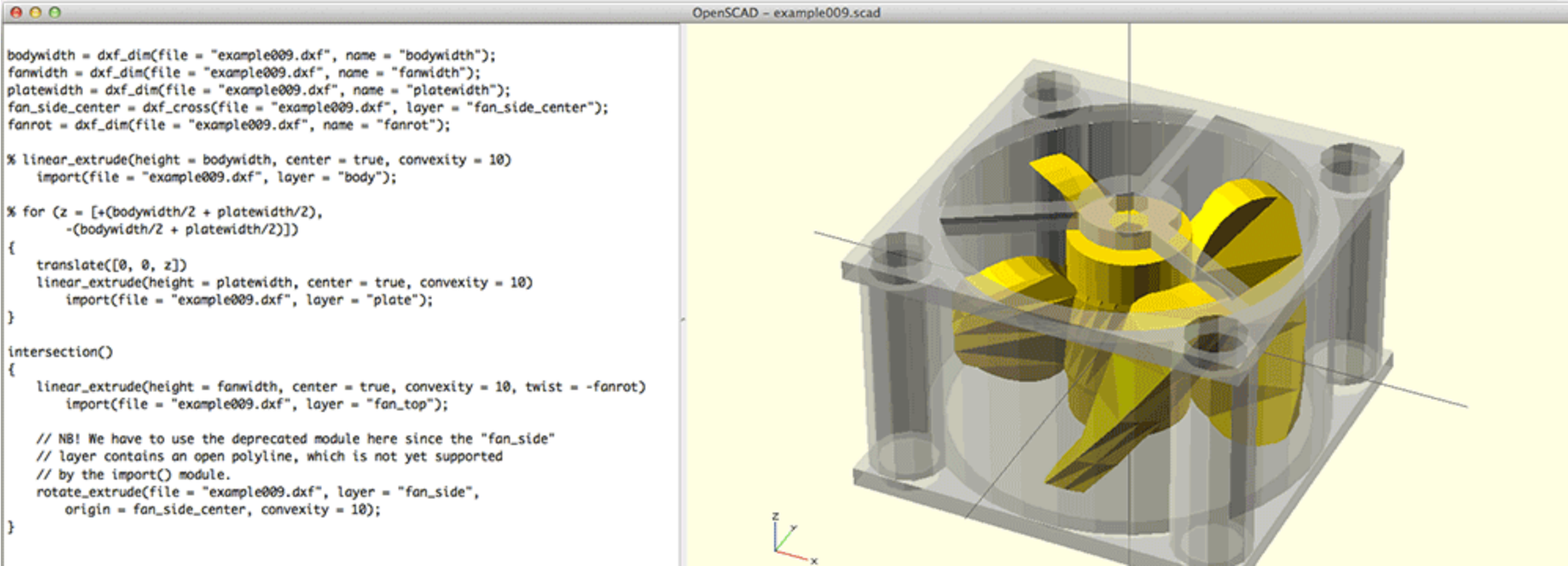
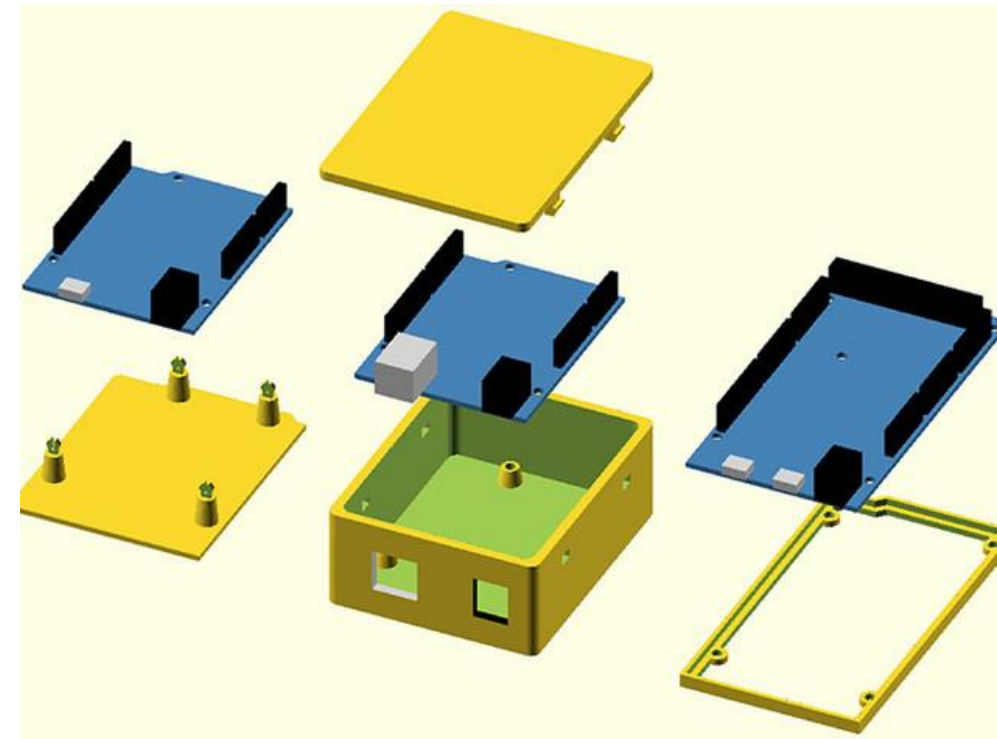


OpenScad



- Why use it?

- Free!
- Large community: <http://forum.openscad.org/>
- Easier to get started than other softwares;

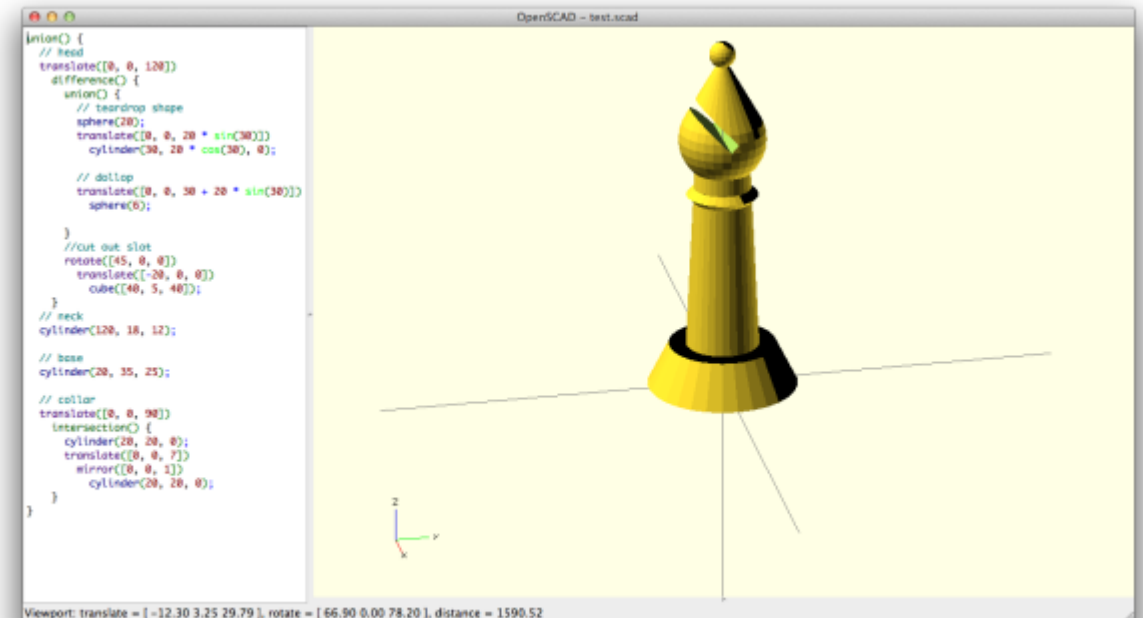


The Basics

- Geometric Figures
 - Cube
 - Sphere
 - Cylinder
- Transformations
 - Translating
 - Scaling
 - Rotating
 - Mirroring

