Polyhedron

Polyeder

Ein Polyeder ist allgemein ein einfacher 3D-Körper. Er kann verwendet werden, um jede regelmäßige oder unregelmäßige Form zu erstellen, einschließlich solcher mit konkaven sowie konvexen Merkmalen. Gekrümmte Oberflächen werden durch eine Reihe flacher Oberflächen angenähert.

polyhedron(points = [[X0, Y0, Z0], [X1, Y1, Z1], ...], faces = [[P0, P1, P2, P3, ...], ...], convexity = n);

Die Parameter haben folgende Bedeutungen

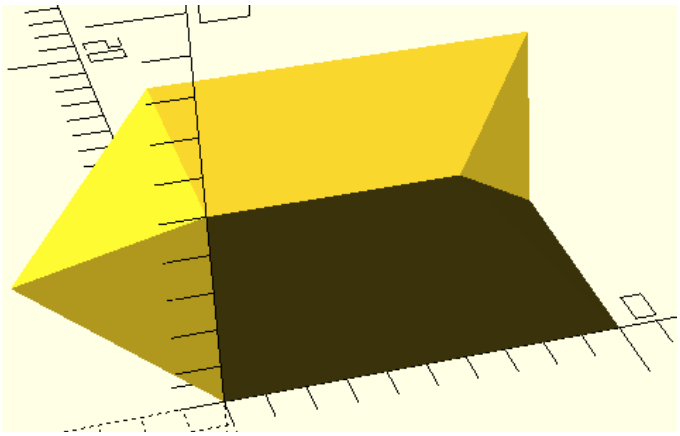
points Es werden die Eck-Punkte im Raum des Koordinatensystems über Vektoren [x,y,z] angegeben. Punkte können in beliebiger Reihenfolge definiert werden. N Punkte werden in der definierten Reihenfolge als 0 bis N-1 referenziert.

faces Flächen, die durch die Punktaufzählung in den eckigen Klammern festgelegt werden. Es müssen mindestens drei Punkte, oder auch mehr, zur Flächenbestimmung aufgezählt werden.
Es sollten genügend Flächen definiert sein, damit der Volumenkörper ohne Überlappung vollständig eingeschlossen wird. Sollten Punkte, die eine einzelne Fläche beschreiben, nicht auf derselben Ebene liegen, wird die Fläche nach Bedarf automatisch in Dreiecke geteilt.

convexity Ganze Zahl. Der Konvexitätsparameter gibt die maximale Anzahl von Flächen an, die ein Strahl, der das Objekt schneidet, durchdringen kann. Dieser Parameter wird nur für die korrekte Anzeige des Objekts im OpenCSG-Vorschaumodus benötigt. Es hat keine Auswirkung auf das Polyeder-Rendering. Bei Anzeigeproblemen sollte die Einstellung auf 10 in den meisten Fällen gut funktionieren.

Es soll konstruiert werden:

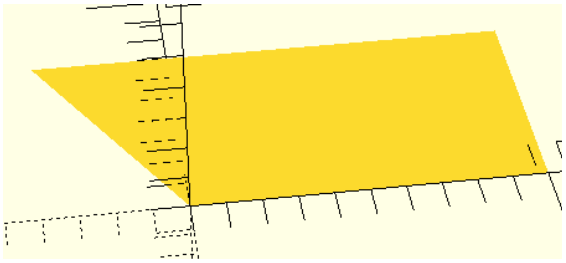
Ein „normaler“ Quader mit zwei versetzten Eckpunkten.



013-02

Es wird eine Fläche nach der anderen gezeichnet. Als Erstes kommen die 4 Punkte (0-3). Danach die Verbindungen der 4 Punkte [0,1,2,3]:

```
points = [
  [   0,  0,  0 ],    //0
  [ 10,  0,  0 ],   //1
  [ 10,  7,  0 ],   //2
  [ -4,  7,  0 ]    //3
  ];
  faces = [
  [0,1,2,3]  // unten
  ];
polyhedron( points, faces );
```

