CGI MINT für Zuhause

Smiley-Stift-Figur

Viele Gegenstände werden in großen Fabriken produziert. Manche Gegenstände sind nützlich und manche sind einfach nur hübsch.

Kann man Gegenstände auch selber herstellen?

Inhalt:

- Konstruktion
- Produktion
- Tipps & Tricks

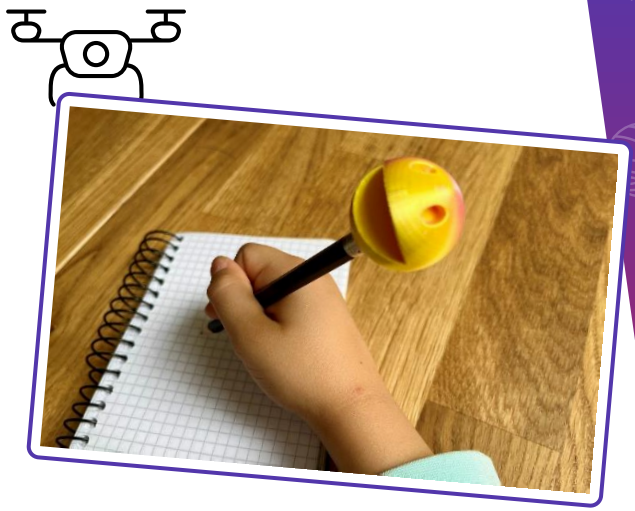
Alter:

Ab 15 Jahre

Dauer:

Ca. 2 Stunden

Stift-Figur mit OpenSCAD & 3D-Drucker



OpenSCAD ist ein kostenloses CAD-Programm, mit dem man mit wenigen Befehlen 3D-Modelle modellieren kann.

Wenn du dich schon mit OpenSCAD auskennst, kannst du direkt ein neues Projekt beginnen und mit der Schritt-für-Schritt-Anleitung in diesem Heft weitermachen.

Falls dies dein erstes Projekt mit OpenSCAD ist, schau dir vorher unseren „OpenSCAD - Schnellstart“ an. Du findest das Heft unter https://www.cgi.com/de/mint im Tab Technik.

Deine Stift-Figur kannst du anschließend mit einem 3D-Drucker ausdrucken. Wenn du keinen eigenen 3D-Drucker hast, haben wir zum Schluss ein paar Tipps für dich.

Das Konstruieren von eigenen Gegenständen ist gar nicht schwer.

Möchtest du wissen, wie das geht?

1. Konstruiere einen Smiley mit OpenSCAD.
2. Drucke den Smiley mit einem 3D-Drucker aus.
3. Überasche deine Freunde mit deiner eigenen Stift-Figur.

Das brauchst du:

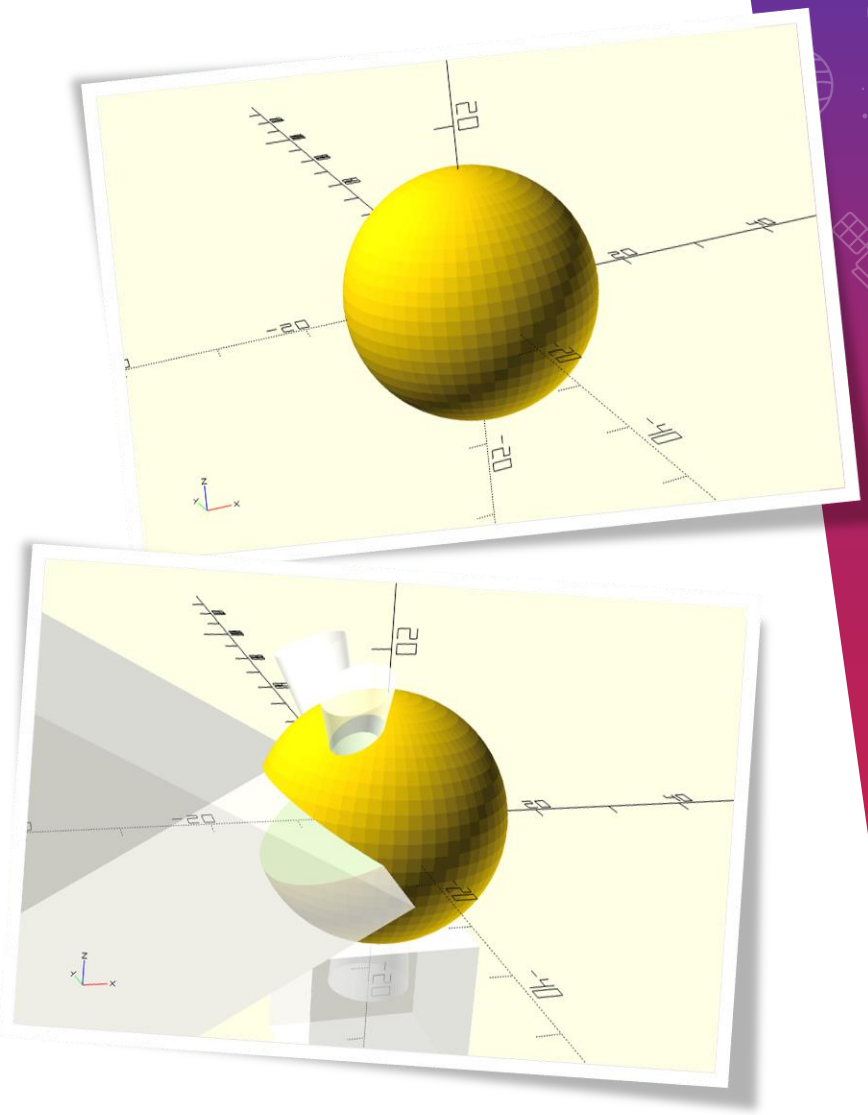
- Computer
- OpenSCAD (Kostenlose Software)
- 3D-Drucker
- Gelbes Filament