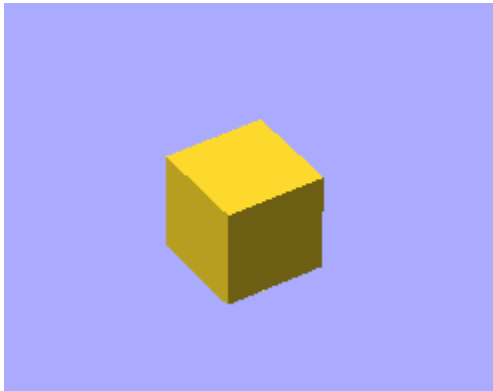
CSG

- Constructive Solid Geometry
 - Union
 - Difference
 - Intersection

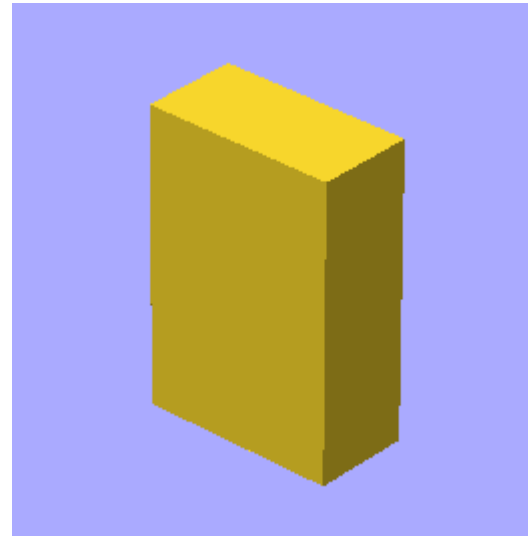


The Basics

- Cube
 - `$ cube([x,y,z]);`
 - `Cube([10,10,10]);`

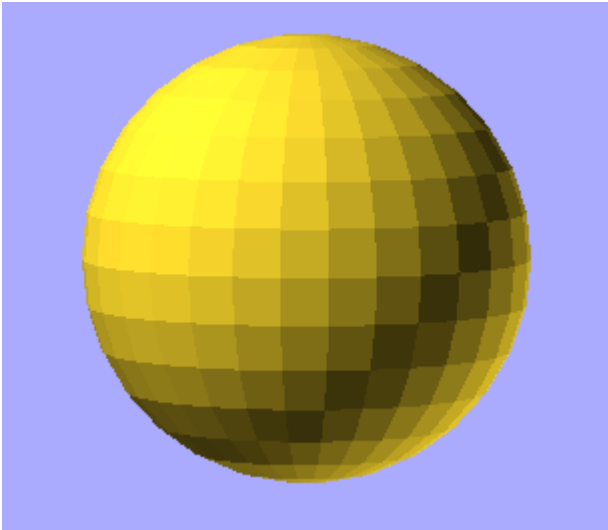


`cube([10,20,30], center = true);`



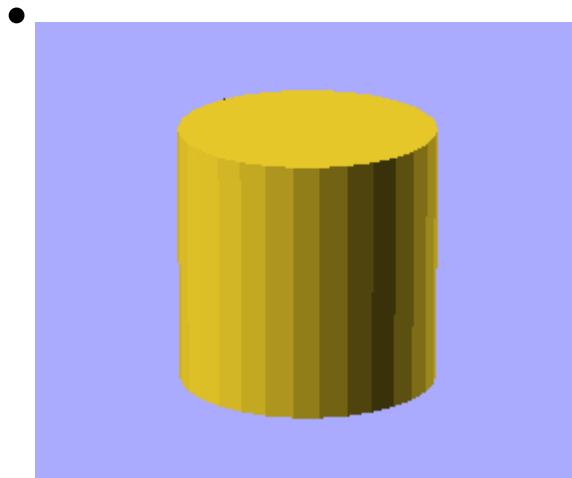
The Basics

- Sphere
 - `$ sphere(r);`
 - `sphere(r=20); // already centered`
 -

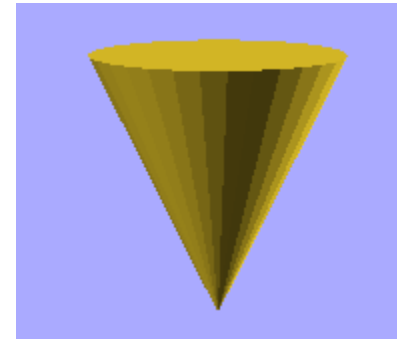


The Basics

- Cylinder
 - `$ cylinder(h,r1,r2);`
 - `Cylinder(h=20,r=10);`

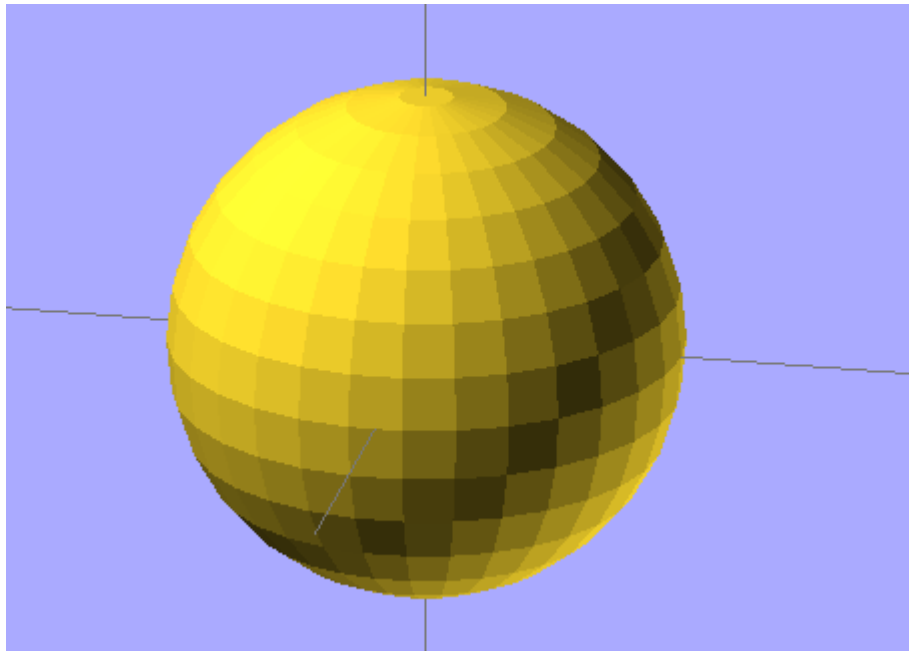


`Cylinder(h=20,r=10, r2=0, center = true);`

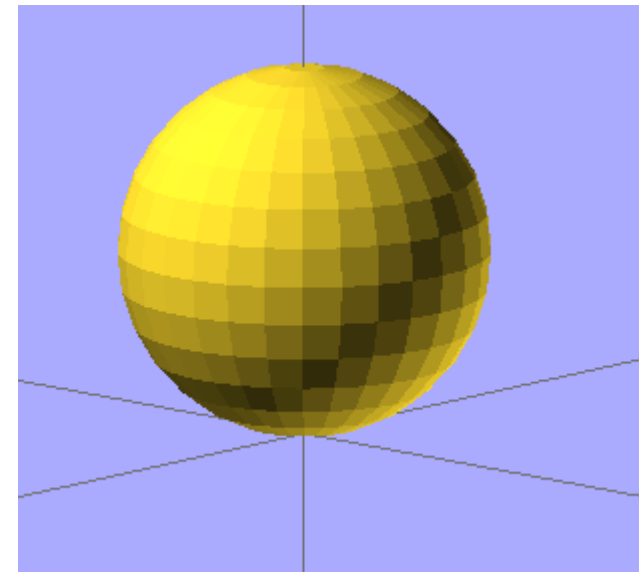


Transformations

- Translation => Move to somewhere else;
- Sphere(r=20);

