**Computer Graphics Assignment 2**

**2015005205**

**최홍규**

1. **실행방법**

python3에서 실행되도록 하였다. python3 assignment1.py을 입력하여 실행한다.

파이썬 버전 : Python 3.7.1

왼쪽,오른쪽 마우스 클릭 드래그와 마우스 스크롤을 통해 카메라를 옮길 수 있다.

OBJ 파일을 window창에 drag and drop하면 window에 도형이 그려진다.

1. **구현사항**
2. **drag and drop rendering(삼각형과 사각형을 모두 구현)**



drop\_callback함수를 통해 drag and drop을 구현하였다. 파일을 읽어 한 줄씩 보면서 v, vn, f를 구별하여 리스트에 따로 따로 저장한다.

threevarr, fourvarr에 각각 삼각형, 사각형에 필요한 vertexarray이다.

threevvarr, fourvvarr은 직접 계산한 normal로 만든 vertexarray이다.

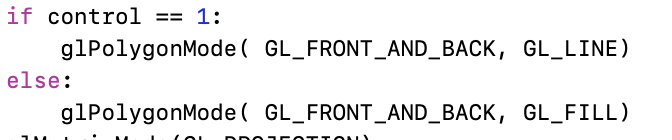
삼각형과 사각형을 rendering하도록 구현하였다. 사각형 이상의 다각형은 처음 세개의 vertex를 이용하여 삼각형만 그렸다.

obj파일을 drag drop할 때 마다 하나의 도형만 화면에 출력한다.

1. **Toggle wireframe / solid mode by pressing Z key**

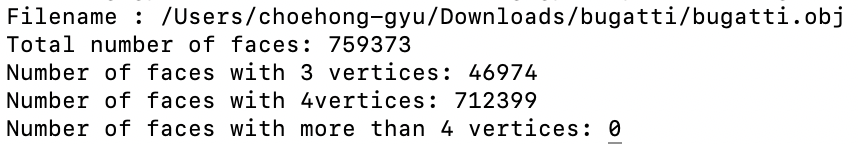
z key를 누르면 control = 0 에서 1 로 바뀌고 wireframe mode가 된다.

default로 solid mode로 되어있다.

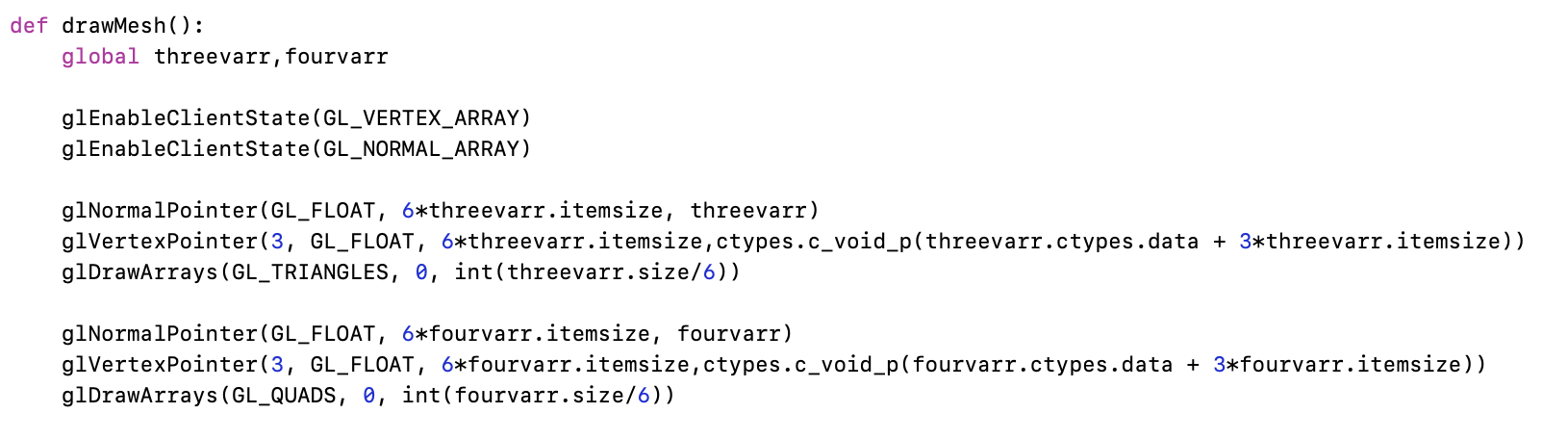


1. **print out information of obj file**

파일의 이름과 전체 face의 개수, 삼각형과 사각형 face의 개수 그 이상의 다각형의 face의 개수를 출력하였다.

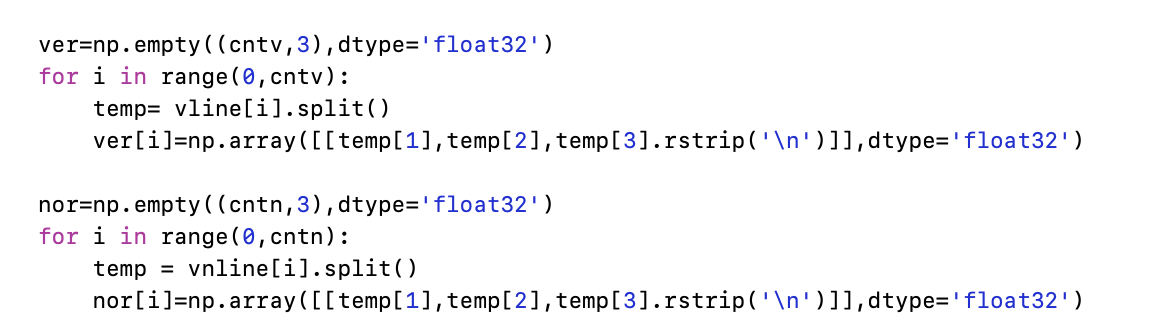


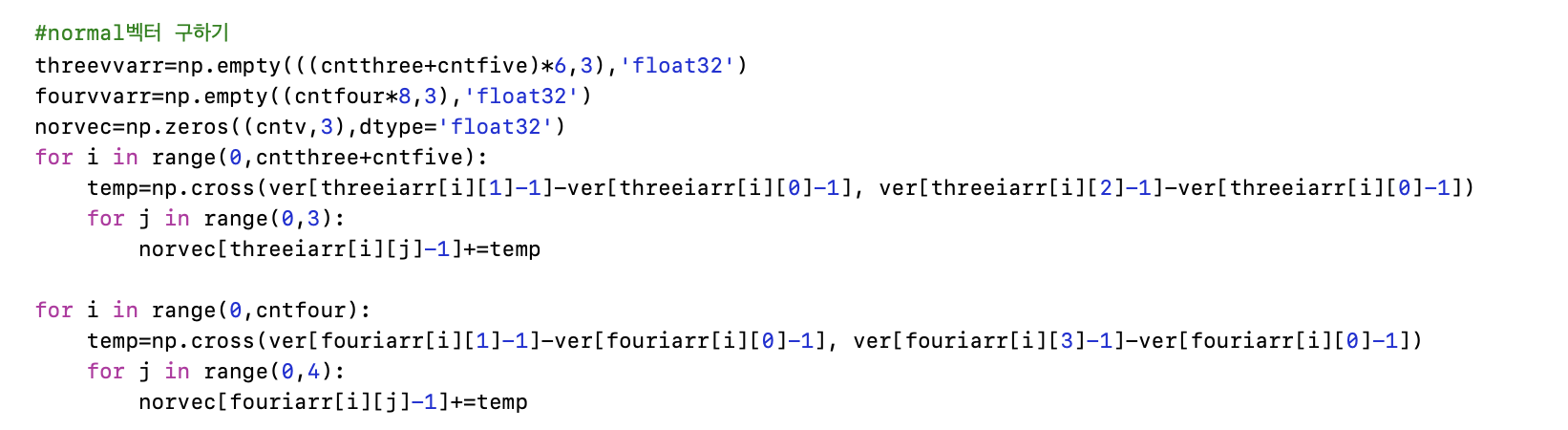
1. **using glDrawArray() to rendering**



도형을 그릴 때 glDrawArray를 이용하였다.

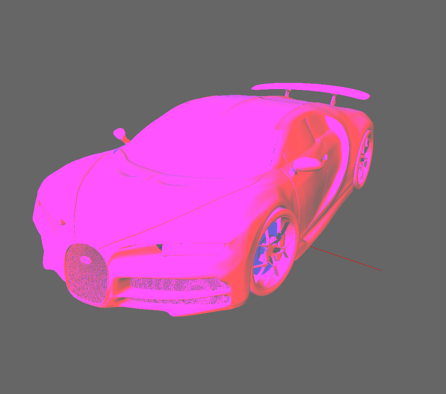
1. **Toggle [shading using normal data in obj file] / [forced smooth shading] by pressing S key**

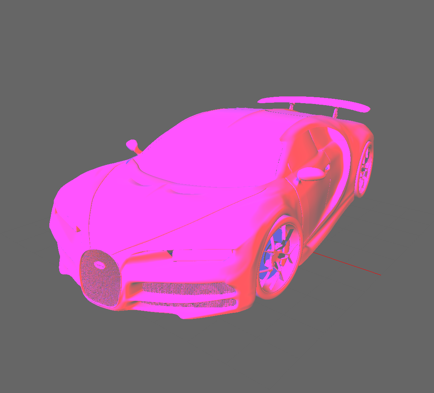




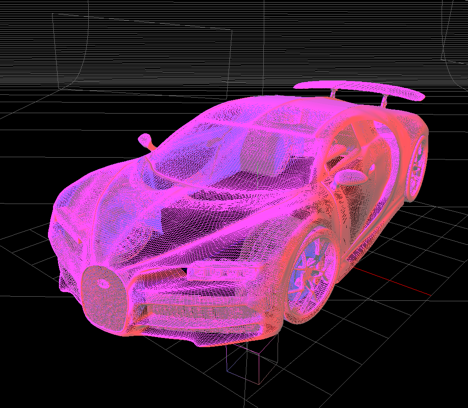
각 점의 normal을 직접 계산하였다. 점 주위의 도형들의 normal벡터(cross product로 계산)의 평균을 그 점의 normal로하였다. s key를 눌러 두가지 모드로 전환할 수 있다. normal이 없는 obj파일의 경우 직접 구한 normal로만 rendering 하였다.

1. **스크린샷**

**[shading using normal data in obj file] [forced smooth shading]**



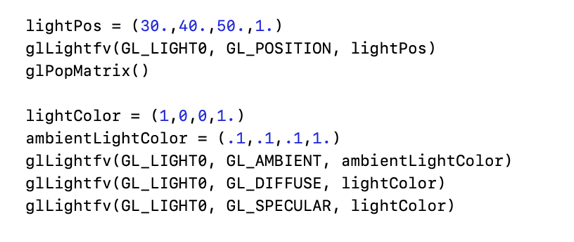
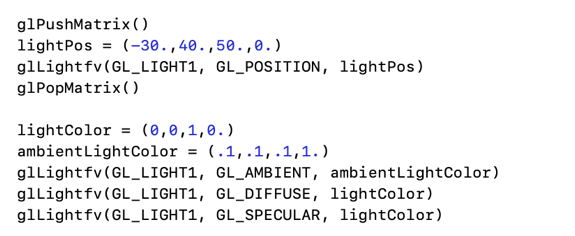
**wireframe mode**

****

1. **빛의 구현**

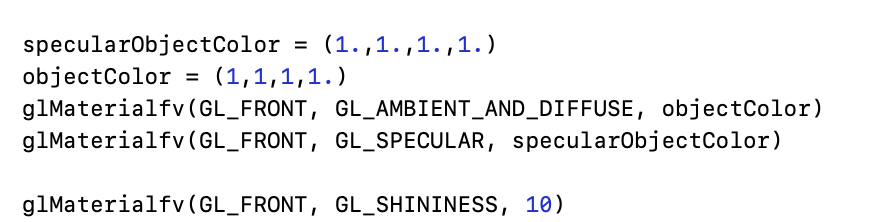
두가지 빛을 구현하였다.

**Light0 Light1**



첫번째 빛[Light0]은 빨간색으로 빛의 위치를 (30,40,50,1)으로 하였다.

두번째 빛[Light1]은 파란색으로 빛의 위치를 (-30,40,50,0)으로 하였다.



오브젝트의 색은 하얀색으로 하였다.