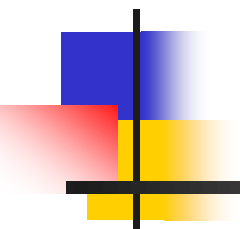
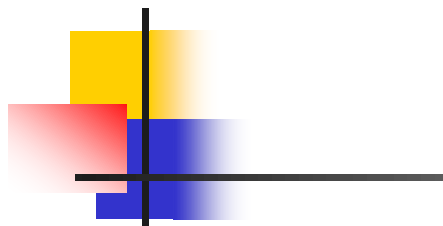


6.1 三维图形的几何造型技术

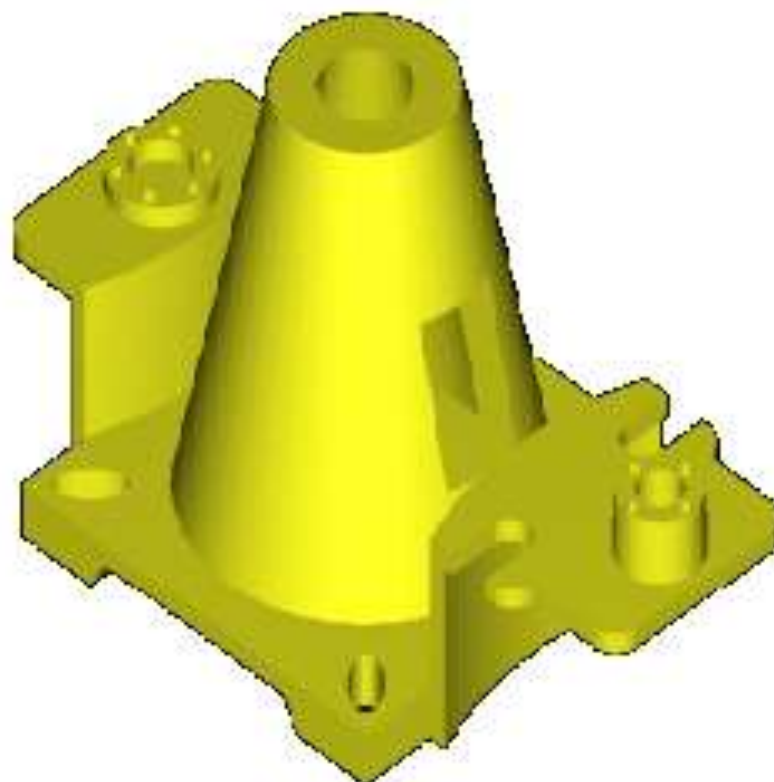


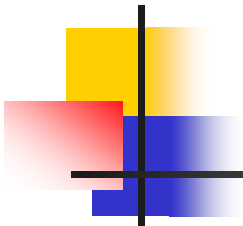
南京农业大学

谢忠红



Solid





引言

■ 三维图形最重要的三个课题。

■ (1) 如何描述一个三维形体？

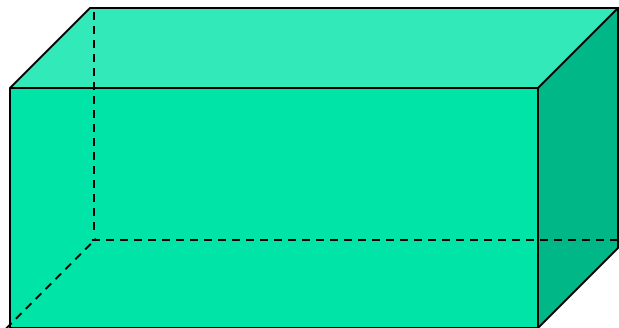
——几何造型技术

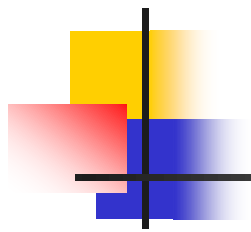
■ (2) 如何将三维形体在二维的图像显示器等输出装置上表示出来？

——三维投影技术

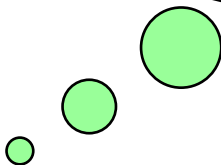
■ (3) 为了增强三维图形在二维显示器上显示的立体感，还有必要对其进行消隐计算。

——消隐技术

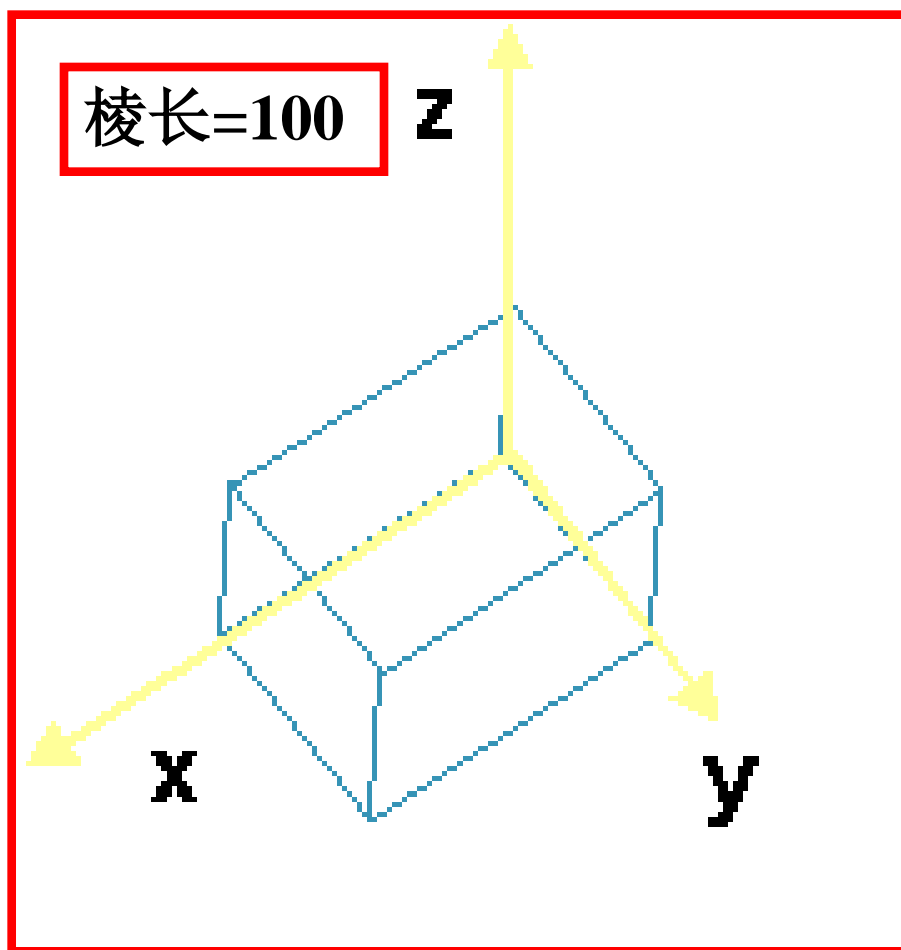




如何定义三维
立体形体？

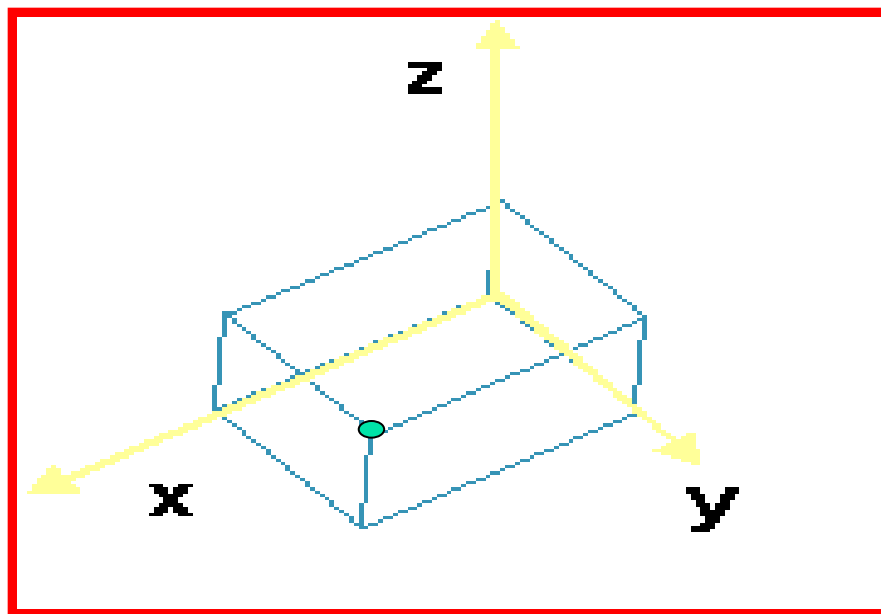


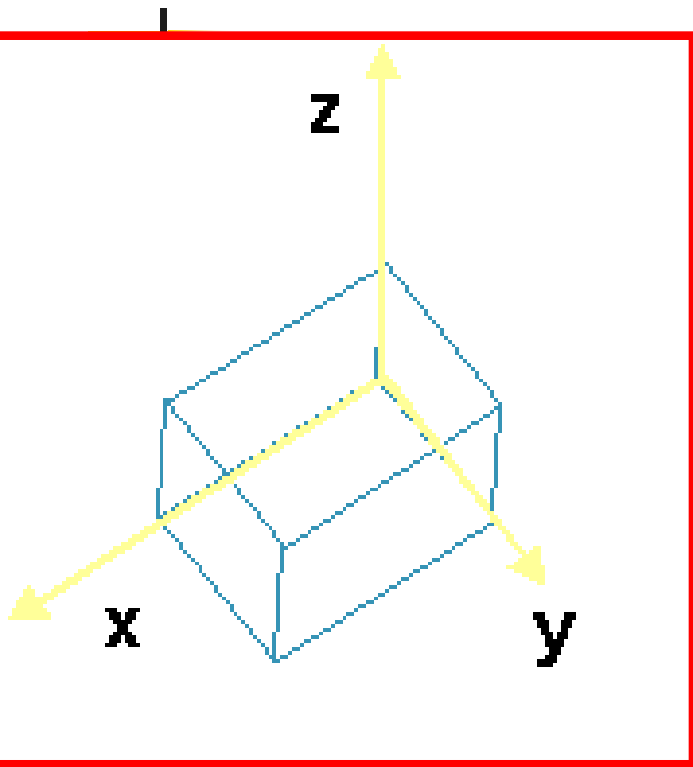
- **声明：** 下面所有有关三维形体的内容本课将均采用下图所示的立方体为例进行讲解，其他图形原理类似。



■ 世界坐标系(右手系)