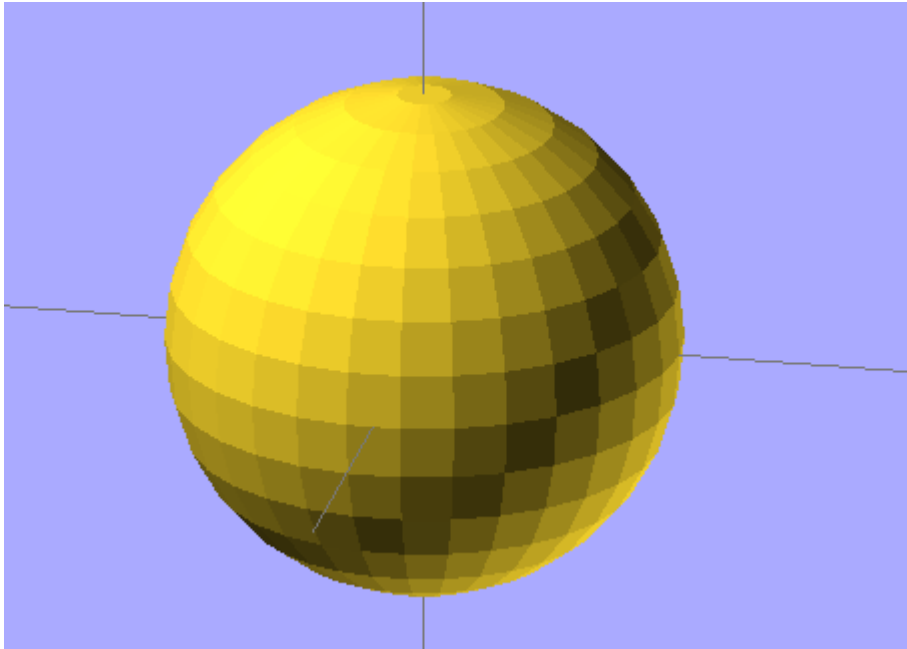


`translate([0,0,20] Sphere(r=20);`

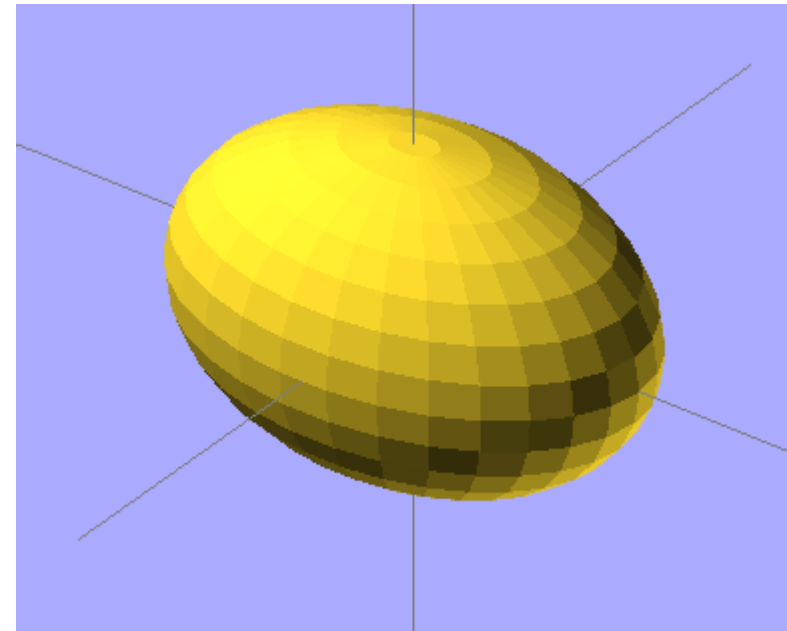


Transformation

- Scaling=> multiply for some value;
- Sphere(r=20);



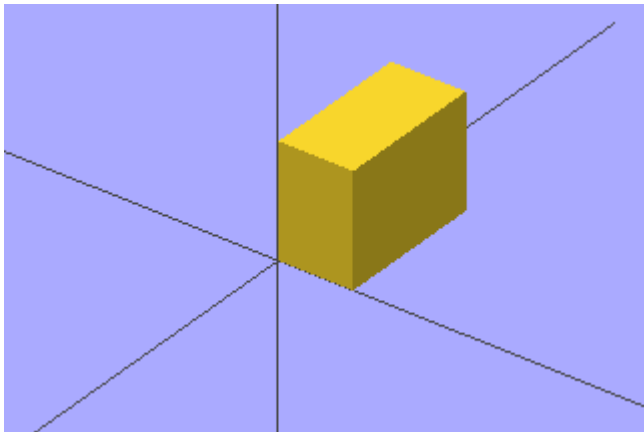
scale([1.5,1,1] Sphere(r=20);



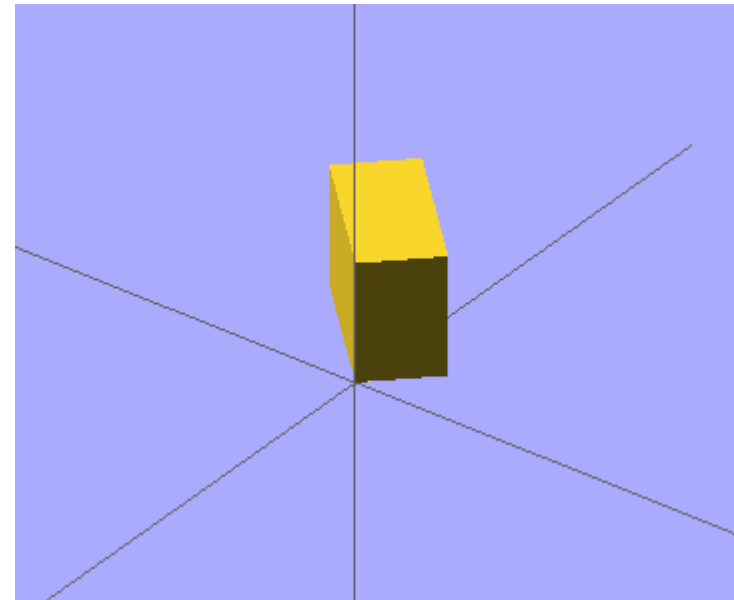
Transformation

- Rotate=> change position in relation to an axis;

