

**Implementation of**

**pocket billiard**

**Subject : Object Oriented Programming and Design**

**Team : 13**

**Student : 이정원**

**이승규**

**조범준 (Presentation speaker)**

**최범수**

**BRIEF description :**

we can play pocket billiard ,applying direct x.

**-Index-**

**1**. How to compile and execute. system requirement for compilation and execution

**2**. Description on functionality (기능) that was implemented in your SW system. - 무슨 기능이 있는지 서술.

**3**. How you implemented (important implementation issues)

**4**. The result of SW system design (it's good to include basic UML diagrams, such as class diagram and use-case diagram, in your SW design result.)

**5**. Execution results : show real examples of program execution. (use screen capture) show that each function(기능) of the SW system is working correctly.

**6**. Explain how you applied object oriented concepts to the development for your project. also explain what you felt and learned from the project.

**7**. Conclusion

**(1)How to compile and execute. system requirement for compilation and execution**

1. Check Direct X version using ‘dxdiag’

2. Download the Direct X Software Development Kit (SDK)

3. you should set up ‘Include Directories’ and ‘Library Directories’.-Edit include Directory

->Choose the folder (where SDK is installed.)\Include.

-Edit Library Directory

-> Choose the folder (where SDK is installed.)\Lib\x86

**(2) Description on functionality (기능) that was implemented in your SW system. - 무슨 기능이 있는지 서술.**

1. Check Foul(free ball)

-when a white ball put in a hole, change the turn and make freeball variable true.

-when player put in other player’s balltype, change the turn and make free ball variable true.

-when a white ball didn’t hit any balls, change the turn and make free ball variable true.

-when a white ball hit other player’s balltype before one’s balltype change

the turn and make free ball variable true.

2. When a ball put in a hole, a ball disappear in display.

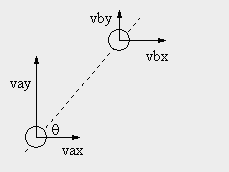
3.When a white ball reach a hole, we can reset the position of a white ball.

4.Check player’s turn

**(3) How you implemented (important implementation issues)**

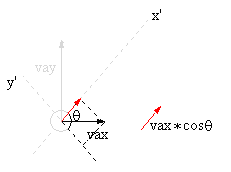
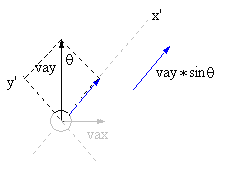
**-Implementation of ball’s strike (hit by function)**

1. **Step of classify velocity**



To see movement in detail, we classify velocity of ball A’s(struck ball)and B(strike ball)’s into vax, vaz and vbx, vbz.

1. **Define new velocity on x’-axis**



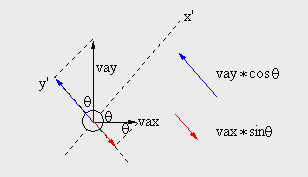
We define the new x-axis(x'), and then we calulate velocity on x'-axis using vax,vaz.

So we calculate ball A's velocity(struck ball) on x'-axis At vaxp(vax\*cosθ+vaz\*sinθ)

In the same way we calculate ball B's velocity(strike ball)on x'-axis at

(vbx\*cosθ+vbz\*sinθ).

1. **Define new velocity on z’-axis**



Also, We define the new z-axis(z'), and then we calulate velocity on z'-axis using vax,vaz.

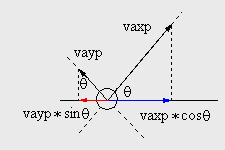
So we calculate ball A's velocity(struck ball) on z'-axis at

vazp (vaz\*cosθ-vax\*sinθ)

In the same way we calculate ball B's velocity(strike ball)on z'-axis at

vbzp (vbz\*cosθ-vbx\*sinθ)

1. **Calculate velocity after collosion**

Since vaxp, vazp, vbxp, vbzp what we used are velocity on x’-axis and z’-axis, we have to change into velocity on x-axis and z-axis. To do this ,

we define velocity after collosion using setPower function.