- 现实的物体具有高度、宽度和深度。三维坐标系是一个直角坐标系；坐标系内任何一点可以由一个有序的三元组(x, y, z)来表示





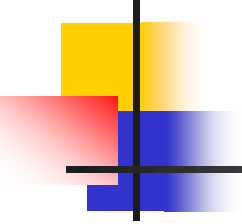
图形数据结构

拓扑信息

几何信息

拓扑信息： 立体的面、边、点之间的关系

几何信息： 点的坐标、边的大小尺寸等信息



如何既能够表示出该立方体的
拓扑信息又能几何信息呢？



——称为三维形体的造型技术

几何形体



三种数据模型

线框模型

表面模型

实体模型

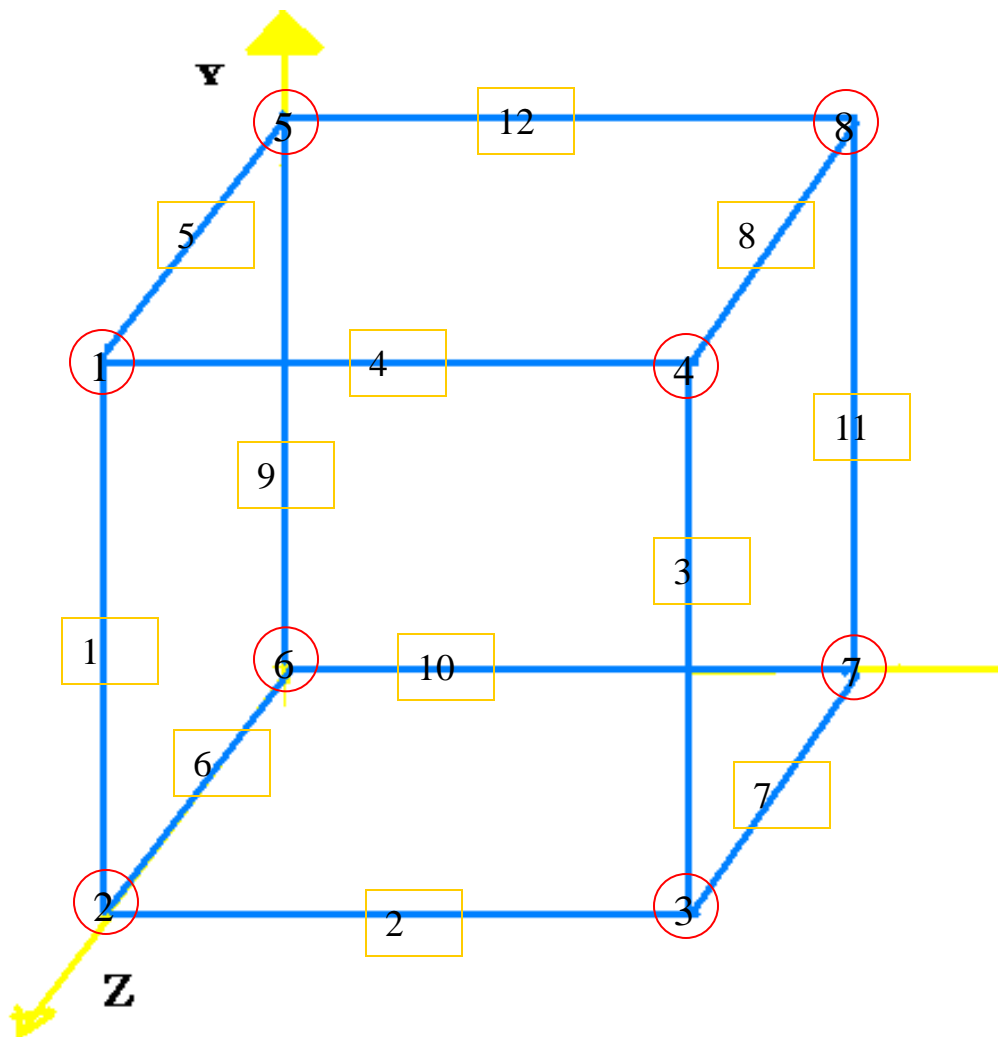
6.1.1 传统几何造型技术

1 表结构表示立体图形

DEF:使用数据表来记录图形元素的数据结构。

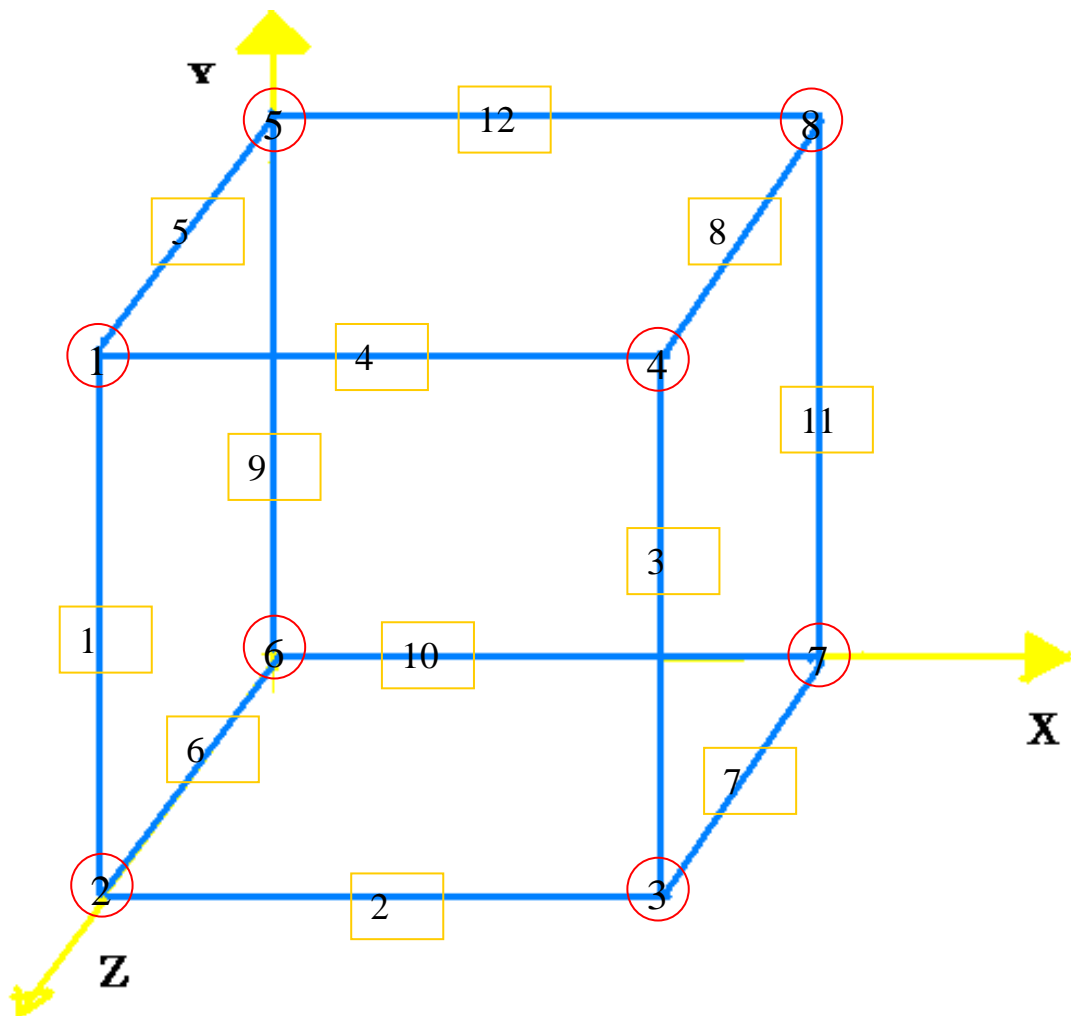
- 表示形体的基本几何元素：
 - 顶点 (**Vertex**) 边 (**Edge**)
 - 面 (**Face**) 环 (**Loop**)
 - 体 (**Body**)

顶点表

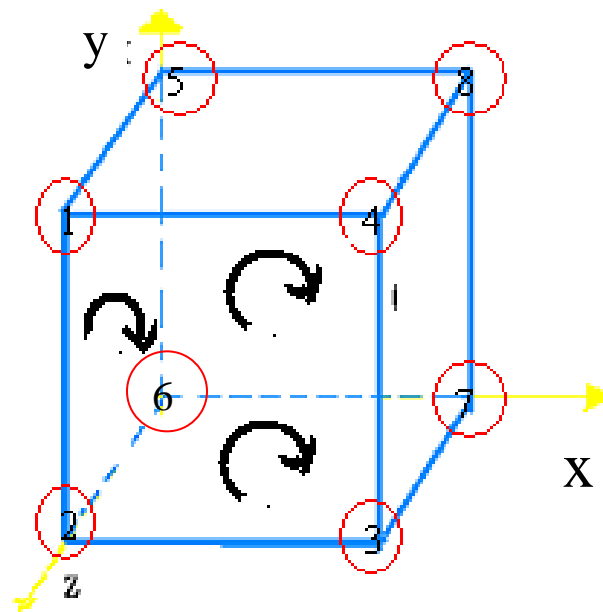
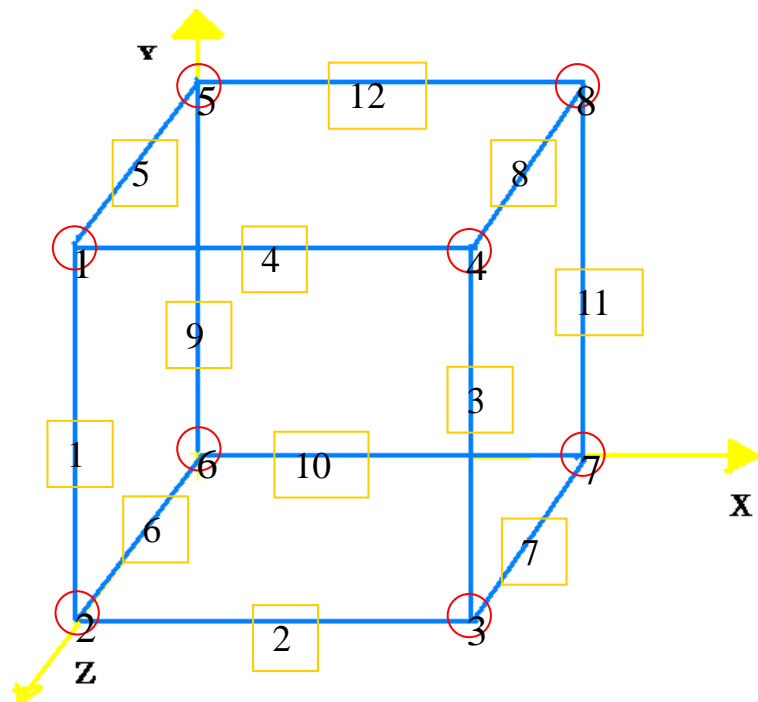
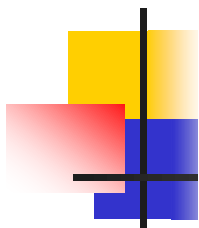


顶点号	坐标值		
	x	y	z
1	0	100	100
2	0	0	100
3	100	0	100
4	100	100	100
5	0	100	0
6	0	0	0
7	100	0	0
8	100	100	0

棱线表(边表)



棱线号	顶点	
1	1	2
2	2	3
3	3	4
4	1	4
5	1	5
6	2	6
7	3	7
8	4	8
9	5	6
10	6	7
11	7	8
12	5	8



面号	边号 (右手顺序)			
1	1	2	3	4
2	3	7	11	8
3	5	4	8	12
4	1	5	9	6
5	2	6	10	7
6	9	12	11	10

