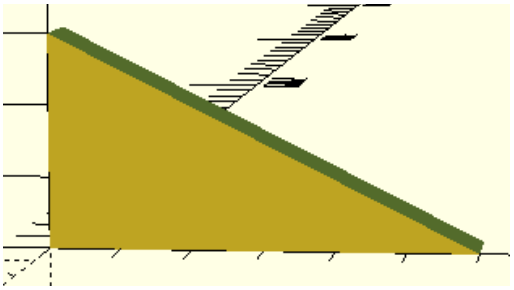
Winkelberechnung 1

Ein Quader soll diagonal geschnitten werden. Seine Ausmaße sind 60 x 30.

Aus :

soll …. werden!



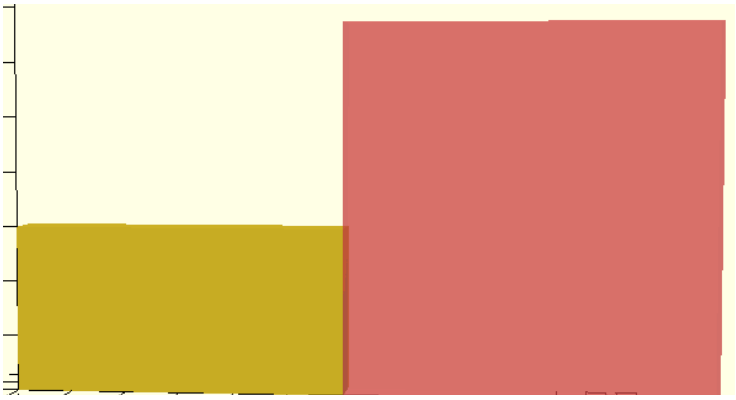
```
wert_x=60; //   X-Länge  vom Quader
wert_z=30; //   Z-Höhe   vom Quader
wert_y=10;  //  Y-Breite vom Quader
//              a² = b² + c²
//    daraus die Wurzel (sqrt) gezogen ergibt:
diagonale   = sqrt( (wert_x ^ 2) + (wert_z ^ 2) );
```

Die Diagonale kann einfach berechnet werden: Diagonale ² = Höhe ² + Breite ²

Dies sind 60² = 3600 und 30² = 900 ergibt: die Wurzel aus 4500 = 67,082.

Dies ist nun die Diagonale.

Nun wird ein Quader mit den Maßen 67,082 x 67,082 erzeugt und an dem zu schneidenden Quader direkt angestellt.
Dies ist gleichzeitig der Punkt zum Kippen.



Über die Winkelfunktion kann jetzt der Drehwinkel berechnet werden.

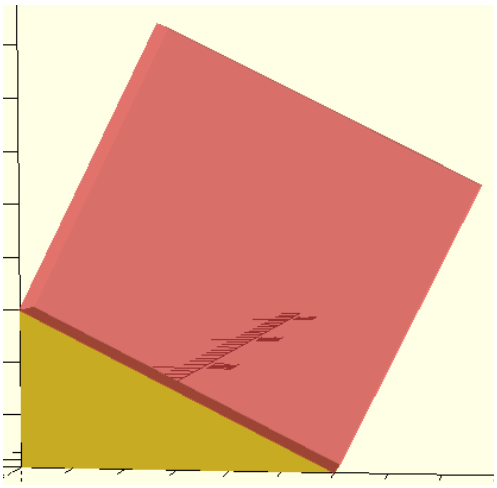
```
winkel = asin(wert_x / diagonale);
```

Die Berechnung ergibt einen Winkel von 63.4349 Grad.

