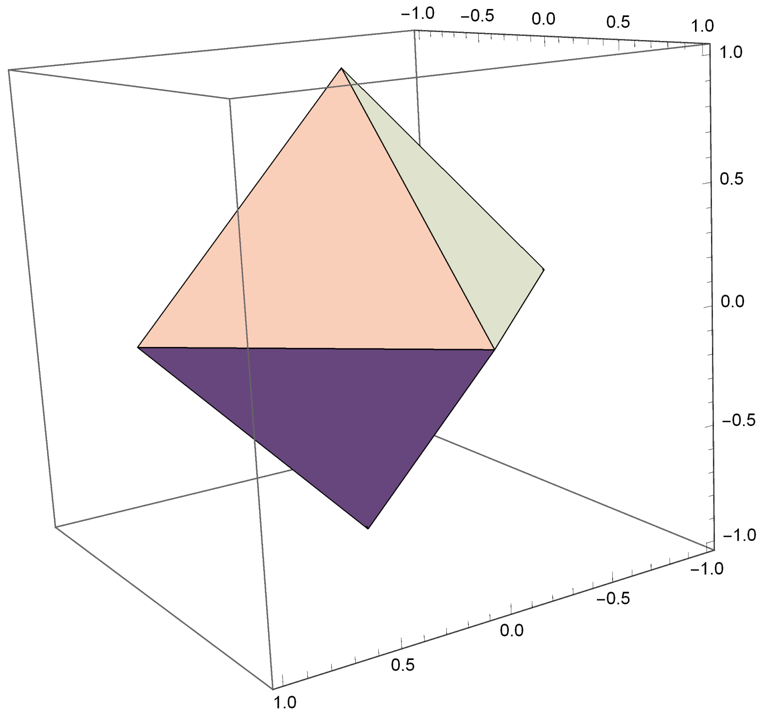
In[5]:=

```
Clear[X, Y, Z];
清除
X = {-1, 0, 1, 0, 0, 0};
Y = {0, -1, 0, 1, 0, 0};
Z = {0, 0, 0, 0, 1, -1};

faces =
  {{1, 2, 5}, {1, 4, 5}, {3, 4, 5}, {2, 3, 5}, {1, 2, 6}, {1, 4, 6}, {2, 3, 6}, {3, 4, 6}};
mesh = Table[{{X[[faces[[i, 1]]]], Y[[faces[[i, 1]]]], Z[[faces[[i, 1]]]]},
        表格
    {X[[faces[[i, 2]]]], Y[[faces[[i, 2]]]], Z[[faces[[i, 2]]]]},
    {X[[faces[[i, 3]]]], Y[[faces[[i, 3]]]], Z[[faces[[i, 3]]]]}}, {i, 8}];
Print[Graphics3D[Polygon[mesh], Axes -> True, BoxRatios → {1, 1, 1}]];
打印  三维图形    多边形            坐标轴  真    边界框比例
```



(*如上图所示，上下两个顶点编号分别是5,6，中间的四个顶点 编号从上往下顺时针来看是1,2,3,4.
程序先随机生成上面六个点的位置，然后由此计算出各边的边长。
取平均边长的根号2倍作为对角线(顶点5,6)的初始长度。之所以有根号2倍，
是因为在上面的例子中，每条边都是$\sqrt{2}$，而对角线的长度则为2.
然后将初始的对角线的长度x和随机生成的6个点points_sol输入dis.m中。
先利用points_sol计算边长length，然后为了方便使用将length存储为l。
length存储的是每个面对应的三角形的三条边，形为f*3，其中f是面的数量。
而l则对应的是任意两点间的距离，形为v*v，其中v为点的数目。
再接下来利用假设的对角线x的长度，固定点5,6在z轴上并关于原点对称，
故坐标分别为[0 0 x/2]和[0 0 -x/2]。接下来计算点1的位置，并将点1固定在x-z的平面上。
由于点1在x-z平面上，故y1=0。这里使用xn，yn，zn，分别表示点n的x，y，z坐标。
于是只剩下x1和z1两个未知数。利用点1到点5，点1到点6这两条边的长度得到两个方程即可算得x1，z1.
接着计算点2的坐标，由于已知点2到点1，点2到点5，点2到点6三条边的边长，
故可得三个方程，因此可以算出x2，y2，z2.同理可得点2对面的点4的位置。
那么只剩下最后一个点3的位置未求。这里我们即可利用点5,6,2算得点3，记为(x3,y3,z3)，
也可利用点5,6,4算得点3,记为(X3,Y3,Z3).通过这两种方式算得的点3往往是不同的。
若它们相同则代表x是合适的对角线长度，即这样得到的所有点组成的图形的边的长度与原来一样。
因为定义error=abs(x3-X3)^2+abs(y3-Y3)^2+abs(z3-Z3)^2.
最后利用系统自带的fminunc将error变小。
当error<10^-5且所得点3的坐标为实数时认为成功得到合适的对角线长度，
即right要加1.正确率定义为right/总次数*100%。*)