06. 몬스터와 사각형 충돌

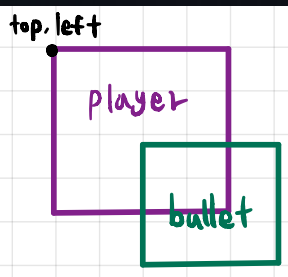
```
typedef struct _tagRectangle
{
    float l, t, r, b;
}RECTANGLE, *PRECTANGLE;

typedef struct _tagBullet
{
    RECTANGLE rc;
    float     fDist;
    float     fLimitDist;
}BULLET, * PBULLET;

typedef struct _tagMonster
{
    RECTANGLE tRC;
    float     fSpeed;      속력
    float     fTime;
    float     fLimitTime;  총알 발사 시간 주기
    int       iDir;        이동 방향
}MONSTER, * PMONSTER;

// Global Variables:
HINSTANCE hInst;                                // current instance
WCHAR szTitle[MAX_LOADSTRING];                  // The title bar text
WCHAR szWindowClass[MAX_LOADSTRING];            // the main window class name

HWND g_hWnd;
HDC  g_hDC;
bool g_bLoop = true;

RECTANGLE g_tPlayerRC = { 100, 100, 200, 200 };
MONSTER g_tMonster;

// 플레이어 총알 리스트
std::list<BULLET> g_PlayerBulletList;

// 몬스터 총알
std::list<BULLET> g_MonsterBulletList;
```

```
/* 몬스터 이동 */
g_tMonster.tRC.t += g_tMonster.fSpeed * g_fDeltaTime * fTimeScale * g_tMonster.iDir;
g_tMonster.tRC.b += g_tMonster.fSpeed * g_fDeltaTime * fTimeScale * g_tMonster.iDir;

if( g_tMonster.tRC.b >= 600 )
{
    g_tMonster.iDir = -1;
    g_tMonster.tRC.b = 600;
    g_tMonster.tRC.t = 500;
}

else if( g_tMonster.tRC.t <= 0 )
{
    g_tMonster.iDir = 1;
    g_tMonster.tRC.b = 100;
    g_tMonster.tRC.t = 0;
}

/* 몬스터 총알 생성 */
g_tMonster.fTime += g_fDeltaTime * fTimeScale;   델타 타임을 누적시킴.

if( g_tMonster.fTime >= g_tMonster.fLimitTime )   총알 발사할 시간이 된 경우
{
    g_tMonster.fTime -= g_tMonster.fLimitTime;   주기 시간을 빼줌.

    BULLET tBullet = {};

    tBullet.rc.r = g_tMonster.tRC.l;
    tBullet.rc.l = tBullet.rc.r - 50.f;
    tBullet.rc.t = (g_tMonster.tRC.t + g_tMonster.tRC.b) / 2.0f - 25.f;
    tBullet.rc.b = tBullet.rc.t + 50.f;
    tBullet.fDist = 0.f;
    tBullet.fLimitDist = 800.f;

    g_MonsterBulletList.push_back( tBullet );
}
```


```
/* 몬스터 총알 이동 */
iterEnd = g_MonsterBulletList.end();   ✓
for( iter = g_MonsterBulletList.begin(); iter != iterEnd; )
{
    iter->rc.l -= fSpeed;
    iter->rc.r -= fSpeed;

    iter->fDist += fSpeed;


    if( iter->fDist >= iter->fLimitDist )
    {
        iter = g_MonsterBulletList.erase( iter );
        iterEnd = g_MonsterBulletList.end();
    }

    else if( iter->rc.r <= 0 )
    {
        iter = g_MonsterBulletList.erase( iter );
        iterEnd = g_MonsterBulletList.end();
    }

    // 충돌 조건
    else if( g_tPlayerRC.l <= iter->rc.r && iter->rc.l <= g_tPlayerRC.r &&
        g_tPlayerRC.t <= iter->rc.b && iter->rc.t <= g_tPlayerRC.b )
    {
        iter = g_MonsterBulletList.erase( iter );
        iterEnd = g_MonsterBulletList.end();
    }

    else
    {
        ++iter;
    }
}
```

```
/* 출력 */
// 몬스터 출력
Rectangle( g_hDC, g_tMonster.tRC.l, g_tMonster.tRC.t, g_tMonster.tRC.r, g_tMonster.tRC.b );

// 플레이어 출력
Rectangle( g_hDC, g_tPlayerRC.l, g_tPlayerRC.t, g_tPlayerRC.r, g_tPlayerRC.b );

// 플레이어 총알 출력
iterEnd = g_PlayerBulletList.end();   ✓
for( iter = g_PlayerBulletList.begin(); iter != iterEnd; ++iter )
{
    Rectangle( g_hDC, iter->rc.l, iter->rc.t, iter->rc.r, iter->rc.b );
}

// 몬스터 총알 출력
iterEnd = g_MonsterBulletList.end();   ✓
for( iter = g_MonsterBulletList.begin(); iter != iterEnd; ++iter )
{
    Rectangle( g_hDC, iter->rc.l, iter->rc.t, iter->rc.r, iter->rc.b );
}

// 800*600 사각형으로 화면 덮기
// Rectangle( g_hDC, 0, 0, 800, 600 );
}
```

✓ : iterEnd 변수하나로 몬스터와 플레이어 총알 리스트의 end()를 가리키므로 계속해서 갱신된 end() 값 넣어줌.
(무엇 가리키는지 & 현재 리스트 end)



player.left ≤ bullet.right
player.right ≥ bullet.left
player.top ≤ bullet.bottom
player.bottom ≥ bullet.top