- `cube([10, 20, 15]);`

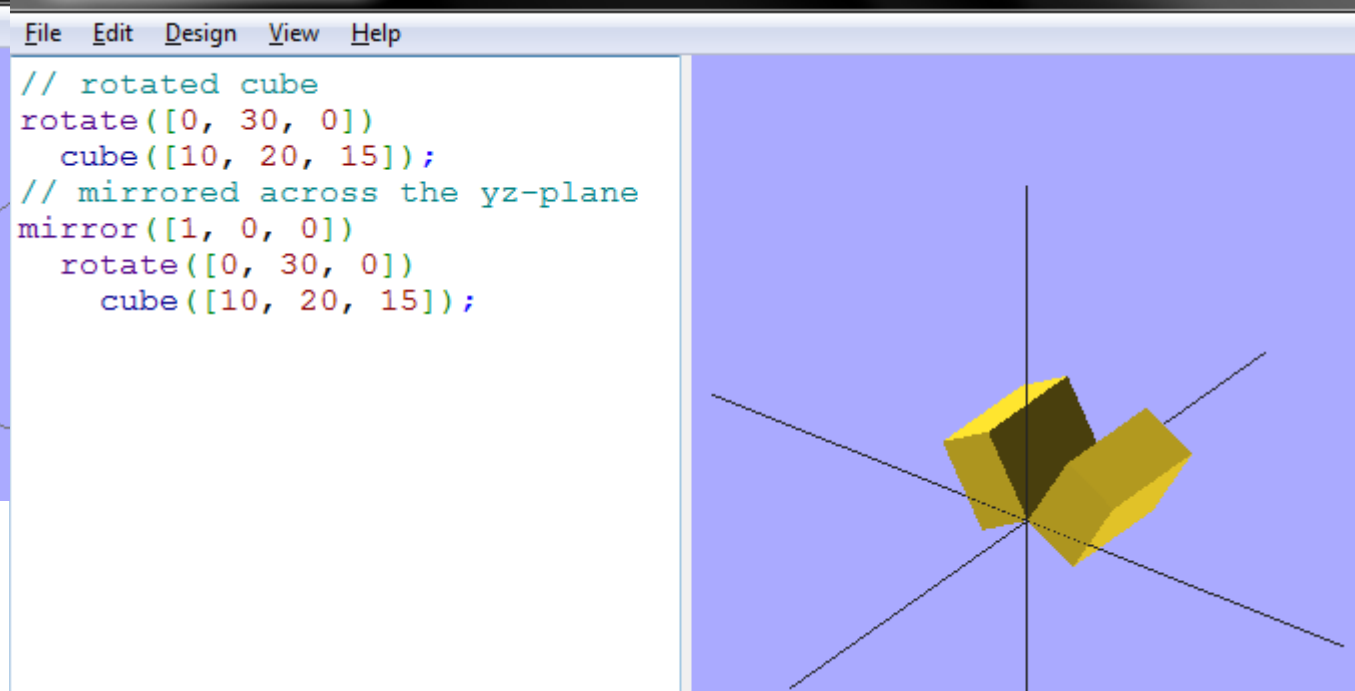
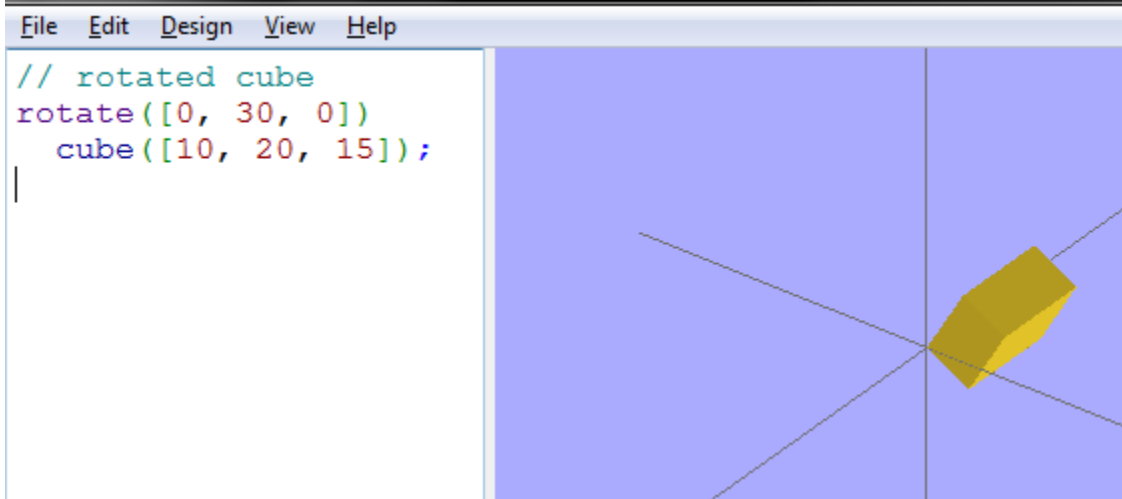


- `rotate([0,0,45] cube([10, 20, 15]));`



Transformation

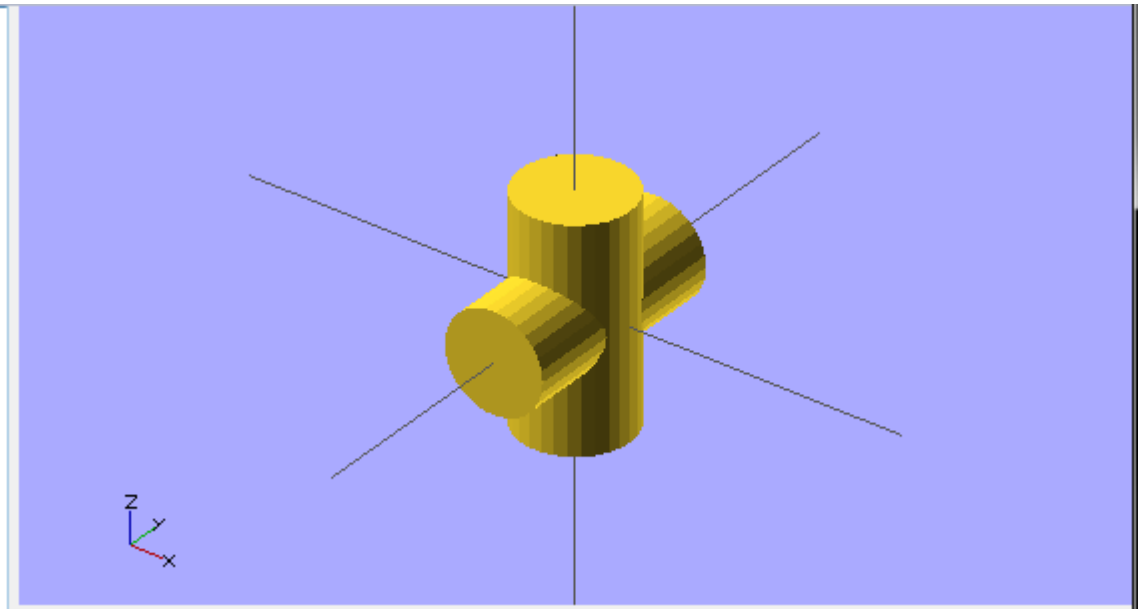
- Mirror=> reflect in relation to an axis;



Constructive Solid Geometry Operations

- Union

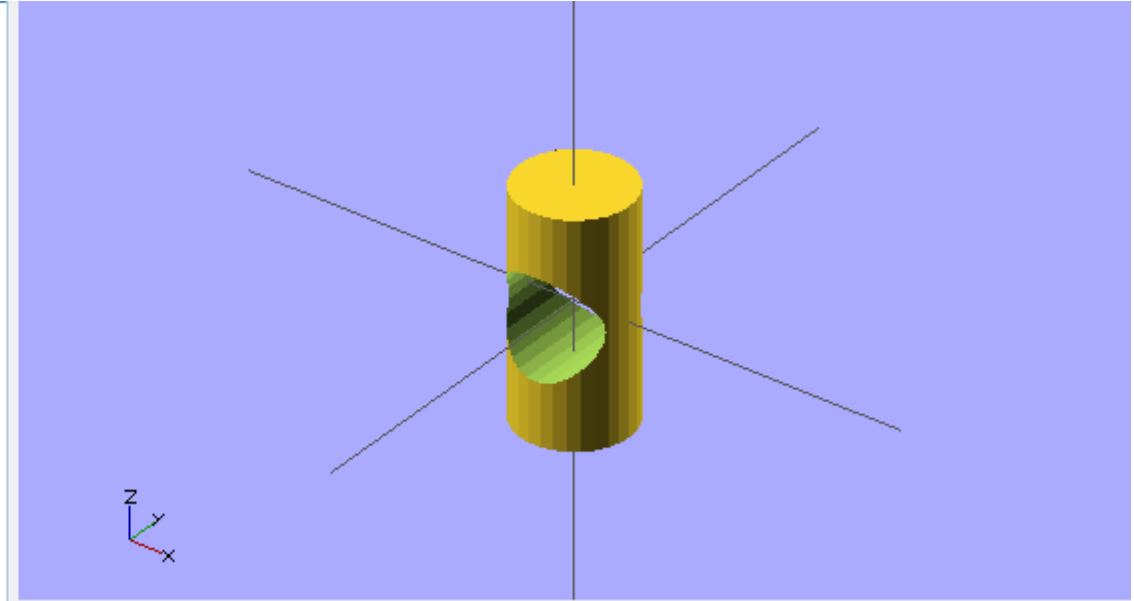
```
union() {  
  cylinder(h = 40, r = 10, center = true);  
  rotate([90, 0, 0])  
    cylinder (h = 40, r = 9, center = true);  
}
```



