오락실의 하키 게임을 SW로 구현

사용하는 헤더 불러오기

In [ ]:
```
#include<stdio.h>
#include<Windows.h>
#include<conio.h>
#include<stdlib.h> // rand 함수 사용
#include<time.h>
```

매크로 선언

In [ ]:
```
#define UP 72       // 키보드 위 방향키
#define DOWN 80     // 키보드 아래 방향키
#define speed_num 10 // 시작할때의 공의 속도 (숫자가 커질수록 느려짐 (3일 경우 3번에 한번 공이 이동한다는 뜻))
#define rally_num 1  // 몇번의 랠리마다 속도를 증가시킬지 설정
#define width 49    // 가로 크기 - 1 (0부터 시작하기 때문)
#define length 19   // 세로 크기 - 1 (0부터 시작하기 때문)
```

사용하는 변수들 선언

In [ ]:
```
int pos_x= width / 2; // 공의 시작 위치
int pos_y=length / 2;
int vel_x=1; // 공의 진행방향
int vel_y=1;
int speed1 = speed_num;
int speed2 = 1;
int speed_flag = 1;
int rally = rally_num;
int rally_flag = 1;
int ball_flag1 = 0;
int ball_flag2 = 0;
int ball_flag3 = 0;
int diff = 0;
int com_speed = 0;
int com_flag = 0;
```

콘솔 창 설정

가로는 50 세로는 20의 크기를 가지는 콘솔 창을 설정해 준다
콘솔창의 이름은 hockey game으로 설정

In [ ]:
```
void SetConsoleView()   // 콘솔 창의 크기와 제목을 지정하는 함수
{
    system("mode con:cols=50 lines=20");
    system("title hockey game");

}
```

커서 위치 이동 함수

In [ ]:
```
void GotoXY(int x, int y) // 커서의 위치를 x, y로 이동하는 함수
{
    COORD Pos;
    Pos.X = x;
    Pos.Y = y;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), Pos);
}
```

Player Bar 함수

|
|
|
|
|
모습으로 게임창의 왼쪽에 Player Bar가 만들어진다

In [ ]:
```
void Drawbar(int y) // player의 bar를 그려주는 함수
{
    GotoXY(0, y-2);
    printf("┃\n");
    GotoXY(0, y-1);
    printf("┃\n");
    GotoXY(0, y);
    printf("┃\n");
    GotoXY(0, y+1);
    printf("┃\n");
    GotoXY(0, y+2);
    printf("┃\n");
}
```

Computer Bar 함수

|
|
|

|
|
모습으로 게임창의 오른쪽에 Computer Bar가 만들어진다

In [ ]:
```
void Drawcom(int y) // 컴퓨터의 bar를 그려주는 함수
{
    GotoXY(width, y-2);
    printf("¦");
    GotoXY(width, y-1);
    printf("¦");
    GotoXY(width, y);
    printf("¦");
    GotoXY(width, y+1);
    printf("¦");
    GotoXY(width, y+2);
    printf("¦");
}
```

레이아웃 함수

게임창의 위와 아래를 구분할 수 있게 레이아웃을 그려준다

In [ ]:
```
void Drawlayout()   // 레이아웃을 그려주는 함수
{
    GotoXY(0, 0);
    printf("------------------------------------------------");
    GotoXY(0, length);
    printf("------------------------------------------------");

}
```

공 함수

In [ ]:
```
void Drawball(int x, int y) // 공을 그려주는 함수
{
    GotoXY(x, y);
    printf("o");
}
```

공 위치 함수

speed1 변수의 값이 1보다 크면 공의 속도는 1/speed1이 되고
speed1 변수의 값이 1보다 크지 않으면 공의 속도는 speed2가 된다
vel_x와 vel_y변수는 공의 방향을 정해준다
vel_x가 음수이면 게임창의 왼쪽으로 vel_x가 양수이면 게임창의 오른쪽으로 공이 이동
vel_y가 음수이면 게임창의 위쪽으로 vel_y가 양수이면 게임창의 아래쪽으로 공이 이동

