Aula 3

Objetos 3D e operadores

O que veremos hoje

- Objetos primitivos
- Operadores de objetos
- Modelagem de um objeto a partir de medidas reais

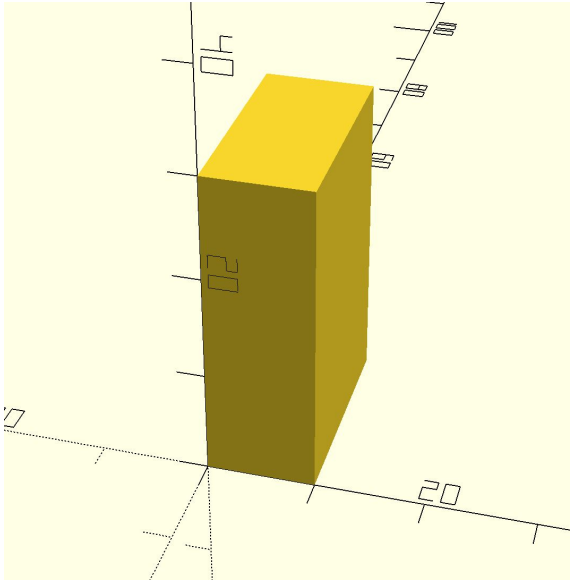
Finalmente vamos modelar!

- Cada arquivo do OpenSCAD gera **um objeto**!
- Esse objeto deve ser declarado no **fim** do arquivo
- Formas de modelagem:
  - Geometria sólida construtiva
  - Extrusão de formas 2D
- Hoje veremos a **primeira**

Poliedros primitivos

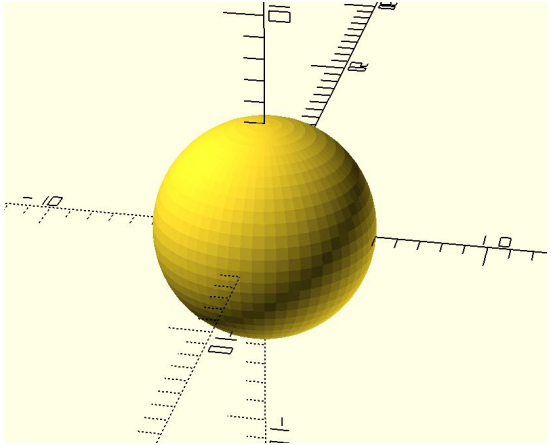
Cubo

- Na verdade, um **paralelepípedo**
- Por padrão, uma das quinas na origem
  - Pode ser alterado usando o parâmetro center
- Usos:
  - `cube([x, y, z]);` `// cubo de dimensões x, y, z`
  - `cube(n);` `// cubo de lado de tamanho n`
  - `cube(n, center=true);` `// cubo centralizado`
- As dimensões são expressas em **milímetros**!



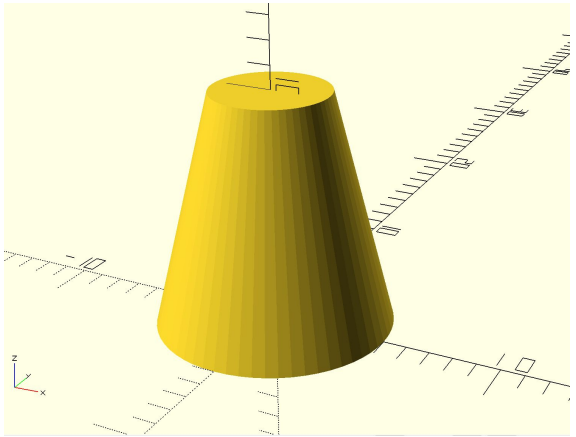
Esfera

- Por padrão, seu **centro** fica na **origem**
  - Ao contrário do cubo!
- Pode ser criada a partir de seu **raio** ou **diâmetro**:
  - `sphere(d = 10); // esfera de 10mm de diametro`
  - `sphere(r = 10); // esfera de 10mm de raio`
- Sua forma é **discretizada**
  - Na prática, é um **poliedro** com muitos lados
  - A variável especial **$fn** define o número de lados
    - Quanto **maior**, mais **suave**
    - Quanto **menor**, mais **rápida** é a renderização
    - Pode ser definida **globalmente**, ou passada por parâmetro