Constructive Solid Geometry Operations

- Difference

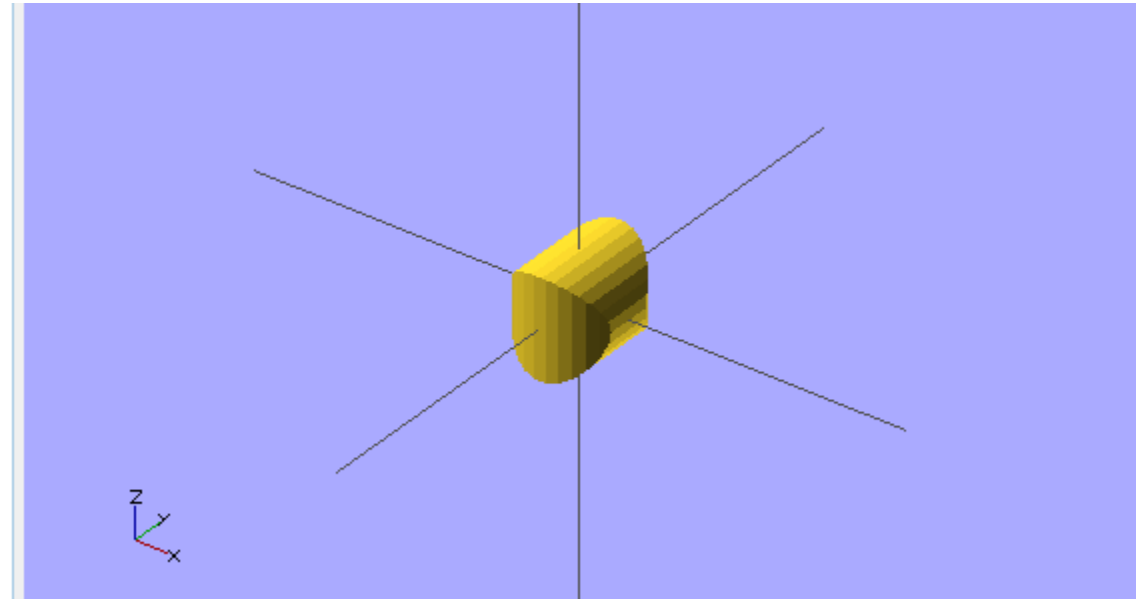
```
difference() {  
  cylinder(h = 40, r = 10, center = true);  
  rotate([90, 0, 0])  
    cylinder(40, r = 9, center = true);  
}
```



Constructive Solid Geometry Operations

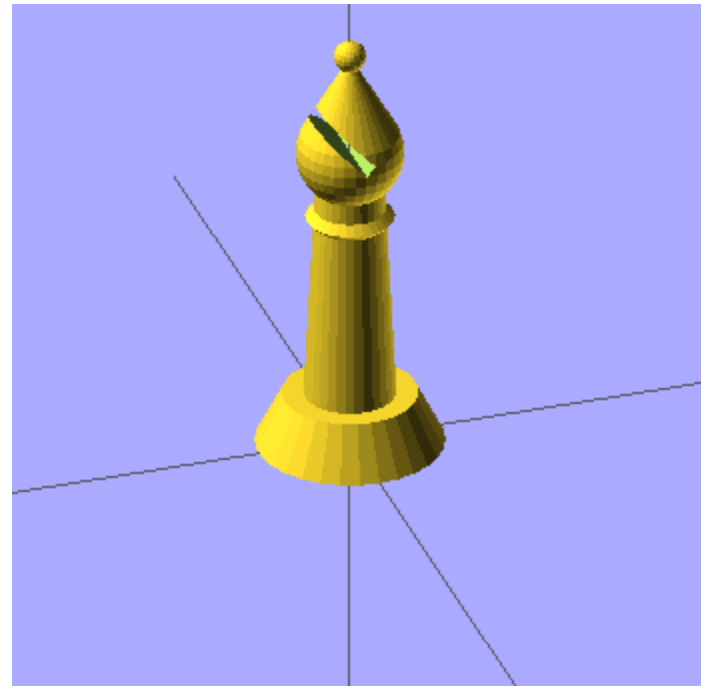
- Intersection => What will overlap?

```
intersection() {  
  cylinder(h = 40, r = 10, center = true);  
  rotate([90, 0, 0])  
    cylinder(40, r = 9, center = true);  
}
```



Got the basics?

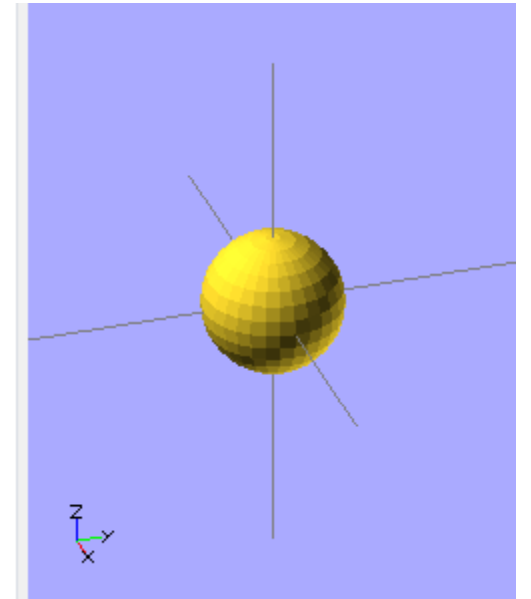
- Practice, practice, practice!
 - And one example



Bishop

- Start with the “teardrop” shape;

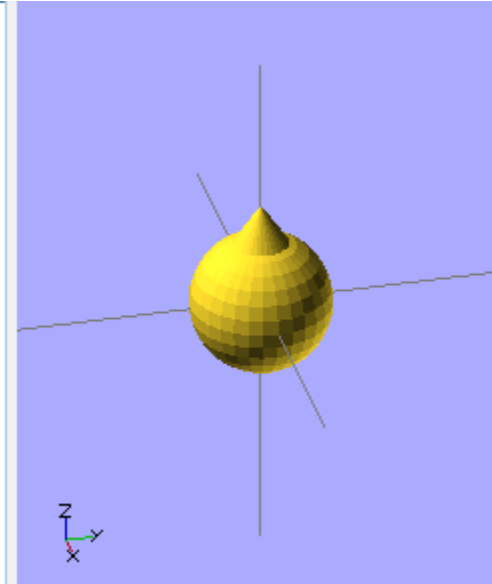
```
sphere(r=20);
```



Bishop

- Start with the “teardrop” shape;
- Add a cone to the sphere;