Grundform [1/8]

Das erste Objekt ist eine Kugel. Diese lässt sich mit der Funktion **sphere()** im Ursprung des Koordinatensystems erzeugen

Der Parameter **d** definiert den Durchmesser der Kugel. Die Angabe ist in Millimetern. Die Kugel hat eine Größe von 30 Millimetern, also 3 cm. Mit dem zweiten Parameter **$fn** gibt man die Anzahl der Fragmente an, aus denen die Kugel aufgebaut sein soll. Je höher die Anzahl desto runder ist die Kugel. 60 ist ein guter Wert. Höhere Werte sind sehr rechenintensiv für den Computer und können die Vorschau verlangsamen.

```
sphere(d=30, $fn=60);
```



Damit ist die Grundform fertig. Alles Weitere werden **Vertiefungen** in diesem Objekt.

Vertiefungen erreicht man, indem man die Formen anderer Objekte mit Booleschen Operationen von der Grundform abzieht. Das zweite Bild zeigt, wie später andere Objekte mit der Operation **Differenz** von der Grundform abgezogen werden.

Nutze die Schreibweise für Kommentare „//“, um die Grundform auszukommentieren. Dadurch wird dieser Befehl bei der Vorschau nicht mehr berücksichtigt und du kannst ihn trotzdem später wiederverwenden. Mache danach mit den anderen Objekten für die Vertiefungen weiter.

```
//sphere(d=30, $fn=60);
```

Mund [2/8]

Die Form für den Mund ist das Ergebnis von vier aufeinander folgenden Befehlen.

Mit der Funktion **cirle()** wird ein zweidimensionaler Kreis mit 30 mm Radius erzeugt. Durch die Angabe, dass der Kreis nur aus drei Fragmenten bestehen soll, wird er zum Dreieck.

```
circle(r=30, $fn=3);
```

Mit der Funktion **linear_extrude()** lässt sich aus dem Dreieck ein Keil machen. Der Parameter definiert die Höhe der Extrusion. Die genaue Höhe spielt hier keine Rolle, da der Keil später nur breiter als die Grundform sein soll.