Cilindro

- Também útil para fazer **cones** e **troncos de cone**
- Por padrão, centralizado na origem nos eixos **x** e **y**
  - Mas não o **z**. Pode ser mudado com o parâmetro **center**
- **r** define o raio, ou **d** define o diâmetro
- Pode receber dois raios, para fazer **cones** e **troncos de cone**
  - **r1** é o raio da parte de baixo
  - **r2** é o raio da parte de cima
  - se um dos raios for zero, temos um **cone**
  - Também é possível usar diâmetros, com **d1** e **d2**
- A variável **$fn** também vale aqui



Operadores de objetos

Operadores de objetos

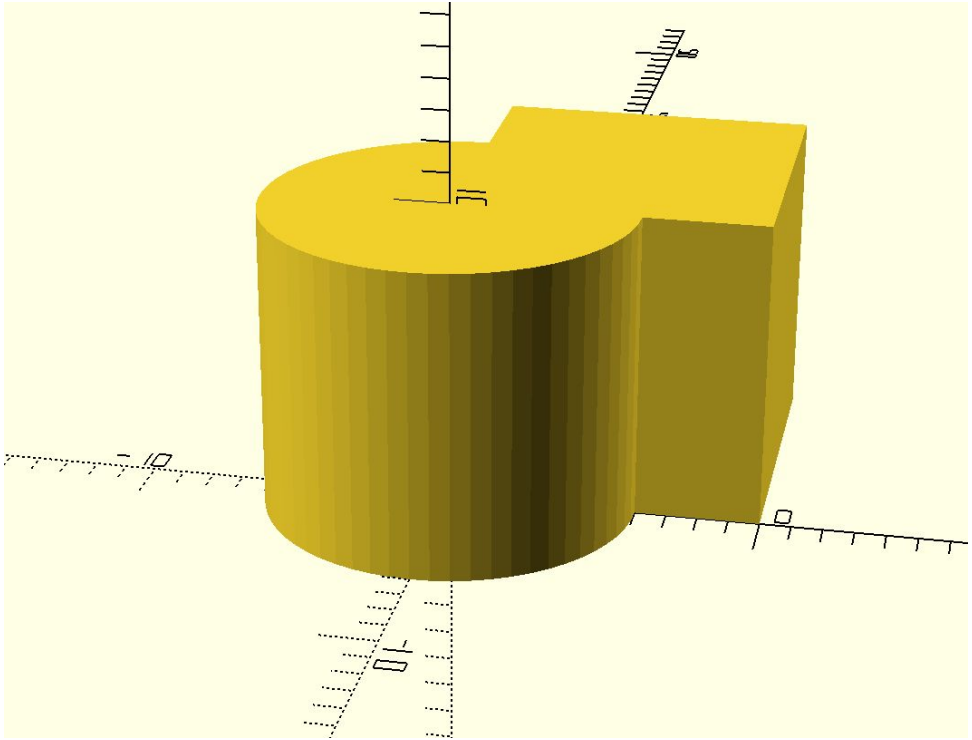
- **Modificam** um ou mais objetos
- Geram um novo objeto
- Operadores booleanos:
  - Baseados na **álgebra booleana** e **teoria dos conjuntos**
  - União
  - Intersecção
  - Diferença
- Transformações:
  - Mudam as propriedades de um objeto
- Sintaxe:

```
operador(parametros) {
   objeto1;
   objeto2;
}
```

União

- **Junta** dois ou mais objetos
- Exemplo:

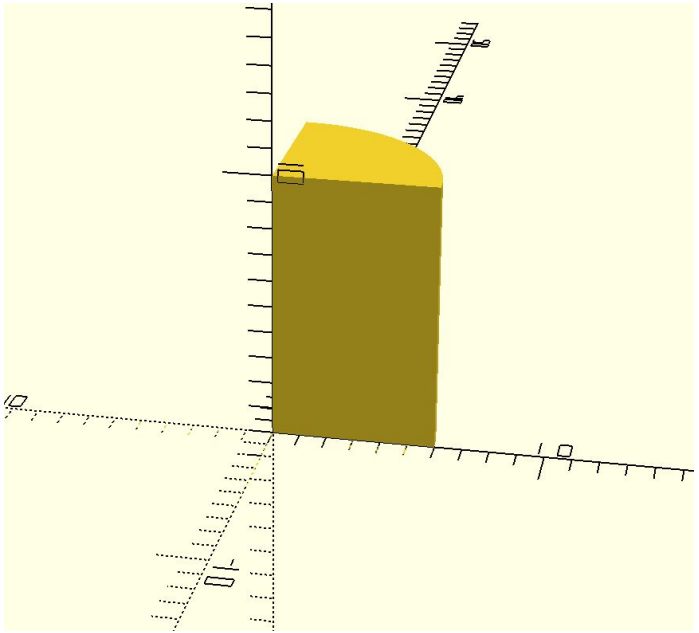
```
union() {
    cylinder(h=10, d=12);
    cube(10);
}
```



Intersecção

- **Seleciona** apenas a parte **compartilhada** pelos objetos
- Exemplo:

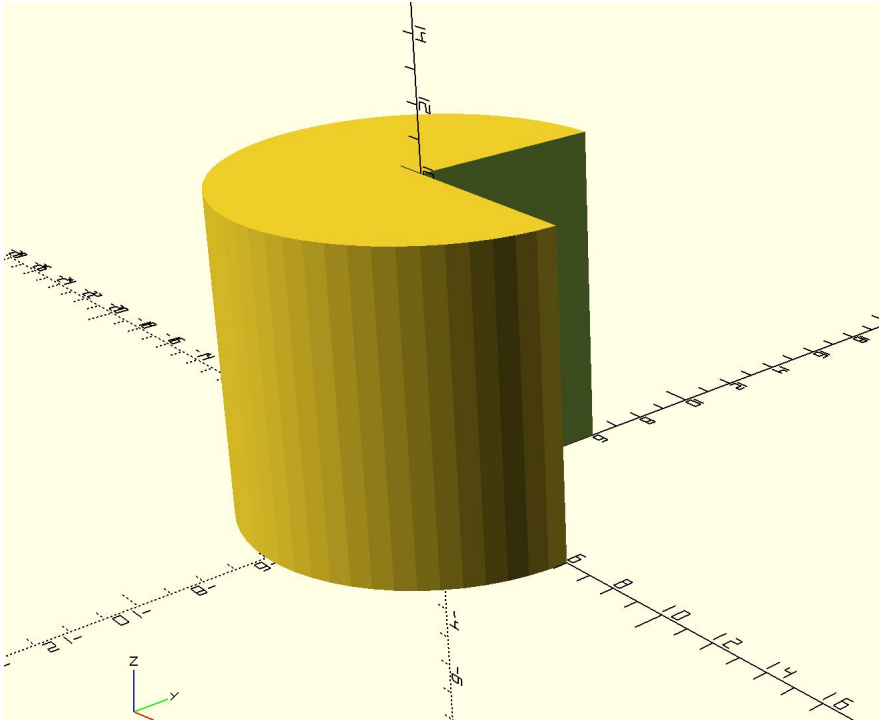
```
intersection() {
    cylinder(h=10, d=12);
    cube(10);
}
```



Diferença

- **Retira** do **primeiro** objeto a parte **compartilhada** com os outros
- Exemplo:

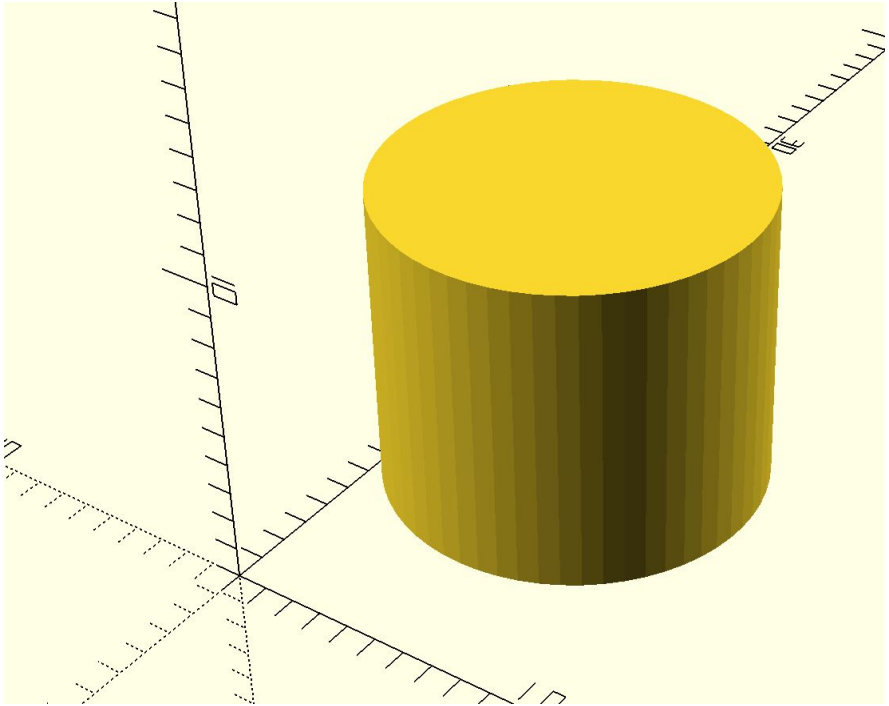
```
difference() {
    cylinder(h=10, d=12);
    cube(10);
}
```



Translação

- **move** o objeto
- Exemplo:

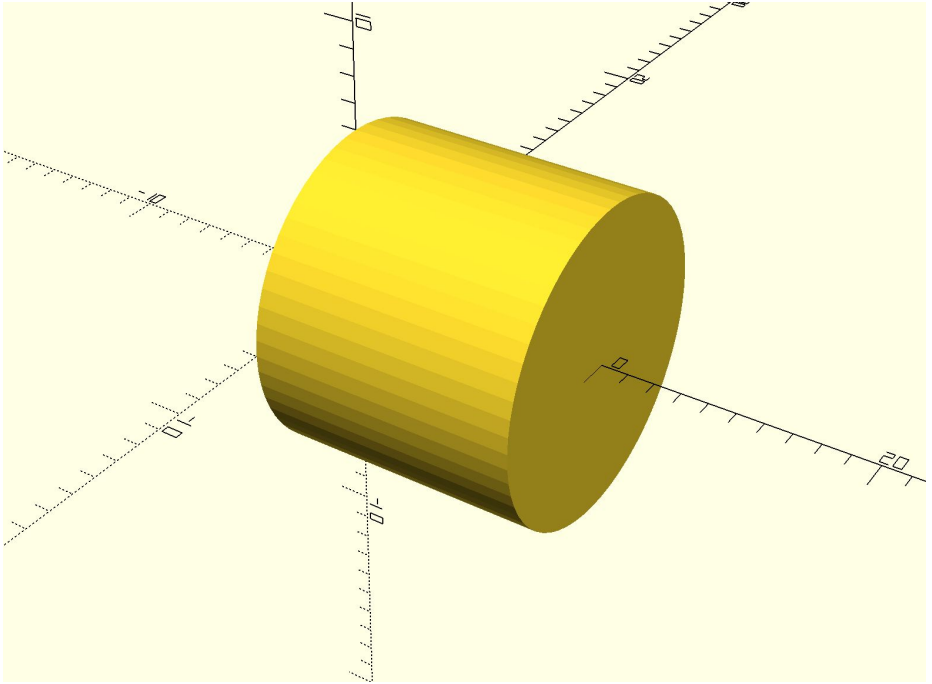
```
translate([5, 10, 0]) {
    cylinder(h=10, d=12);
}
```



Rotação

- **Rotaciona** o objeto
- Em relação aos **eixos**
- Exemplo:

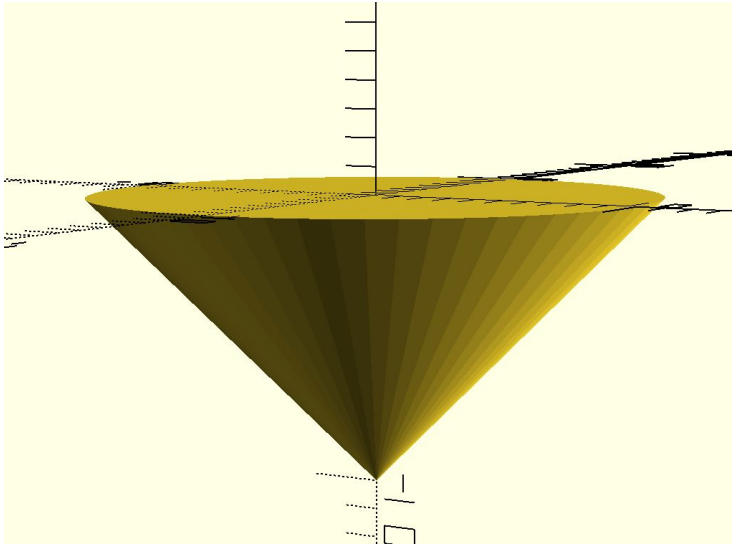
```
rotate([0, 90, 0]) {
    cylinder(h=10, d=12);
}
```



Espelho

- **Rotaciona** o objeto
- Em relação aos **eixos**
- Exemplo:

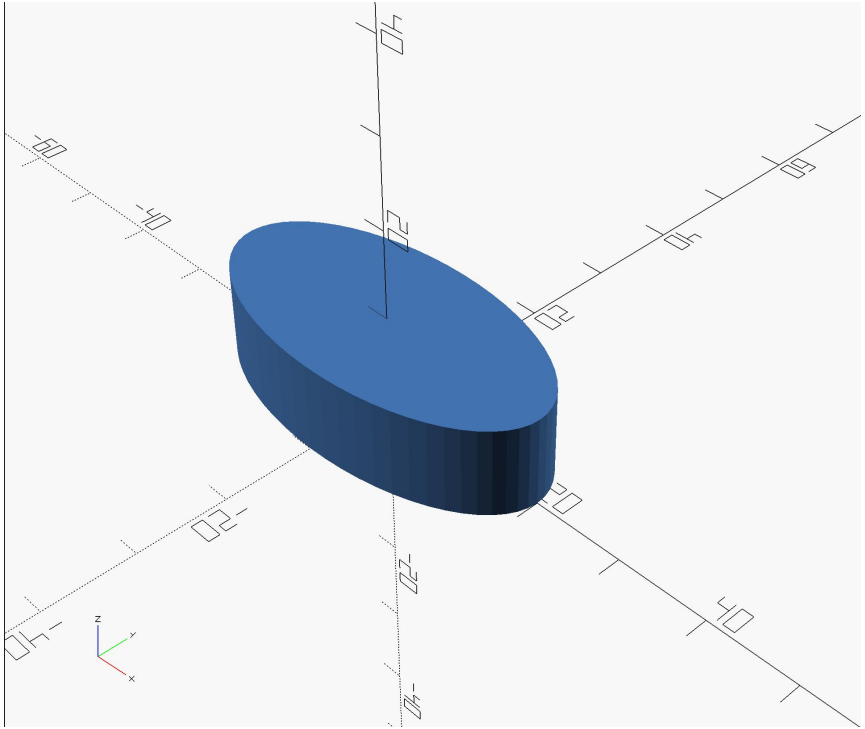
```
mirror([0, 0, 1]) {
    cylinder(h=10, r1=10, r2=0);
}
```



Escalonamento

- **Aumenta** ou **diminui** o objeto, proporcionalmente
- Exemplo:

```
scale([2, 1, 1]) {
    cylinder(h=10, r1=10,
r2=0);
}
```



