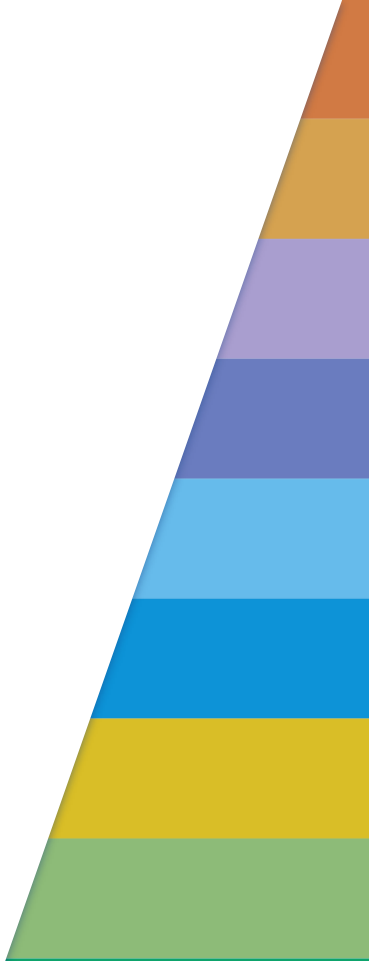


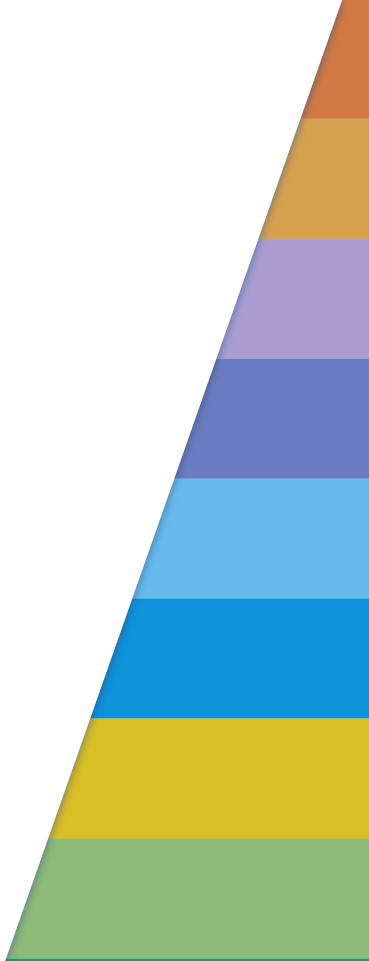
3D Graphics Programming

T163 - Game Programming

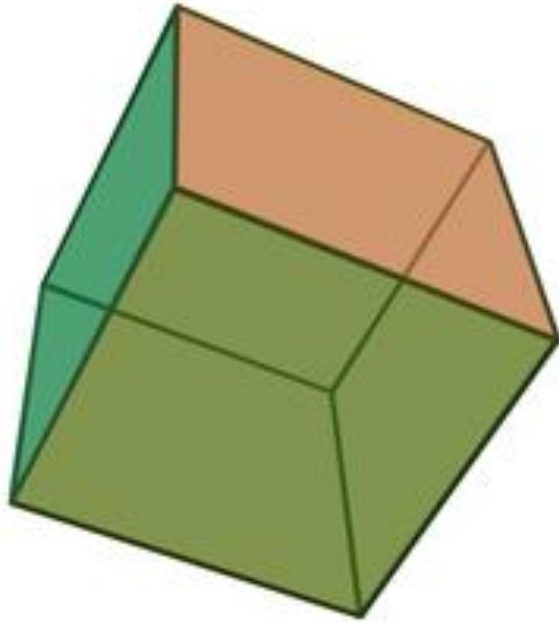
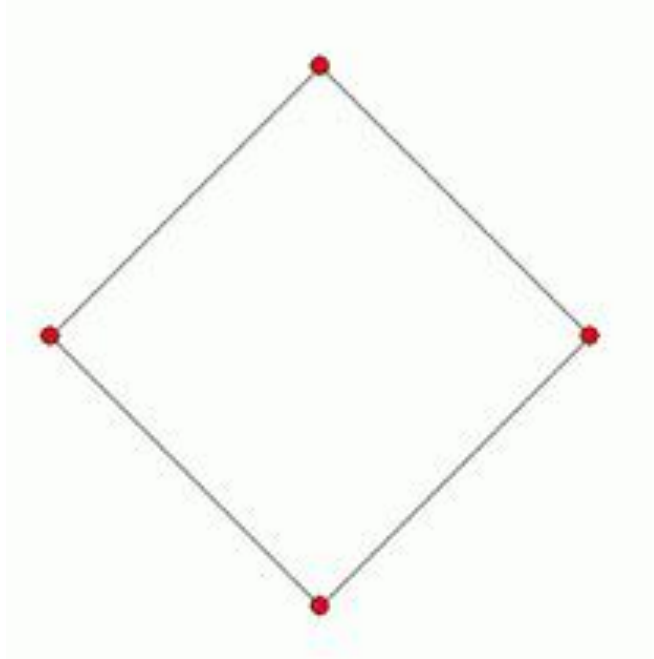