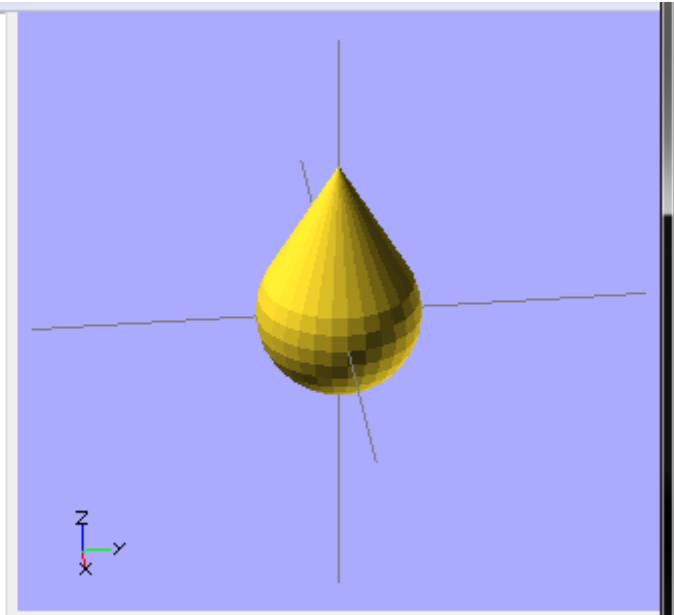
```
union() {  
  | sphere(r = 20);  
  | cylinder(h = 30, r1 = 20, r2 = 0);  
}
```



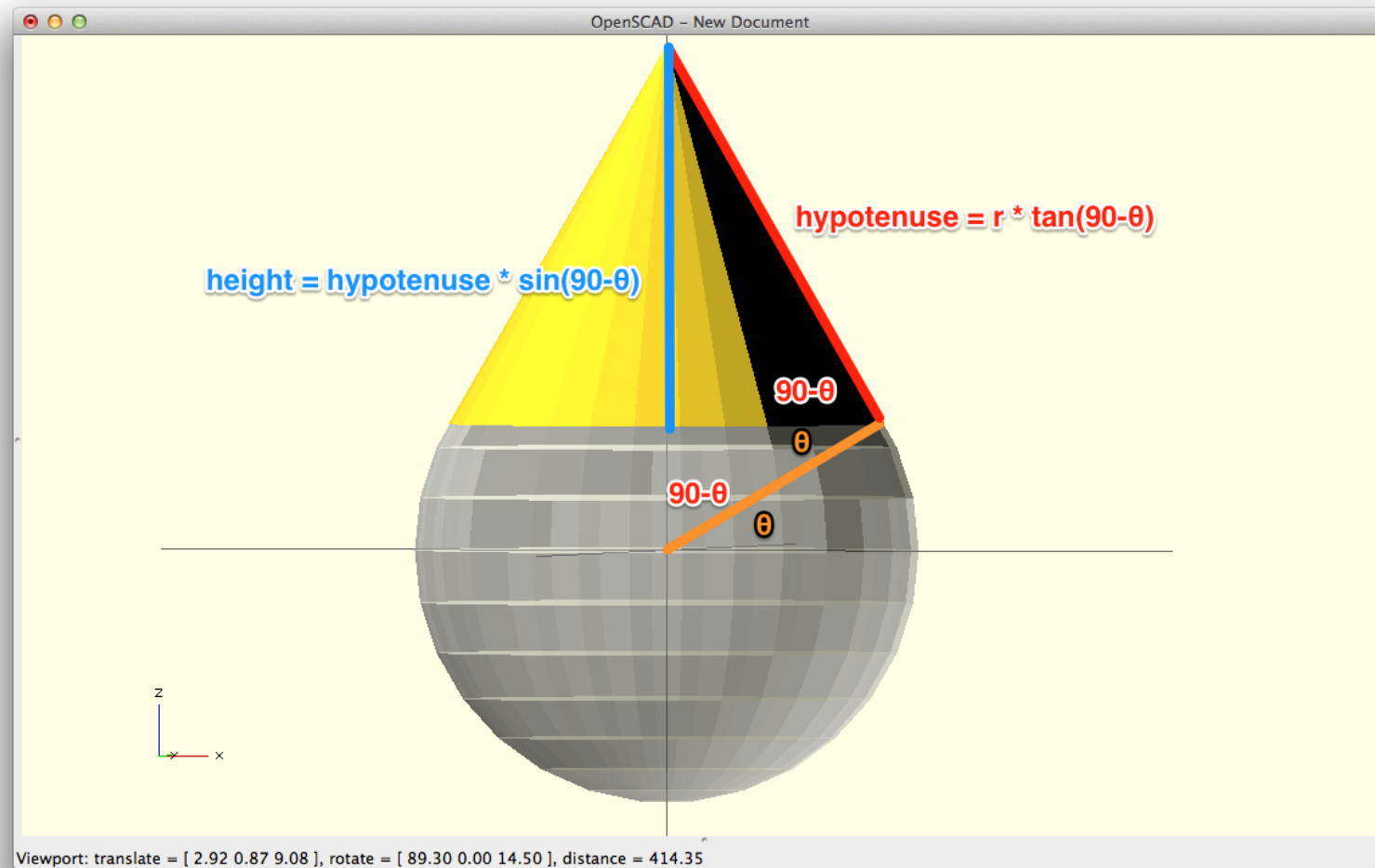
Bishop

- Start with the “teardrop” shape;
- Add a cone to the sphere;
- Match the cone with the sphere

```
union() {  
  sphere(r = 20);  
  translate([0, 0, 20 * sin(30)])  
    cylinder(h = 30, r1 = 20 * cos(30), r2 = 0);  
}
```

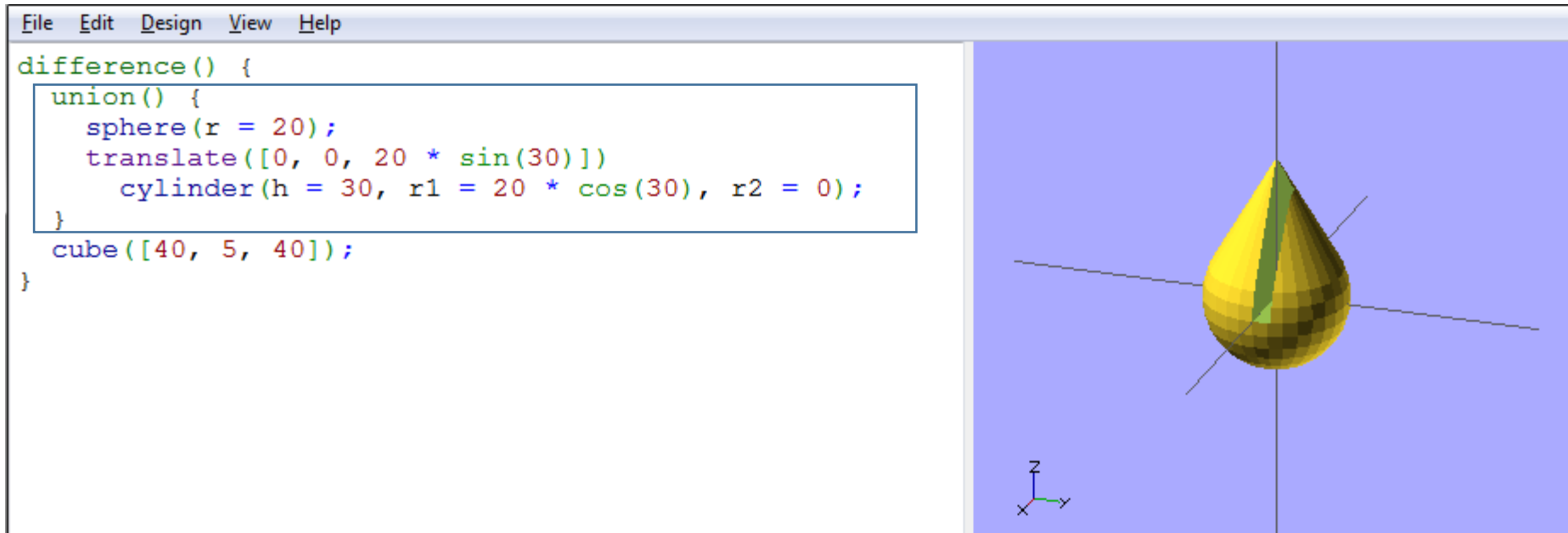


Why $\sin(30)$???