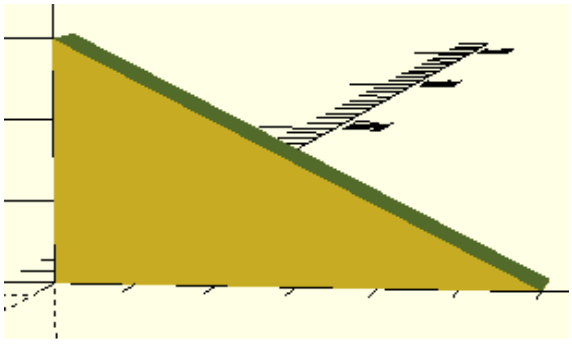
012-02

```
echo(" ---------------------------------------- ");
echo("Mein Quader hat eine Laenge von ", wert_x, " und eine Hoehe von ", wert_z);
echo("Die Diagonale ist: ",diagonale, "lang.");
echo("Der Kipp-Winkel betraegt: ", winkel, " Grad.");
echo(" ---------------------------------------- ");

difference(){
cube([wert_x,wert_y, wert_z]) ;

#translate([wert_x,-1,0]) rotate([0,-winkel,0])
  cube([wert_x, wert_y +2, diagonale]) ;
}
```

Da der Winkel nach links kippen soll, also gegen den Uhrzeigersinn, muss dieser mit einem Minus versehen werden. Außerdem ist der abschneidende Quader noch mit einem **#** versehen, damit dieser noch halb durchscheinend ersichtlich ist.



Wird das Rautenzeichen entfernt, ist somit der Quader über die Diagonale abgeschnitten.



Hier noch das gesamte Listing:

```
// Winkelberechnung 1

wert_x=60; //   X-Länge  vom Quader
wert_z=30; //   Z-Höhe   vom Quader
wert_y=10;  //  Y-Breite vom Quader

diagonale   = sqrt( (wert_x ^ 2) + (wert_z ^ 2) );
  winkel = asin(wert_x / diagonale);

echo(" ---------------------------------------- ");
echo("Mein Quader hat eine Laenge von ", wert_x, " und eine Hoehe von ",
wert_z);
echo("Die Diagonale ist: ",diagonale, "lang.");
echo("Der Kipp-Winkel betraegt: ", winkel, " Grad.");
echo(" ---------------------------------------- ");

difference(){
cube([wert_x,wert_y, wert_z]) ;

translate([wert_x,-1,0]) rotate([0,-winkel,0])
cube([wert_x, wert_y +2, diagonale]) ;
}
```

Bei den beiden Variablen „wert_x=60; //   X-Länge vom Quader“ und „wert_z=30; //   Z-Höhe   vom Quader“ können jederzeit andere Werte eingetragen werden.

Es wird dann berechnet und gekippt und abgeschnitten!

Ausprobieren.

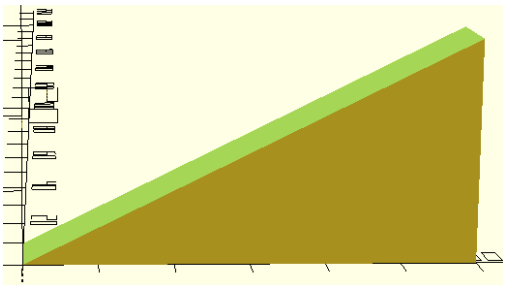
012-04

Winkelberechnung 2

Im vorhergehenden Beispiel wurde ein Winkel berechnet. Nun soll der Quader nochmals diagonal geschnitten werden. Seine Ausmaße sind 60 x 30.

Aus :

soll diesmal …. werden!



Die Abmessungen sind wie vorher:

```
wert_x=60; //   X-Länge  vom Quader
wert_z=30; //   Z-Höhe   vom Quader
wert_y=10;  //  Y-Breite vom Quader

diagonale   = sqrt( (wert_x ^ 2) + (wert_z ^ 2) );
```

Die Diagonale kann auch hier einfach berechnet werden:
$a^2 = b^2 + c^2$ entspricht: Diagonale ^{2} = Höhe ^{2} + Breite ^{2}

Dies sind 60^2 = 3600 und 30^2 = 900 ergibt: die Wurzel aus 4500 = 67,082.
Und schon ist das Maß der Diagonalen / Hypotenuse vorhanden!

Nun geht es los mit der Rechnerei: man müsste im Tabellenbuch nachschlagen unter Trigonometrische Funktionen und dann dort nach dem Winkel sehen ….

Vergessen! Es wird mit OpenSCAD gearbeitet!