Week 4

3D Geometry

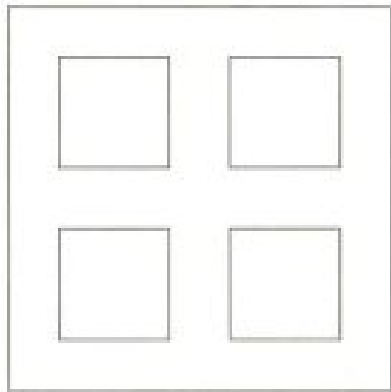


The Third Dimension



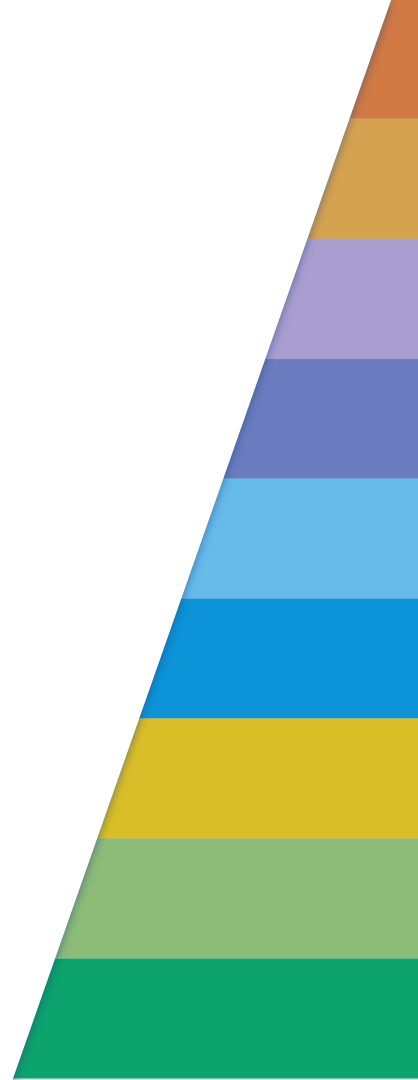
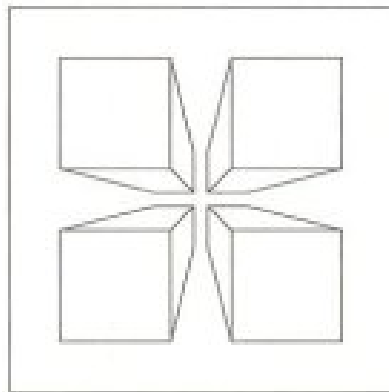
Camera Projections