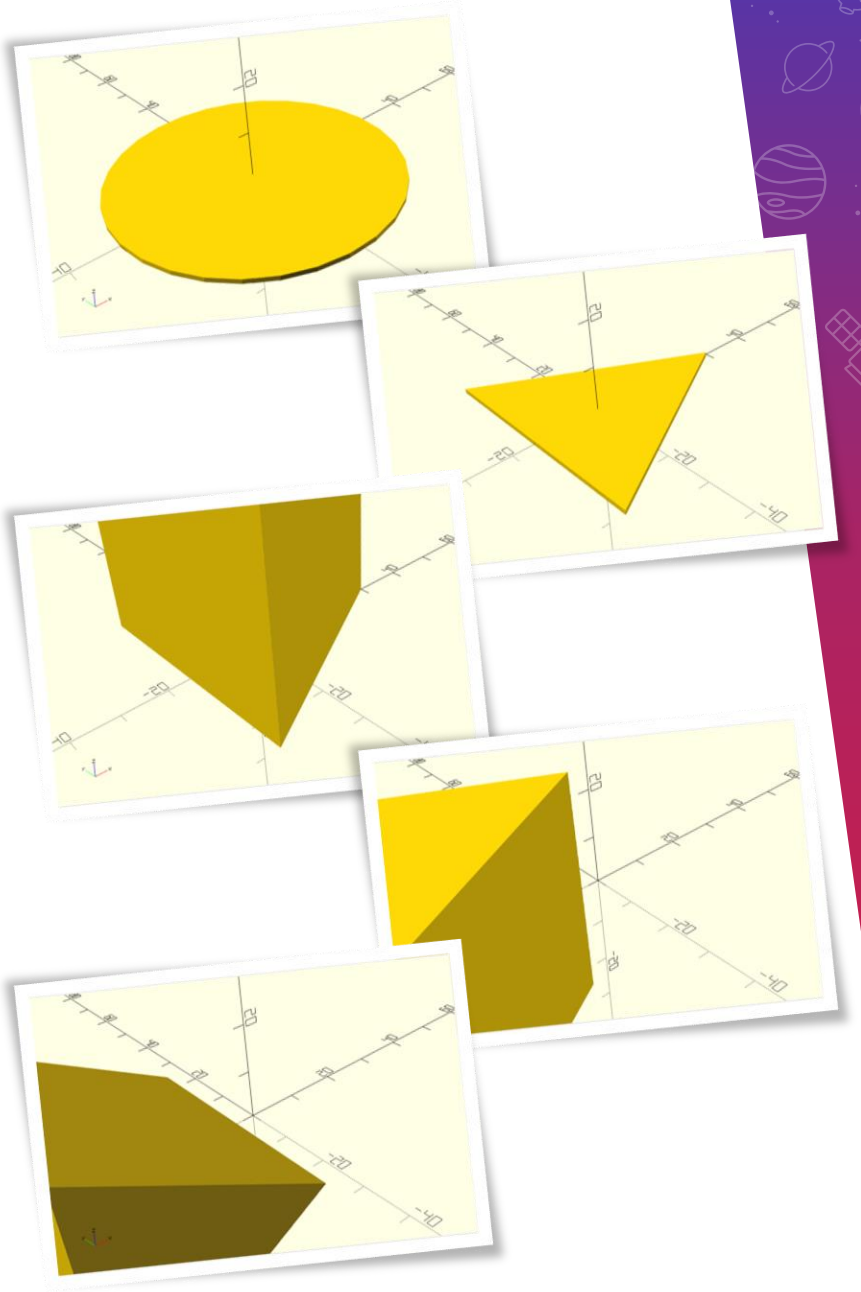
```
linear_extrude(height=50)
```

Mit der **translate()**-Funktion lässt sich das Keil-Objekt im Koordinatensystem verschieben. Hierfür wird als Parameter ein Vektor **[x,y,z]** übergeben. Damit wird die neue Position des Objektes bestimmt.

```
translate([-34,0,-25])
```

Mit der **rotate()**-Funktion wird das Keil-Objekt um die X-Achse gedreht, indem pro Achse **[x,y,z]** die Gradzahl bestimmt wird.

```
rotate([90,0,0])
```



Eigene Funktion [3/8]

Damit man das Keil-Objekt später einfach wieder aufrufen kann, wird die gesamte Befehlskette in eine eigene Funktion mit dem Schlüsselwort **module** eingebettet und **create_mouth()** benannt.

```
module create_mouth() {
      rotate([90,0,0])
      translate([-34,0,-25])
      linear_extrude(height=50)
      circle(r=30, $fn=3);
}
```

Die Zeilenumbrüche und die Einzüge dienen nur der besseren Lesbarkeit. Das Programm wertet die Befehlskette vom Semikolon ausgehend von *unten nach oben* aus, genauso als hätte man die Befehle hintereinander geschrieben.

Linkes Auge [4/8]

Mit fünf aufeinander folgenden Befehlen wird die Form für das linke Auge erzeugt.

Den Kreis mit 2 mm Radius kann man mit der **scale()**-Funktion zu einer Ellipse ausdehnen. Mit dem Vektor **[x,y,z]** gibt man an, dass der Kreis sich auf der Y-Achse um das Eineinhalbfache ausdehnt. Dadurch sehen die Augen nicht aus wie Knopfaugen.

```
scale([1,1.5,1])
circle(r=2, $fn=60);
```

Mit der Funktion **linear_extrude()** entsteht wieder ein dreidimensionales Objekt. Durch den Parameter **scale** lässt sich die obere Fläche des Objektes erneut skalieren und wird um das Eineinhalbfache vergrößert. Die untere Fläche bleibt unverändert. Dadurch entsteht keine zylindrische, sondern eine Kegel-ähnliche Form, die später einen schönen optischen Effekt erzielt.

