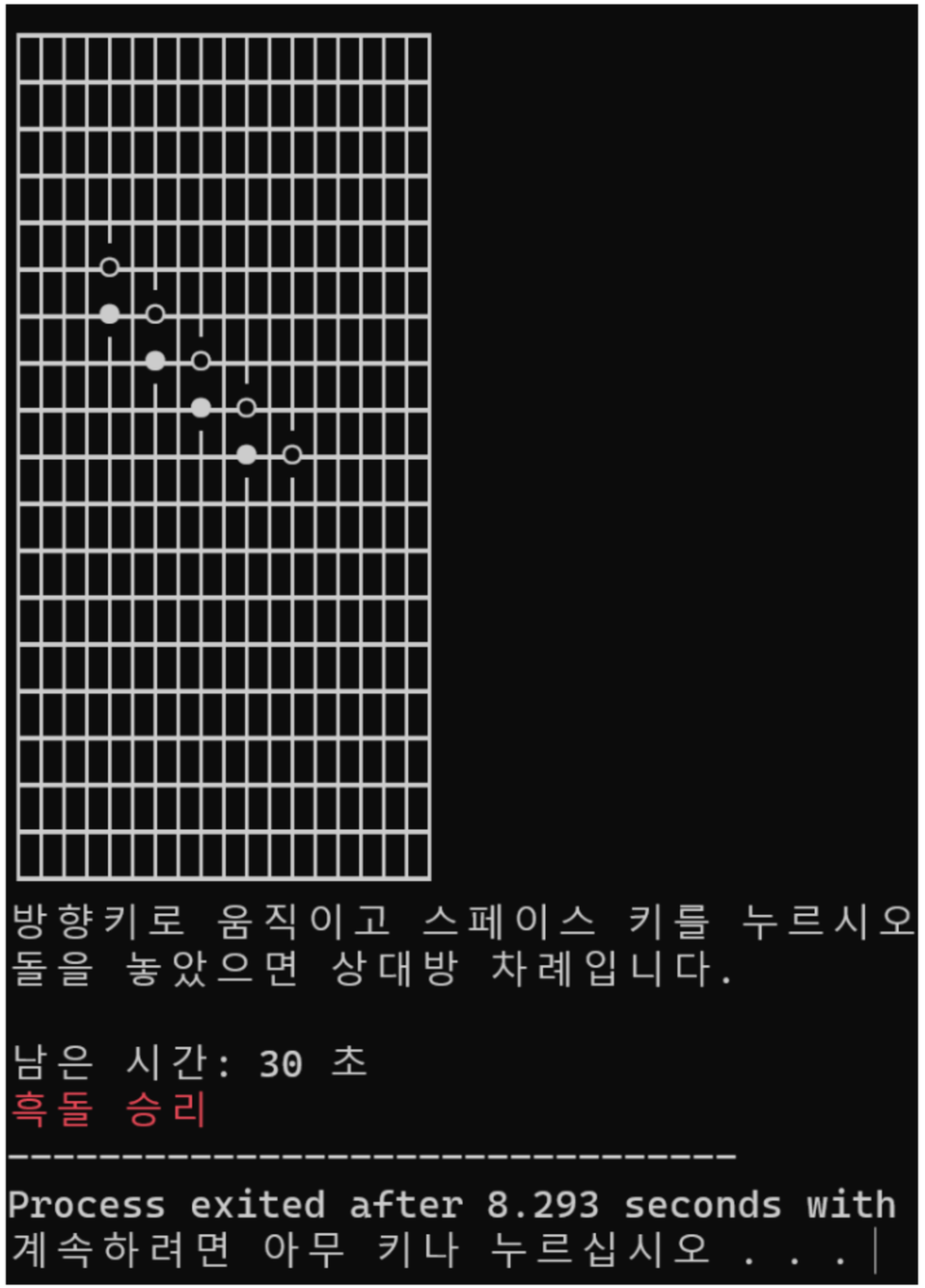
게임프로그래밍

2019775009
소프트웨어학과
김수민

게임 소개

오목

응용8_1_1의 상위버전 8_1_3 참고



기존 코드 문제점

1. 같은 자리에 다른 색깔의 돌로 덮어 버릴 수가 있다

2. 오목 게임의 승리 조건이 없다

3. 바둑돌이 바둑판 우측을 벗어남(move_arrow)

그 외 추가기능

1. 바둑 돌을 놓을 때마다 beep 출력

2. 승리한 player의 바둑돌 색 출력 및 색상변경

3. 각 플레이어당 30초의 제한시간(Timer) 추가

game_control 함수(1)

clock() -> #include <time.h>을
상단에 선언해야 쓸 수 있습니다

clock_t start_time = clock() 코드의 의미는
현재 시간을 측정하여 변수 start_time에

저장하는 코드입니다. clock() 함수는 프로그램이
시작된 후 경과한 CPU 시간을 반환합니다.

```
void game_control(void)
{
    int x=1, y=1, other=0; // 순서 추가
    int matrix[2][20][20]={0}; // plyaer,x,y
    char key;
    char *stone[2]={"○ ","● "}; // 바둑돌 색깔
    clock_t start_time = clock();

    do
    {
     gotoxy(1,1);
     draw_check01(18, 18);
     gotoxy(x, y);
     printf("%s", stone[other]);
     display_stone(matrix);
     gotoxy(1, 20);
     printf("방향키로 움직이고 ");
     printf("스페이스 키를 누르시오.");
     gotoxy(1,21);
         printf("돌을 놓았으면 상대방 차례입니다.
```

game_control 함수(2)

