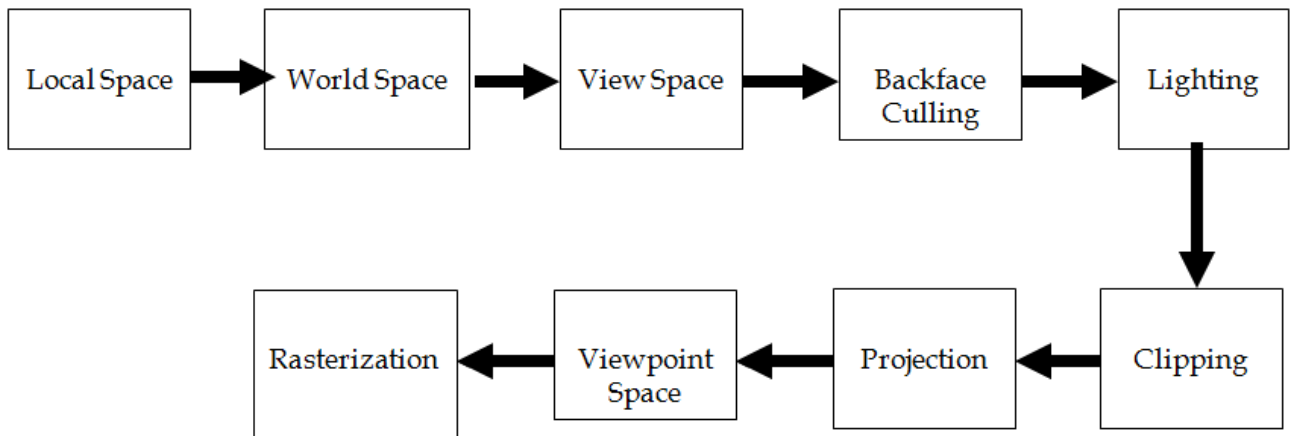


중간고사

담당교수: 단국대학교 멀티미디어공학전공 박경신

- 답은 반드시 답안지에 기술할 것. 공간이 부족할 경우 반드시 답안지 몇 쪽의 뒤에 있다고 명기한 후 기술할 것. 그 외의 경우의 답안지 뒤쪽이나 연습지에 기술한 내용은 답안으로 인정 안 함. 답에는 반드시 네모를 쳐서 확실히 표시할 것.
- 답안지에 학과, 학번, 이름 외에 본인의 암호를 기입하면 성적공고시 학번 대신 암호를 사용할 것임.

1. 다음은 기하학적 객체(모델)들로 3차원 장면을 화면에 출력하기까지의 렌더링 파이프라인 (Rendering Pipeline)을 보여주고 있다. 각각의 단계를 간단히 설명하라. (20점)



Local Space:

World Space:

View Space:

Backface Culling:

Lighting:

Clipping:

Projection:

Viewpoint Space:

Rasterization:

2. 다음은 D3DMATERIAL9 구조체를 보여주고 있다. Diffuse, Ambient, Specular, Emissive, Power가 무엇인지 구체적으로 설명하라 (10점).

```
typedef struct _D3DMATERIAL9 {  
    D3DCOLORVALUE    Diffuse;  
    D3DCOLORVALUE    Ambient;  
    D3DCOLORVALUE    Specular;  
    D3DCOLORVALUE    Emissive;  
    float             Power;  
} D3DMATERIAL9;
```

3. 다음은 D3DLIGHT9 구조체를 보여주고 있다. D3DLIGHT9이 지원하는 세 가지 광원의 타입과 각 광원을 지정하는데 필요한 변수를 설명하라 (5점). 그리고 Attenuation0, Attenuation1, Attenuation2가 무엇인지 구체적으로 설명하라 (5점).

```
typedef struct _D3DLIGHT9 {  
    D3DLIGHTTYPE      Type;  
    D3DCOLORVALUE      Diffuse;  
    D3DCOLORVALUE      Specular;  
    D3DCOLORVALUE      Ambient;  
    D3DVECTOR          Position;  
    D3DVECTOR          Direction;  
    float              Range;  
    float              Falloff;  
    float              Attenuation0;  
    float              Attenuation1;  
    float              Attenuation2;  
    float              Theta;  
    float              Phi;  
} D3DLIGHT9;
```