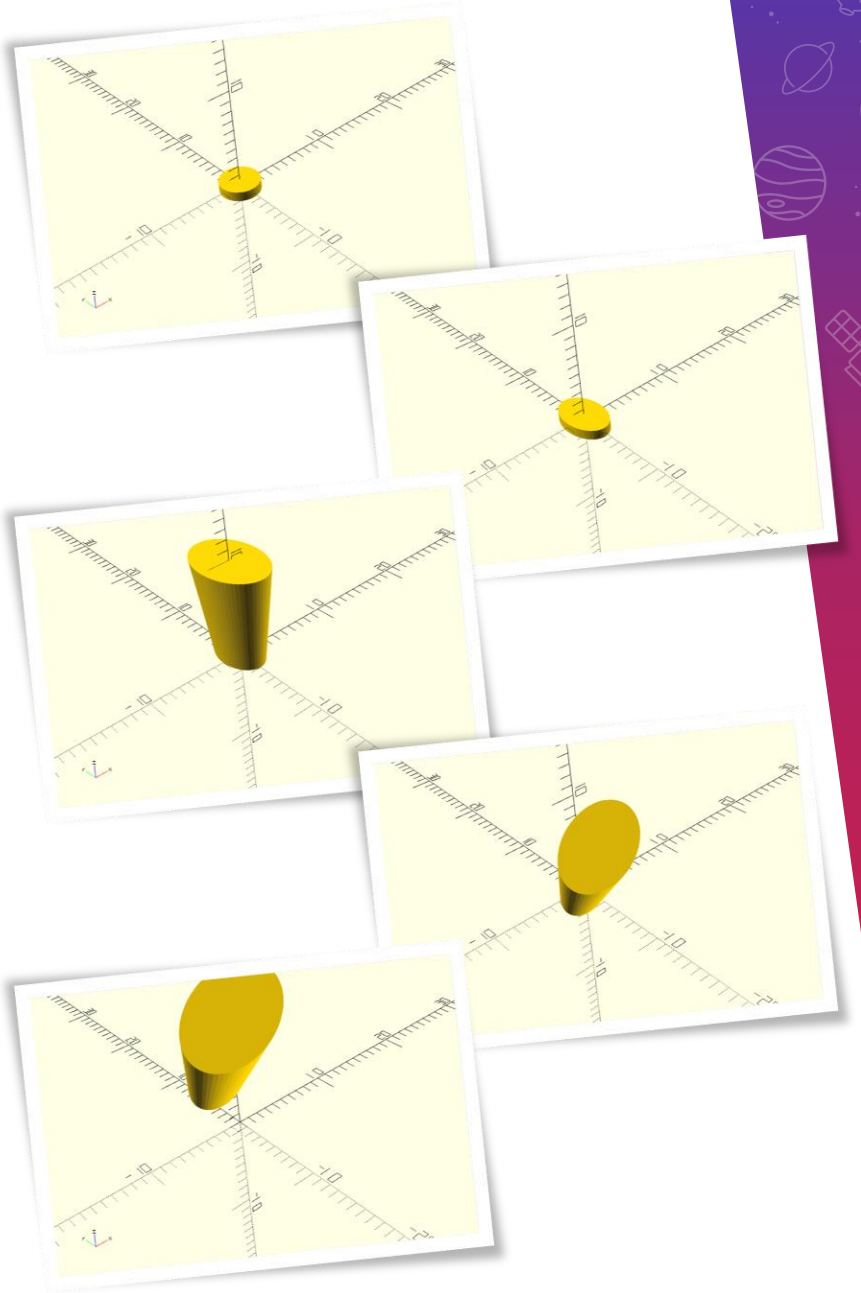
```
linear_extrude(height=10, scale=1.5)
```

Mit der **rotate()**-Funktion wird das Objekt richtig ausgerichtet.

```
rotate([-25,-25,120])
```

Mit der **translate()**-Funktion wird das Objekt an die richtige Position bewegt.

```
translate([-5,-5,10])
```