출력 printf문의 Color를 바꿀 때 사용한 GetStdHandle은 상단에 #include <windows.h> 선언 필수

```
// 시간제한 추가
while (!_kbhit()){ // 제한시간동안 입력버퍼가 없는동안 반복
    int remaining_time = (clock() - start_time) / CLOCKS_PER_SEC; // 현재시간-시작시간 / 초
    gotoxy(1, 23);
    printf("남은 시간 : %d 초      ", 30 - remaining_time); // 뒤의 공백들은 이전에 출력된 남은 시간을 지우는 역할을 합니다.
        if (remaining_time >= 30){
            HANDLE hConsole = GetStdHandle(STD_OUTPUT_HANDLE);
                if(other == 0){
                    SetConsoleTextAttribute(hConsole, FOREGROUND_BLUE | FOREGROUND_INTENSITY);
                        printf("\n백돌 승리");
                }else if(other == 1){
                    SetConsoleTextAttribute(hConsole, FOREGROUND_RED | FOREGROUND_INTENSITY);
                    printf("\n흑돌 승리");
                }

                SetConsoleTextAttribute(hConsole, FOREGROUND_RED | FOREGROUND_GREEN | FOREGROUND_BLUE); // 원래 색상으로 복구
                exit(0);
            }
}
```

game_control 함수(3)

바둑돌을 해당 좌표에 놓을 때 검사 조건문

```
else if(key==32)
    {
        // x,y좌표값에 돌이 놓여있는지 여부 체크
        if (matrix[0][(x+1)/2][y] == 0 && matrix[1][(x+1)/2][y] == 0) {
            Beep(1000, 200); //  ( 주파수 ,  시간 )
            matrix[other][(x+1)/2][y]=1;
            start_time = clock(); // 시간 재설정
        // 승리조건 추가
        int winner = check_win(matrix, other);
        if(winner != -1)
        {
            HANDLE hConsole = GetStdHandle(STD_OUTPUT_HANDLE);
            if(winner == 0) {
                SetConsoleTextAttribute(hConsole, FOREGROUND_RED | FOREGROUND_INTENSITY);
                printf("\n흑돌 승리");
            } else if (winner == 1){
                SetConsoleTextAttribute(hConsole, FOREGROUND_BLUE | FOREGROUND_INTENSITY);
                printf("\n백돌 승리");
            }
            SetConsoleTextAttribute(hConsole, FOREGROUND_RED | FOREGROUND_GREEN | FOREGROUND_BLUE); // 원래 색상으로 복구
           exit(0);
        }
        //차례 넘김
        other=1-other;
            }
            else {
            printf("\n이미 돌이 있는 위치입니다.\n");
            Sleep(1000); // 1초 대기후
            system("cls"); // 화면 지움
        }
    }
```

display_stone 함수


```
void display_stone(int matrix[][20][20])
{
    int i, x, y;
    char  *stone[2]={"o ", "● "};

    for(i=0;i<2;i++)
        for(x=1;x<20;x++)
            for(y=1;y<20;y++)
            {
                if (matrix[i][x][y]==1)
                {
                    gotoxy(x*2-1, y);
                    printf("%s", stone[i]);
                }
            }
}
```

matrix 배열에 담긴 정보값으로
실질적으로 바둑판에 바둑돌을
표시해줄 함수

check_win 함수(1)


```
int check_win(int matrix[][20][20], int player) {
    int i, j;

    // 세로 체크
    for (i = 1; i < 20; i++) {
        for (j = 1; j <= 16; j++) {
            if (matrix[player][i][j] == 1 &&
                matrix[player][i][j+1] == 1 &&
                matrix[player][i][j+2] == 1 &&
                matrix[player][i][j+3] == 1 &&
                matrix[player][i][j+4] == 1 &&
                (j==1 || matrix[player][i][(j-1)]==0) &&
                (j==16 || matrix[player][i][(j+5)]==0)) {
                    return player;
            }
        }
    }
```

check_win 함수(2)


```
// 가로 체크
for (i = 1; i <= 16; i++) {
    for (j = 1; j <= 20; j++) {
        if (matrix[player][i][j] == 1 &&
            matrix[player][i+1][j] == 1 &&
            matrix[player][i+2][j] == 1 &&
            matrix[player][i+3][j] == 1 &&
            matrix[player][i+4][j] == 1 &&
            (i==1 || matrix[(player)][(i-1)][j]==0) &&
            (i==16 || matrix[(player)][(i+5)][j]==0)) {
                return player;
        }
    }
}
```

check_win 함수(3)


```
// 좌측하단에서 우측상단으로 체크
for (i = 1; i <= 16; i++) {
    for (j = 1; j <= 16; j++) {
        if (matrix[player][i][j] == 1 &&
            matrix[player][i+1][j+1] == 1 &&
            matrix[player][i+2][j+2] == 1 &&
            matrix[player][i+3][j+3] == 1 &&
            matrix[player][i+4][j+4] == 1 &&
            ((i==16 || j==16) || matrix[(player)][(i+5)][(j+5)]==0) &&
            ((i==1 || j==1) || matrix[(player)][(i-1)%20][(j-1)]==0)) {
                return player;
        }
    }
}
```

check_win 함수(4)


```
    // 우측상단에서 좌측하단으로 체크
    for (i = 20; i >= 5; i--) {
        for (j = 1; j <= 16; j++) {
            if(matrix[player][i][j] == 1 &&
               matrix[player][i-1][j+1] == 1 &&
               matrix[player][i-2][j+2] == 1 &&
               matrix[player][i-3][j+3] == 1 &&
               matrix[player][i-4][j+4] == 1 &&
               ((i==20 || j==1) || matrix[(player)][(i+1)%20][(j-1)]==0) &&
               ((i<=5 || j>=16) || matrix[(player)][(i-5)][(j+5)]==0)) {
                  return player;
            }
        }
    }

    return -1;
}
```

출처

https://chat.openai.com/auth/login?next=%2F

감사합니다