面表

面号	顶点号 (右手顺序)			
1	1	2	3	4
2	4	3	7	8
3	5	1	4	8
4	2	1	5	6
5	2	6	7	3
6	6	5	8	7

- 
- 如何使用C语言
 - 描述三表结构呢？



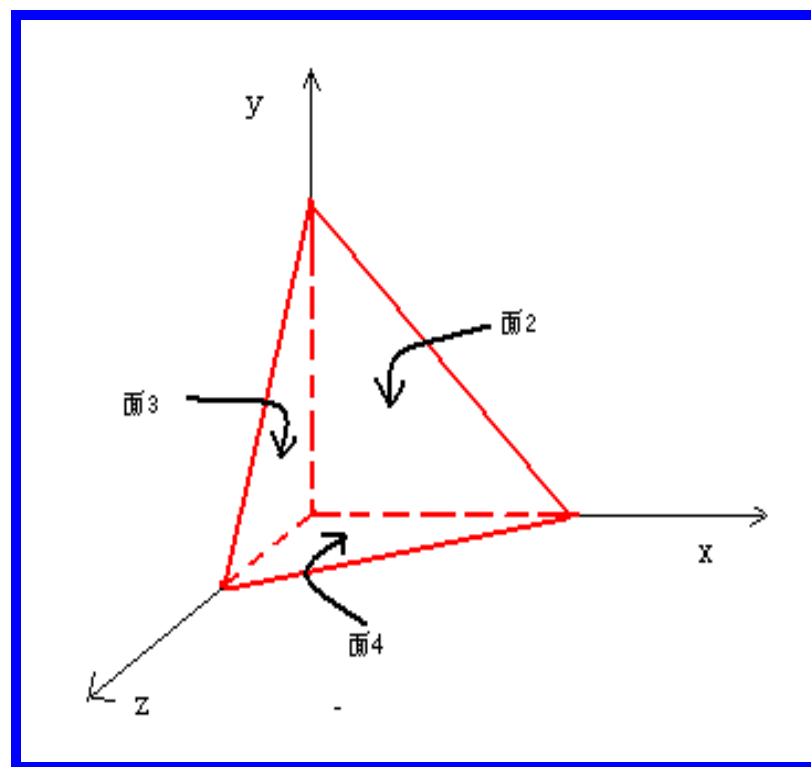
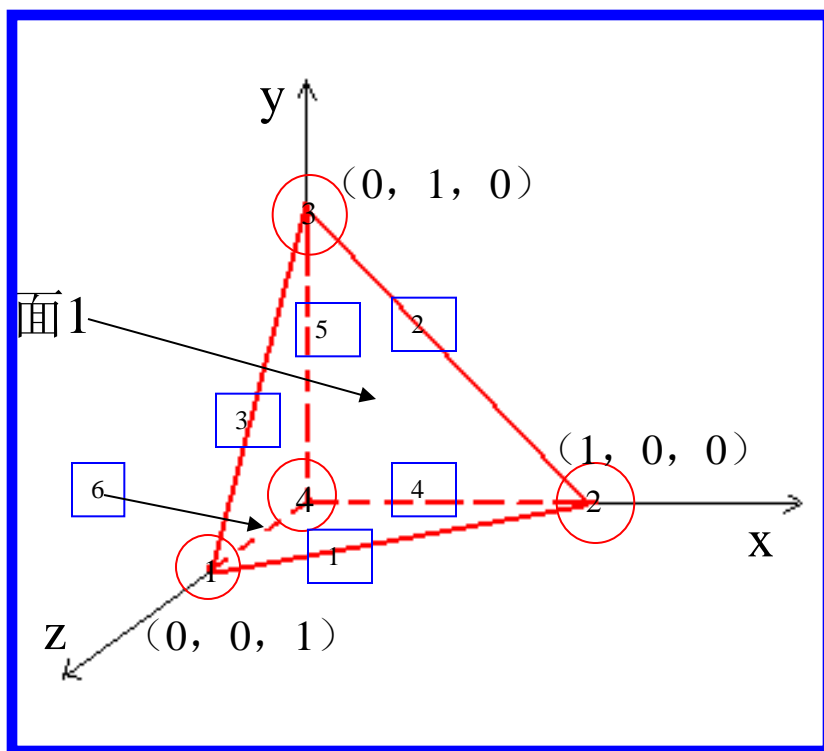
•C语言描述三表结构

- **typedef**
- **struct point**
- **{ int x;**
- **int y;**
- **int z;**
- **}Cpoint;**
- **typedef**
- **struct edge**
- **{int st_point;**
- **int end_point;**
- **} Cedge;**

- typedef**
- struct face**
- **{int fst_pt;**
- **int snd_pt;**
- **int thd_pt;**
- **int forth_pt;**
- **}Cface;**
- main()**
- **{**
- **Cpoint tab_point[9];**
- **Cedge tab_edge[13];**
- **Cface tab_face[7];**
- **}**

■ 例题:

- 将下图所示得四面体的立体模型当作形状信息，用表结构这种数据结构来表示该四面体。



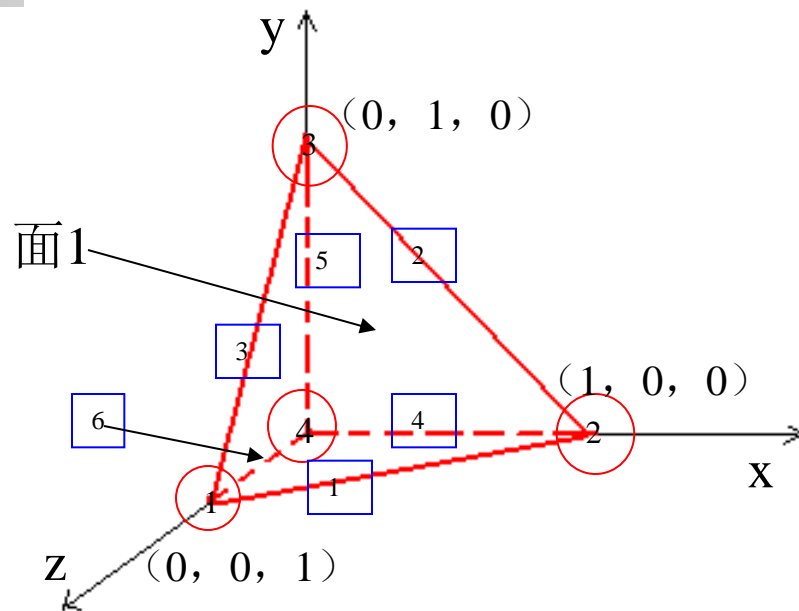
■ 解答:

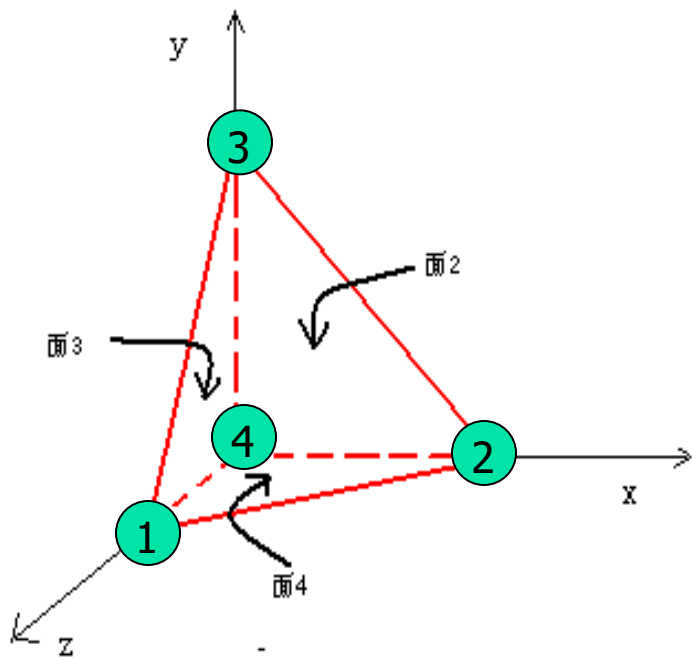
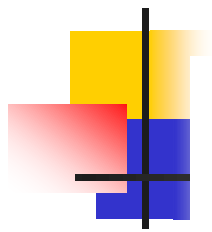
■ 顶点表

棱线表

顶点号	坐标值		
	x	y	z
1	0	0	1
2	1	0	0
3	0	1	0
4	0	0	0

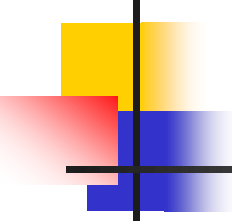
棱线号	顶 点	
1	1	2
2	2	3
3	1	3
4	2	4
5	3	4
6	1	4