Bishop

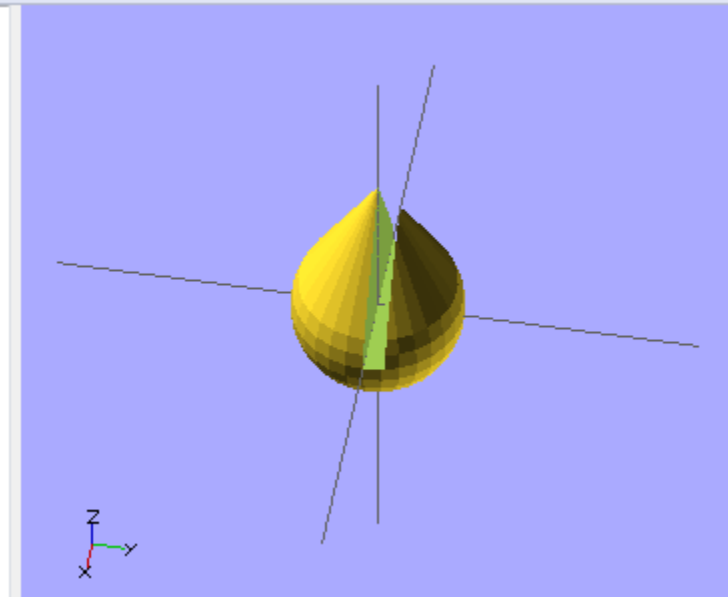
- Cut the piece (difference)



Bishop

- Correct the cut position

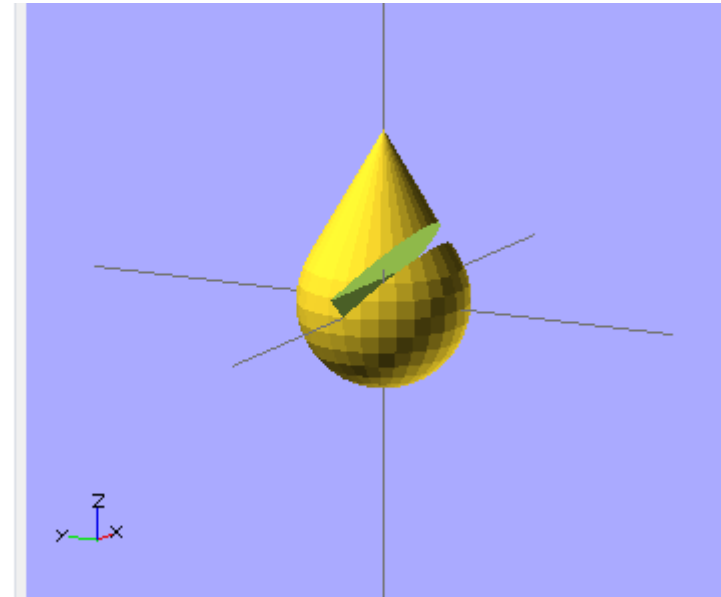
```
difference() {  
  union() {  
    sphere(r = 20);  
    translate([0, 0, 20 * sin(30)])  
    cylinder(h = 30, r1 = 20 * cos(30), r2 = 0);  
  }  
  translate([-20, 0, 0])  
  cube([40, 5, 40]);  
}
```



Bishop

- Still missing the rotation!

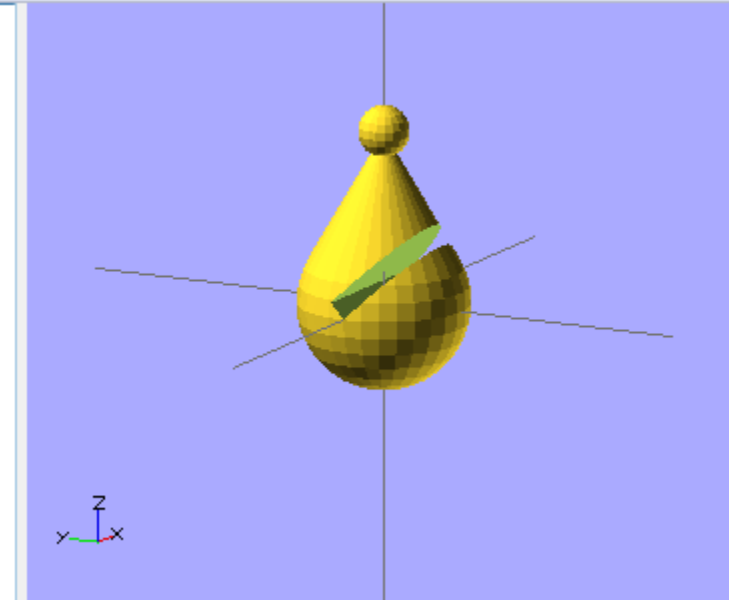
```
difference() {  
  union() {  
    sphere(r = 20);  
    translate([0, 0, 20 * sin(30)])  
    cylinder(h = 30, r1 = 20 * cos(30), r2 = 0);  
  }  
  rotate([45, 0, 0])  
  translate([-20, 0, 0])  
  cube([40, 5, 40]);  
}
```



Bishop

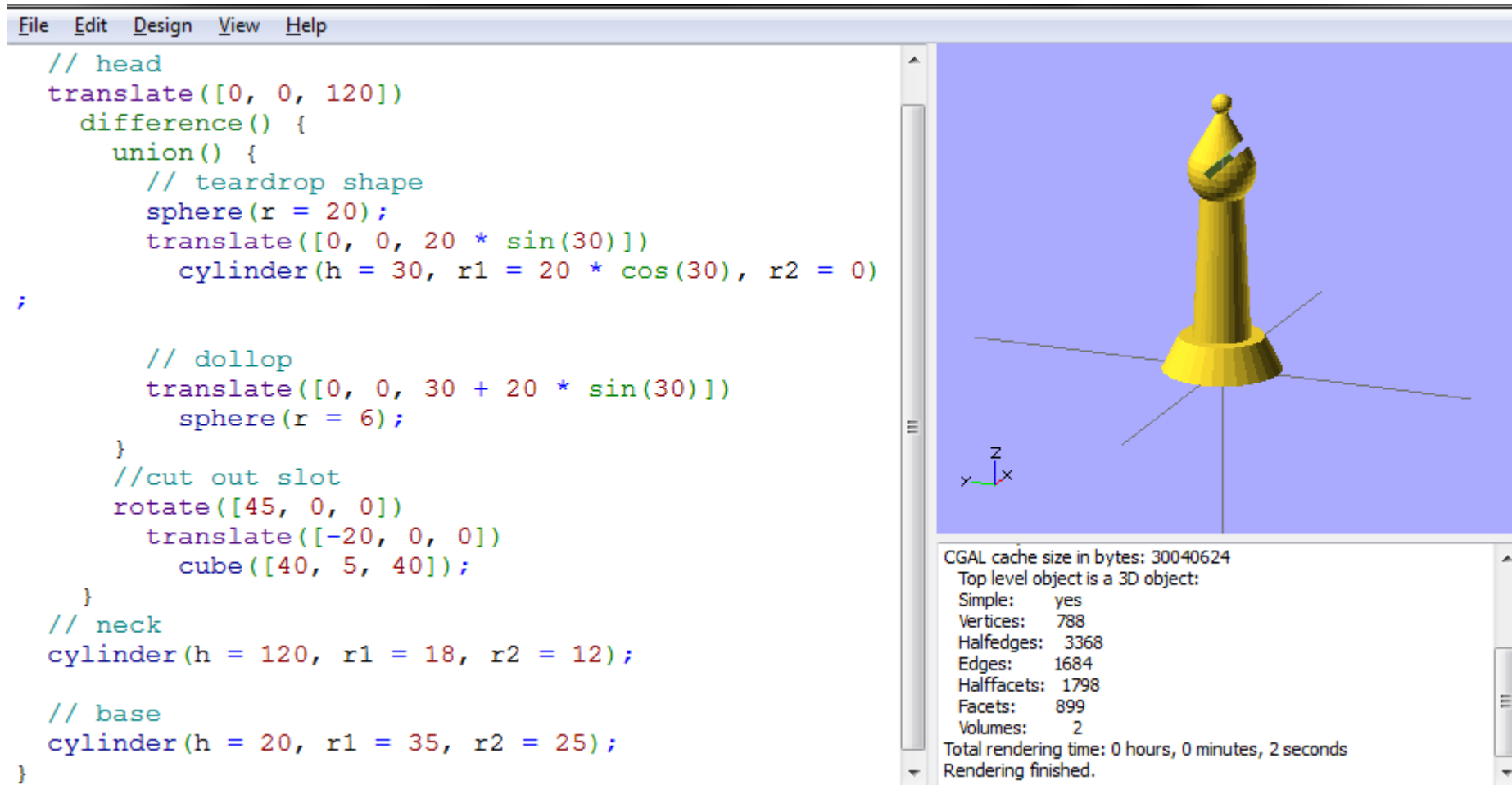
- Now, let's add a dollop on top. Since we know the height of the cone is 30, and we moved it up $20 * \sin(30)$, we'll need to translate the dollop $30 + 20 * \sin(30)$. We'll also comment the parts so we don't get confused.

```
difference() {  
  union() {  
    // teardrop shape  
    sphere(r = 20);  
    translate([0, 0, 20 * sin(30)])  
    cylinder(h = 30, r1 = 20 * cos(30), r2 = 0);  
  
    // dollop  
    translate([0, 0, 30 + 20 * sin(30)])  
    sphere(r = 6);  
  }  
  //cut out slot  
  rotate([45, 0, 0])  
  translate([-20, 0, 0])  
  cube([40, 5, 40]);  
}
```