Orthographic

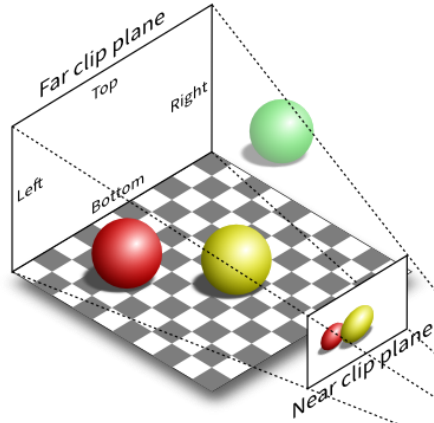


VS

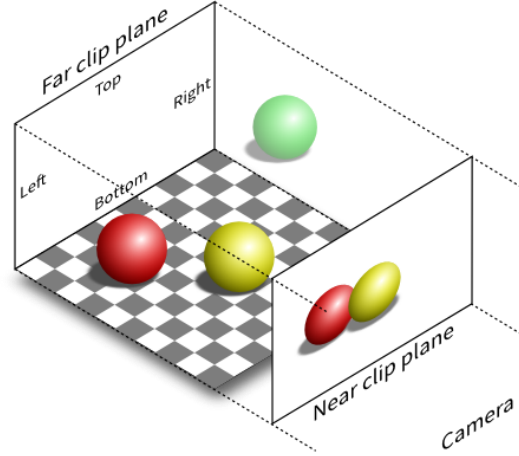
Perspective



Camera Projections



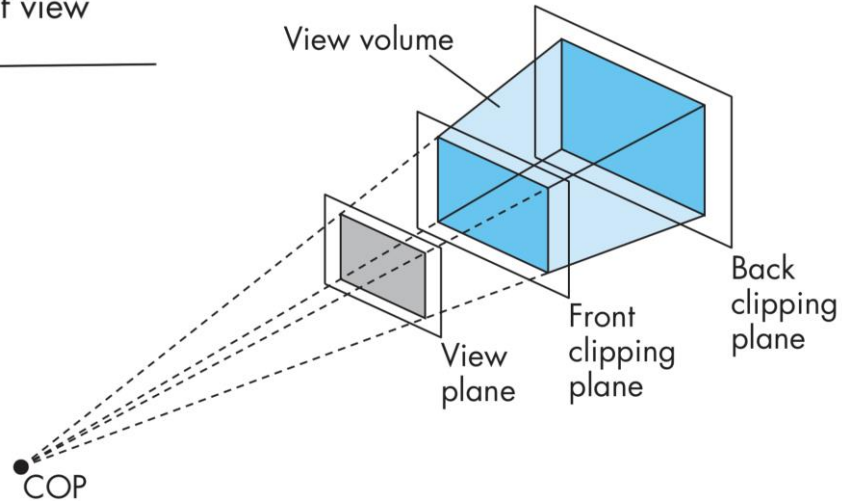
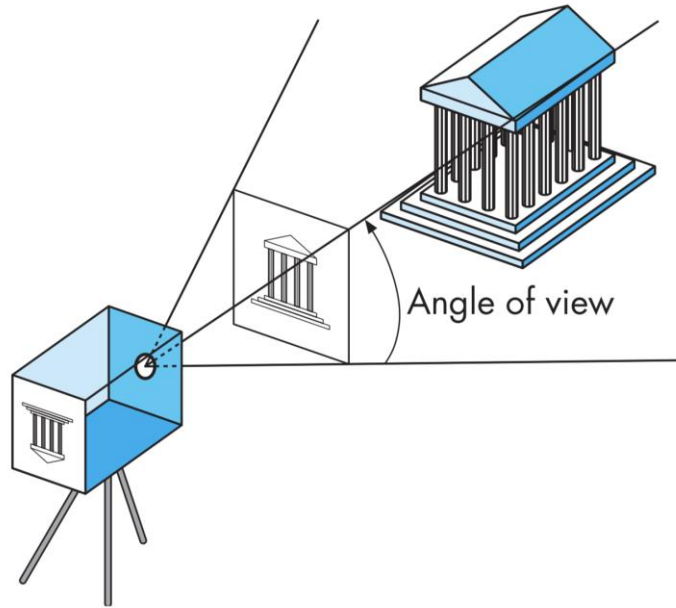
Perspective projection (P)



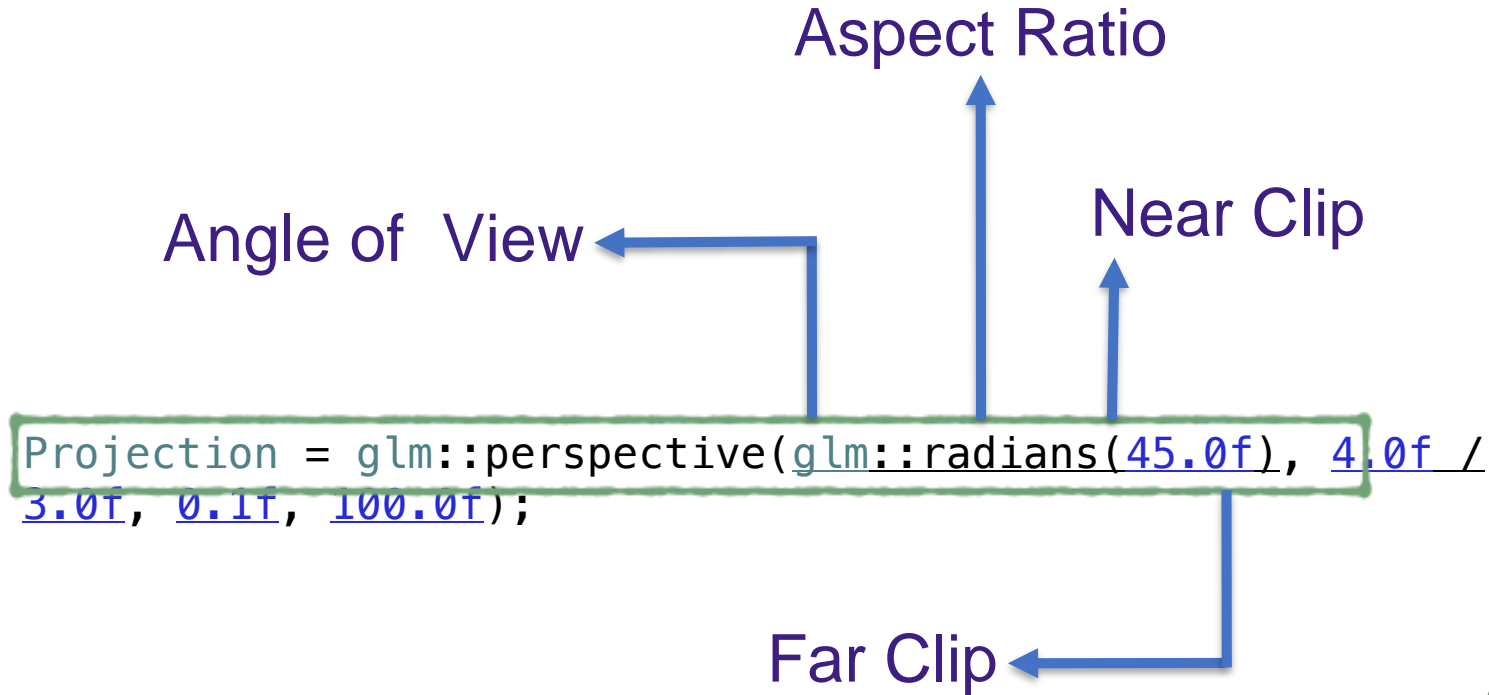
Orthographic projection (O)