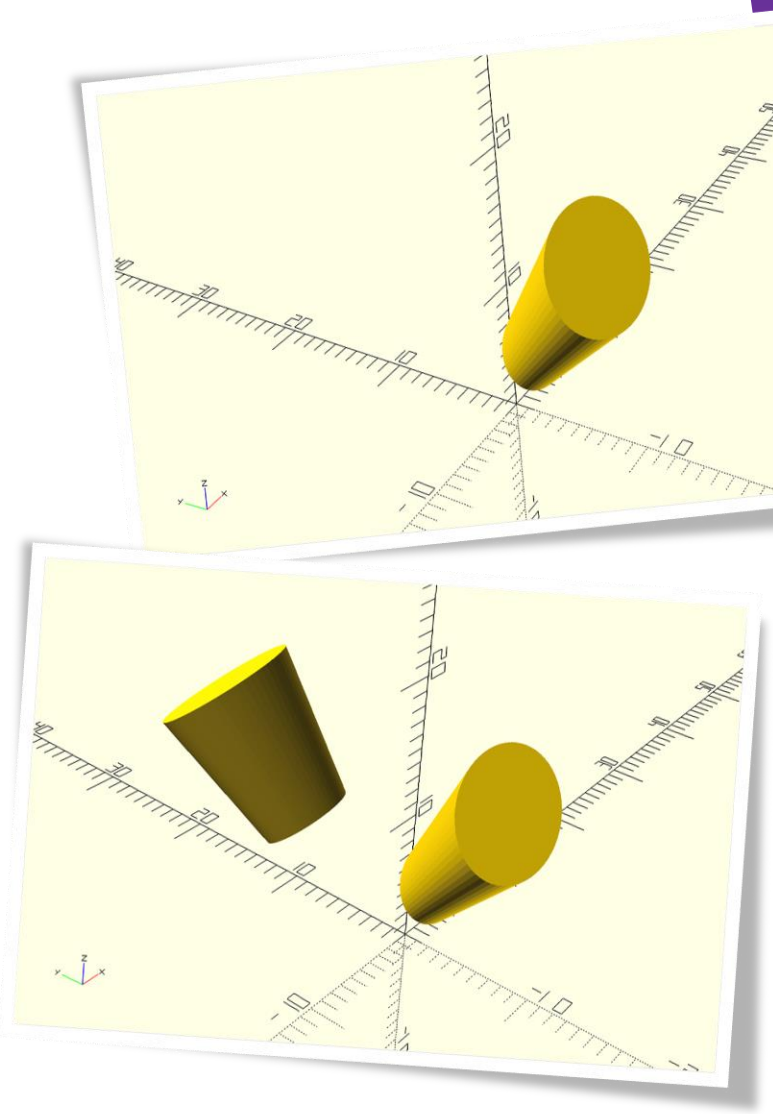
Rechtes Auge [5/8]

Für das zweite Auge werden einfach alle Befehle wiederholt und nur die Parameter für Ausrichtung und Position verändert. So entsteht mit wenigen Anpassungen das rechte Auge.

Für einen späteren Aufruf beider Augen-Objekte wird schließlich die gesamte Befehlskette in eine eigene Funktion eingebettet und **create_eyes()** benannt. Kommentare innerhalb der Funktion helfen den Code leserlich zu gestalten.

```
module create_eyes() {
  // Linkes Auge
  translate([-5,-5,10])
  rotate([-25,-25,120])
  linear_extrude(height=10, scale=1.5)
  scale([1,1.5,1])
  circle(r=2, $fn=60);

  // Rechtes Auge
  translate([-5,5,10])
  rotate([25,-25,-120])
  linear_extrude(height=10, scale=1.5)
  scale([1,1.5,1])
  circle(r=2, $fn=60);
}
```



Loch [6/8]

Um den Smiley auf einen Stift setzen zu können fehlt noch ein Loch. Das Loch soll eine konische Form haben, damit man den Stift in die Vertiefung schieben kann bis er stecken bleibt. Auch diese Form wird später von der Grundform abgezogen.

Mit der **cylinder()**-Funktion gibt es eine weitere Möglichkeit eine konische Form zu erzeugen. Der Durchmesser für die untere Fläche (Parameter **d1**) und der Durchmesser für die obere Fläche (Parameter **d2**) lässt sich jeweils einzeln einstellen.

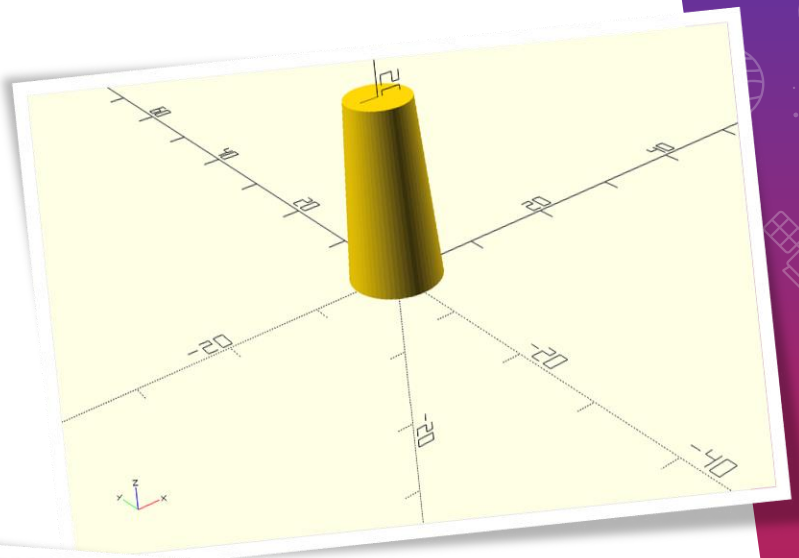
```
cylinder(d1=10, d2=7, h=20, $fn=60);
```