Bishop

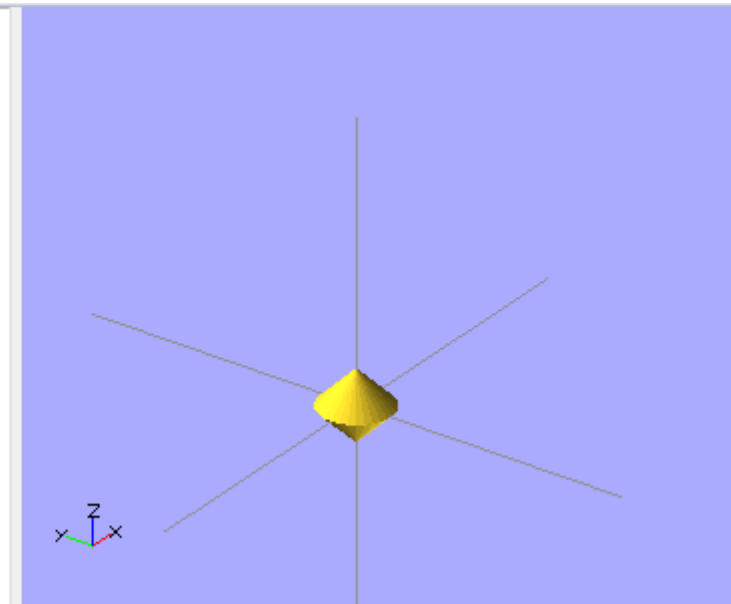
- Add the neck and the base!



Bishop

- Making a fancy collar

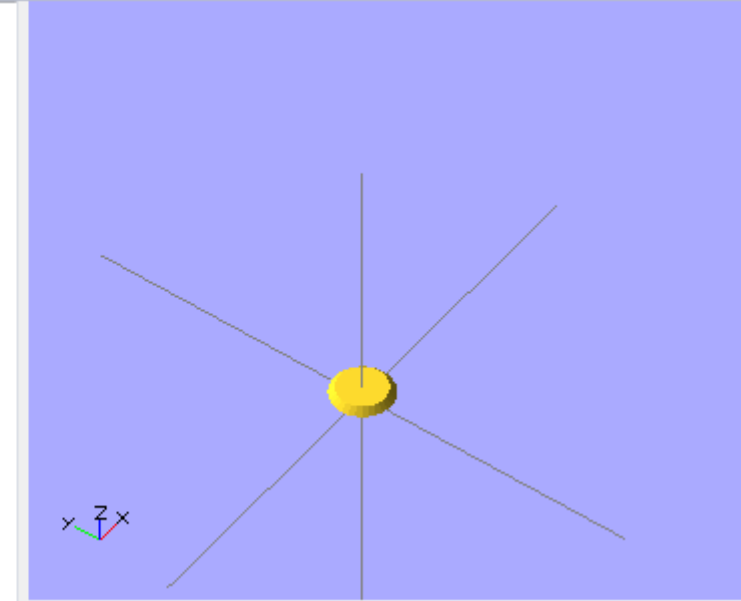
```
cylinder(h = 20, r1 = 20, r2 = 0);  
mirror([0, 0, 1])  
  cylinder(h = 20, r1 = 20, r2 = 0);
```



Bishop

- Making a fancy collar

```
intersection() {  
  cylinder(h = 20, r1 = 20, r2 = 0);  
  translate([0, 0, 7])  
    mirror([0, 0, 1])  
      cylinder(h = 20, r1 = 20, r2 = 0);  
}
```



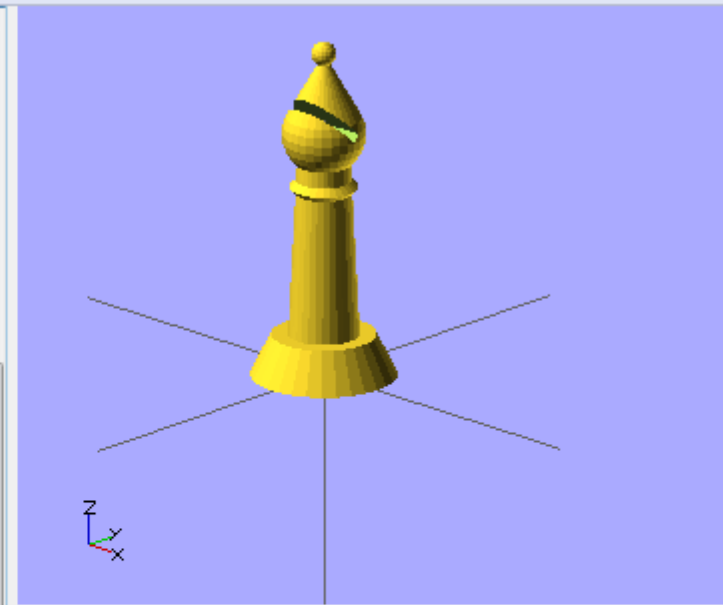
Bishop

- Everything together!

```
// dollop
translate([0, 0, 30 + 20 * sin(30)])
sphere(r = 6);
}
//cut out slot
rotate([45, 0, 0])
translate([-20, 0, 0])
cube([40, 5, 40]);
}
// neck
cylinder(h = 120, r1 = 18, r2 = 12);

// base
cylinder(h = 20, r1 = 35, r2 = 25);

// collar
translate([0, 0, 90])
intersection() {
  cylinder(h = 20, r1 = 20, r2 = 0);
  translate([0, 0, 7])
  mirror([0, 0, 1])
  cylinder(h = 20, r1 = 20, r2 = 0);
}
}
```



Saved backup file: E:/Victor/My Documents/OpenSCAD/backups/unsaved-backup-hgqH1736.scad
Compiling design (CSG Tree generation)...
Rendering Polygon Mesh using CGAL...
PolySets in cache: 4
PolySet cache size in bytes: 87936
CGAL Polyhedrons in cache: 72
CGAL cache size in bytes: 31935080
Top level object is a 3D object:
Simple: yes
Vertices: 990
Halfedges: 4008