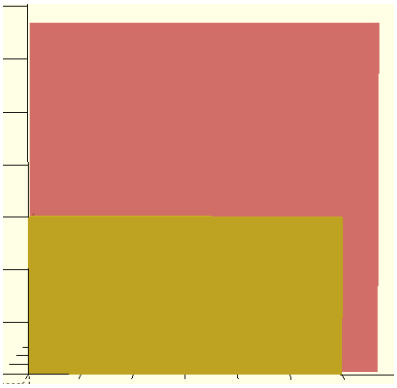
OpenSCAD hat wunderbare Funktionen bereits eingebaut: „asin“ und „acos“.
Damit werden die Ergebnisse ohne Tabellenbuch gleich ausgegeben – und zwar in Winkelgrade.

```
  w1 = asin(wert_x / diagonale);
  w2 = acos(wert_x / diagonale);
```

Als Muster werden die Ergebnisse in der Konsole ausgegeben:

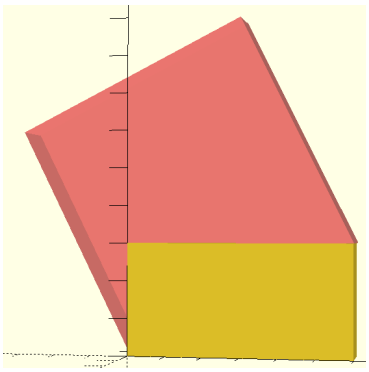
```
ECHO: " ---------------------------------------- "
ECHO: "Der Quader hat eine Laenge von ", 60, " und eine Hoehe von ", 30
ECHO: "Die Diagonale ist: ", 67.082, "lang."
ECHO: ""
ECHO: "Oberer Winkel von: ", 63.4349, " Grad."
ECHO: "-asin- ", 63.4349, " Grad."
ECHO: ""
ECHO: "Untere Winkel von: ", 26.5651, " Grad."
ECHO: "-acos- ", 26.5651, " Grad."
ECHO: " ---------------------------------------- "
```

Hier sieht man gleich sehr schön, was „asin“ und „acos“ leisten: die Ergebnisse in Grad.

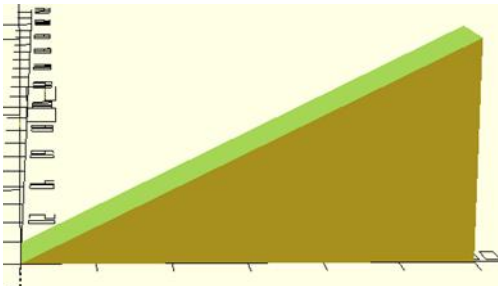


So kann nun der mit „**#**“ in rot markierte Quader im Hintergrund gekippt werden:

```
difference(){
cube([wert_x,wert_y, wert_z]) ;

translate([0,5,0])
   #translate([0,0,0]) rotate([0,-w2,0])
cube([diagonale, wert_y +2, diagonale]) ;
}
```



Nun wird der hintere rote Quader nach vorn gesetzt: translate([0,-1,0])
Dieser steht nun auf jeder Seite um 1 über, damit sauber geschnitten werden kann.
Nach Entfernen der Raute „#“ sieht dann alles ganz gut aus!



012-06

Es kann mit „asin“ gerechnet werden: 90 – 63,4349 Grad ergibt 26,5651
oder gleich mit „acos“, das den Wert sofort präsentiert: 26,5651.

Das Dreieck hat eine Länge von 60 und eine Höhe von 30.
Die Diagonale / Hypotenuse ist 67,082.

Der kleine Winkel wird über „sin α = Gegenkathete durch Hypotenuse“ berechnet:
(30 : 67,082) = 0,44721.
Nach Tabellenbuch sind dies etwa 26 ° 34‘ = 26 Grad 34 Minuten.

Gehen wir aus von:
100/100stel Minuten entsprechen 60/60stel Minuten

so werden aus 10/100stel > 6/60stel.
Oder 1,6666/100stel entsprechen 1/60stel.