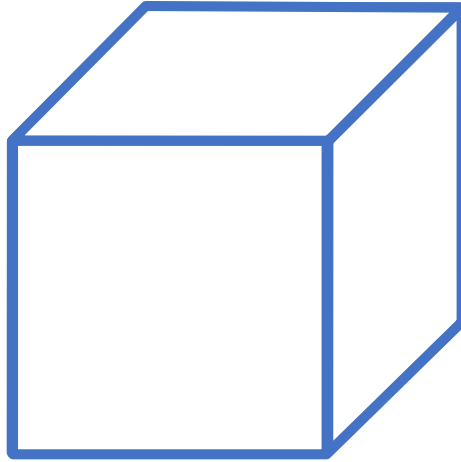
Camera Projections



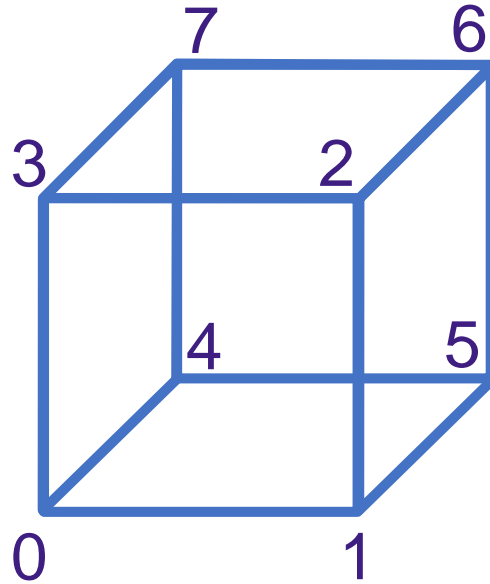
Camera Projections



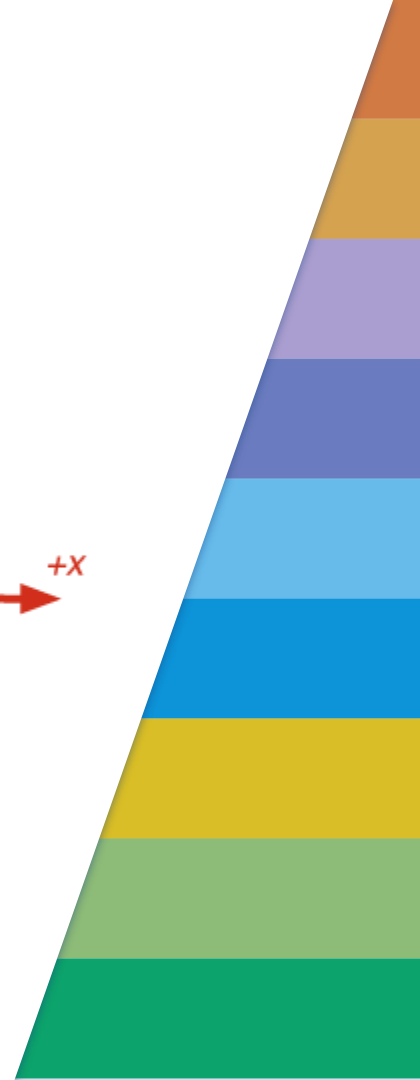
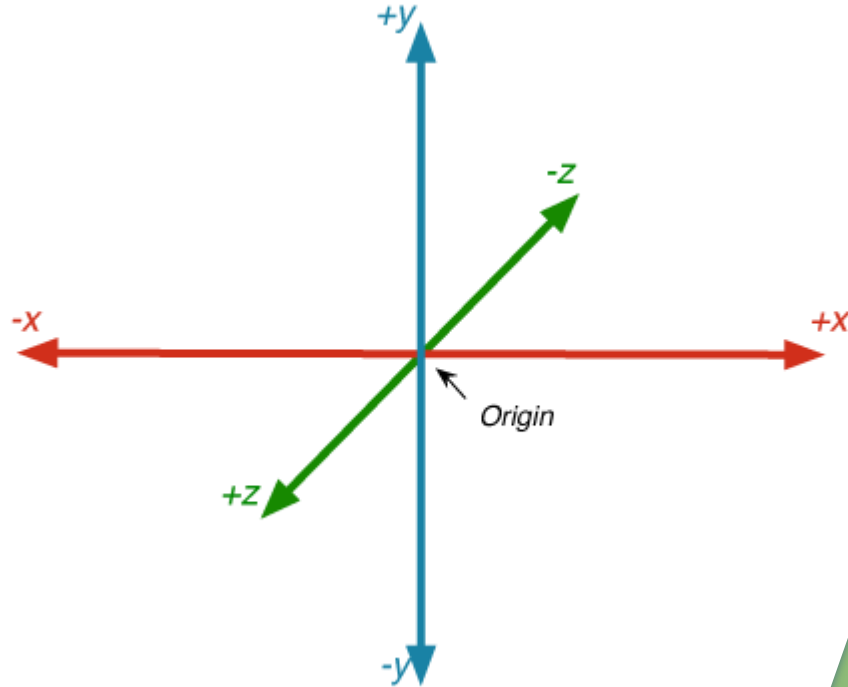