Hull

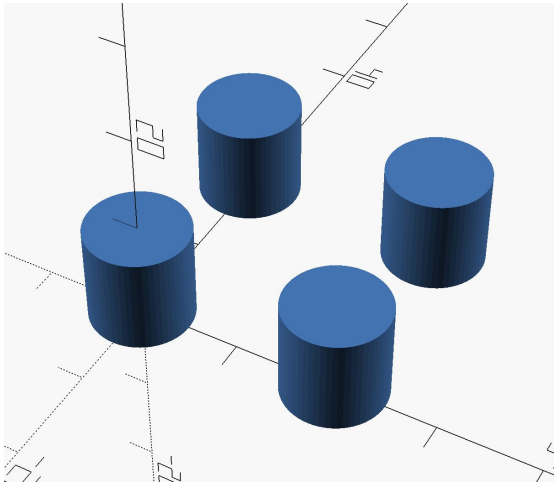
- **Conecta** objetos
- Exemplo:

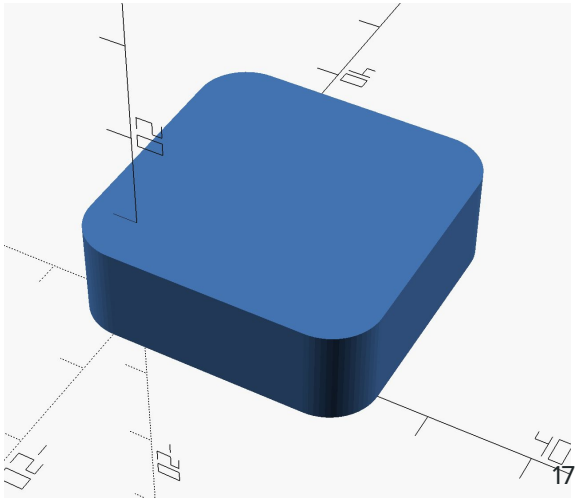
```
hull() {
    cylinder(h=10, d=10);

    translate([0, 20, 0]) cylinder(h=10, d=10);

    translate([20, 0, 0]) cylinder(h=10, d=10);

    translate([20, 20, 0]) cylinder(h=10, d=10);
}
```

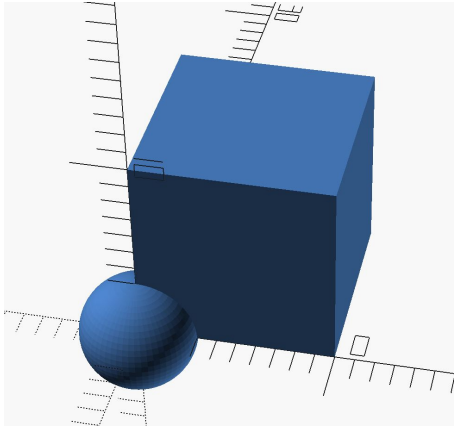


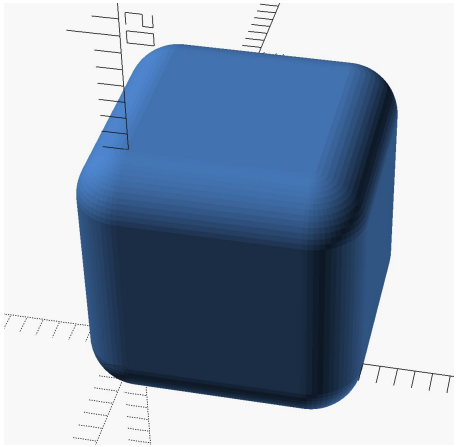


Soma de Minkowski

- Definição **matemática**: https://pt.wikipedia.org/wiki/Adição_de_Minkowski
- Passa um objeto **em volta** do outro
- Exemplo:

```
minkowski() {
    cube(10);
    sphere(r=3);
}
```





Cor

- **Colore** o objeto
- Várias formas:
  - nome da cor
  - rgba

```
color(“green”) {
    cylinder(h=10, r1=10, r2=0);
}
```

Combinação de transformações

Vamos modelar este objeto: