심화전공실습 (CGL)

HW05_3D Programming

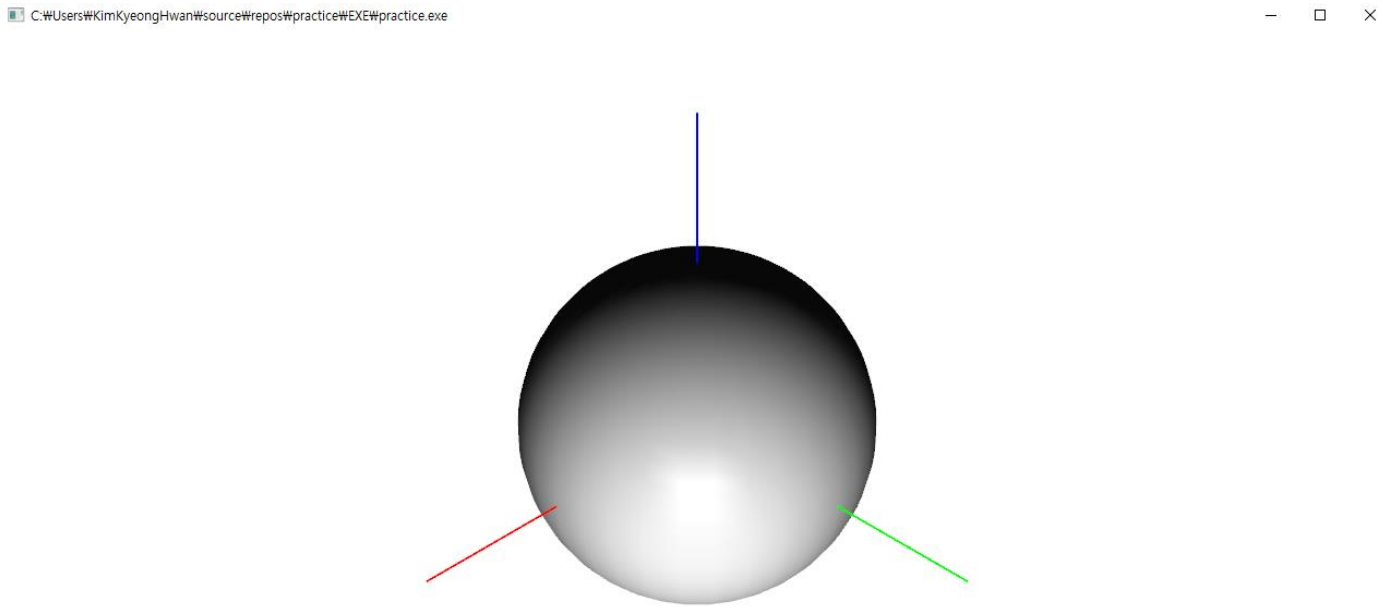
Self-scoring table

	P1	P2	P3	E1	E2	Total
Score	**1**	**1**	**1**	**1**	**1**	**5**

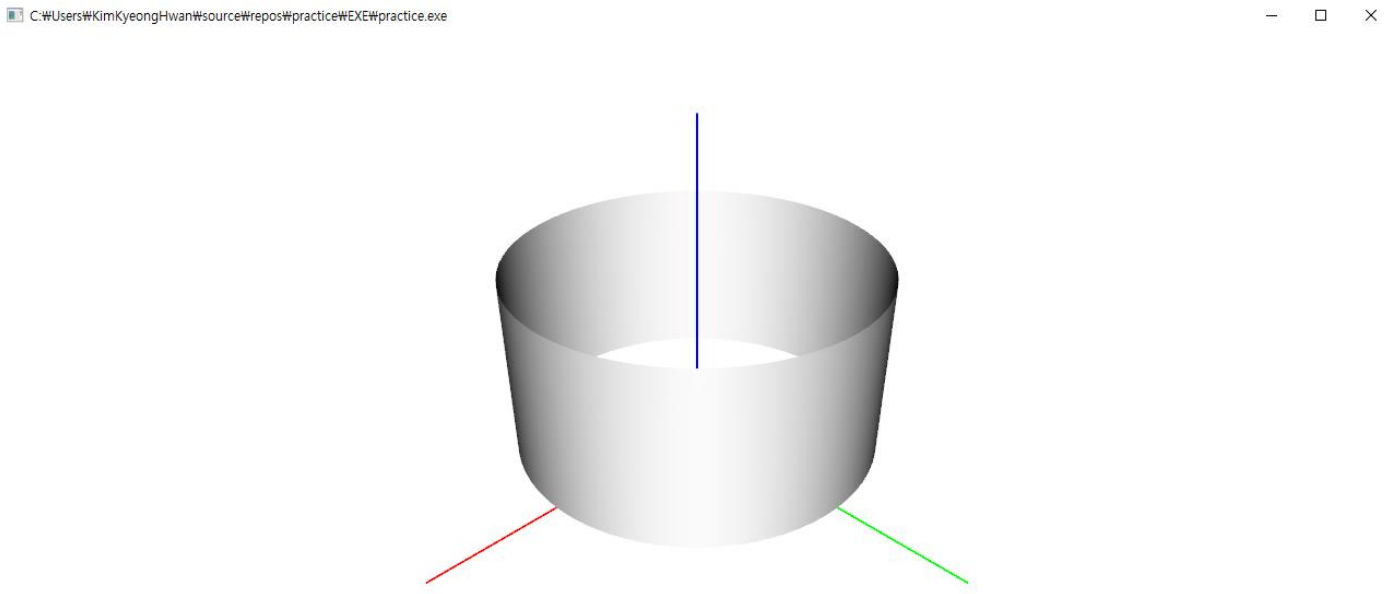
2018707068 김경환

KWANGWOON UNIVERSITY

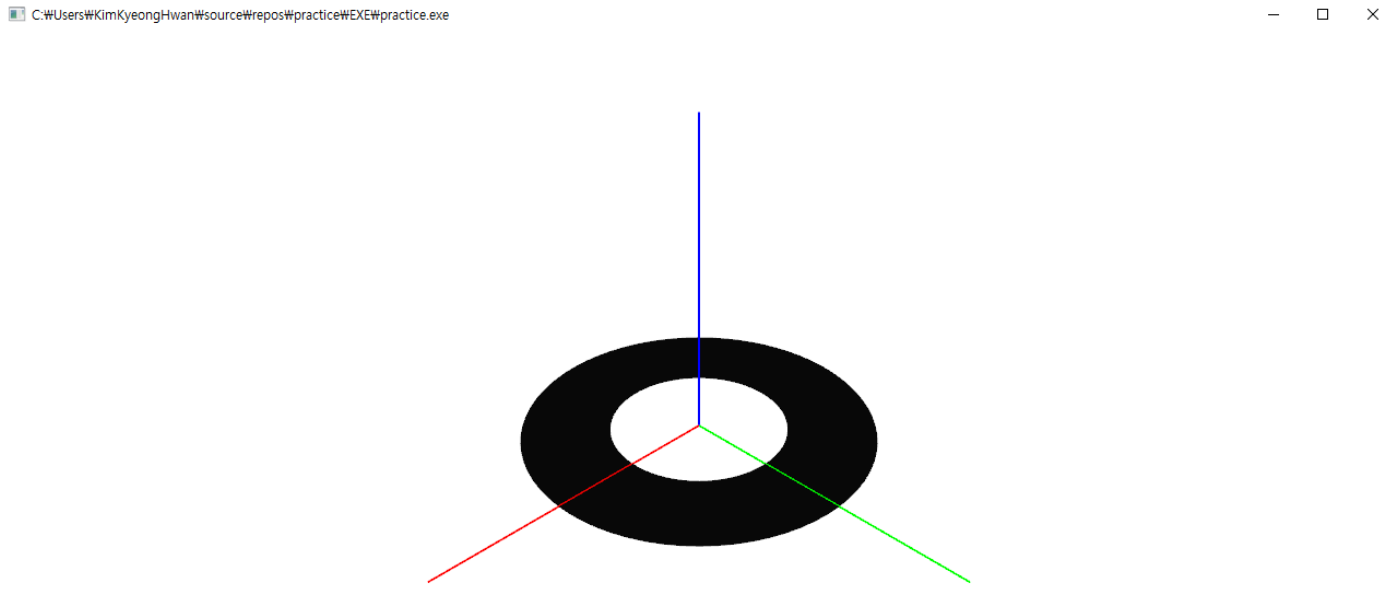
Practice01 Snapshot, Explanation:



main문의 Argument로 1을 준 상태로, OpenGL의 quardric objects인 sphere를 Polygon fill on 상태로 그리고 있다.

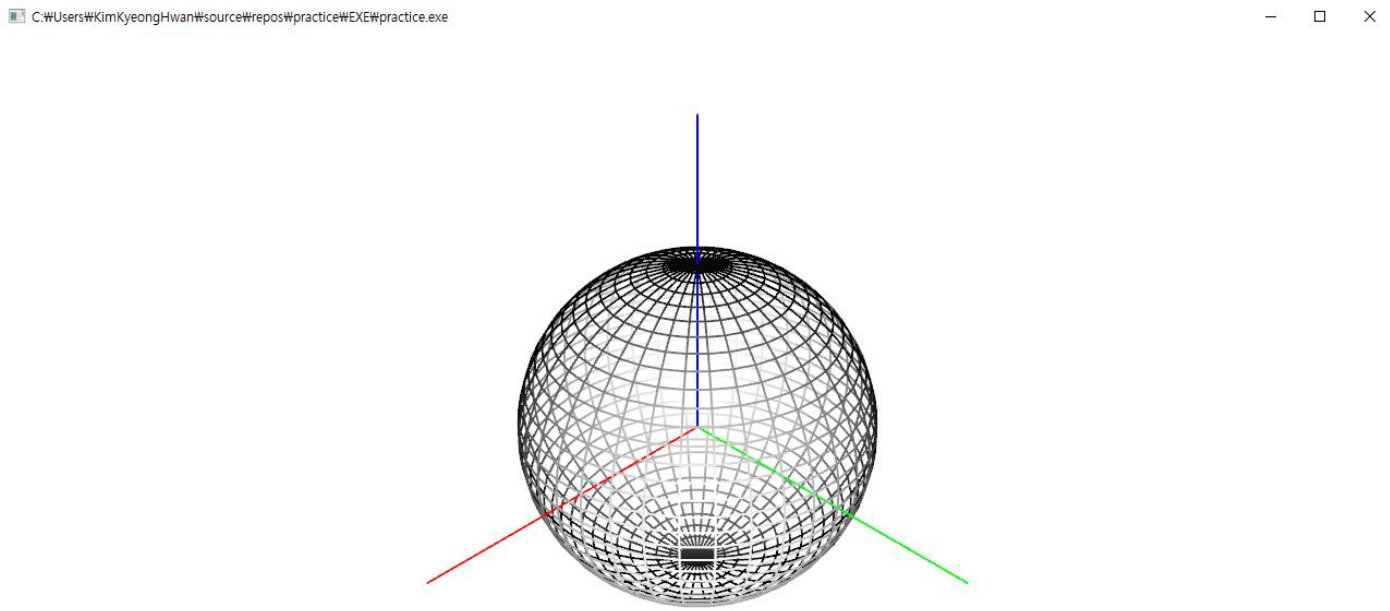


main문의 Argument로 2을 준 상태로, OpenGL의 quardric objects인 cylinder를 Polygon fill on 상태로 그리고 있다.

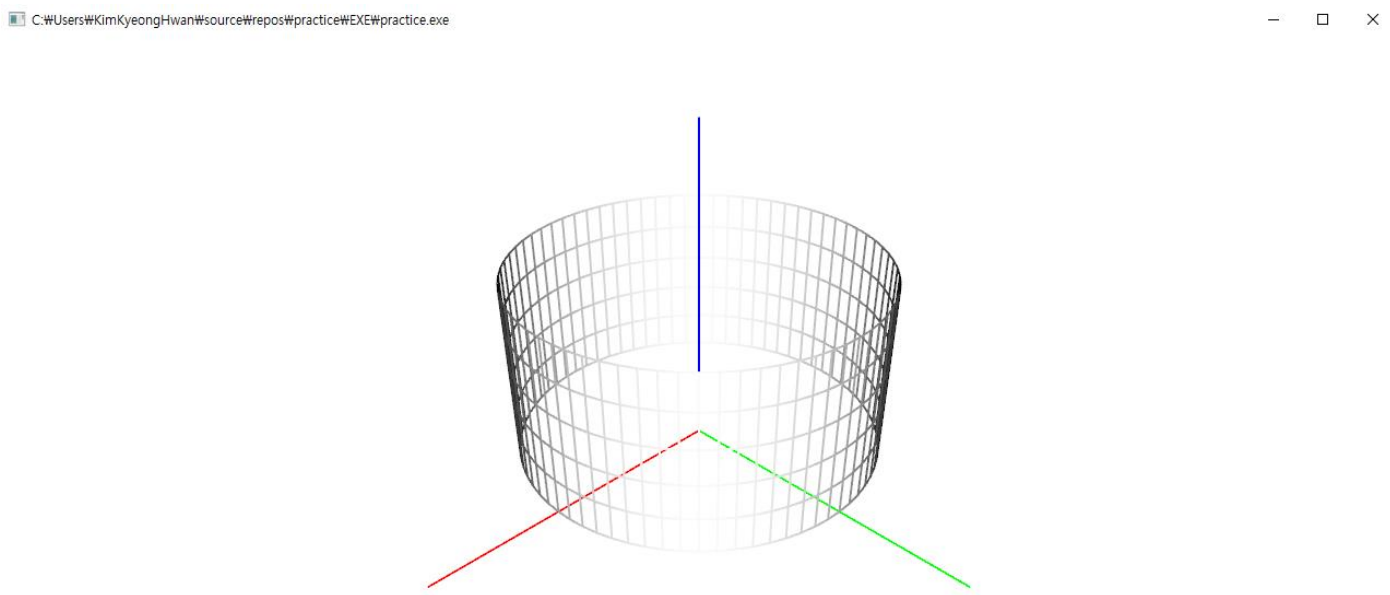


main문의 Argument로 3을 준 상태로, OpenGL의 quardric objects인 disk를 Polygon fill on 상태로 그리고 있다.

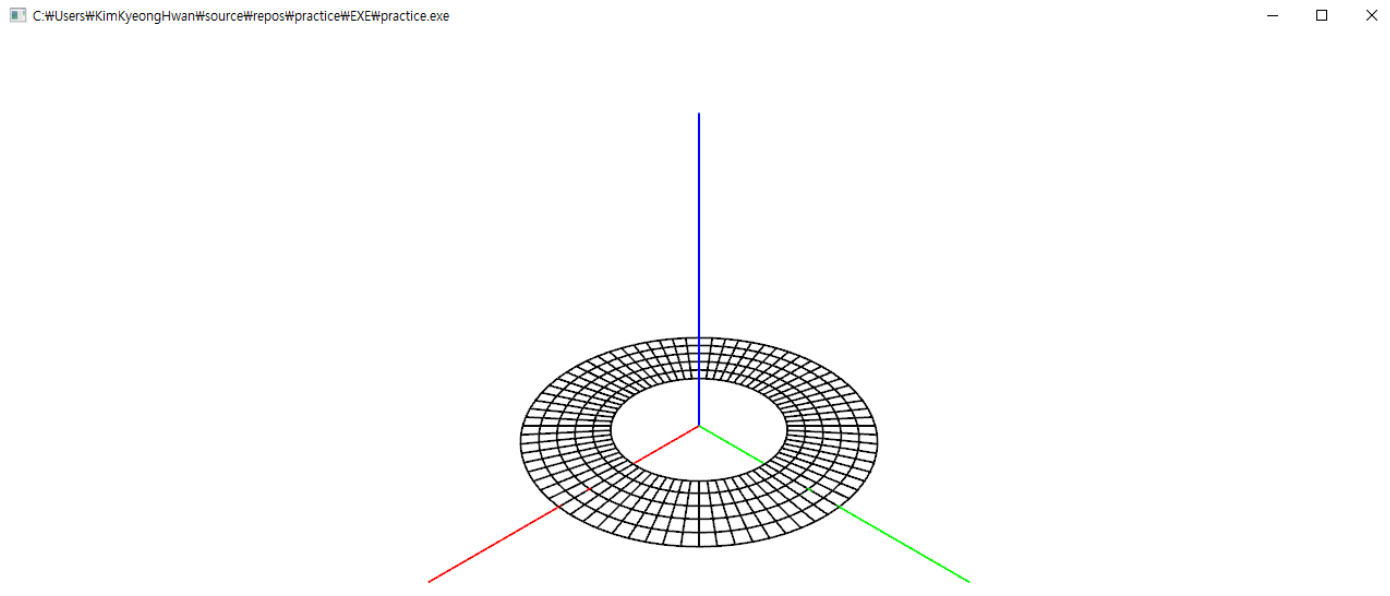
Practice02 Snapshot, Explanation:



main문의 Argument로 1을 준 상태로, OpenGL의 quardric objects인 sphere를 Polygon fill off 상태로 그리고 있다.

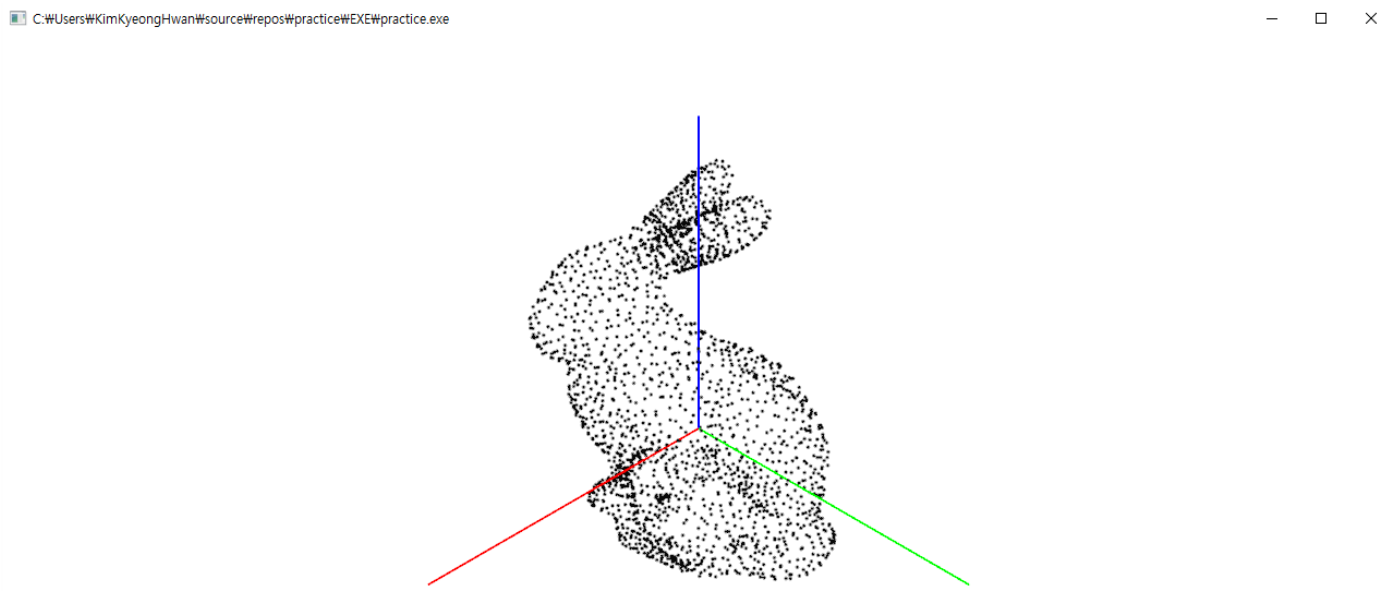


main문의 Argument로 2을 준 상태로, OpenGL의 quardric objects인 cylinder를 Polygon fill off 상태로 그리고 있다.



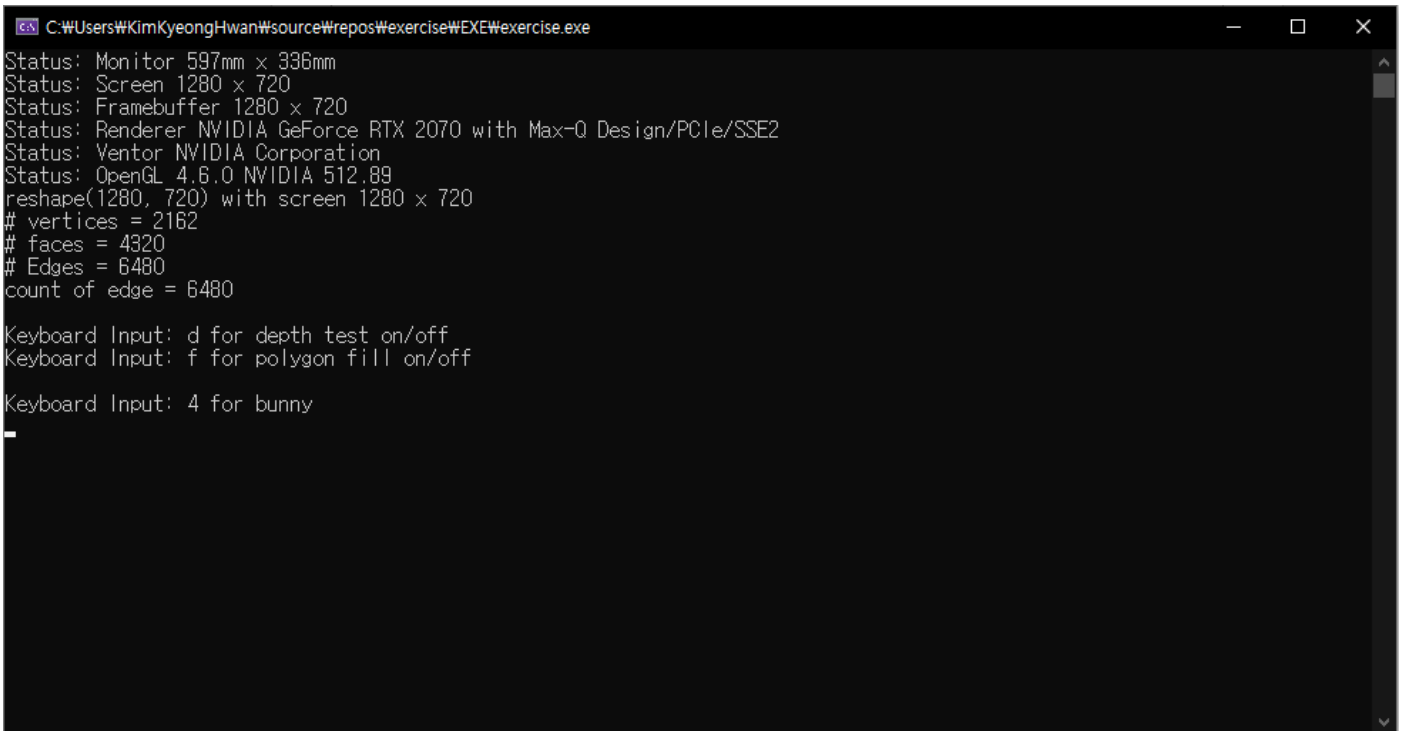
main문의 Argument로 3을 준 상태로, OpenGL의 quardric objects인 disk를 Polygon fill off 상태로 그리고 있다.

Practice03 Snapshot, Explanation:



Bunny model의 Polygonal Mesh file을 read하고 points를 사용해 이를 그린 모습이다.

Exercise01 SnapShot, Explanation:

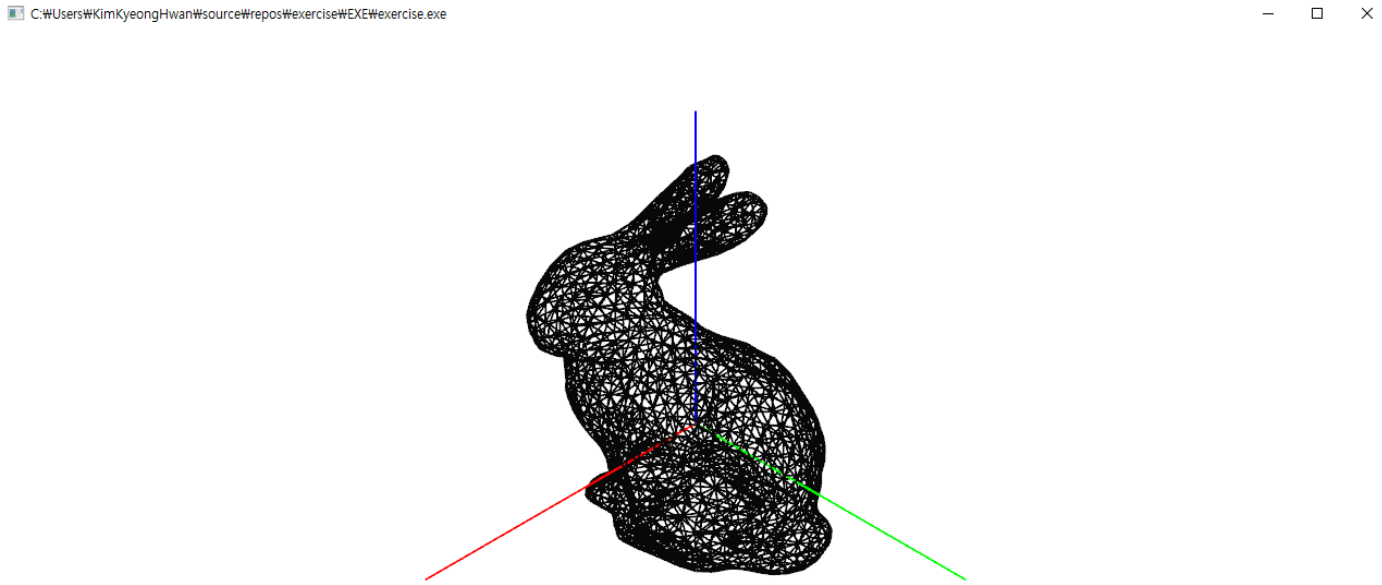


우선 # Edges = 6480의 출력문을 보면 이는 m01_bunny.off file에 나와있는 # edges를 읽어 출력한 것이다.

또한, 그 밑에 출력문인 count of edge = 6480을 보면 이는 m01_bunny.off file의 다각형 vertex 개수, vertex index를 통해 edge의 개수를 추출해낸 것이다.

추출하는 과정에 대해 간략하게 설명하자면 vector안에 vector가 있는 2차원 vector를 사용하였다. 여기서 vertices의 개수는 program이 실행되고 나서 알 수 있음으로 내부 vector의 크기는 동적할당이 되야 한다. 그리하여 readMesh 함수의 vertex x, y, z를 저장하는 반복문에서 빈 vector를 추가하였다.
그 이후 2차원 vector의 모든 내부 vector에 접근하여 해당 size를 합하고 이를 출력하였다.

Exercise02 SnapShot, Explanation:



위 과정에서 추출해낸 겹치지 않는 모든 edge의 vertex 정보가 저장되어 있는 2차원 vector의 모든 element에 접근해 이를 GL_LINES로 출력한 모습이다.
Bunny model를 이루는 모든 edge를 한 번씩만 세어 2차원 vector에 저장하였고, 이러한 정보만을 이용하여 bunny를 그렸으므로 중복되게 그려진 부분은 없다.

이번 과제는 기본적인 3D quardric object를 그려보고, Polygon Mesh File을 읽어 bunny를 그려보았다. 그리고 여기서 Polygon Mesh File에 저장된 polygon 정보를 통해 모든 edge의 정보를 단 한 번씩만 저장하였고, 이를 통해 GL_LINES로 그려지는 bunny를 출력해보았다.
최근 사용하는 언어가 c++이 아닌 python이었기 때문에 해당 자료구조를 사용하는데 어려움이 있었던 것 같다. 하지만, 다시 c++를 공부하면서 이에 대한 내용을 상기할 수 있어서 좋았던 것 같다.