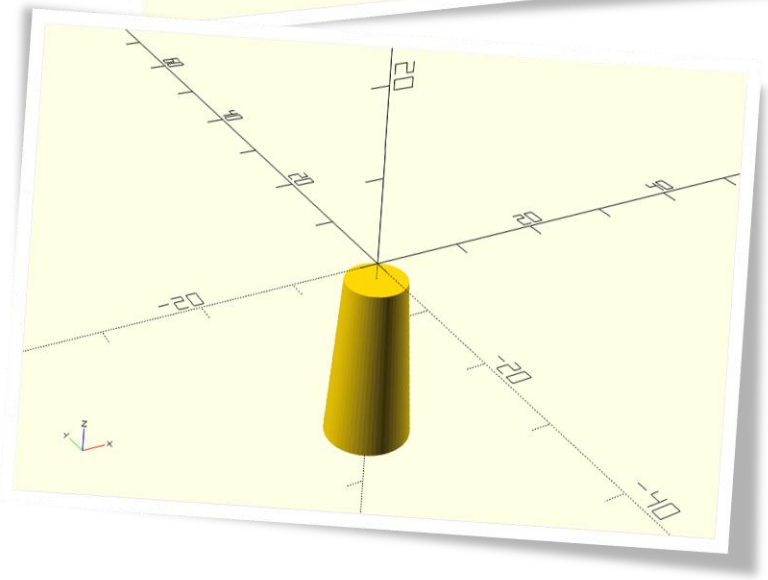
Mit der **translate()**-Funktion wird der Zylinder entlang der Z- Achse bewegt und so in die richtige Position gebracht.

```
translate([0,0,-22])
```

Auch diese Befehlskette wird wieder in eine eigene Funktion eingebettet und **create_pencil_hole()** benannt.

```
module create_pencil_hole() {
     translate([0,0,-22])
     cylinder(d1=10, d2=7, h=20, $fn=60);
}
```





Haftung [7/8]

Damit der Smiley später beim Drucken nicht von der Druckplatte gerissen wird, wird die Grundfläche etwas vergrößert indem ein Teil der unteren Grundform abgeschnitten wird. Dadurch hat das Objekt während des Drucks genug Halt.

Mit der **cube()**-Funktion entsteht ein Würfel. Der erste Parameter bestimmt die Kantenlänge für jede Achse **[x,y,z]**. Der Parameter **center** gibt an, dass das Objekt im Ursprung des Koordinatensystems erzeugt werden soll.