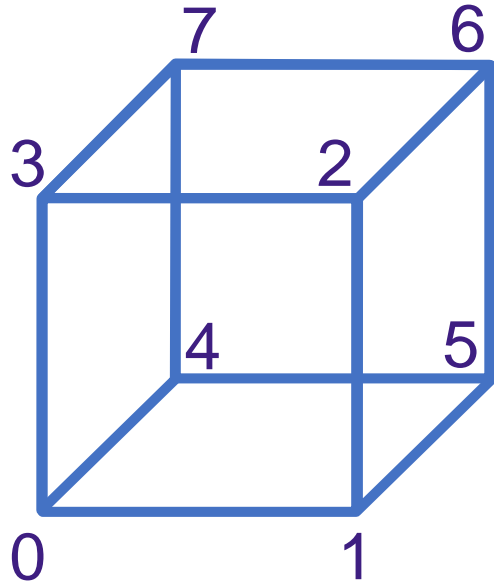
Rendering a 3D Object



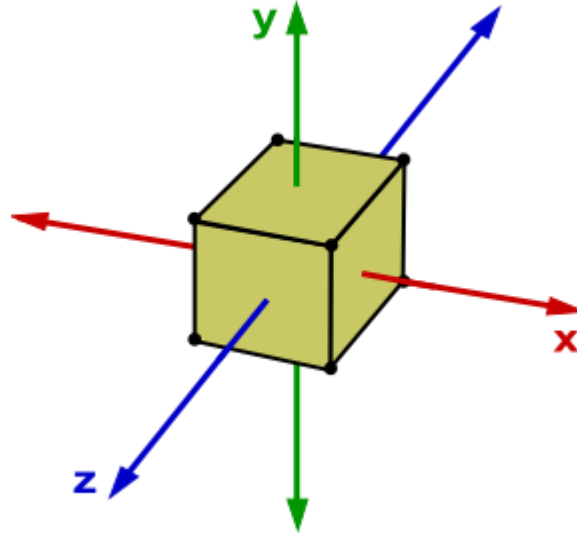
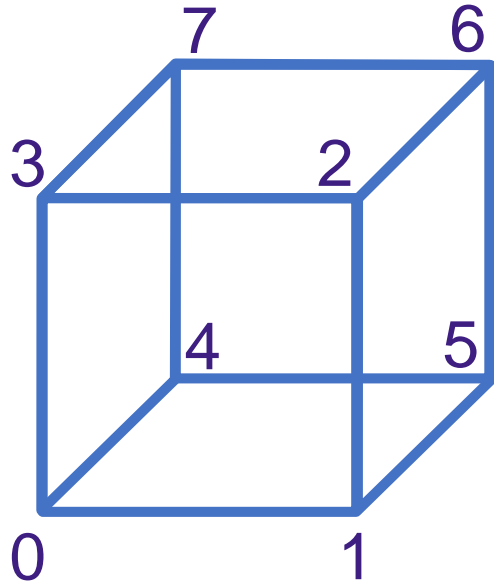
Rendering a 3D Object



