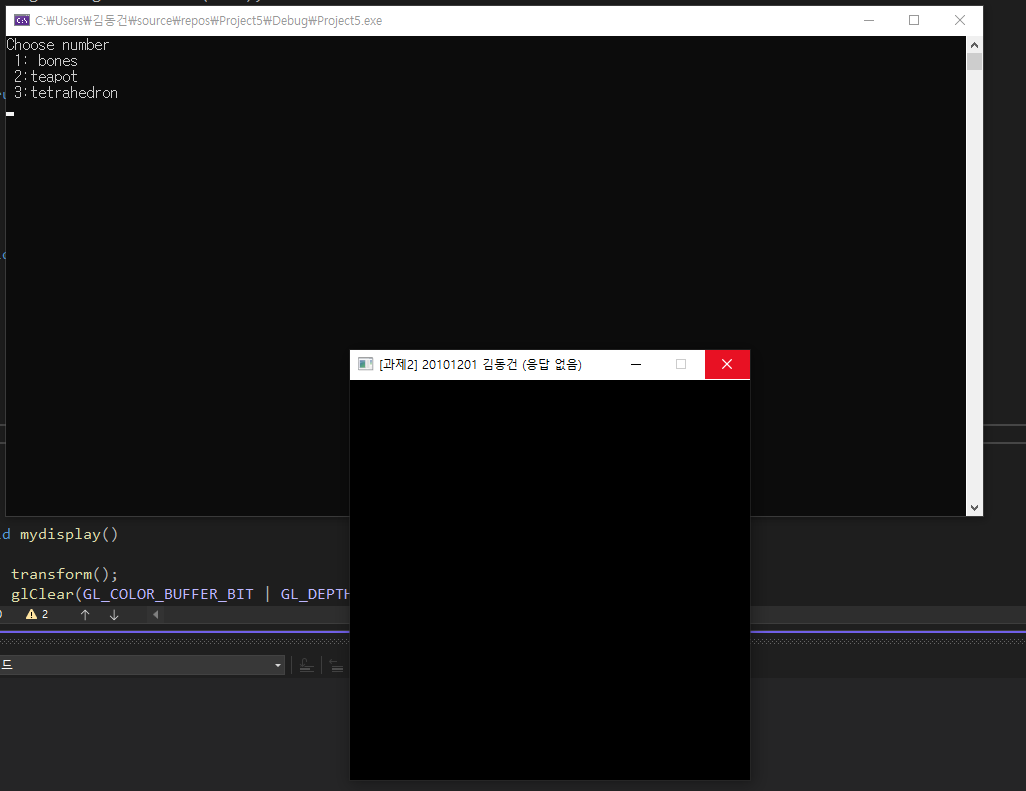
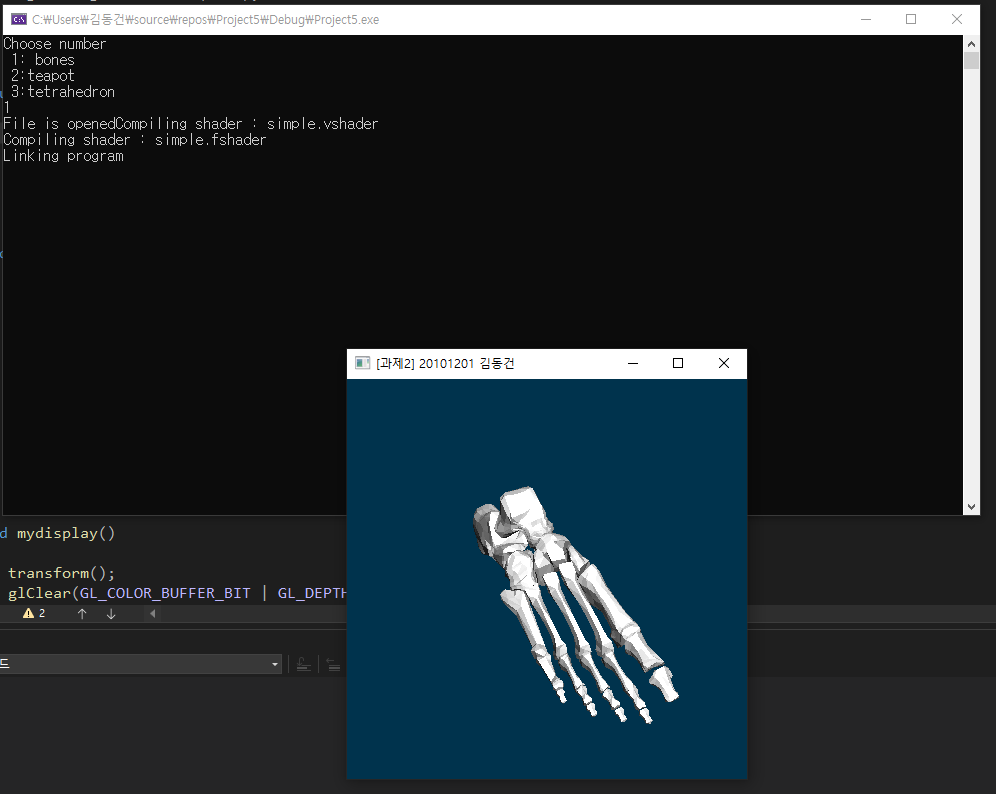
Computer Graphics Assignment #2

20101201 김동건

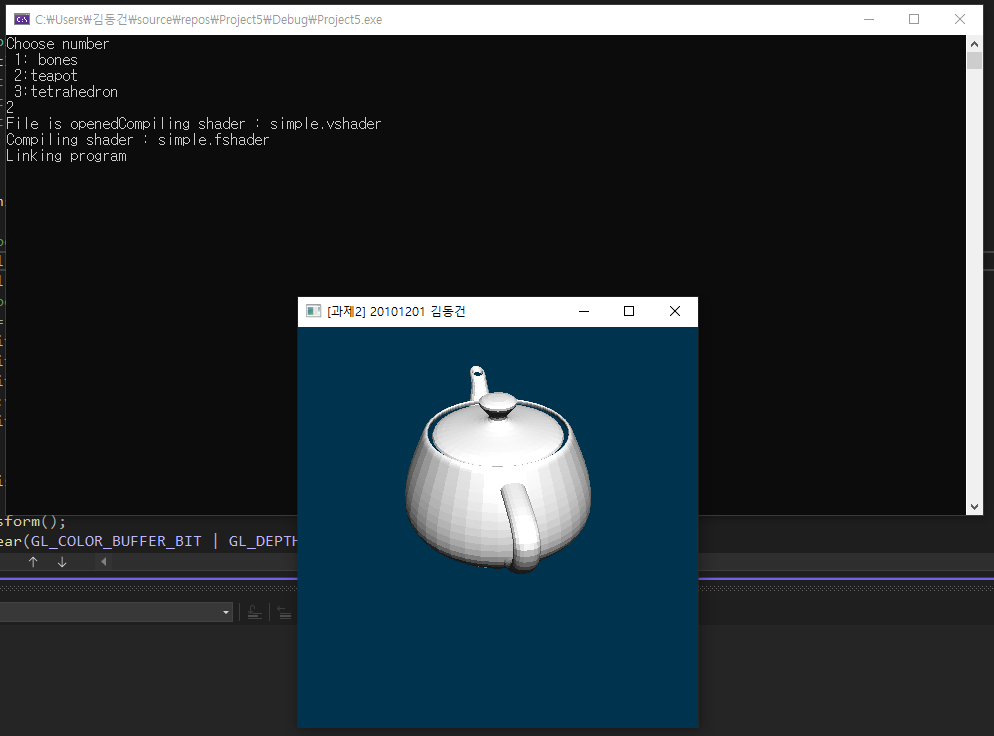
* 실행 시 첫 화면



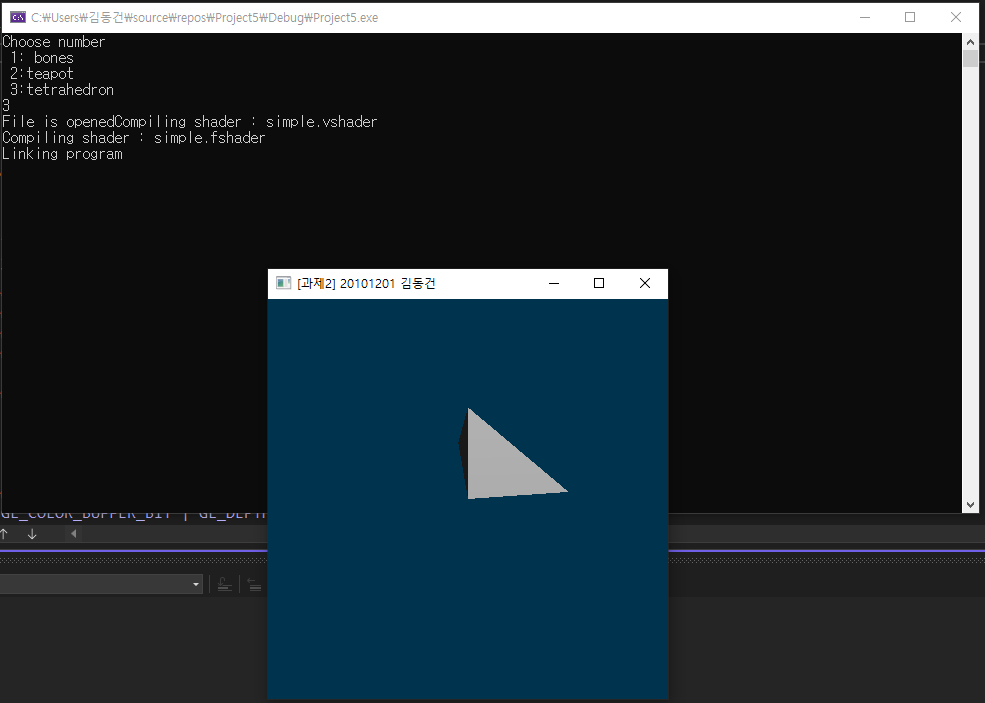
* 1 입력(bones.obj)



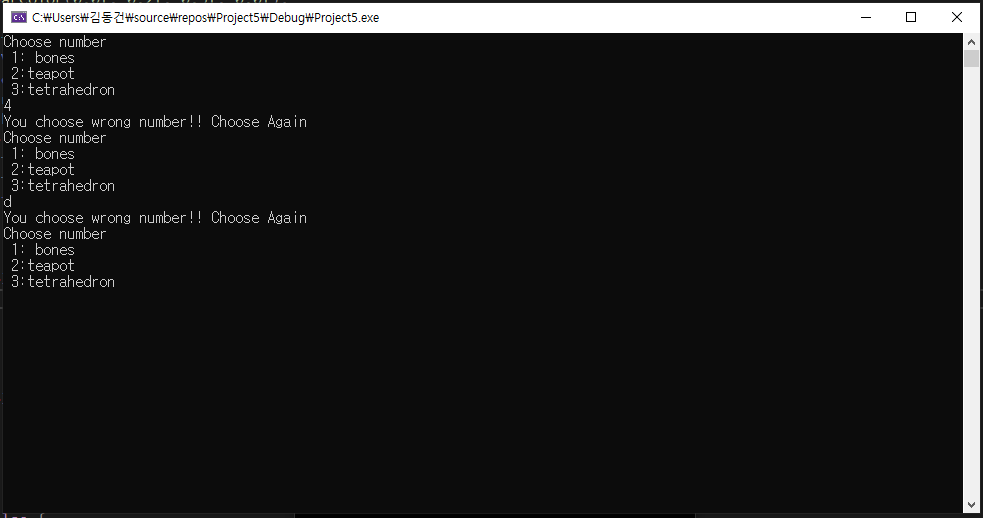
* 2 입력(teapot.obj)



* 3 입력(tetrahedron.obj)



* 그 외의 입력들일 경우는 다시 고르도록 반복



void init()

{

…

char num;

while (1) {

std::cout << "Choose number\n 1: bones\n 2:teapot\n 3:tetrahedron\n";

std::cin >> num;

if (num == '1') {

loadOBJ("bones.obj", vertices);

break;

}

else if (num == '2') {

loadOBJ("teapot.obj", vertices);

alpa = 5;

beta = 7;

break;

}

else if (num == '3') {

loadOBJ("tetrahedron.obj", vertices);

alpa = 2;

beta = 3;

break;

}

else {

std::cout << "You choose wrong number!! Choose Again" << std::endl;

}

}

glGenBuffers(1, &VertexBufferID);

glBindBuffer(GL\_ARRAY\_BUFFER, VertexBufferID);

glBufferData(GL\_ARRAY\_BUFFER, sizeof(glm::vec3) \* vertices.size(), vertices.data(), GL\_STATIC\_DRAW);

auto getNormal = [](const glm::vec3& point1, const glm::vec3& point2, const glm::vec3& point3)

{

glm::vec3 edge1 = point2 - point1;

glm::vec3 edge2 = point3 - point2;

return glm::normalize(glm::cross(edge1, edge2));

};

for (int i = 0; i < (vertices.size() - 2); i += 3) {

glm::vec3 normal = getNormal(vertices[i], vertices[i + 1], vertices[i + 2]);

normals.push\_back(normal);

normals.push\_back(normal);

normals.push\_back(normal);

}

glGenBuffers(1, &NormalBufferID);

glBindBuffer(GL\_ARRAY\_BUFFER, NormalBufferID);

glBufferData(GL\_ARRAY\_BUFFER, sizeof(glm::vec3) \* normals.size(), normals.data(), GL\_STATIC\_DRAW);

…

}

While 문으로 번호에 따라 해당 오브젝트 출력하도록 하였다. 1,2,3 외의 숫자나 char가 들어오면 다시 질문으로 돌아간다. 2와 3 입력 시에는 alpa와 beta를 바꾸는데 이는 Viewpoint 를 바꿔주는 것이다. Bones와 teapot은 뷰볼륨을 벗어나서 시점을 더 뒤로 하여 뷰볼륨 안에 들어가게 하였다. 그 이후는 loadOBJ로 obj파일을 읽은 후 vertices에 값들을 집어넣고 normal벡터도 구하여 버퍼 안에 입력하였다. for문을 사용하여 vertices의 크기만큼 normal의 값을 구하고 normals 벡터안에 집어넣었다. 총 3개씩 넣기 때문에 i값은 3씩 증가시켰다.

void myreshape(int w, int h)

{

…

View = glm::lookAt(

glm::vec3(alpa, beta, beta),

glm::vec3(0, 0, 0),

glm::vec3(0, 1, 0)

);

transform();

}

alpa=7, beta=11로 초기 세팅되어있다.

Simple.vshader #version 330 core

layout(location = 0) in vec3 vertexPosition;

layout(location = 1) in vec3 vertexColor;

layout(location = 2) in vec3 vertexNormal;

out vec3 fragmentColor;

out vec3 positionWorldspace;

out vec3 normalViewspace;

out vec3 eyeDirectionViewspace;

out vec3 lightDirectionViewspace;

uniform mat4 MVP;

uniform mat4 V;

uniform mat4 M;

uniform vec3 LightPosition;

void main() {

gl\_Position = MVP \* vec4(vertexPosition, 1.0);

positionWorldspace = (M \* vec4(vertexPosition, 1.0)).xyz;

vec3 vertexPositionViewspace = (V \* M \* vec4(vertexPosition, 1.0)).xyz;

eyeDirectionViewspace = vec3(0, 0, 0) - vertexPositionViewspace;

vec3 lightPositionViewspace = (V \* vec4(LightPosition, 1.0)).xyz;

lightDirectionViewspace = lightPositionViewspace - vertexPositionViewspace;

normalViewspace = (V \* M \* vec4(vertexNormal, 0.0)).xyz;

fragmentColor = vertexColor + 1.0;

}

main에서 fragmentColor를 하얀색으로 바꿔주기 위해 1.0을 더해주었다. 원래 0.0f였기 때문에 1.0을 더하여 흰색으로 쉐이딩하였다.