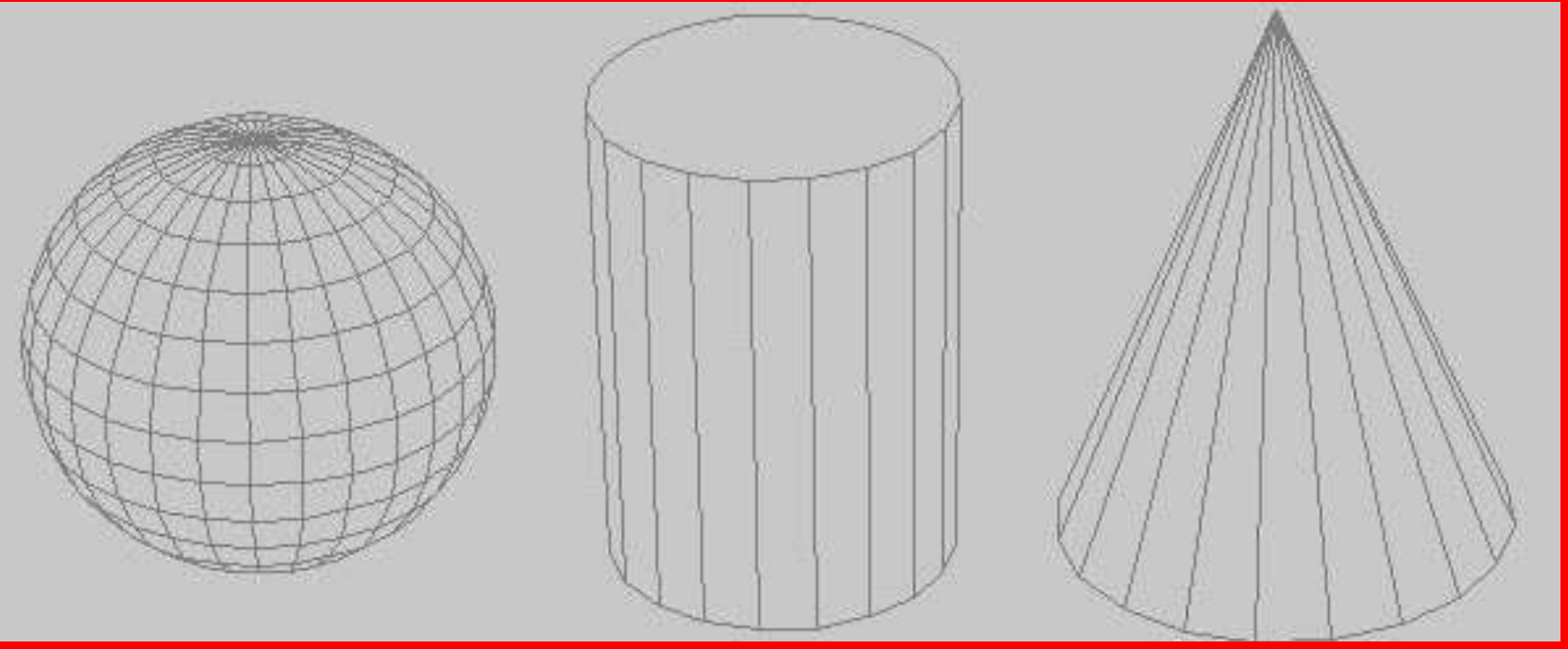
■面表

面号	顶点号 (右手顺序)		
	1	2	3
1	1	2	3
2	3	2	4
3	3	4	1
4	1	4	2

- 
- 所有形体都可以使用
 - 表结构来描述吗？



如果我们能够将三维形体表示
为一系列平面多边形的集合

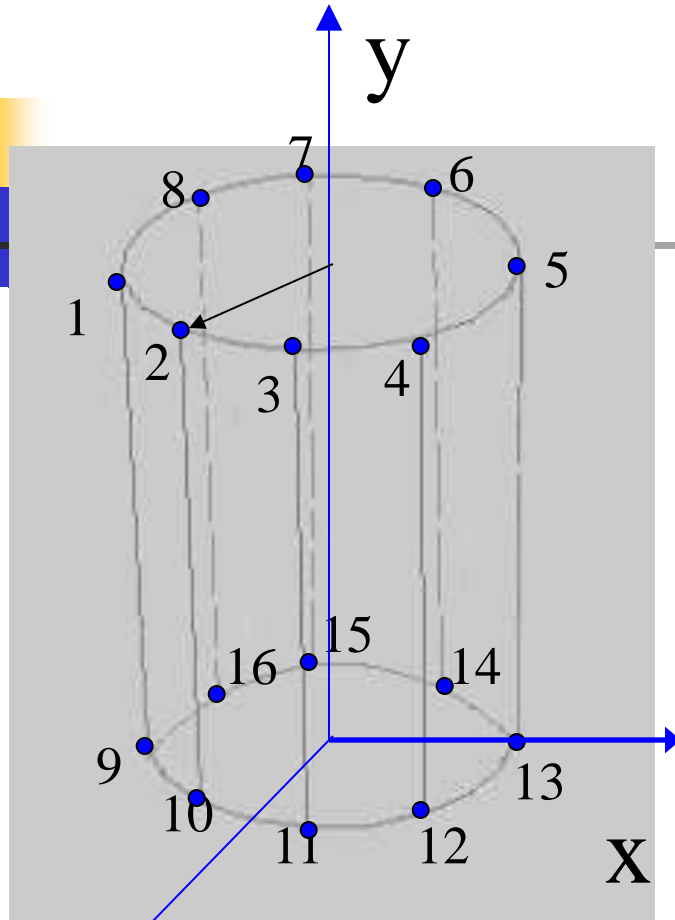


形体的多边形表示

优点： 适用性广，实现简单，可以表达复杂的形体；

缺点： 只能逼近原形体，不适合于精确计算，

顶点表

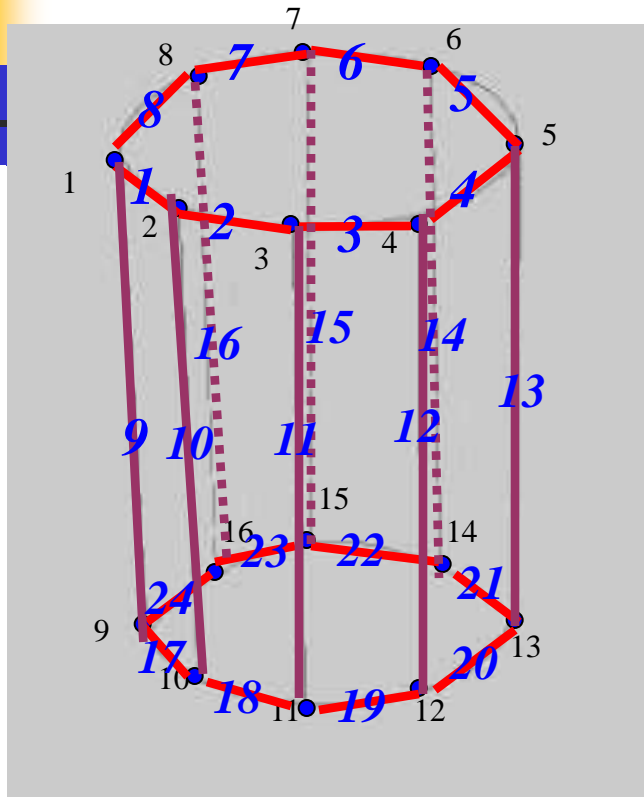


顶点号	坐标值		
	x	y	z
1			
2			
3			
.....
15			
16			

Z

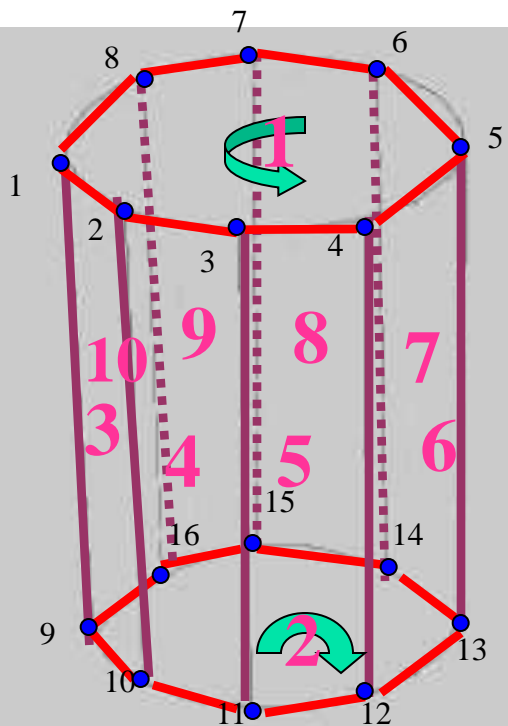
半径: R 高: H

棱线表



棱线号	顶点	
1	1	2
2	2	3
3	3	4
4	4	5
.....
21	13	14
22	14	15
23	15	16
24	16	9

面表

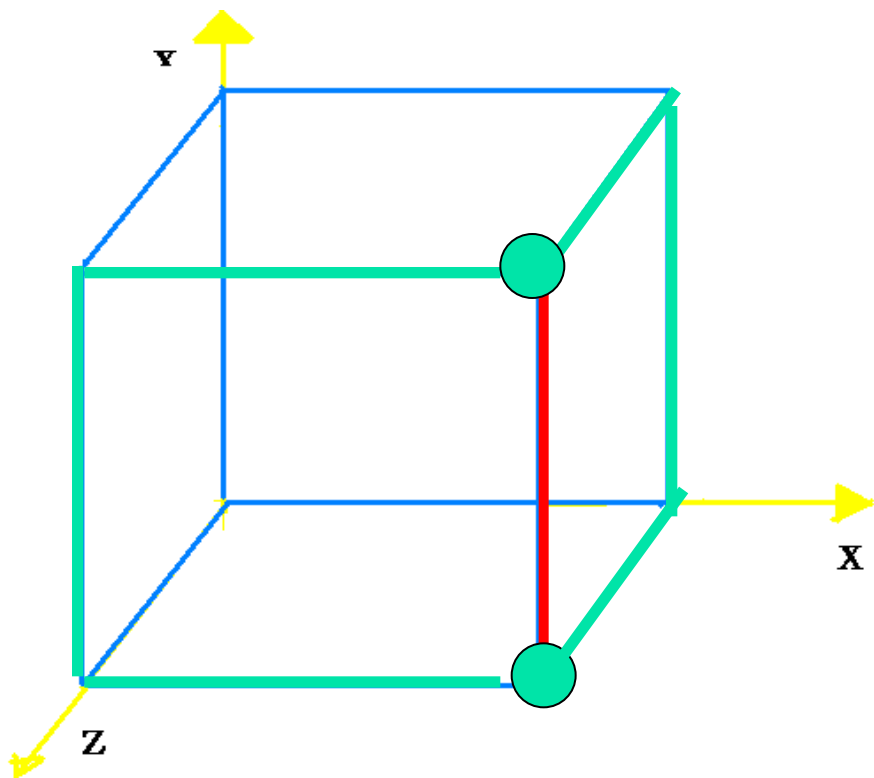


面号	顶点号			
1	1	2	8
2	16	15	9
3	2	1	9	10
4	3	2	10	11
5	4	3	11	12
6	5	4	12	13
7	6	5	13	14
8	7	6	14	15
9	8	7	15	16
10	1	8	16	9

2 翼边结构表示立体形体



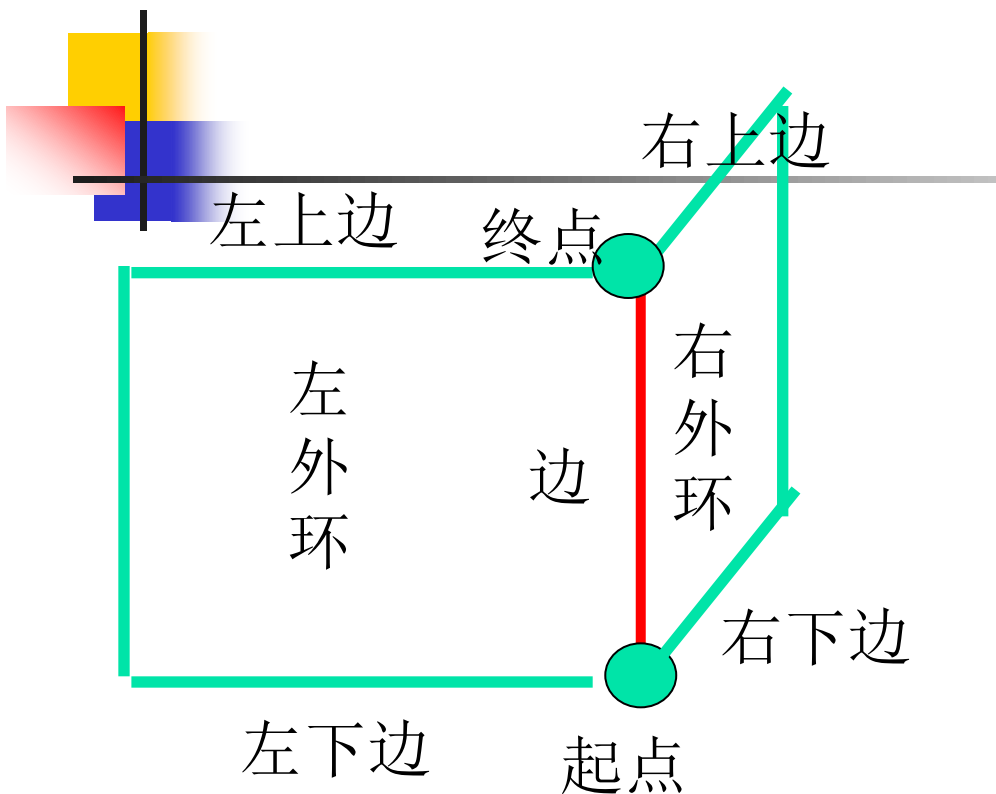
美国斯坦福大学**G.Baumgart**提出了一种翼边结构表示三维形体



以边为核心

- (1) 连接边的两个顶点
- (2) 连接边的两个面
- (3) 从形体外侧观看边、点、面还有一些拓扑信息

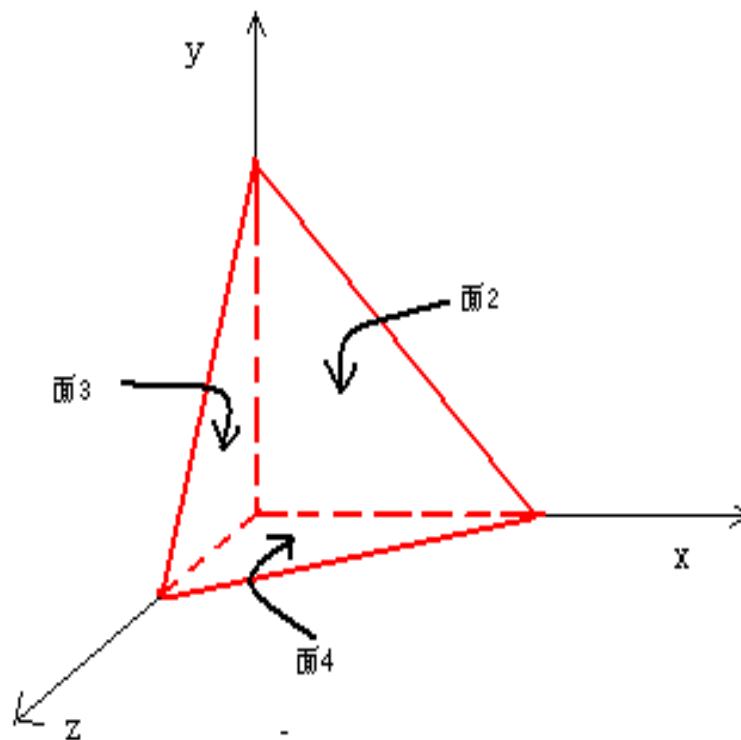
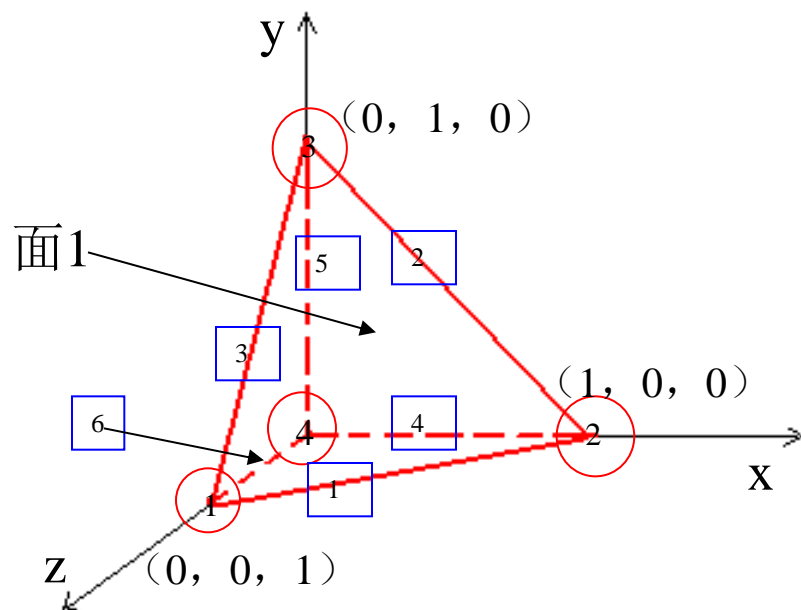
翼边结构