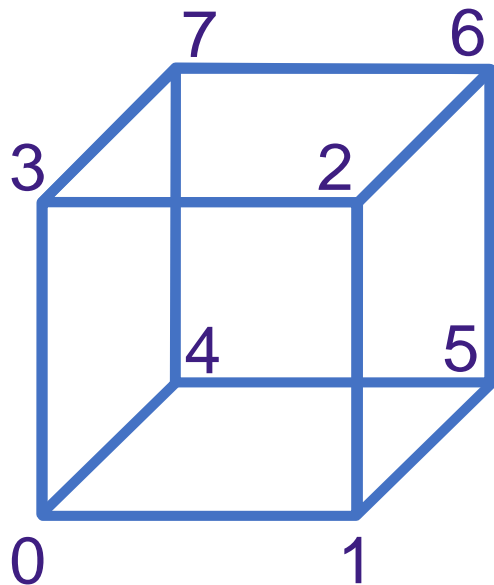
Rendering a 3D Object



Rendering a 3D Object



Rendering a 3D Object



Vertex List

$$V_0 = (-1, -1, 1)$$

$$V_1 = (1, -1, 1)$$

$$V_2 = (1, 1, 1)$$

$$V_3 = (-1, 1, 1)$$

$$V_4 = (-1, -1, -1)$$

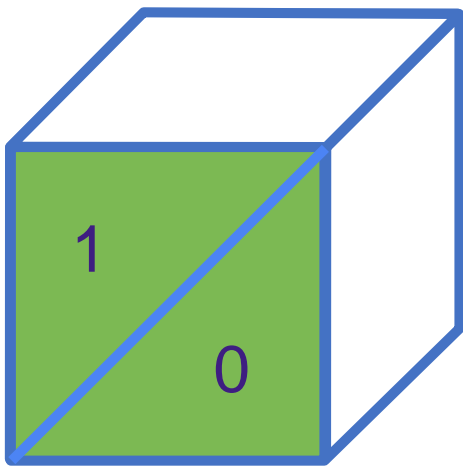
$$V_5 = (1, -1, -1)$$

$$V_6 = (1, 1, -1)$$

$$V_7 = (-1, 1, -1)$$



Rendering a 3D Object



Index List

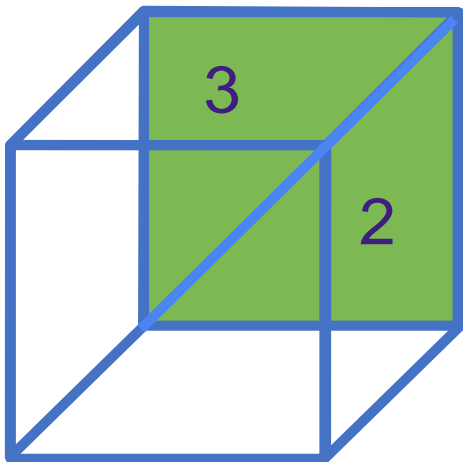
Front Face

Triangle0 = (0, 1, 2)

Triangle1 = (2, 3, 0)



Rendering a 3D Object



Index List

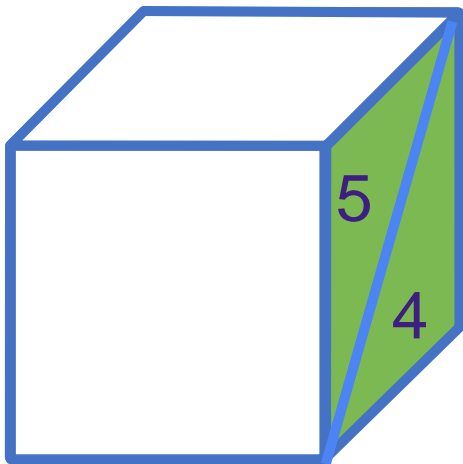
Back Face

Triangle2 = (4, 5, 6)

Triangle3 = (6, 7, 4)



Rendering a 3D Object



Index List

Right Face

Triangle4 = (1, 5, 6)

Triangle5 = (6, 2, 1)



Rendering a 3D Object

Index List

```
GLushort cube_index_array[] = {  
    // front  
    0, 1, 2,  
    2, 3, 0,  
    // top  
    1, 5, 6,  
    6, 2, 1,  
    .  
    .  
    .  
};
```

Vertex List

```
float cube_vertices[] = {  
    -0.45, -0.45,  0.45,  
    0.45, -0.45,  0.45,  
    .  
    .  
    .  
};
```