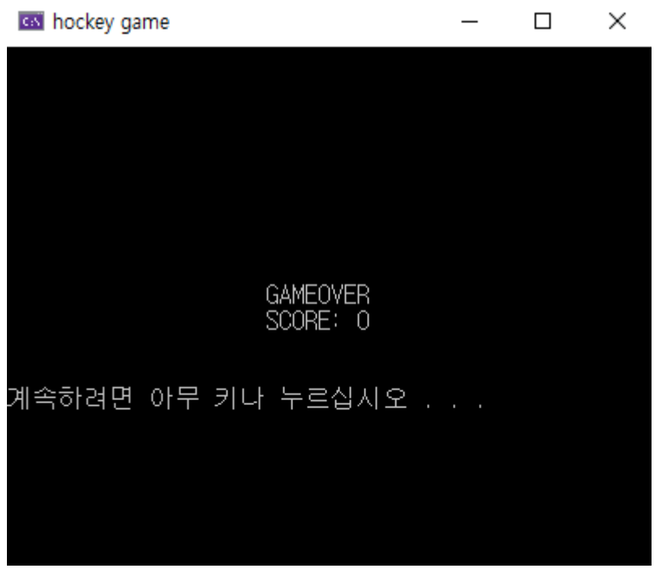
In [ ]:
```
void update() // 공의 위치를 지정해 주는 함수
{
    if (speed1 > 1)
    {
        if (speed_flag == speed1)
        {
            pos_x = pos_x + vel_x;
            pos_y = pos_y + vel_y;
            speed_flag = 0;
        }

        speed_flag = speed_flag + 1;
    }
    else
    {
        pos_x = pos_x + (vel_x * speed2);
        pos_y = pos_y + (vel_y * speed2);
    }


    Drawball(pos_x, pos_y);
}
```

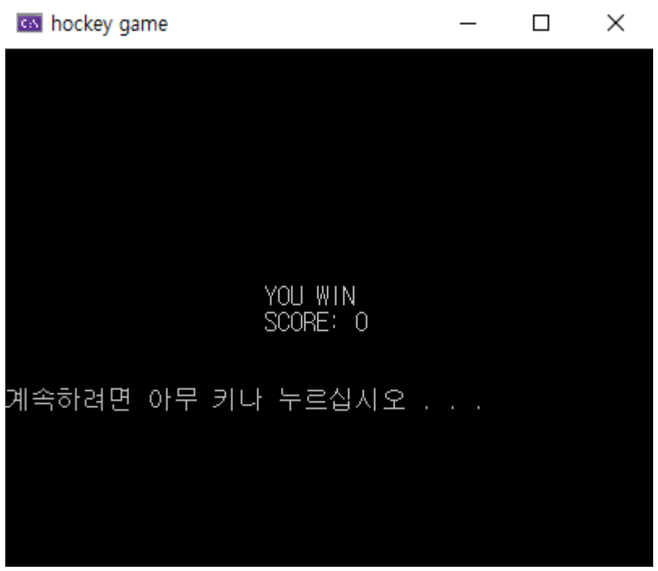
게임 종료 창

졌을 때



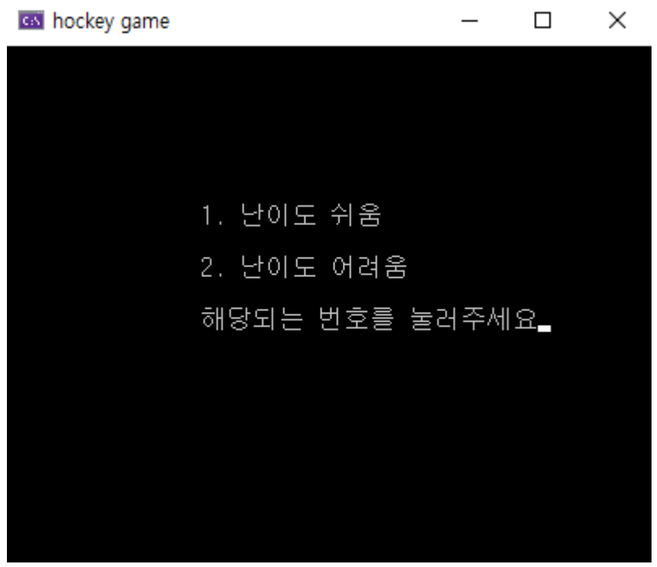
In [ ]:
```
void Drawgameover(int score) // 게임종료창
{
    system("cls"); // 창을 초기화
    GotoXY(20, length / 2);
    printf("GAMEOVER\n");
    GotoXY(20, (length / 2)+1);
    printf("SCORE: %d\n\n\n", score);
    system("pause");
}
```