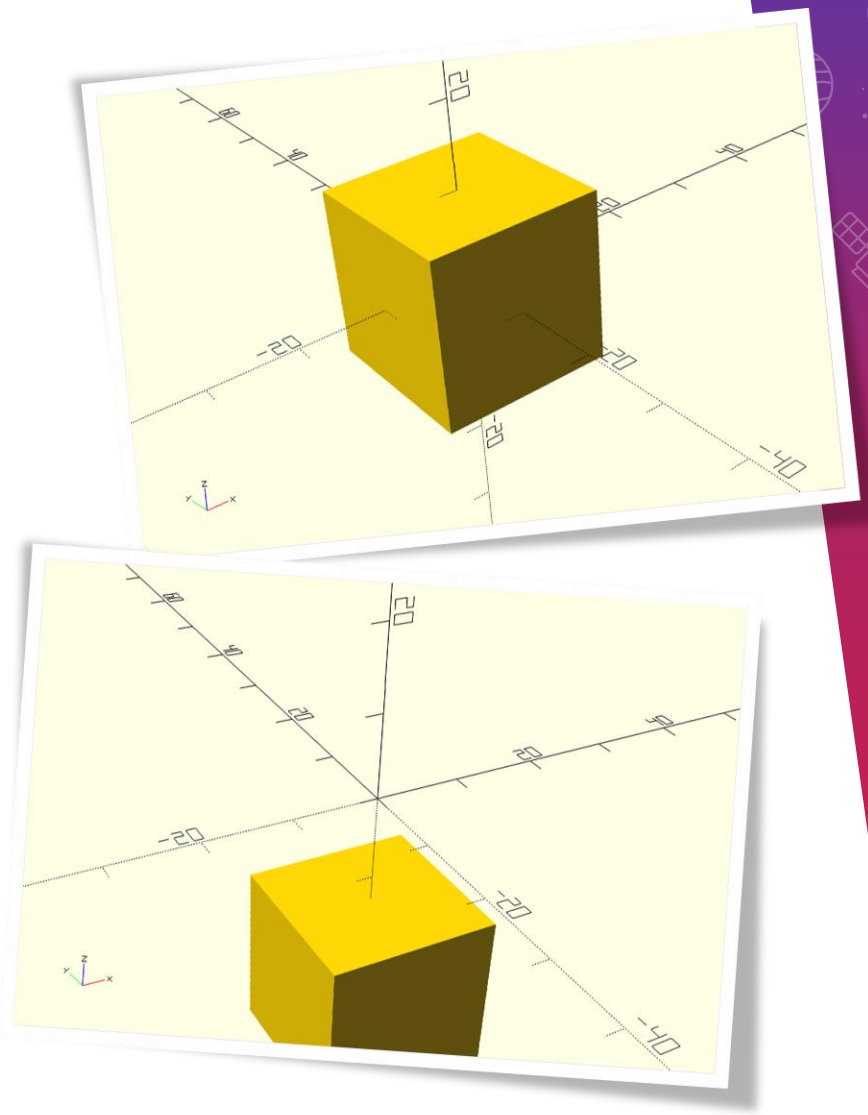
```
cube([20,20,20], center=true);
```

Mit der **translate()**-Funktion wird der Würfel an die richtige Position bewegt.

```
translate([0,0,-23])
```

Auch diese Befehlskette wird wieder in eine eigene Funktion eingebettet und **create_stand()** benannt.

```
module create_stand() {
      translate([0,0,-23])
      cube([20,20,20], center=true);
}
```

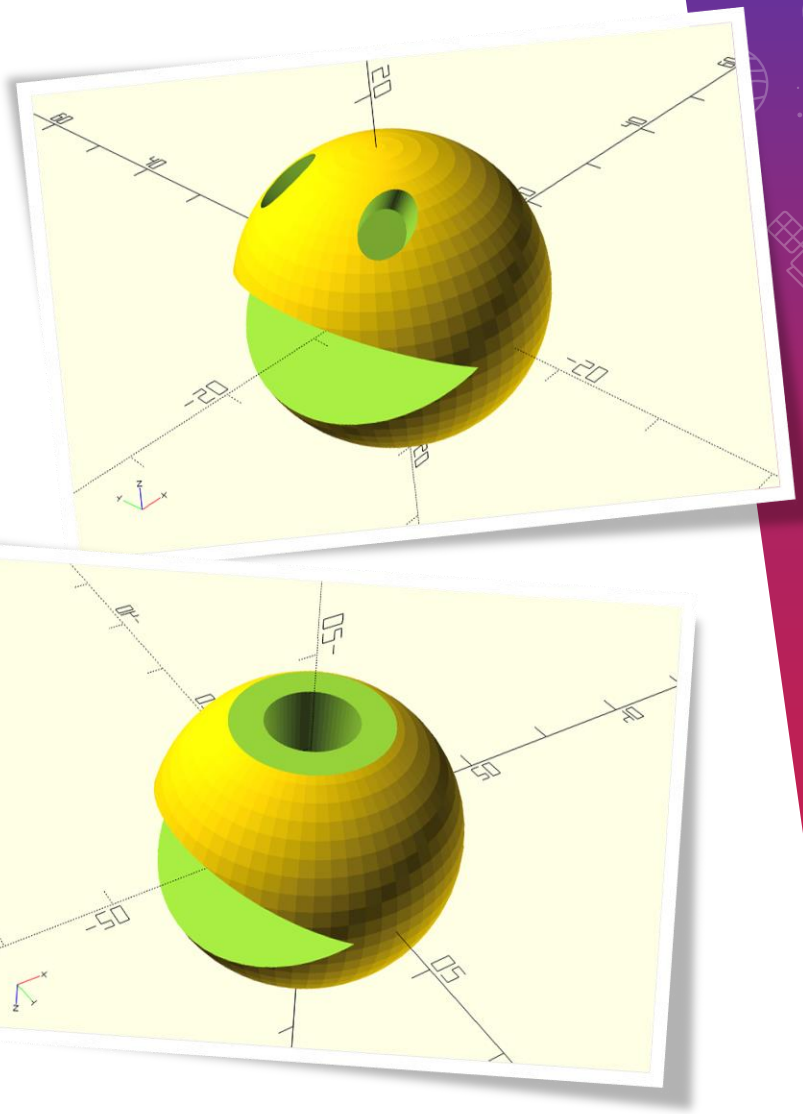


Smiley [8/8]

Nachdem die Grundform und alle Vertiefungs-Objekte gebaut wurden, können alle Objekte über Funktionsnamen aufgerufen werden.