从体外由下-上或左-右

任意一边都是可以构成一个翼边结构

举例：用翼边结构表示下列几何形体



■ 解答:

■ 顶点表

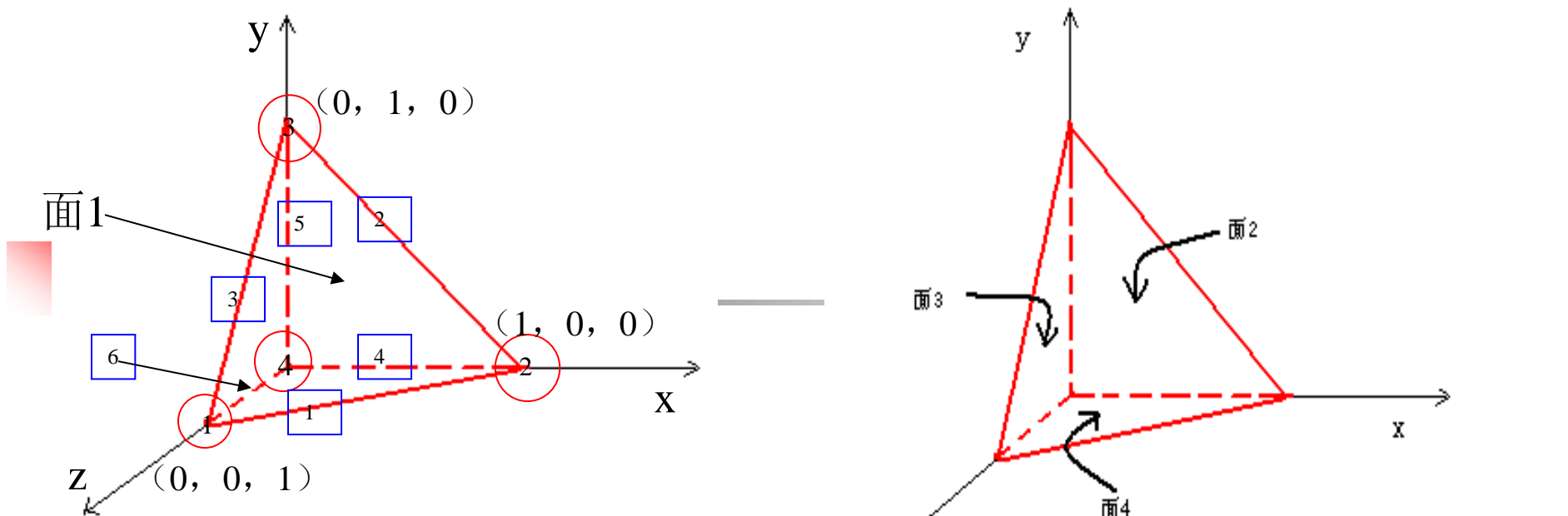
顶点号	坐标值		
	x	y	z
V1	0	0	1
V2	1	0	0
V3	0	1	0
V4	0	0	0

棱线表

棱线号	顶 点	
E1	1	2
E2	2	3
E3	1	3
E4	2	4
E5	3	4
E6	1	4

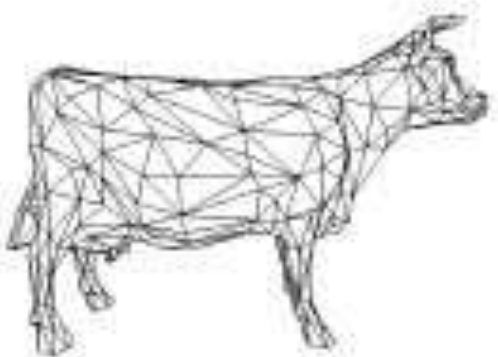
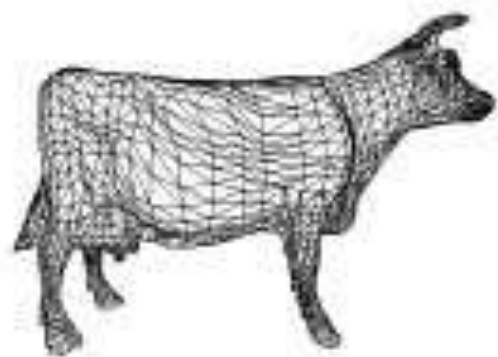
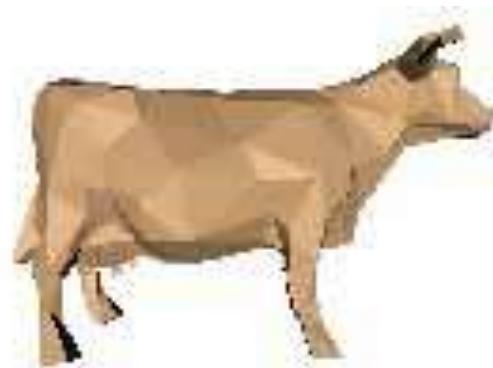
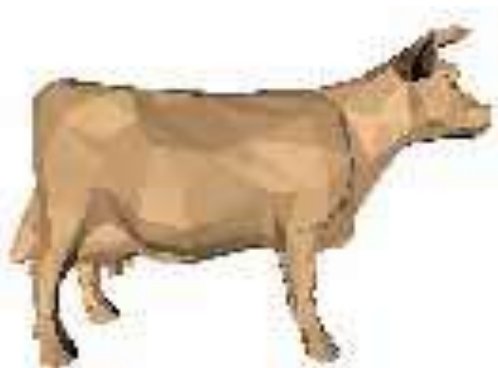
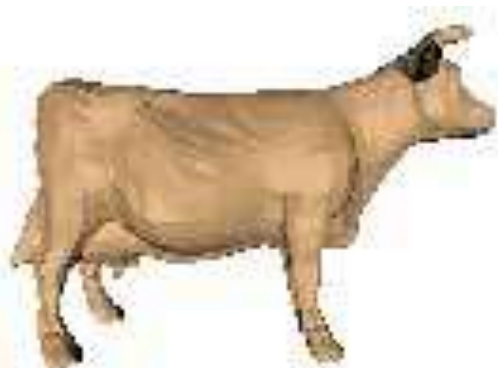
面表

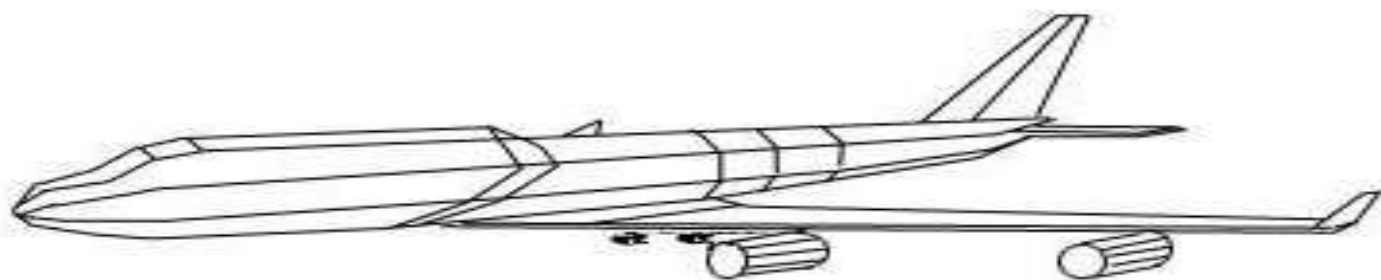
面号	顶点号 (右手顺序)		
F1	1	2	3
F2	3	2	4
F3	3	4	1
F4	1	4	2



边	起点	终点	右外环	右上边	右下边	左外环	左上边	左下边
E1	v1	v2	f4	e4	e6	f1	e2	e3
E2	v2	v3	f2	e5	e4	f1	e3	e1
E3	v1	v3	f1	e2	e1	f3	e5	e6
E4	v2	v4	f1	e6	e1	f2	e5	e2
E5	v4	v3	f3	e3	e6	f2	e2	e4
E6	v1	v4	f3	e5	e3	f4	e4	e1

更加 复杂的图形怎么表示呢？



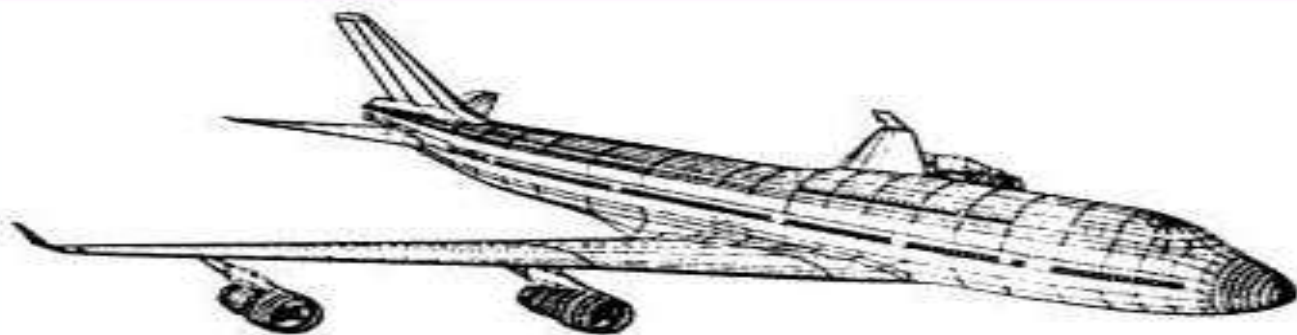


747-400-S

520
406

Vertices
Polygons

顶点
多边形



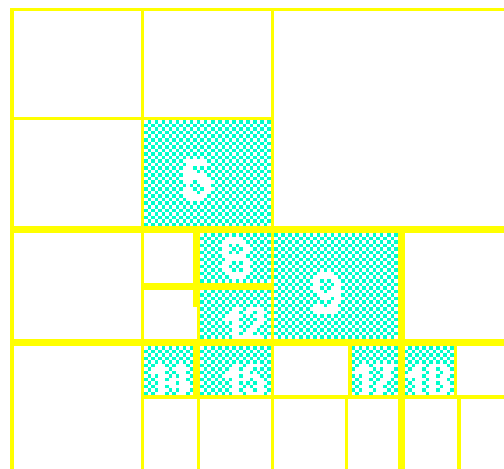
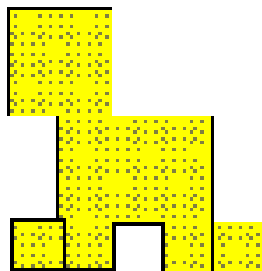
3D动力