이겼을 때



In [ ]:
```
void Drawwin(int score)
{
    system("cls");
    GotoXY(20, length / 2);
    printf("YOU WIN\n");
    GotoXY(20, (length / 2) + 1);
    printf("SCORE: %d\n\n\n", score);
    system("pause");
}
```

게임 시작 난이도 선택 창



In [ ]:
```
void Drawdiff()
{
    system("cls");
    GotoXY(15, 6);
    printf("1. 난이도 쉬움");
    GotoXY(15, 8);
    printf("2. 난이도 어려움");
    GotoXY(15, 10);
    printf("해당되는 번호를 눌러주세요");
}
```

메인 함수

In [ ]:
```
int main(void)
{
```

콘솔창 띄움

In [ ]:
```
    SetConsoleView();

```

무한루프 실행 이 무한루프는 게임이 끝나면 다시 시작할 수 있도록 해준다

In [ ]:
```
while (1)
{
```

변수들 초기화

In [ ]:
```
    // 초기화
    int bar_y = length/2;
    int com_y = length/2;
    int score = 0;
    pos_x = width/2;
    pos_y = length/2;
    vel_x = 1; // 공의 진행방향
    vel_y = 1;
    speed1 = speed_num; // 공의 속도 (숫자가 커질수록 느려짐 (3일 경우 3번에 한번 공이 이동한다는 뜻))
    speed_flag = 1;
    rally = rally_num; // 몇번의 랠리마다 속도를 증가시킬지 설정
    rally_flag = 1;
    ball_flag1 = 0;
    ball_flag2 = 0;
    ball_flag3 = 0;
    com_speed = 0;
    com_flag = 0;
    int rand1 = 0;
    int rand2 = 0;
```

난수 생성기

rand의 seed값을 현재 시간으로 설정해주어 실행하는 시간에 따라 rand 값이 다르게 나오도록 설정해준다 rand값의 범위는 0 ~ 32767이다
이 값을 2로 나눈 나머지 값은 0 아니면 1이 나오게 된다
이 두 값을 가지고 vel_x와 vel_y에 1이나 -1을 랜덤으로 설정해주어 게임 시작시 공의 방향을 랜덤으로 설정시켜 준다

In [ ]:
```
////////////////////// 난수생성기 //////////////////////////////
srand((unsigned int)time(NULL)); // rand의 seed를 현재 시간으로 입력해준다
rand1 = rand() % 2; // rand값의 범위는 0~32767이다 2로 나눈 나머지를 구하게 되면 0과 1 두가지 값이 나오게 된다
rand2 = rand() % 2;

if (rand1 == 0)
{
    vel_x = -1;
}
else
{
    vel_x = 1;
}

if (rand2 == 0)
{
    vel_y = -1;
}
else
{
    vel_y = 1;
}
////////////////////////////////////////////////////////
```

난이도 선택 창을 띄운다

In [ ]:
```
Drawdiff();
```

난이도 입력받기

이 무한루프는 난이도를 입력받는다
키보드 1을 누르게 되면 난이도는 쉬움
키보드 2를 누르게 되면 난이도는 어려움으로 설정된다
다른 키를 누르게 되면 1, 2 중 원하는 난이도를 눌러주세요 라는 안내가 뜨게 된다

In [ ]:
```
        while (1)
        {
            diff = 0;
            if (_kbhit()) {                // 1. 키보드 입력이 감지되면
                int pressedKey = _getch();  // 2. 아스키 코드 값을 가져오고
                switch (pressedKey) {       // 3. 입력받은 키에 해당하는 동작을 수행한다.
                case '1':   // 난이도 쉬움
                    diff = 1;
                    break;

                case '2':   // 난이도 어려움
                    diff = 2;
                    break;

                default:
                    GotoXY(10, 12);
                    printf("1, 2 중 원하는 난이도를 눌러주세요");
                }
            }
            if (diff == 1 || diff == 2)
            {
                break;
            }
        }
```