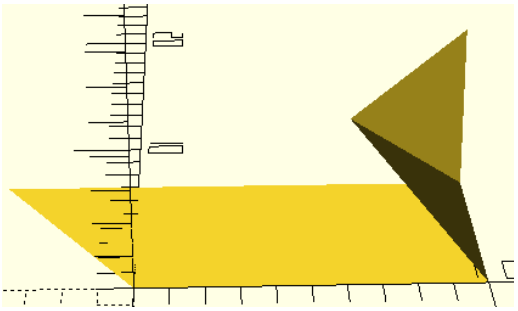


Nun folgen die oberen 4 Punkte mit ihren Angaben. Dann sieht es schon so aus:

```
points = [
  [   0,  0,  0 ],  //0
  [ 10,  0,  0 ],  //1
  [ 10,  7,  0 ],  //2
  [  -4,  7,  0 ],  //3
  [   0,  0,  5 ],  //4
  [   6,  0,  5 ],  //5
  [ 10,  7,  5 ],  //6
  [   0,  7,  5 ]   //7
  ];
```

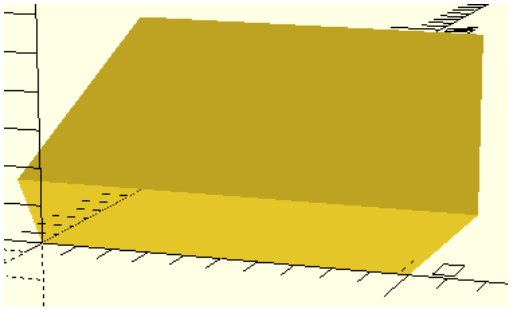
Hinter den Zahlen des Koordinatensystems stehen jeweils die Nummern der Punkte, die später miteinander verbunden werden sollen.

```
faces = [
  [0,1,2,3],  // unten
  [5,6,2,1]  // rechts
  ];

polyhedron( points, faces );
```



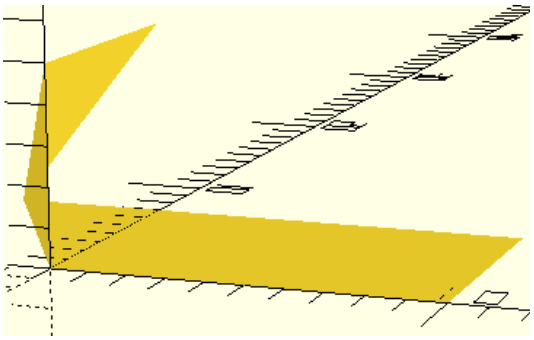
Die rechte Seite entsteht.

```
faces = [
  [0,1,2,3],  // unten
  [6,7,3,2]  // hinten
  ];
```



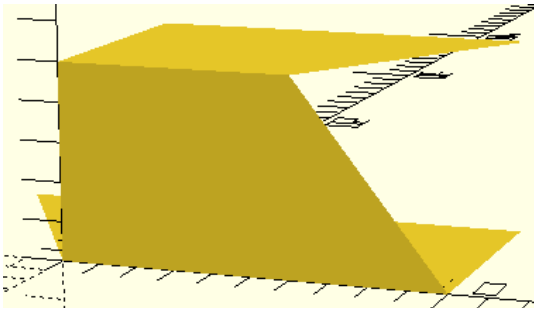
Auch die hintere Wand wird sichtbar. Anschließend folgt die zweigeteilte Wand auf der linken Seite:

```
  [0,1,2,3],  // unten
  [0,3,4],    // links1
  [7,4,3]     // links2
  ];
```



Die letzten zwei Wände: vorn und oben

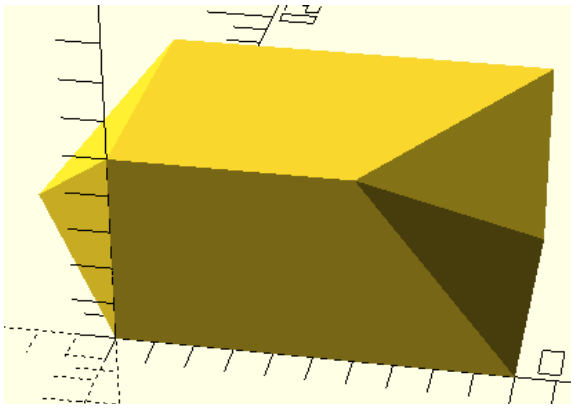
```
faces = [
  [0,1,2,3],  // unten
  [4,5,1,0],  // vorn
  [7,6,5,4]  // oben
  ];
```

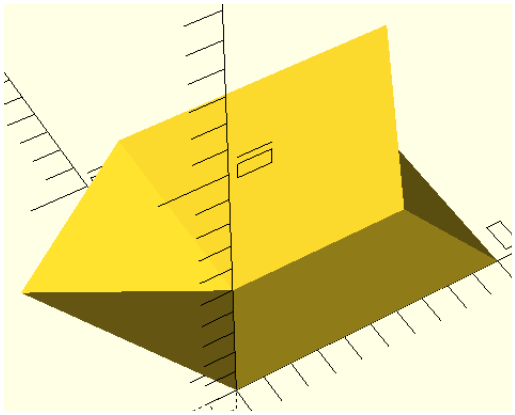


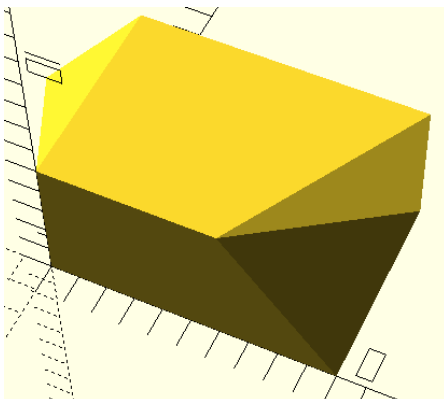
013-04

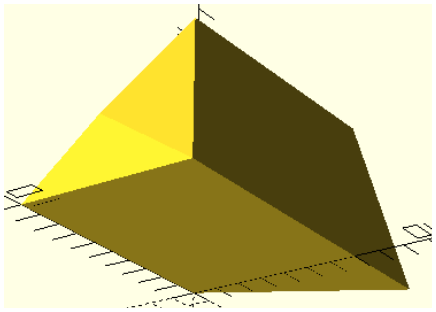
Im folgenden Listing sind nun alle Wände vereint dargestellt.

```
// polyhedron 1

points = [
  [  0,  0,  0 ],  //0
  [ 10,  0,  0 ],  //1
  [ 10,  7,  0 ],  //2
  [ -4,  7,  0 ],  //3
  [  0,  0,  5 ],  //4
  [  6,  0,  5 ],  //5
  [ 10,  7,  5 ],  //6
  [  0,  7,  5 ]   //7
  ];

faces = [
  [0,1,2,3],  // unten
  [4,5,1,0],  // vorn
  [7,6,5,4],  // oben
  [5,6,2,1],  // rechts
  [6,7,3,2],  // hinten
  [0,3,4],   // links1
  [7,4,3]    // links2
  ];

polyhedron( points, faces );
```





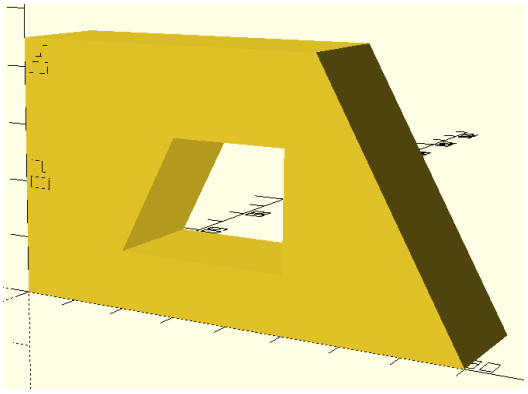




polyhedron 2

Ein Polyeder mit Durchbruch

Ein Polyeder (Vielflächen-Element, engl.: polyhedron) wird wie ein Polygon erstellt: definierte Punkte im Raum werden verbunden und dann zu einer Fläche zusammengefasst.



Hier kommt nun jedoch der Unterschied: jede Fläche hat zwei Seiten – eine Äußere in Gelb und eine Innere in Pink. Somit kann festgelegt werden, wo außen und innen liegen soll.

Nun wird ein Polyeder mit einem Durchbruch erstellt.

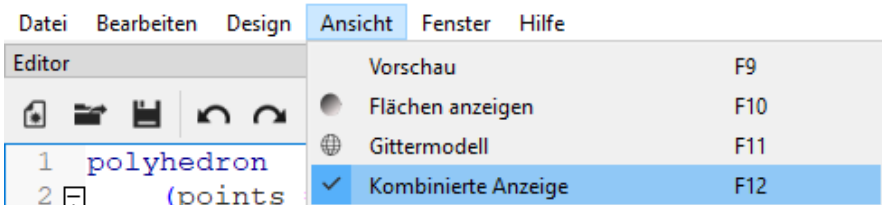
Zur Vereinfachung wurden einige Flächen vorläufig weggelassen. So werden die einzelnen Punkte sichtbar und können besser zugeordnet werden. Eine Handskizze mit Nummerierung der Punkte schafft schnelle Hilfe!

points = [
[0, 0, 0], //0
[0, 20, 0], //1
[0, 0, 45], //2
[0, 20, 45], //3
 	
[80, 0, 0], //4
[80, 20, 0], //5
[55, 0, 45], //6
[55, 20, 45]]; //7
 	
faces = [
[0,1,3,2], // linke Außenseite
[1,0,4,5], // untere Außenseite
[4,6,7,5], // rechte Außenschräge
[2,3,7,6]]; // obere Außenseite
 	
polyhedron(points, faces);
	Zuerst werden die Punkte im Raum definiert und mit Kommentarzeichen nummeriert.
	
	Anschließend werden einzelne Punkte nacheinander aufgerufen und damit zu einer Fläche gestaltet.
	
	OpenSCAD verbindet zwischen der letzten und der ersten Zahl von selbst und schließt damit die Ringform.

Es ist empfehlenswert über das obere Hauptmenu über Ansicht die „Kombinierte Anzeige“ auszuwählen – oder einfach über F12 aufrufen.

013-06

Damit erscheinen die Flächen in der gelben Außenfarbe und der pinken Innenfarbe.



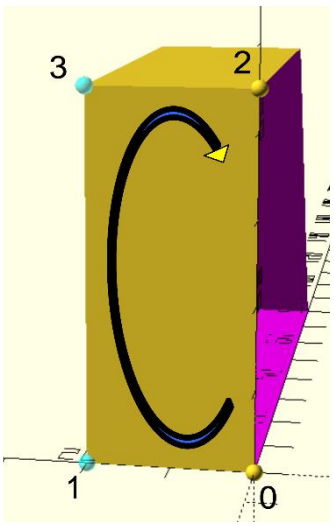
So wird sehr schnell ein Fehler ersichtlich. Es wird trotzdem die Fläche angezeigt – jedoch wird es sich nicht Rendern lassen! Zurück mit F9=Vorschau.

[F9] / [F12]

Technik im Uhrzeigersinn

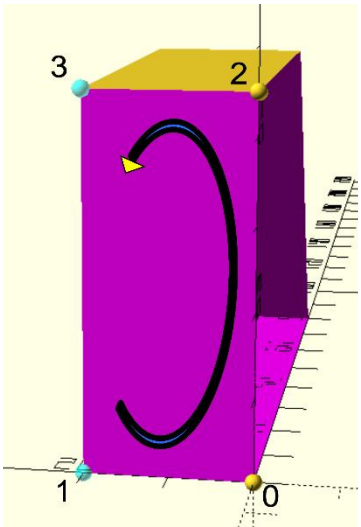
Wird eine Fläche von außen betrachtet und dazu deren Punktauflistung, so wird eindeutig, dass die Aufzählung im Uhrzeigersinn der Fläche läuft.

Wird die diese Technik im Uhrzeigersinn verwendet sind die Flächen immer außen der eigentlichen Form, also in der Farbe Gelb dargestellt.
Wird das Element gedreht, erscheint auf der gegenüberliegenden Seite die Fläche in Pink, also innen in der späteren Form.



Im linken Bild wird im Uhrzeigersinn aufgezählt: [0,1,3,2] . Hierbei ist es egal, bei welcher Zahl man beginnt, es gilt demnach ebenso [1,3,2,0] oder [3,2,0,1] oder [2,0,1,3] !

Im Bild rechts wird es zur Außenseite: [2,3,1,0]. Auch hier ist der Beginn unerheblich, Hauptsache die Richtung stimmt.



Regel für die linke Hand:

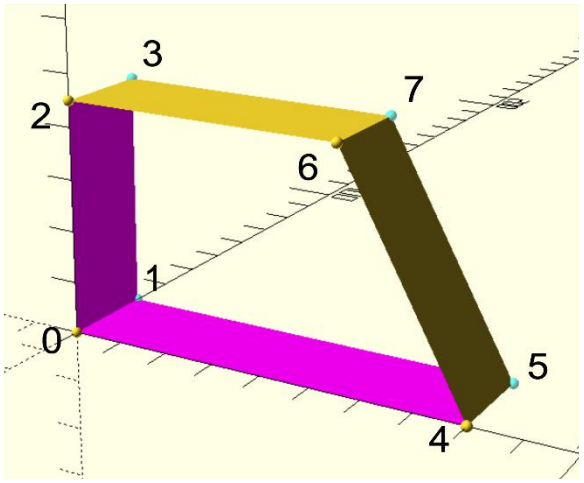
Wird die linke Hand auf die Fläche gelegt und die Finger in Richtung der Reihenfolge der Punkte gekrümmt, sollte der Daumen nach außen zeigen. So ist die Außenseite in Gelb erstellt.

Sollte diese Fläche innen, also in Pink erscheinen, ist die Aufzählung umzukehren - also gegen den Uhrzeigersinn auflisten.

Vorerst wurde nur mit der äußeren Hülle begonnen.

Die Punkte wurden wie folgt erstellt: gelbe Punkte und gerade Zahlen sind vorn, die ungeraden mit den hellblauen Punkten sind hinten.

Es werden mindestens drei oder mehr Punkte miteinander vernetzt.




```
points = [
  [  0,  0,  0 ],       //0
  [  0, 20,  0 ],       //1
  [  0,  0, 45 ],       //2
  [  0, 20, 45 ],       //3

  [ 80,  0,  0 ],       //4
  [ 80, 20,  0 ],       //5
  [ 55,  0, 45 ],       //6
  [ 55, 20, 45 ]];      //7

faces = [
  [0,1,3,2],  // linke Außenseite
  [1,0,4,5],  // untere Außenseite
  [4,6,7,5],  // rechte Außenschräge
  [2,3,7,6]]; // obere Außenseite

polyhedron( points, faces );
```

Über eine kleine Handskizze mit den Nummernangaben der Punkte freut sich bestimmt jeder, denn seine Arbeit wird dadurch erheblich erleichtert.

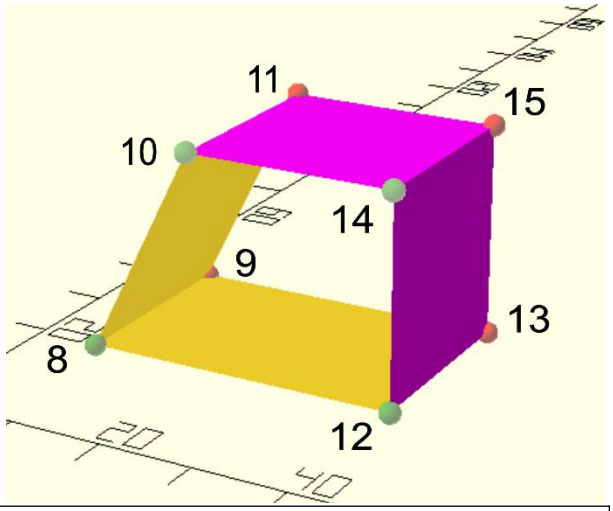
013-08

Oben unter „points“ sind die acht Punkte im Raum mit (x,y,z) jeweils aufgelistet, sowie deren Nummer als Kommentar.

Danach kommen die Flächen unter „faces“. Es werden in eckigen Klammern die Punkte durch Kommata getrennt angegeben die verbunden werden sollen. Somit entsteht eine Fläche nach der anderen.

Als Nächstes werden die Flächen des Durchbruches erzeugt. Diese sollten im Kernbereich innere Flächen in gelb aufweisen, sowie nach außen in pink.

Hier wurden die vorderen Punkte nun in Hellgrün und die hinteren in Rot dargestellt.