- ❖ Note that while the vertices array is 1D, OpenGL will know to render them three at a time, so the index 0 represents the 1st three floats, 1 represents the 2nd group of floats, etc.
- ❖ It is very helpful to write out your vertex data in code with comments and line breaks



Rendering a 3D Object

New IBO for the Index List

```
glBindBuffer(GL_ARRAY_BUFFER, ...);  
glBufferData(GL_ARRAY_BUFFER, ...);
```



```
glBindBuffer(GL_ELEMENT_ARRAY_BUFFER, ...);  
glBufferData(GL_ELEMENT_ARRAY_BUFFER, ...);
```



Rendering a 3D Object

New IBO for the Index List

```
GLuint cube_IB0;  
glGenBuffers(1, &cube_IB0);  
glBindBuffer(GL_ELEMENT_ARRAY_BUFFER, cube_IB0);  
glBufferData(GL_ELEMENT_ARRAY_BUFFER, sizeof(index_list), index_list, GL_STATIC_DRAW);
```



Rendering a 3D Object

New IBO for the Index List

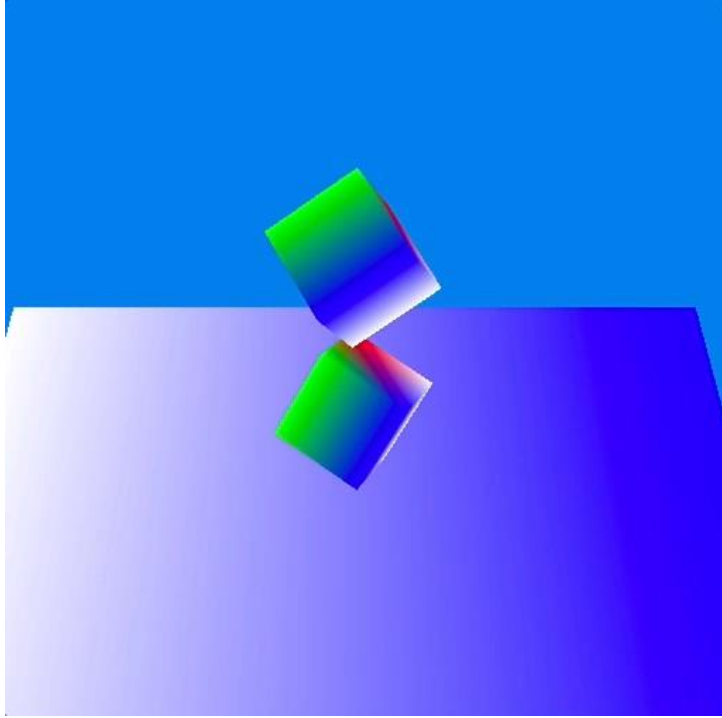
```
glDrawArrays(GL_LINE_STRIP, 0, NumVertices);
```



```
glDrawElements(GL_TRIANGLES, 36, GL_UNSIGNED_SHORT, 0);
```



Depth Testing



Depth Testing

When Initializing, Enable Depth Mode

```
glutInitDisplayMode(GLUT_RGBA);
```

```
glutInitDisplayMode(GLUT_RGBA | GLUT_DEPTH);
```



Depth Testing

When Initializing, Enable Depth Test

```
glEnable(GL_DEPTH_TEST);
```



Depth Testing

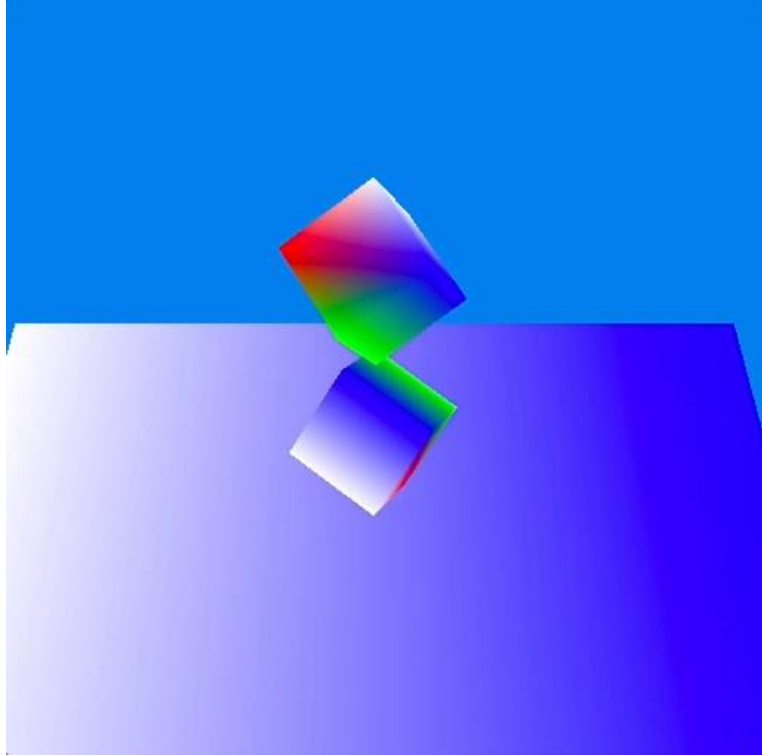
When Drawing, Clear Depth Buffer

```
glClear(GL_COLOR_BUFFER_BIT);
```

```
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
```