게임 진행

이 무한 루프는 게임이 진행될 수 있도록 해준다

In [ ]:
```
        while (1)
        {
```

키보드 입력 받기

위 방향키가 입력되면 Player bar를 위로 이동시켜준다
아래 방향키가 입력되면 Player bar를 아래로 이동시켜준다
Player bar의 속도는 2다 (한번 입력시 2칸씩 이동)

In [ ]:
```
        if (_kbhit()) {                // 1. 키보드 입력이 감지되면
            int pressedKey = _getch();  // 2. 아스키 코드 값을 가져오고
            switch (pressedKey) {       // 3. 입력받은 키에 해당하는 동작을 수행한다.
            case UP:
                bar_y -= 2;
                break;
                // 위로 이동
            case DOWN:
                bar_y += 2;
                break;
                // 아래로 이동
            }
        }
```

Player Bar 최대 최소 값

Player Bar가 게임창 밖으로 나가지 않도록 해준다
Player Bar는 bar_y를 기준으로 2칸씩 더 확장되어있다 (Drawbar 함수 참고)
따라서 가장자리 레이아웃 자리까지 생각해서 위 아래 가장 끝에서 3번째에 bar_y의 최소 최대 값을 지정해준다

In [ ]:
```
        if (bar_y <= 3)     // player bar의 위 아래 최소 최대 값을 정해준다
        {
            bar_y = 3;
        }
        else if (bar_y >= length - 3)
        {
            bar_y = length - 3;
        }
```

update 함수를 실행시켜 공의 위치를 설정해 준다

In [ ]:
```
        update(); // 공의 위치를 변경해주는 함수
```

난이도 적용

난이도가 쉬움이면 Computer Bar의 이동 속도는 0.5가 된다
난이도가 어려움이면 Computer Bar의 이동 속도는 1이 된다

In [ ]:
```
        ////////////////////////////////////////////////////////////////////////////////////////

        if (diff == 1)  // 난이도 쉬움
        {
            com_speed = 2;
        }
        else if (diff == 2) // 난이도 어려움
        {
            com_speed = 1;
        }
```

Computer Bar 이동

공의 위치에 따라 움직인다
이때 Computer Bar의 이동속도는 난이도에 따라 다르다

In [ ]:
```
        if (com_flag == com_speed)
        {
            if (com_y > pos_y)  // 컴퓨터 bar의 위치를 설정해준다
            {
                com_y = com_y - 1;
            }
            else if (com_y < pos_y)
            {
                com_y = com_y + 1;
            }
            com_flag = 0;
        }
        com_flag = com_flag + 1;
```

Computer Bar 최대 최소 값

Computer Bar가 게임창 밖으로 나가지 않도록 해준다
Computer Bar는 com_y를 기준으로 2칸씩 더 확장되어있다 (Drawcom 함수 참고)
따라서 가장자리 레이아웃 자리까지 생각해서 위 아래 가장 끝에서 3번째에 com_y의 최소 최대 값을 지정해준다

In [ ]:
```
        if (com_y <= 3)     // 컴퓨터 bar의 위 아래 최소 최대 값을 정해준다
        {
            com_y = 3;
        }
        else if (com_y >= length - 3)
        {
            com_y = length - 3;
        }
        ////////////////////////////////////////////////////////////////////////////
```

공 방향전환

공이 위나 아래 부분에 닿으면 방향을 전환해 안쪽으로 튀도록 만들어준다
공의 속도가 빨라지면 공이 레이아웃을 통과할 수도 있기 때문에 공의 y값 pos_y는 위 부분일 경우 1로 아래 부분일 경우 length - 1로 지정해준다
여기서 ball_flag를 사용하는 이유는 공의 속도가 느리면 닿은 뒤 바로 튀어나가지 않기 때문에 여러번 실행되는 것을 방지하기 위함이다 (ball_flag1 ~ ball_flag3까지 모두 같은 용도이다)
vel_y에 -1을 곱해주게 되면 양수는 음수로 음수는 양수로 바뀌면서 현재 y 진행 방향의 반대로 공이 이동하게된다