Teilen wir nun die Kommazahl von 0,5651 durch 1,666
so erscheint als Ergebnis: 0,339073 ….

Dies ist nun 0,3390 Minuten in 60/60stel – also 33,90 Minuten!

So schlecht rechnet demnach OpenSCAD gar nicht!

Das komplette Listing:

```
// Winkelberechnung  2

wert_x=60; //   X-Länge  vom Quader Ank
wert_z=30; //   Z-Höhe   vom Quader Gegenk
wert_y=10;  //  Y-Breite vom Quader

diagonale   = sqrt( (wert_x ^ 2) + (wert_z ^ 2) );

  w1 = asin(wert_x / diagonale);
  w2 = acos(wert_x / diagonale);

echo(" ---------------------------------------- ");
echo("Der Quader hat eine Laenge von ", wert_x, " und eine Hoehe von ", wert_z);
echo("Die Diagonale ist: ",diagonale, "lang.");
echo("" );
echo("Oberer Winkel von: ", w1, " Grad.");
echo("-asin- ", w1, " Grad.");
echo("" );
echo("Untere Winkel von: ", w2, " Grad.");
echo("-acos- ", w2, " Grad.");

echo(" ---------------------------------------- ");

difference(){
cube([wert_x,wert_y, wert_z]) ;

translate([0,-1,0])
   translate([0,0,0]) rotate([0,-w2,0])
// entspricht auch:
//    translate([0,0,0]) rotate([0,-(90-w1),0])
cube([diagonale, wert_y +2, diagonale]) ;

}
```

w2 entspricht auch dem Winkel w1 von 90 subtrahiert. Ein positiver Wert lässt das Objekt im Uhrzeigersinn drehen, ein negativer Wert gegen ihn.

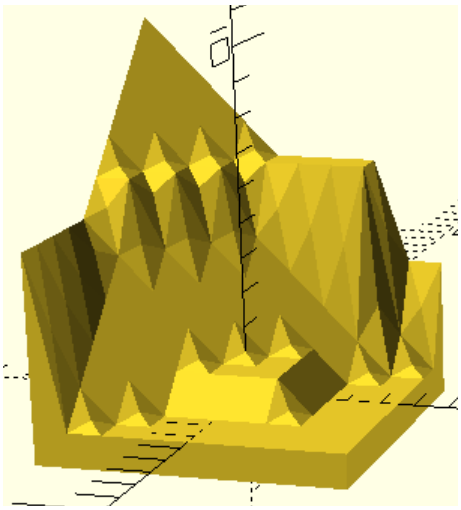
012-08

surface (file, center, invert, convexity);

3D-Form aus Bilddatei/Zahlendatei

Als erstes wird unter dem Namen „muster.dat“ die folgende Datei abgespeichert.	10 9 8 7 6 5 5 5 5 5 9 8 7 6 6 4 3 2 1 0 8 7 6 6 4 3 2 1 0 0 7 6 6 4 3 2 1 0 0 0 6 6 4 3 2 1 1 0 0 0 6 6 3 2 1 1 1 0 0 0 6 6 2 1 1 1 1 0 0 0 6 6 1 0 0 0 0 0 0 0 3 1 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0

```
surface(file = "012-muster.dat", center =
true, convexity = 5);
```

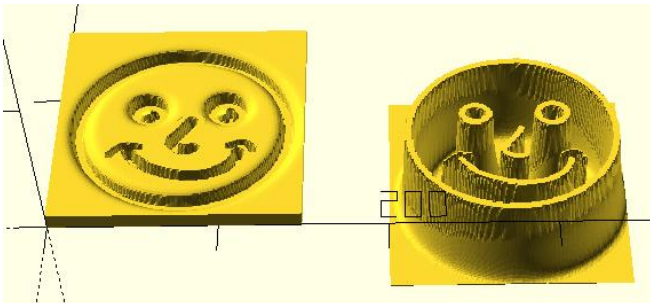


Mit der obigen Zeile wird die Datei *muster.dat* eingelesen und die Zahlenwerte als Muster dargestellt.

surface 2

Es können ebenso PNG-Dateien eingelesen und auch invertiert werden:

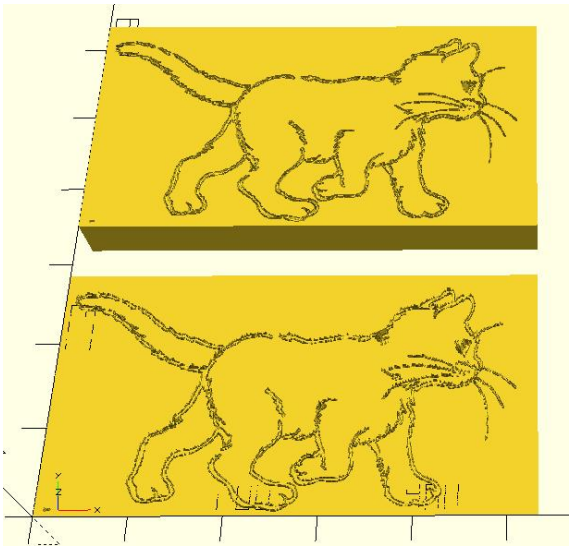
```
// surface_2

scale([1, 1, 0.1])
  surface(file = "012-smiley.png");

translate([200,0,0])    scale([1, 1, 0.6])
  surface(file = "smiley.png", invert = true);
```



scale([1,1,0.1]) lässt den Smiley nach oben wachsen (links im Bild),

dagegen scale([1, 1, 0.6]) und surface(file = "012-smiley.png", invert = true); lässt den Smiley auf der Oberfläche, zieht jedoch das Bild nach unten!

katze.png

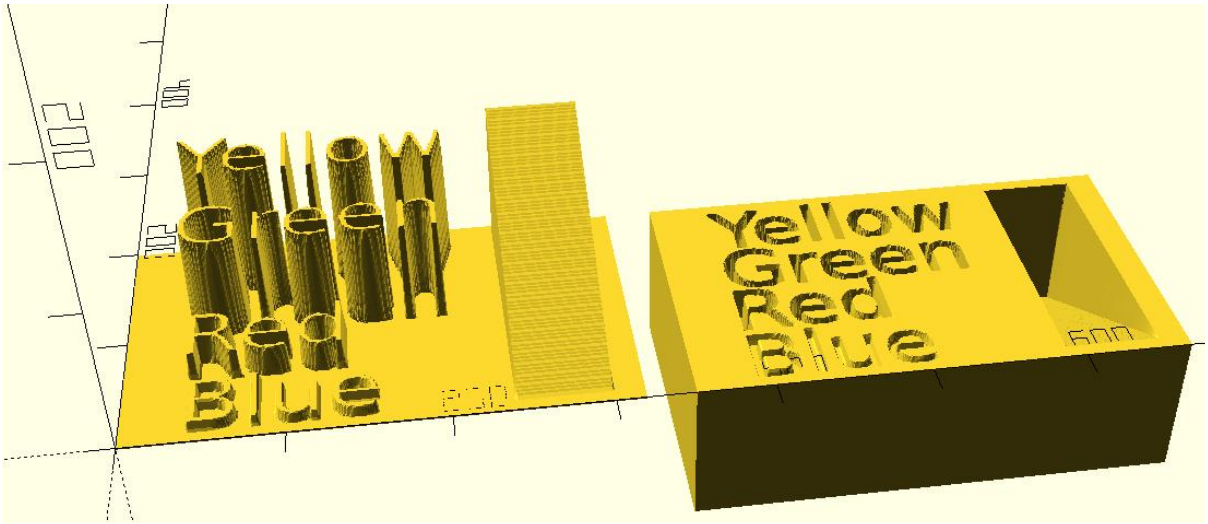


012-10

surface 3

Auch eine PNG-Datei mit RGB und Schwarz-Weiß-Abstufung lässt sich als Bild darstellen:


```
surface(file = "012-color.png", invert = false);

translate([350,0,0])
surface(file = "012-color.png", invert = true);
```



„invert=false“ ergibt das linke Teil im Bild
„invert=true“ zieht wiederum alles in die Tiefe: rechts im Bild