D3DLIGHT9이 지원하는 3가지 광원의 타입과 각 광원을 지정하는데 필요한 변수:

Attenuation0/1/2:

4. D3DXMatrixRotationX/Y/Z와 D3DXMatrixRotationYawPitchRoll 함수를 각각 설명하고, 아래의 간단한 코드를 참조하여 차이점을 설명하라 (10점).

```
D3DXMATRIX R1, Rx, Ry;
D3DXMatrixRotationX(&Rx, D3DXToRadian(45.0));
D3DXMatrixRotationY(&Ry, D3DXToRadian(45.0));
R1 = Ry * Rx ;

D3DXMATRIX R2;
D3DXMatrixRotationYawPitchRoll(&R2, D3DXToRadian(45.0), D3DXToRadian(45.0), 0.0);
```

5. 다음은 간단한 Geometry 프로그램을 보여주고 있다. 출력화면의 그림을 그리시오. 그림에 Geometry의 Vertex를 표시할 것 (10점).

```
IDirect3DVertexBuffer9* g_pGeometry_VB = 0;
struct Vertex
{
    float x, y, z;
    float nx, ny, nz;
    static const DWORD FVF;
};
const DWORD Vertex::FVF = D3DFVF_XYZ | D3DFVF_NORMAL;
Vertex g_pGeometryVertices[] =
{
    {-1.0f, 0.0f, -1.0f, 0.0f, 0.707f, -0.707f},
    { 0.0f, 1.0f,  0.0f, 0.0f, 0.707f, -0.707f},
    { 1.0f, 0.0f, -1.0f, 0.0f, 0.707f, -0.707f},
    {-1.0f, 0.0f,  1.0f, -0.707f, 0.707f, 0.0f},
    { 0.0f, 1.0f,  0.0f, -0.707f, 0.707f, 0.0f},
```

```

{-1.0f, 0.0f, -1.0f, -0.707f, 0.707f, 0.0f},
{ 1.0f, 0.0f, -1.0f, 0.707f, 0.707f, 0.0f},
{ 0.0f, 1.0f, 0.0f, 0.707f, 0.707f, 0.0f},
{ 1.0f, 0.0f, 1.0f, 0.707f, 0.707f, 0.0f},
{ 1.0f, 0.0f, 1.0f, 0.0f, 0.707f, 0.707f},
{ 0.0f, 1.0f, 0.0f, 0.0f, 0.707f, 0.707f},
{-1.0f, 0.0f, 1.0f, 0.0f, 0.707f, 0.707f},
{-1.0f, 0.0f, -1.0f, 0.0f, -1.0f, 0.0f},
{ 1.0f, 0.0f, -1.0f, 0.0f, -1.0f, 0.0f},
{-1.0f, 0.0f, 1.0f, 0.0f, -1.0f, 0.0f},
{ 1.0f, 0.0f, -1.0f, 0.0f, -1.0f, 0.0f},
{ 1.0f, 0.0f, 1.0f, 0.0f, -1.0f, 0.0f},
{-1.0f, 0.0f, 1.0f, 0.0f, -1.0f, 0.0f}
};

bool Setup()
{
    // Create vertex and index buffers for GEOMETRY
    Device->CreateVertexBuffer(
        18 * sizeof(Vertex),
        D3DUSAGE_WRITEONLY,
        Vertex::FVF,
        D3DPOOL_MANAGED,
        &g_pGeometry_VB,
        0);

    void * pVertices = NULL;
    g_pGeometry_VB->Lock( 0, sizeof(g_pGeometryVertices), (void**)&pVertices, 0 );
    memcpy( pVertices, g_pGeometryVertices, sizeof(g_pGeometryVertices) );
    g_pGeometry_VB->Unlock();
    // 중간생략...
    return true;
}

void Cleanup()
{
    d3d::Release<IDirect3DVertexBuffer9*>(g_pGeometry_VB);
}

```

```
bool Display(float timeDelta)
```

```
{
```

```
    if( Device )
```

```
    {
```

```
        Device->Clear(0, 0, D3DCLEAR_TARGET | D3DCLEAR_ZBUFFER, 0xffffffff, 1.0f, 0);
```

```
        Device->BeginScene();
```

```
// 중간생략...
```

```
        // Draw GEOMETRY.
```

```
        Device->SetStreamSource(0, g_pGeometry_VB, 0, sizeof(Vertex));
```

```
        Device->SetFVF(Vertex::FVF);
```

```
        Device->DrawPrimitive(D3DPT_TRIANGLELIST, 0, 6);
```

```
        Device->EndScene();
```

```
        Device->Present(0, 0, 0, 0);
```

```
    }
```

```
    return true;
```

```
}
```