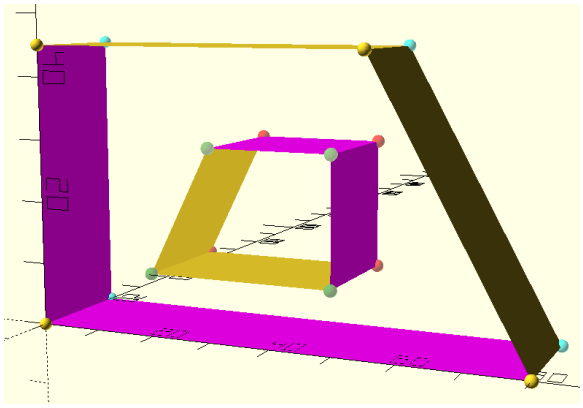

```
points = [
  [  0,  0,  0 ],  //0
  [  0, 20,  0 ],  //1
  [  0,  0, 45 ],  //2
  [  0, 20, 45 ],  //3

  [ 80,  0,  0 ],  //4
  [ 80, 20,  0 ],  //5
  [ 55,  0, 45 ],  //6
  [ 55, 20, 45 ],  //7

  [ 20,  0, 10 ],  //8
  [ 20, 20, 10 ],  //9
  [ 30,  0, 30 ],  //10
  [ 30, 20, 30 ],  //11

  [ 50,  0, 10 ],  //12
  [ 50, 20, 10 ],  //13
  [ 50,  0, 30 ],  //14
  [ 50, 20, 30 ]]; //15

faces = [
   /*
```

```
  [0,1,3,2],  // linke Außenseite
  [1,0,4,5],  // untere Außenseite
  [4,6,7,5],  // rechte Außenschräge
  [2,3,7,6]]; // obere Außenseite
  */

  [8,10,11,9],    // linke Innennseite
  [8,9,13,12],    // untere Innenseite
  [15,14,12,13],  // rechte Innenschräge
  [10,14,15,11]]; // obere Innenseite

polyhedron( points, faces );
```

Das Listing: über die Kommentarzeichen /* und */ wurden die äußeren Flächen ausgeblendet und zusätzlich die der inneren Flächen angefügt. So erscheinen vorläufig nur die inneren Flächen mit den richtigen Farben.

Werden nun die Kommentarzeichen entfernt und die eckigen Klammern nebst Semikolon richtig gesetzt, erscheinen beide Formen ineinander gesetzt:

Nun liegt es daran, die äußeren Punkte mit den inneren zu verbinden, damit ein geschlossenes Element entstehen kann.

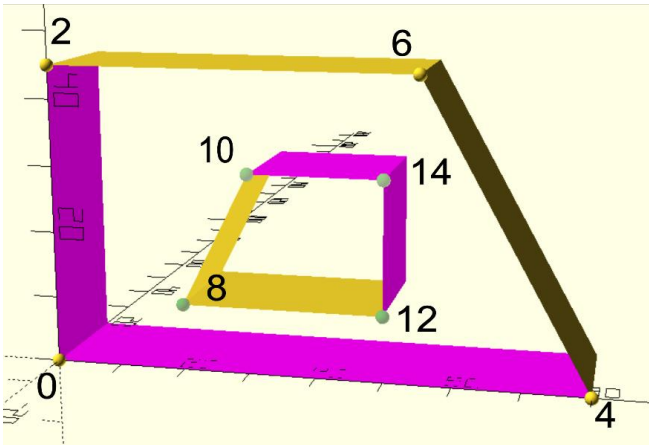


Zur besseren Übersicht werden nur die vorderen Punkte angezeigt.

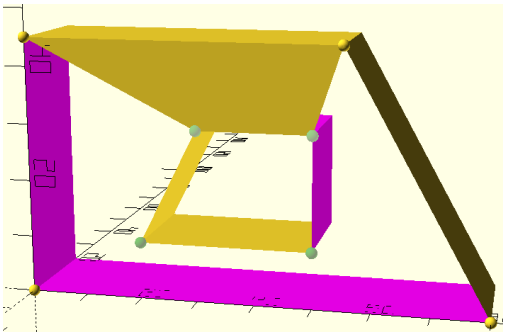
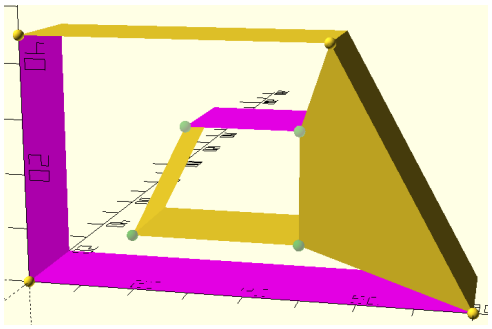
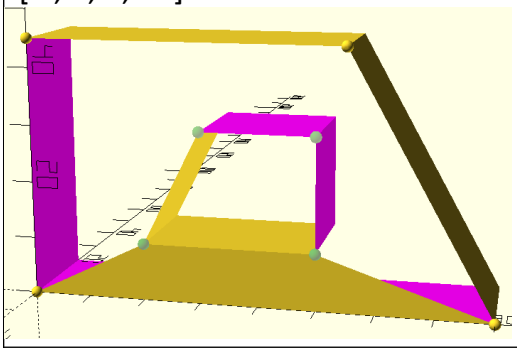
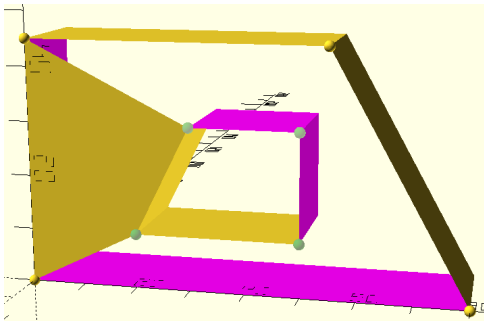
Die neuen Flächen werden jetzt zu trapezförmigen Darstellungen:

[2,6,14,10], [6,4,12,14],
[4,0,8,12] und [0,2,10,8]

Alle Flächen zeigen zum Betrachter hin nach außen in Gelb.

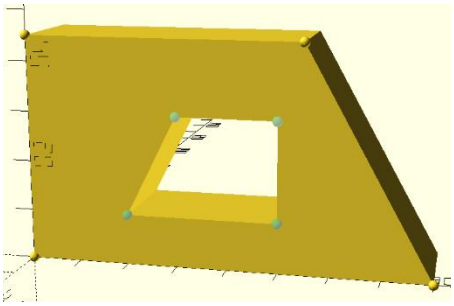


013-10

[2, 6, 14 ,10] 	[6, 4, 12, 14] 
[4, 0, 8, 12] 	[0, 2, 10, 8] 

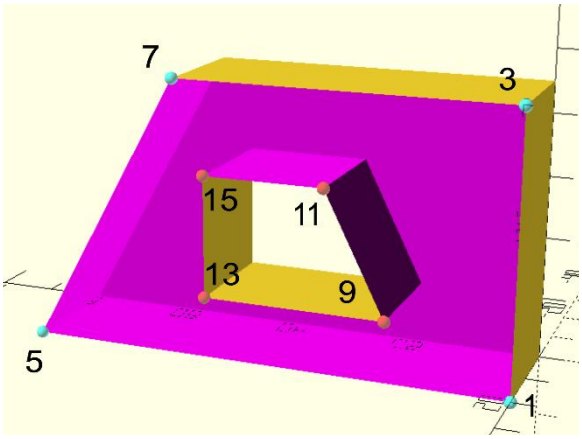
So werden die einzelnen Flächen nacheinander geschaffen. Nimmt man nun alle vier Flächen gemeinsam in das Listing auf, erscheint dafür folgendes Ergebnis:

So sieht nun die vordere Seite aus: alles in gelb und in richtiger Reihenfolge der Zahlangaben.



Nun wird das gesamte Objekt nach hinten gedreht. Auf der Rückseite ist noch alles offen. Die pinken Flächen zeugen von inneren Seiten, also alles bestens!

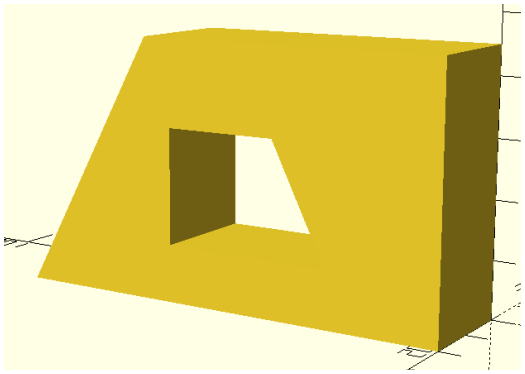
Die vorderen Punkte in gelb und grün wurden entfernt, dafür die roten und hellblauen für die hintere Seite eingefügt.

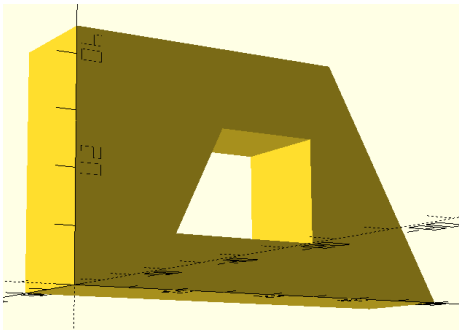


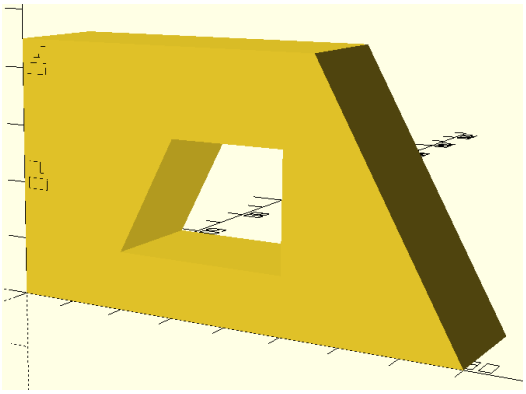
Auch hier werden wieder die einzelnen Flächen (diesmal mit den ungeraden Zahlen) erzeugt:

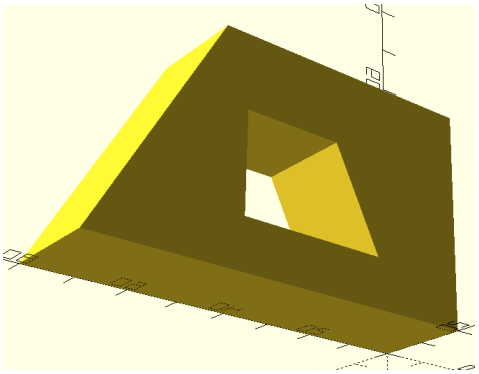
[5, 7, 15, 13], [7, 3, 11, 15],
[11, 3, 1, 9] und [13, 9, 1, 5].

Nach Fertigstellung und Entfernung der Punkte zur Orientierung:









013-12

Gesamtes Listing des Polyhedron 2

```
//  polyhedron II - Polyeder2

points = [
  [  0,  0,  0 ],    // 0
  [  0, 20,  0 ],   // 1
  [  0,  0, 45 ],   // 2
  [  0, 20, 45 ],  // 3

  [ 80,  0,  0 ],   // 4
  [ 80, 20,  0 ],  // 5
  [ 55,  0, 45 ],  // 6
  [ 55, 20, 45 ], // 7

  [ 20,  0, 10 ],   // 8
  [ 20, 20, 10 ],  // 9
  [ 30,  0, 30 ],  // 10
  [ 30, 20, 30 ], // 11

  [ 50,  0, 10 ] , // 12
  [ 50, 20, 10 ],  // 13
  [ 50,  0, 30 ],   // 14
  [ 50, 20, 30 ]];// 15


faces = [
  [ 0, 1, 3, 2], // linke Außenseite
  [ 1, 0, 4, 5], // untere Außenseite
  [ 4, 6, 7, 5], // rechte Außenschräge
  [ 2, 3, 7, 6], // obere Außenseite

  [ 8,10,11, 9],   // linke Innennseite
  [ 8, 9,13,12],   // untere Innenseite
  [15,14,12,13], // rechte Innenschräge
  [10,14,15,11], // obere Innenseite

  [ 2, 6,14,10],  // vorn oben
  [ 6, 4,12,14],  // vorn rechts
  [ 4, 0, 8,12],   // vorn unten
  [ 0, 2,10, 8],   // vorn links

  [ 5, 7,15,13],  // hinten oben
```

```
  [ 7, 3,11,15],  // hinten rechts
  [11, 3, 1, 9],  // hinten unten
  [13, 9, 1, 5]   // hinten links

  ];

polyhedron( points, faces );
```

Als weiteren Versuch die Zahlen ändern …..

```
points = [
  [  0,  0,  0 ], // 0
  [  0, 20,  0 ], // 1
  [  0,  0, 45 ], // 2
  [  0, 20, 45 ], // 3

  [ 80,  0,  0 ], // 4
  [ 80, 20,  0 ], // 5
  [ 55,  0, 45 ], // 6
  [ 55, 20, 45 ], // 7

  [ 10, -5, 10 ], // 8
  [ 20, 17, 10 ], // 9
  [ 30, -5, 30 ], // 10
  [ 10, 17, 30 ], // 11

  [ 60, -5, 10 ], // 12
  [ 40, 17, 10 ], // 13
  [ 50, -5, 30 ], // 14
  [ 50, 17, 30 ]];// 15
```