3 八叉树表示三维形体

四叉树的表示用于二维形体的分解

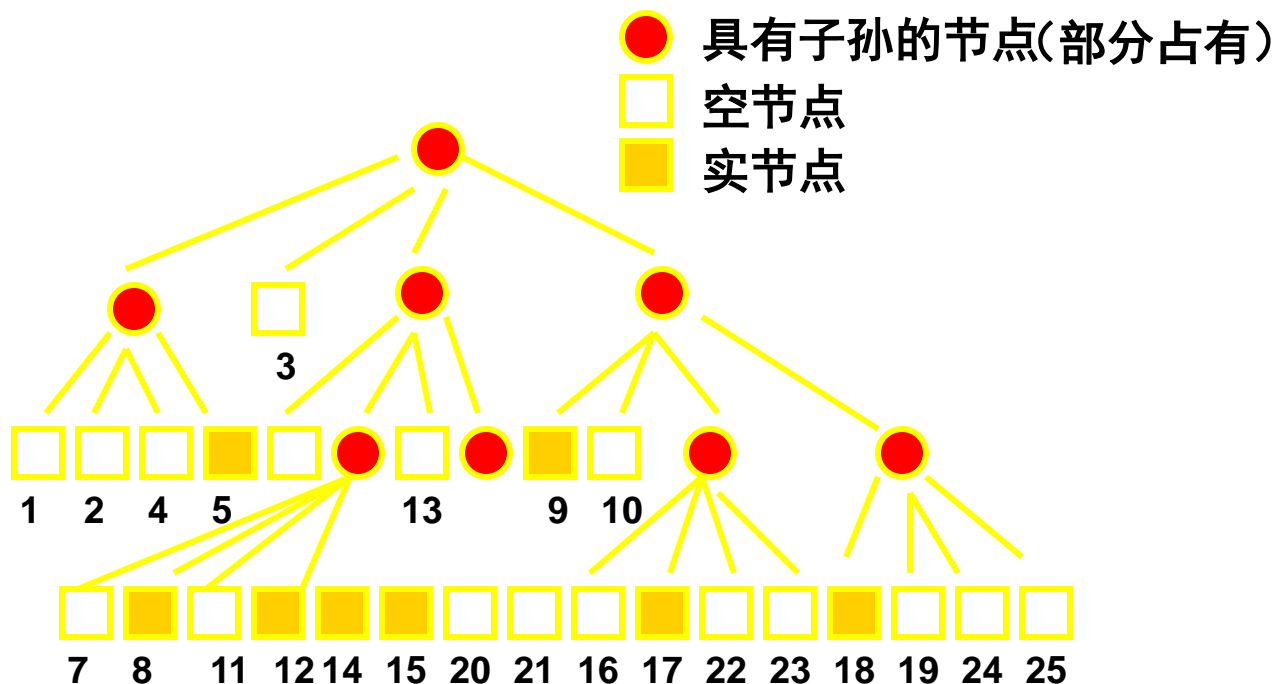
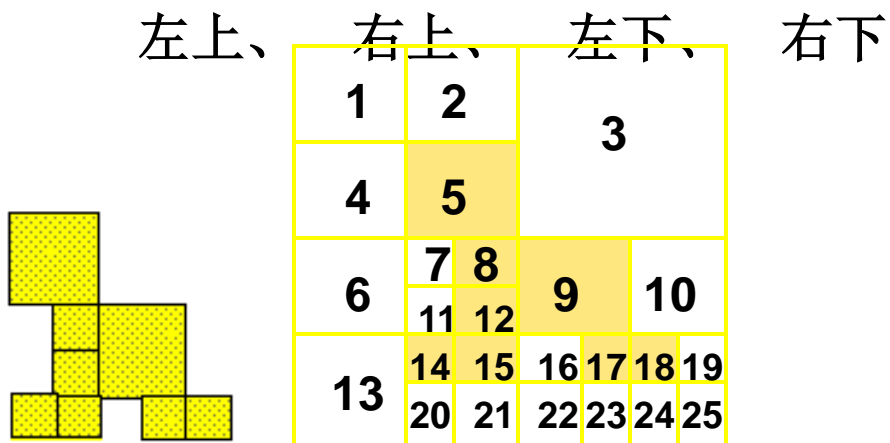
步骤:(1)将形体所在的外接正方形递归地等分成4个小正方形,

- 若小正方形是空则不再分(形体不在其中)、
- 若小正方形满(形体完全充满其中), 则不再分解;
- 若一个小正方形被形体部分的占有, 则需将它再一分为四, 这个分解是递归的。

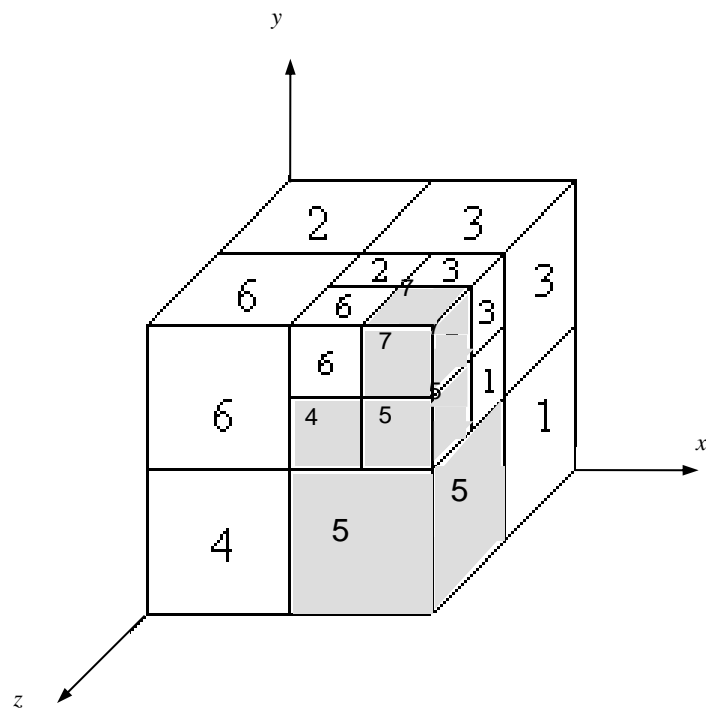
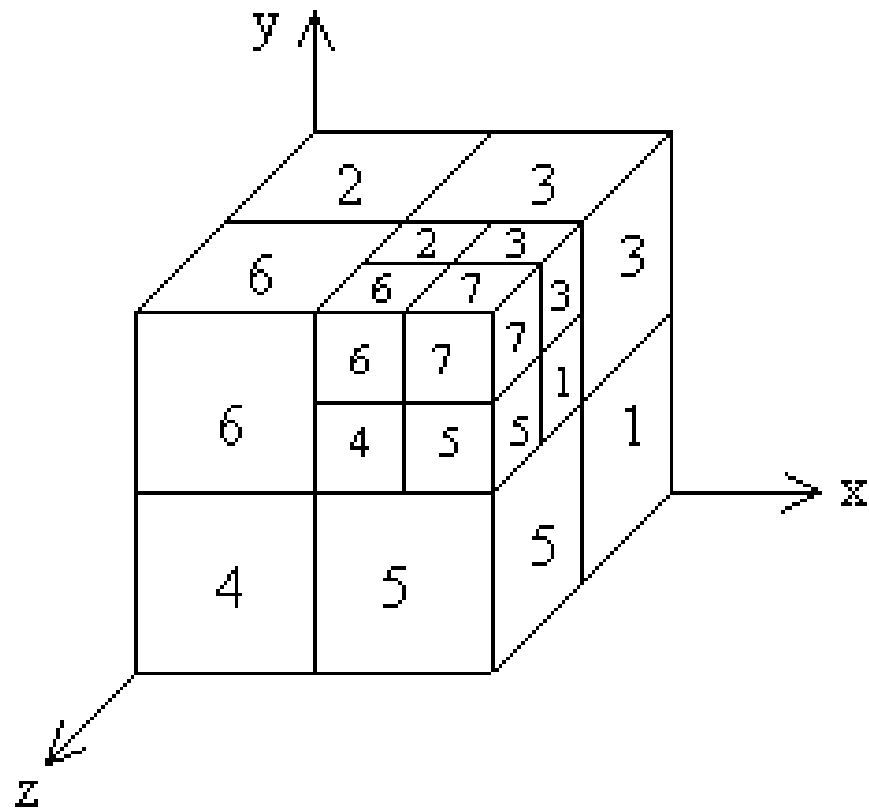


树

二维形体的四叉树表示



八叉树的表示用于二维形体



■ 八叉树建立步骤:

首先 确定该形体的外接立方体

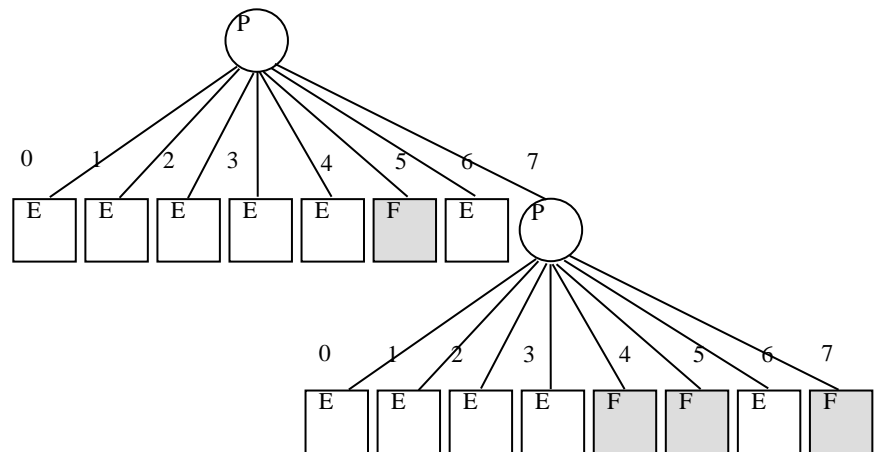
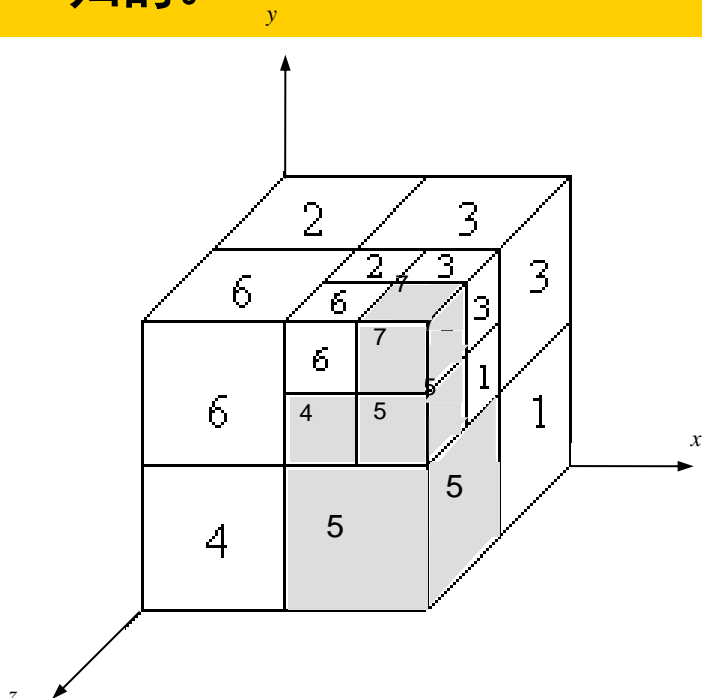
然后分别沿长、宽、高三个方向将该立方体二等分

判定分割出来的小立方体

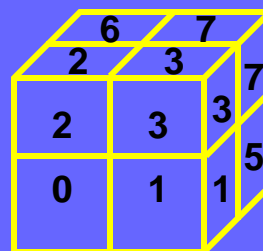
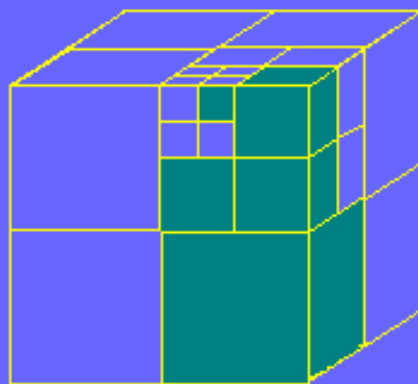
若小正方体是空 (E) 则不再分(形体不在其中)、

若小正方体满(F)(形体完全充满其中), 则不分解;

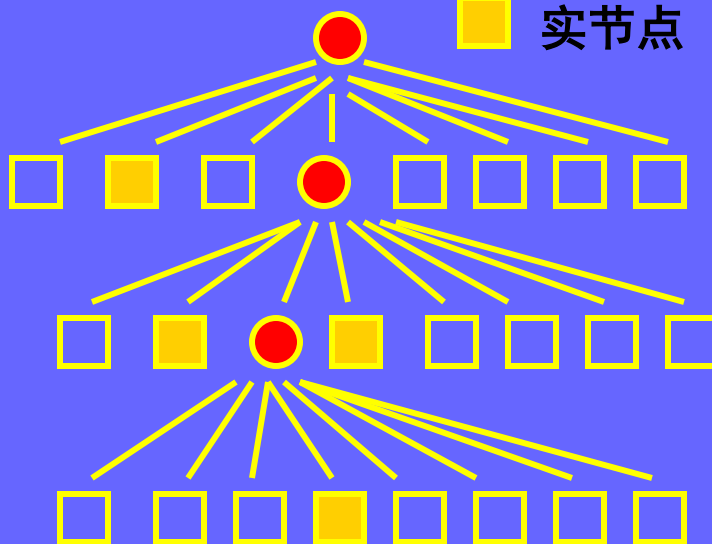
若一个小正方体被形体部分的占有(P), 则需将它再一分为八, 这个分解是递归的。



三维形体的八叉树表示



- 具有子孙的节点
- 空节点
- 实节点



- 
- 如何使用C语言
 - 描述这个八叉树结构呢？



struct **TreeNode**{

- char type; // (‘E’空、 ‘F’满、 ‘P’部分占有)
- ElementType data;
- struct **TreeNode** *furc1, *furc2, *furc3,
- *furc4, *furc5, *furc6, *furc7, *furc8;
- }
- **TreeNode** *eighthFurcTree ;
- CreateEightFurcTree(**TreeNode** &T){
- }



4 CSG (Constructive solid geometry) 表示方法

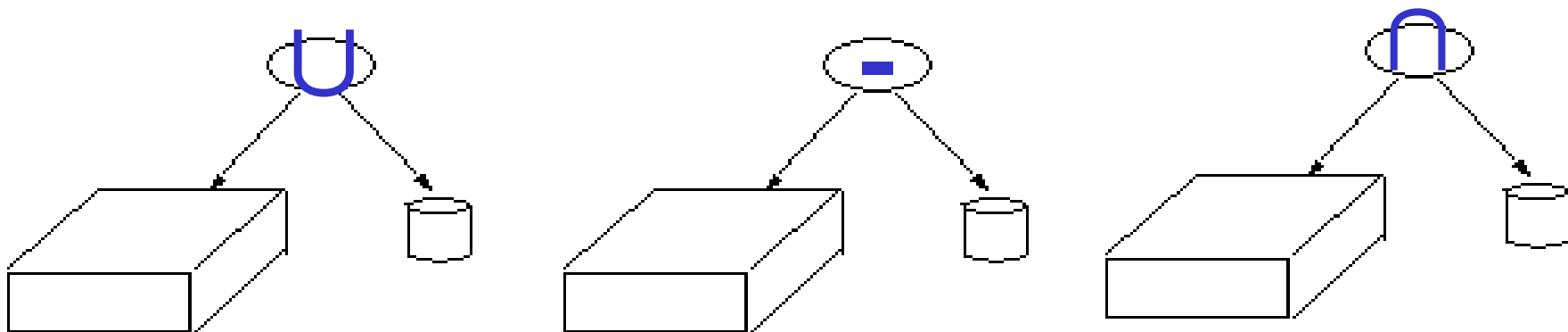
- **定义：**使用集合运算符和体素表示几何形体的方法称为**CSG**表示法。
- **(1) 什么是体素呢？**
- 一般是指比较简单的立体形体：例正方体、圆柱体、球体、多棱柱等

(2) 常见的集合运算

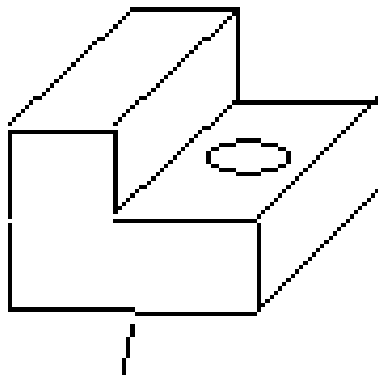
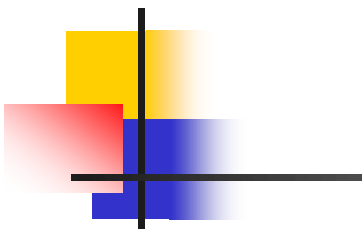
和运算 \cup : 两个形体求和

差运算 $-$: 两个形体求差

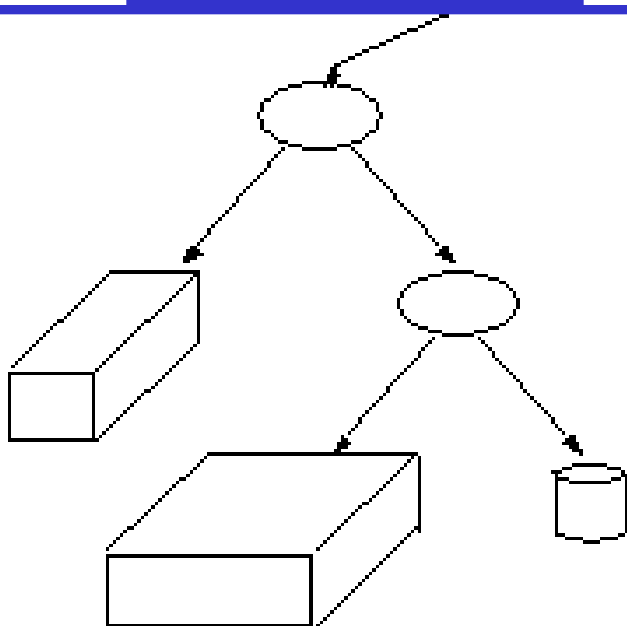
交运算 \cap : 两个形体求相交部分



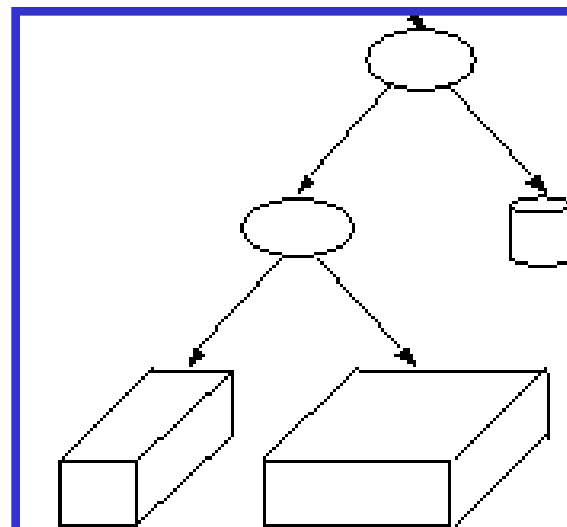
如何使用**CSG**方法表示下列形体呢？



方法**1**



方法**2**





■ CSG方法

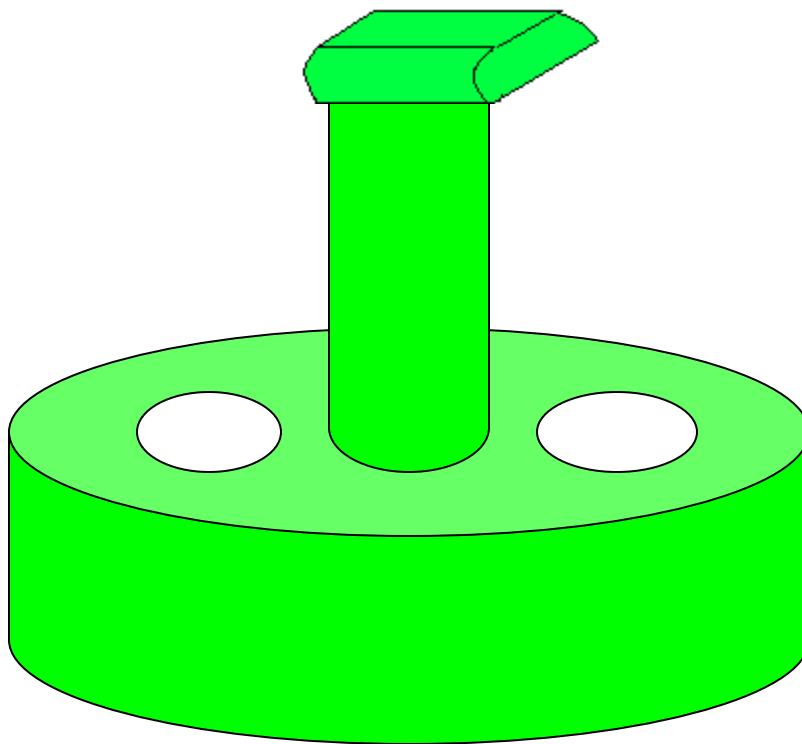
■ 优点

- 表示简单、直观，无二义性
- 数据量比较小，内部数据的管理比较容易
- 形体形状容易被修改

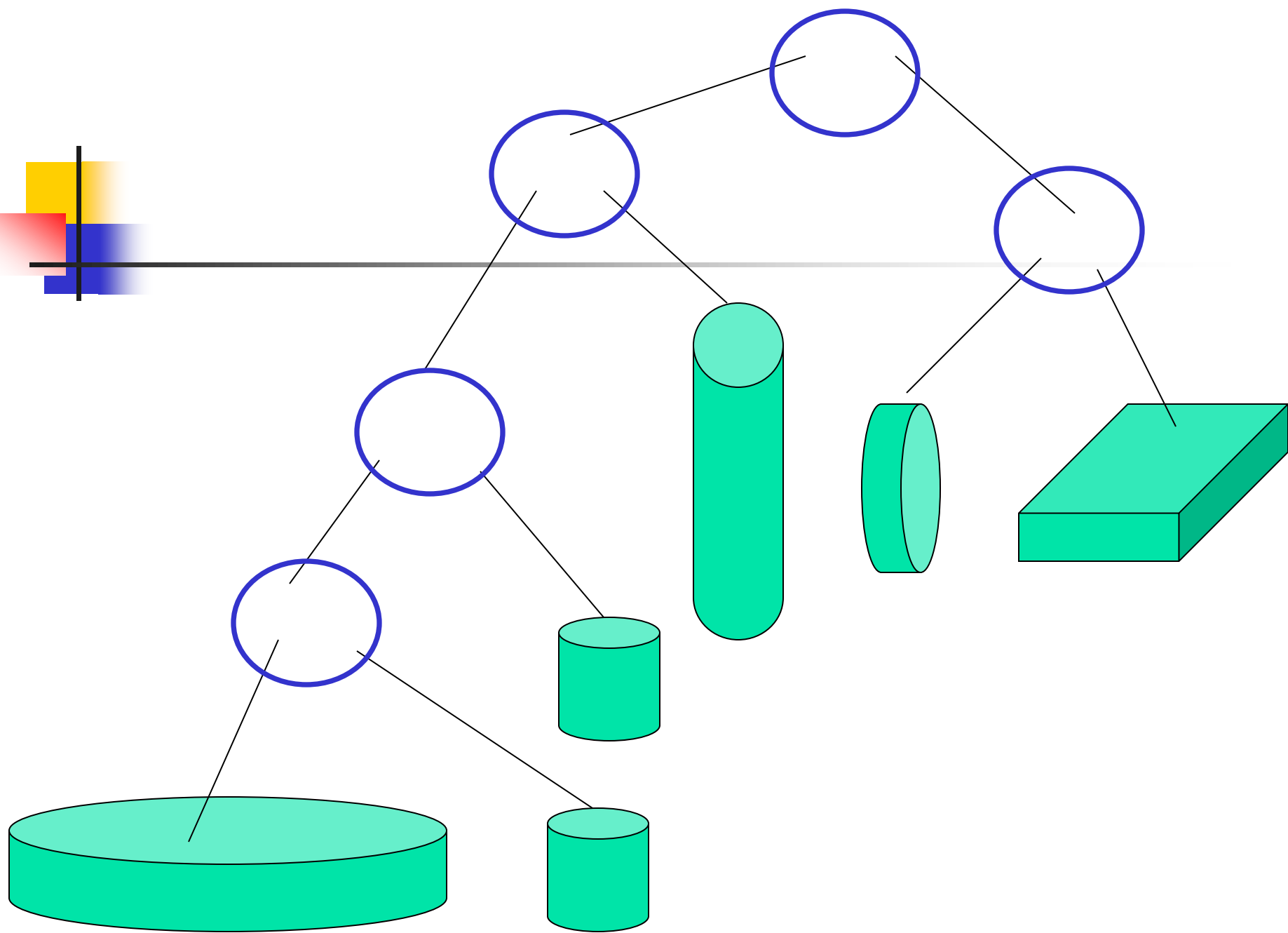
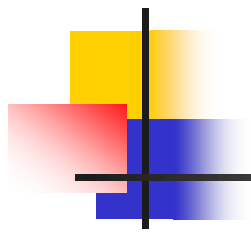
■ 缺点：