6. Single-Pass Multi-texturing과 Multi-Pass Multi-texturing의 차이점을 간단히 설명하라 (10점). 5번 문제의 Geometry 프로그램에서 두 개의 텍스처 이미지를 읽어 들여와서 Single-Pass Multi-texturing을 하려면 추가해야 할 부분을 간단히 적어라 (10점).

7. 다음은 사각형을 블렌딩(Blending)하여 화면에 출력하는 프로그램 코드의 일부를 보여주고 있다. 빈 칸에 블렌딩 방식을 간단히 설명하고, 블렌딩 최종 값을 식으로 나타내라. 이 때 srcPixel은 C_s 로, destPixel은 C_d 로 표현한다 (10점).

블렌딩 공식: $\text{outputPixel} = \text{srcPixel} * \text{srcBlendFactor} + \text{destPixel} * \text{dstBlendFactor}$

알파 블렌딩: $\text{outputPixel} = C_s * A_s + C_d * (1 - A_s)$

bool Display(float timeDelta)

```
{
    if( Device ) {
        Device->Clear(0, 0, D3DCLEAR_TARGET | D3DCLEAR_ZBUFFER, 0xffffffff, 1.0f, 0);
        Device->BeginScene();

        // _____
        // _____

        Device->SetRenderState(D3DRS_SRCBLEND, D3DBLEND_ONE);
        Device->SetRenderState(D3DRS_DESTBLEND, D3DBLEND_ZERO);
        Device->SetTexture(0, Tex1);

        Device->SetStreamSource(0, Quad, 0, sizeof(Vertex));
        Device->SetFVF(Vertex::FVF);
        Device->DrawPrimitive(D3DPT_TRIANGLELIST, 0, 2);

        // _____
        // _____

        Device->SetRenderState(D3DRS_SRCBLEND,D3DBLEND_ZERO);
        Device->SetRenderState(D3DRS_DESTBLEND,D3DBLEND_SRCCOLOR);
        Device->SetTexture(0, Tex2);

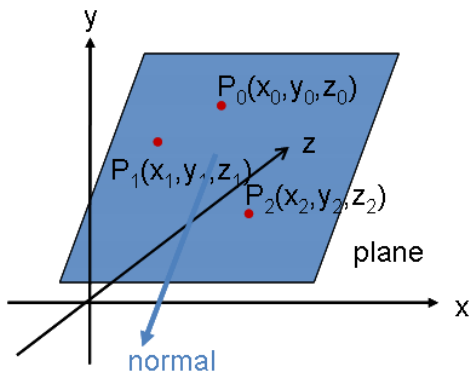
        Device->SetStreamSource(0, Quad, 0, sizeof(Vertex));
        Device->SetFVF(Vertex::FVF);
        Device->DrawPrimitive(D3DPT_TRIANGLELIST, 0, 2);

        Device->EndScene();
        Device->Present(0, 0, 0, 0);
    }
    return true;
}
```


8. 스텐실 버퍼 (Stencil buffer)에 대해 설명하고, 스텐실 버퍼를 활용하는 대표적인 사례를 명시하고 구현 방법을 간단히 적어라 (10점).

9. 다음은 평면을 만들어내는 Direct3D 코드의 일부이다. 빈 칸을 완성하시오 (extra 10점).

주의: 평면의 법선 벡터를 고려하여 winding order를 잡아야 함.



```
void GetPlaneFromPoints(D3DXVECTOR3 p0, D3DXVECTOR3 p1, D3DXVECTOR3 p2, D3DXPLANE& out)
{
    D3DXVECTOR3 u, v, n;
    u = _____;
    v = _____;
    D3DXVec3Cross(_____, _____);
    D3DXVec3Normalize(_____, _____);
    FLOAT d = -D3DXVec3Dot(_____, _____);
    out.a = n.x;
    out.b = n.y;
    out.c = n.z;
    out.d = d;
}
```