In [ ]:
```
        ////////////////////// 공의 방향전환 ////////////////////////////
        if (pos_y <= 1) // 공이 위 부분에 닿으면 방향을 전환해준다
        {
            pos_y = 1;
            if (ball_flag1 == 0)    // ball_flag가 0이면 방향을 바꾸어 준다
            {
                vel_y = vel_y * -1;
                ball_flag1 = 1;
            }

        }
        else if (pos_y >= length - 1) // 공이 아래 부분에 닿으면 방향을 전환해준다
        {
            pos_y = length - 1;
            if (ball_flag1 == 0)    // ball_flag가 0이면 방향을 바꾸어 준다
            {
                vel_y = vel_y * -1;
                ball_flag1 = 1;
            }
        }
        else
        {
            ball_flag1 = 0;
        }
```

공이 Computer Bar와 만났을 때

공이 Computer Bar와 만나게 되면 방향을 전환해 안쪽으로 공이 튀도록 만들어준다
Computer Bar는 com_y를 기준으로 2칸씩 더 확장되어있다 (Drawcom 함수 참고)
공의 속도가 빨라지면 공이 bar를 통과할 수도 있기 때문에 공의 x값 pos_x는 width - 1로 지정해준다
vel_x에 -1을 곱해주게 되면 양수는 음수로 음수는 양수로 바뀌면서 현재 x 진행 방향의 반대로 공이 이동하게된다

In [ ]:
```
        if (pos_x >= width - 1)         // 공이 컴퓨터 bar와 만나면 방향을 전환해준다
        {
            if ((pos_y >= com_y - 2) && (pos_y <= com_y + 2))
            {
                pos_x = width - 1;
                if (ball_flag2 == 0)
                {
                    vel_x = vel_x * -1;
                    ball_flag2 = 1;
                }

            }
        }
        else
        {
            ball_flag2 = 0;
        }
```

공이 Player Bar와 만났을 때

공이 Player Bar와 만나게 되면 방향을 전환해 안쪽으로 공이 튀도록 만들어준다
Player Bar는 bar_y를 기준으로 2칸씩 더 확장되어있다 (Drawbar 함수 참고)
공의 속도가 빨라지면 공이 bar를 통과할 수도 있기 때문에 공의 x값 pos_x는 1로 지정해준다
vel_x에 -1을 곱해주게 되면 양수는 음수로 음수는 양수로 바뀌면서 현재 x 진행 방향의 반대로 공이 이동하게된다
또한 점수를 추가해 주고 공의 속도를 높여준다

In [ ]:
```
        if (pos_x <= 1)         // 공이 player bar와 만나면 방향을 전환해준다 player bar와 만날 때 마다 점수가 올라가고 속도가 빨라진다
        {
            if ((pos_y >= bar_y - 2) && (pos_y <= bar_y + 2))
            {
                pos_x = 1;
                if (ball_flag3 == 0)
                {
                    vel_x = vel_x * -1;
                    score = score + 1; // 점수를 추가
                    if (rally_flag == rally) // 설정한 랠리만큼 진행이되면 속도를 높여줌
                    {
                        if (speed1 > 1)
                        {
                            speed1 = speed1 - 1;
                        }
                        else
                        {
                            speed2 = speed2 + 1;
                        }
                        rally_flag = 0;
                    }
                    rally_flag = rally_flag + 1;
                    ball_flag3 = 1;
                }

            }
        }
        else
        {
            ball_flag3 = 0;
        }
        ////////////////////////////////////////////////////////////////////////////
```

공이 아웃됐을 때

공이 게임창을 벗어날 경우
왼쪽(Player Bar 위치)으로 아웃되면 gameover창을 띄우고 오른쪽(Computer Bar 위치)으로 아웃되면 Win창을 띄운다

In [ ]:
```
            if (pos_x <= 0) // 공이 아웃될 때
            {
                Drawgameover(score);
                break;
            }
            else if (pos_x >= width)
            {
                Drawwin(score);
                break;
            }
```

화면 그리기

In [ ]:
```
            Drawbar(bar_y); // player bar를 그려준다
            Drawcom(com_y); // 컴퓨터의 bar를 그려준다
            Drawlayout();   // 레이아웃을 그려준다
            Sleep(10);
            system("cls");  // 화면을 초기화 해준다
        }


    }
    return 0;
}
```