Depth Testing



Depth Testing

Use Double Buffering

```
glutInitDisplayMode(GLUT_RGBA);
```



```
glutInitDisplayMode(GLUT_RGBA | GLUT_DOUBLE);
```



Depth Testing

```
glFlush();
```

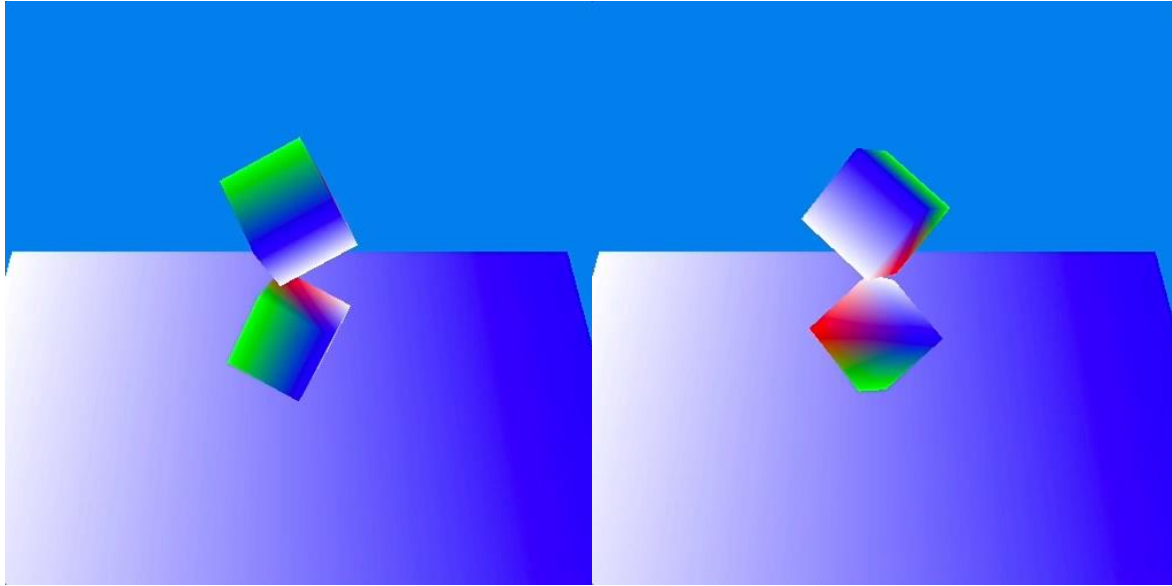


```
glutSwapBuffers();
```

Results in Smoother Animations?

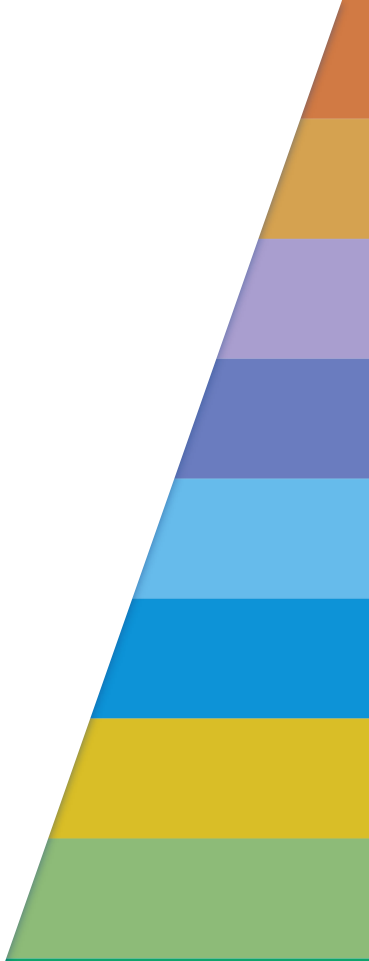


Depth Testing



Week 4

Lab Activities



Week 4 Lab

- ❖ For the lab, see Hooman's material (with video)
- ❖ OpenGL examples covered:
 - More indexed draws
 - Projections
 - Orthographic and perspective



Week 4

End