Mit der **difference()**-Funktion lassen sich die Vertiefungen in der Grundform erzeugen. Bei dieser Booleschen Operation werden alle Objekte miteinander kombiniert. Dabei ist das erste Objekt das Objekt, auf dem die Operation angewendet wird. D.h. alle folgenden Objekte werden aus dem ersten Objekt herausgeschnitten.

```
difference() {
      sphere(d=30, $fn=60);
      create_mouth();
      create_eyes();
      create_pencil_hole();
      create_stand();
}
```



Das zweite Bild zeigt den Smiley von unten. Aus dieser Perspektive lässt sich gut die breitere Grundfläche und das Loch für den Stift erkennen.

3D-Druck

Bitte einen Erwachsenen, dass er dir bei der Bedienung des 3D-Druckers hilft.

Fasse **nicht** in den Drucker während er druckt!

1. Exportiere dein 3D-Modell in OpenSCAD als STL-Datei.
2. Lade die STL-Datei in eine Slicing-Software. Die Slicing-Software teilt das 3D-Modell in Schichten auf, die ein 3D-Drucker dann nacheinander drucken kann.
3. Speichere alle Einstellungen in einer G-code-Datei. Standardeinstellungen sind ausreichend. Supportstruktur ist bei diesem Modell nicht notwendig.
4. Übertrage die G-code-Datei an einen 3D-Drucker.
5. Lege gelbes Filament in den 3D-Drucker ein.
6. Starte den Druck. Der Druck wird ein paar Minuten dauern. Während des Drucks kannst du beobachten, wie der Smiley Schicht für Schicht aufgebaut wird.

Tipps & Tricks

Ich habe keinen eigenen 3D-Drucker!

Wenn du keinen eigenen 3D-Drucker hast gibt es mehrere Möglichkeiten, die du ausprobieren kannst:

- Frag in deinem **Freundeskreis** nach. Evtl. hat jemand einen 3D-Drucker und druckt dir gerne deine Datei aus.
- Nutze einen **Onlinedienst**. Dort kannst du deine 3D-Datei hochladen und nach ein paar Tagen wird dir der fertige Ausdruck zugeschickt. Meistens sind diese Dienste kostenpflichtig.
- Prüfe das Angebot deiner **Schule** oder **Stadtbibliothek**. Manche städtischen Einrichtungen haben einen 3D-Drucker, den man nach Anmeldung sogar kostenlos benutzen kann.
- Erkundige dich, ob es eine **DIY-Community** in deiner Umgebung gibt. Viele Maker-Spaces haben nicht nur handwerkliche Werkzeuge, sondern auch 3D-Drucker.

OpenSCAD zeigt mir Fehlermeldungen an!

Vergleiche dein Coding mit dem Code auf der rechten Seite, um deinen Fehler zu finden. Du kannst den Code auch mit Copy & Paste ins Programm kopieren.

Vollständiger OpenSCAD Code

```
difference() {

    sphere(d=30, $fn=60);

    create_mouth();
    create_eyes();
    create_pencil_hole();
    create_stand();
}

module create_mouth() {

    rotate([90,0,0])
    translate([-34,0,-25])
    linear_extrude(height=50)
    circle(r=30, $fn=3);
}

module create_eyes() {

    // Linkes Auge
    translate([-5,-5,10])
    rotate([-25,-25,120])
    linear_extrude(height=10, scale=1.5)
    scale([1,1.5,1])
    circle(r=2, $fn=60);

    // Rechtes Auge
    translate([-5,5,10])
    rotate([25,-25,-120])
    linear_extrude(height=10, scale=1.5)
    scale([1,1.5,1])
    circle(r=2, $fn=60);
}

module create_pencil_hole() {

    translate([0,0,-22])
    cylinder(d1=10, d2=7, h=20, $fn=60); }

module create_stand() {

    translate([0,0,-23])
    cube([20,20,20], center=true);
}
```

CGI setzt sich dafür ein, junge Menschen für MINT*-Themen zu begeistern.

CGI MINT für Zuhause sind unterhaltsame Aktivitäten, mit denen man zuhause interessante Aspekte von Technologie, Wissenschaft und Nachhaltigkeit erkunden kann … und dabei jede Menge Spaß hat.

Die Aktivitäten richten sich an Kinder im Alter von 5 bis 15 Jahren.
Auf unserer **Website** können weitere Module kostenlos heruntergeladen werden.

* **MINT** ist eine zusammenfassende Bezeichnung für unterschiedliche, aber verwandte technische Disziplinen. **M**athematik, **I**nformatik, **N**aturwissenschaft und **T**echnik.

1976 gegründet und nach wie vor familiengeführt, ist CGI heute einer der weltweit größten unabhängigen Anbieter von IT und Business Consulting. Ein hohes Maß an Eigenverantwortung, Teamwork, Respekt und Zusammenhalt machen das Arbeiten bei uns besonders. Bei uns kannst du dein volles Potenzial entfalten!