And also, we calculate ball A's (struck ball) velocity after collosion at

(vaxp \* cos - vazp \* sin, vaxp \* sin + vazp \* cos).

In the same way we calculate ball B's velocity(strike ball) after collosion at (vbxp \* cos - vbzp \* sin, vbxp \* sin + vbzp \* cos).

**(4) The result of SW system design (it's good to include basic UML diagrams, such as class diagram and use-case diagram, in your SW design result.)**

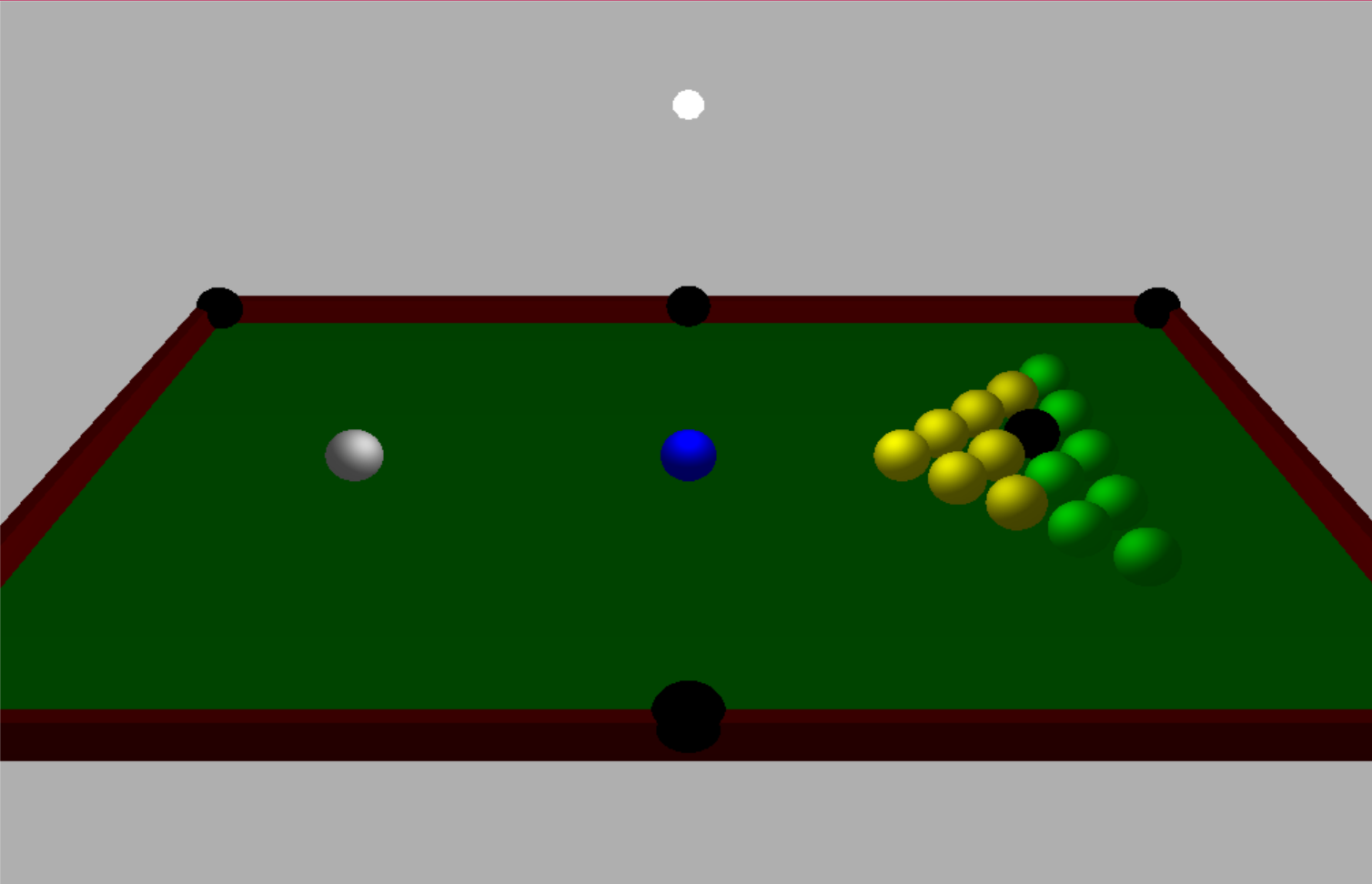
|  |
| --- |
| CSphere |
| /\*공의정보\*/  Center x,y,z  Radius  Velocity x,z  checkFoul  /\*DirectX관련\*/  m\_mLocal  m-mtrl  m\_pSphereMesh |
| Bool create()  Void destroy()  Void draw()  Bool hasIntersected()  Bool hitby()  Bool hasIntersectedToHole()  Bool hitbyHole()  getFoul()  setFoul()  ballUpdate()  getVelocity x,z()  setCenter()  setPower()  getRadius()  getLocalTransform()  getCenter() |

|  |
| --- |
| CPlayer |
| MyTurn  MyballType  LeftBall  Result  Foul |
| setMyTurn()  getMyTurn()  setMyBallType()  getMyBallType()  setLeftBall()  getLeftBall()  setresult()  getresult()  setFoul()  getFoul()  searchFirst() |

Display

**(5) Execution results : show real examples of program execution. (use screen capture) show that each function(기능) of the SW system is working correctly.**

-Initial display-



**<Element what compose the display>**

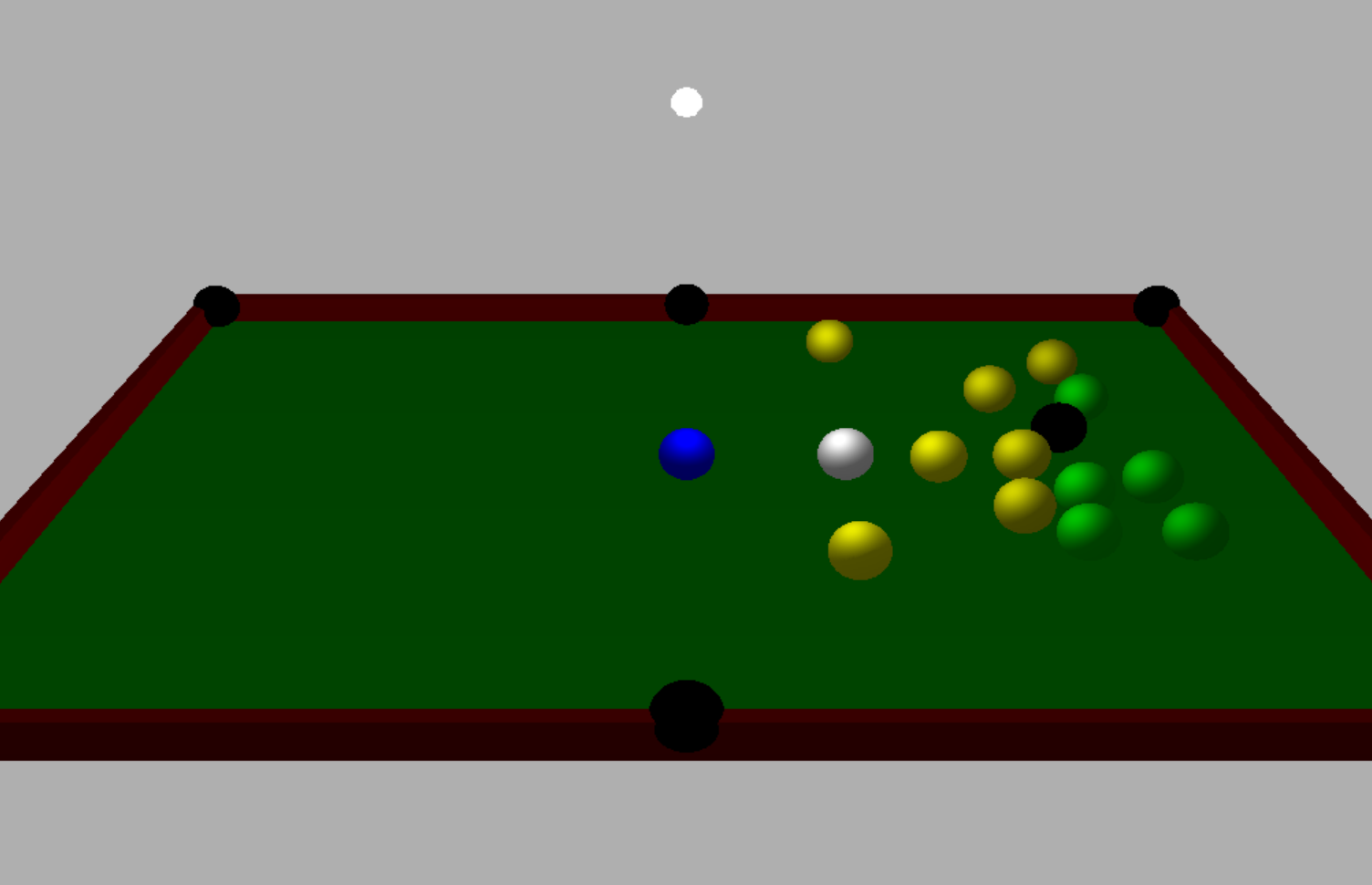
**(ball)**

In pocketball game, it has two ball type(stripe type,color type), but we simply make two bal type(green ball, yellow ball),and then We set 15 balls on the right side of plane in pyramids type, and we contain black ball in a pyramid what is important in pocketball.

**(hole)**

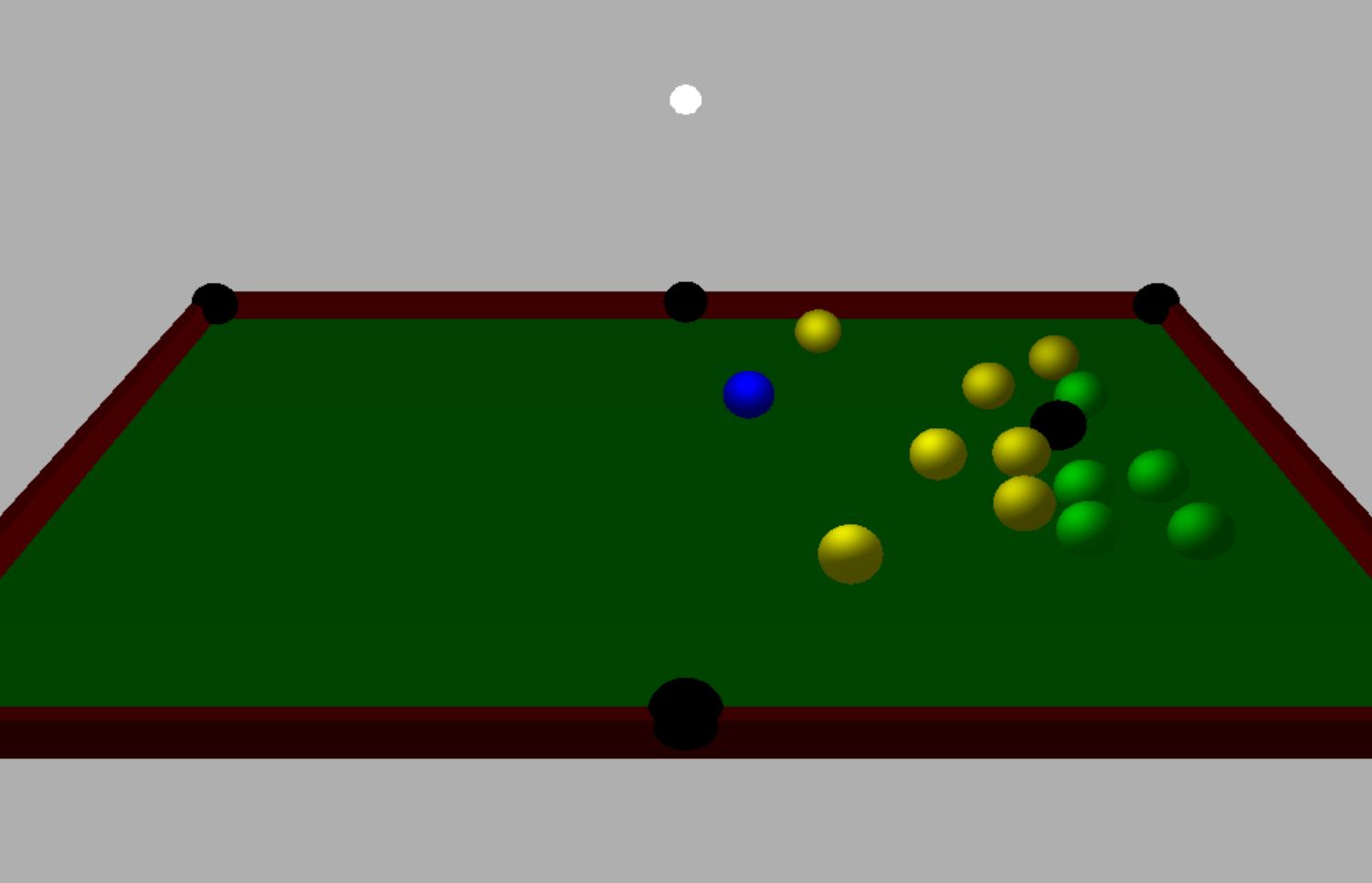
We set 6 holes on the wall using sphere type. When a ball reach a hole, It disappear in display.But when white ball reach a hole, we can reset the position of white ball using a blue ball.(free ball)

-After a collosion-



When a collosion happen, balls which reach holes disappeared in display.(There only remains 13 balls on the plane)

-When a white ball reached a hole-



When a white ball reached a hole, a white ball disappeared in display.

(Free ball)

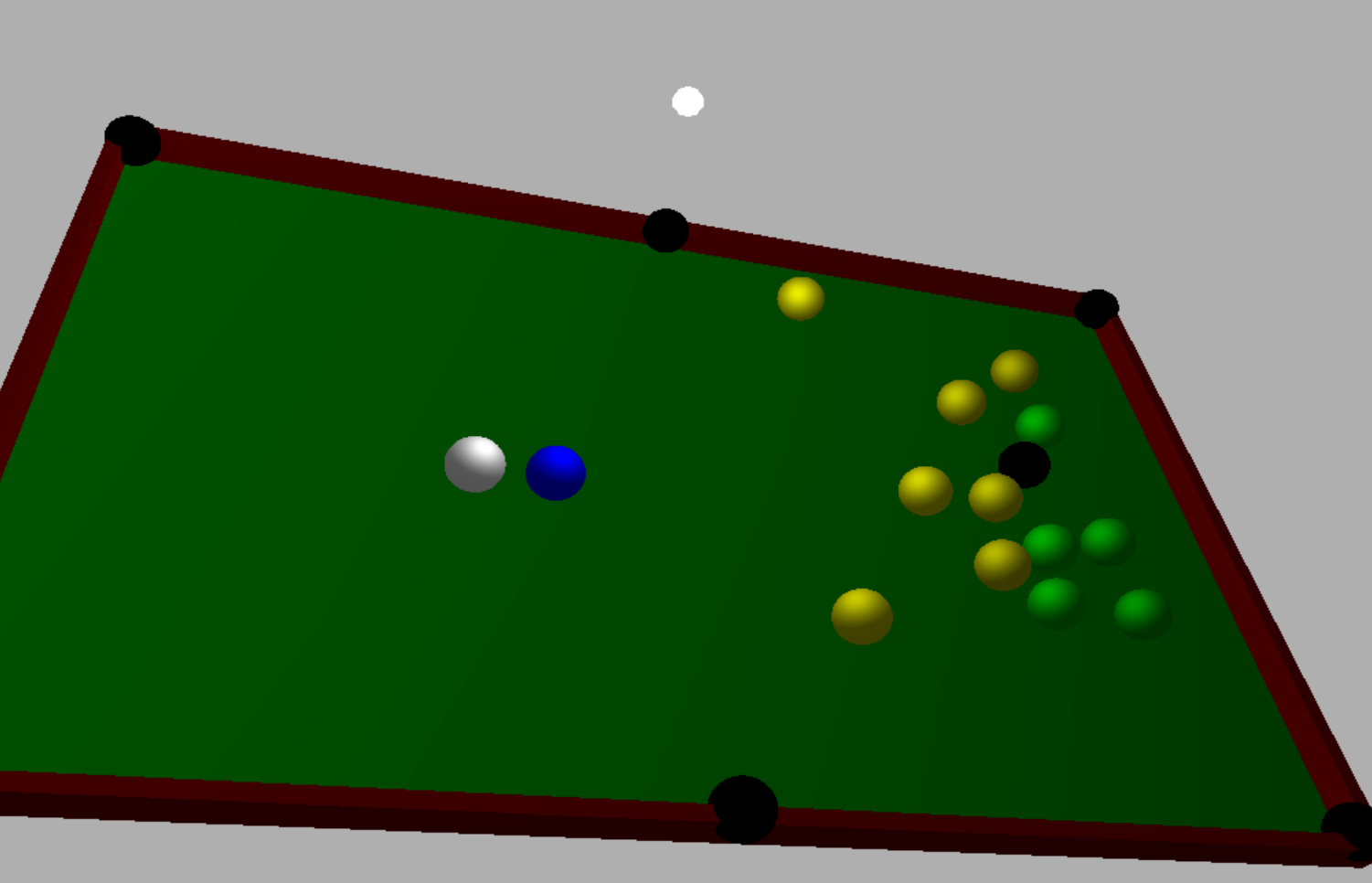
We made a hitbyhole function what reset a ball’s center and make ball’s checkfoul Boolean variable true.

(ex)

ball.setCenter(10000, 10000, 10000);

ball.checkFoul = true;

-Reset the position of white ball-



When a white ball reach a hole, we can reset the position of white ball using a blue ball as you can see in display.

(reference)

if (g\_sphere[15].getFoul() || FREEBALL == true)

{ D3DXVECTOR3 targetpos = g\_target\_blueball.getCenter();

g\_sphere[15].setCenter(targetpos.x, targetpos.y, targetpos.z);

g\_sphere[15].setPower(0.0f, 0.0f);

g\_sphere[15].setFoul();

FREEBALL = false;

//g.spherer[15]->white ball

**(6) Explain how you applied object oriented concepts to the development for your project. also explain what you felt and learned from the project.**

Data encapsulation

-  implement a class with public and private members is an example of data encapsulation and data abstraction.

Ex)

class CSphere {

private:

float center\_x, center\_y, center\_z;

float m\_radius;

float m\_velocity\_x;

float m\_velocity\_z;

bool checkFoul = false;

public:

CSphere(void)

{

D3DXMatrixIdentity(&m\_mLocal); //D3DXMatrixIdentity : 단위행렬로 초기화

ZeroMemory(&m\_mtrl, sizeof(m\_mtrl)); //zeromemory : 문자열을 NULL로 초기화

m\_radius = 0;

m\_velocity\_x = 0;

m\_velocity\_z = 0;

m\_pSphereMesh = NULL;

}

**(7) Conclusion**

When we make this program, we had to know the knowledge of direct x what is a collection of application programming interfaces (APIs) for handling tasks related to multimedia. So, In the process of applying direct x we had many problems. We realize what Direct X provide is very useful way to define multimedia.