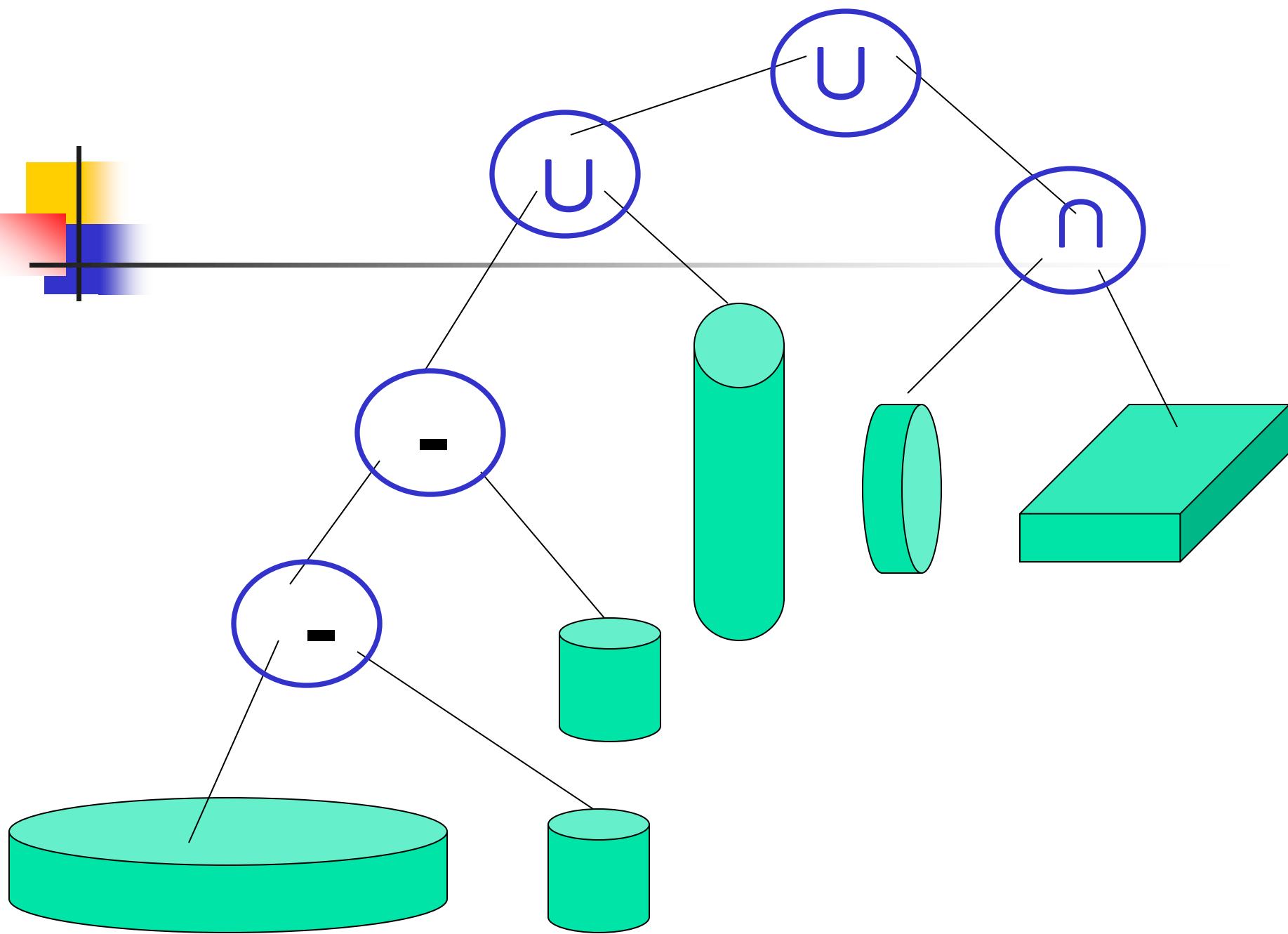
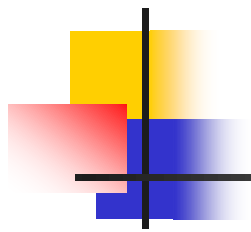
- 表示物体的**CSG**树不唯一
- 受体素种类和对体素操作种类的限制，**CSG**方法表示形体的覆盖域有较大的局限性。

课堂练习题



使用**CSG**方法表示该形体





■ 5表示三维形体的其他方法

■ ——扫描表示

■ **定义：** 一个基体（一般为封闭的二维区域）沿某一路径运动而产生形体。

■ 两个元素：

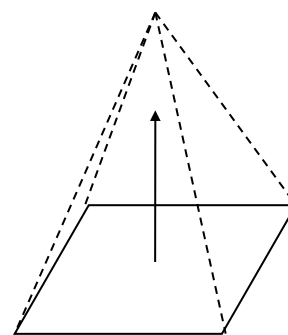
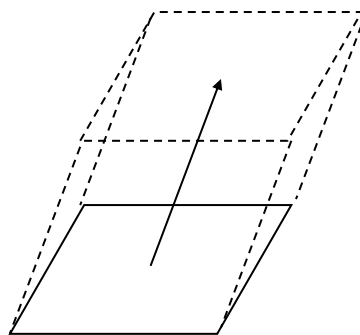
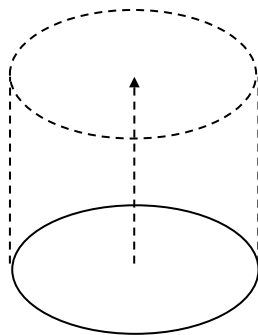
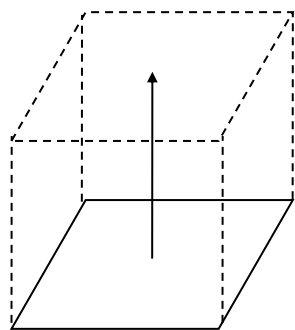
- （1）被运动的基体
- （2）基体运动的路径

- 
- 根据扫描路径和方式的不同，可以分成下列三种扫描的形体：

- **(1) 平移扫描体**
 - **(2) 旋转扫描体**
 - **(3) 广义扫描体**

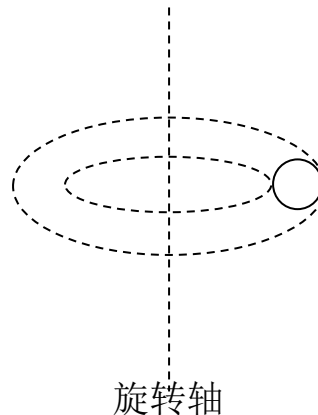
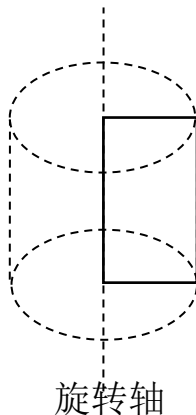
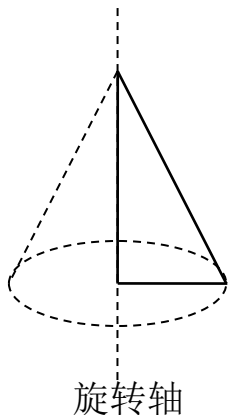
■ 平移扫描

定义： 将一个二维区域沿着一个矢量方向（线性路径）推移，拉伸曲面



旋转扫描

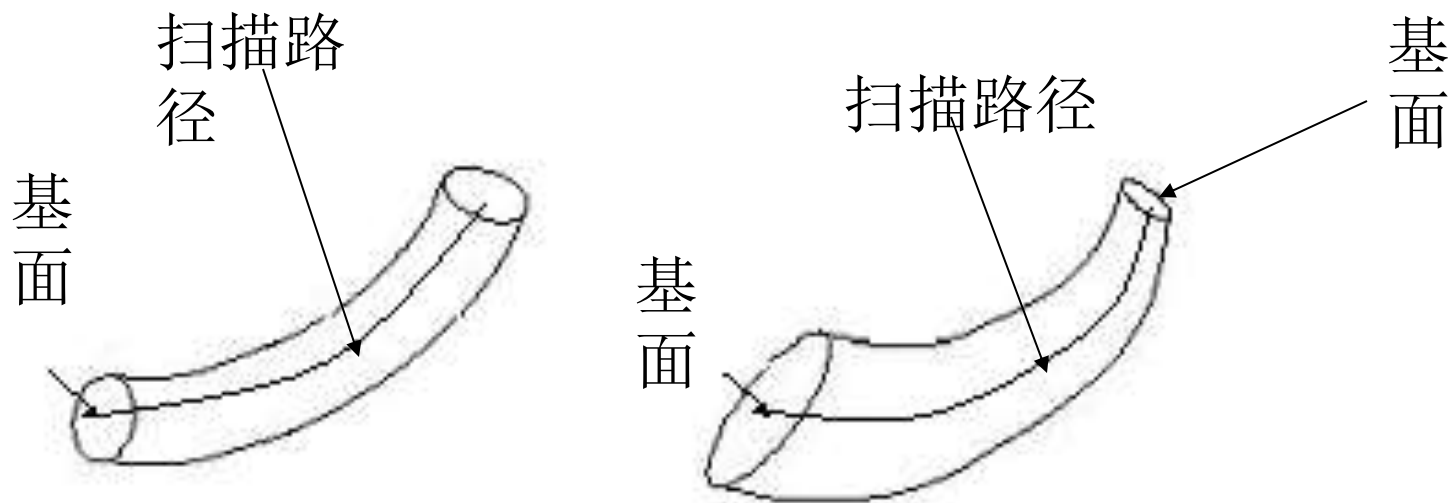
定义： 将一个二维区域绕旋转轴旋转一特定角度（如一周），旋转曲面



广义扫描

定义(1)任意剖面沿着任意轨迹扫描指定的距离，

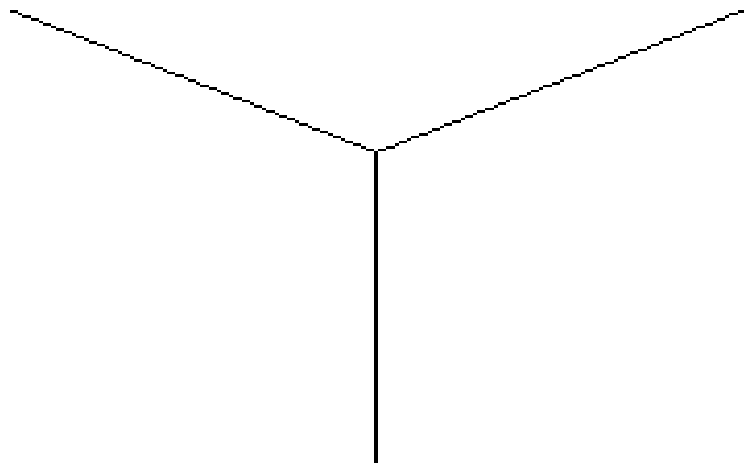
- **(2)**扫描路径可以用曲线函数来描述
- **(3)**可以沿扫描路径变化剖面的形状和大小

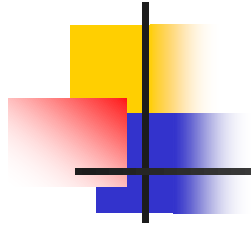


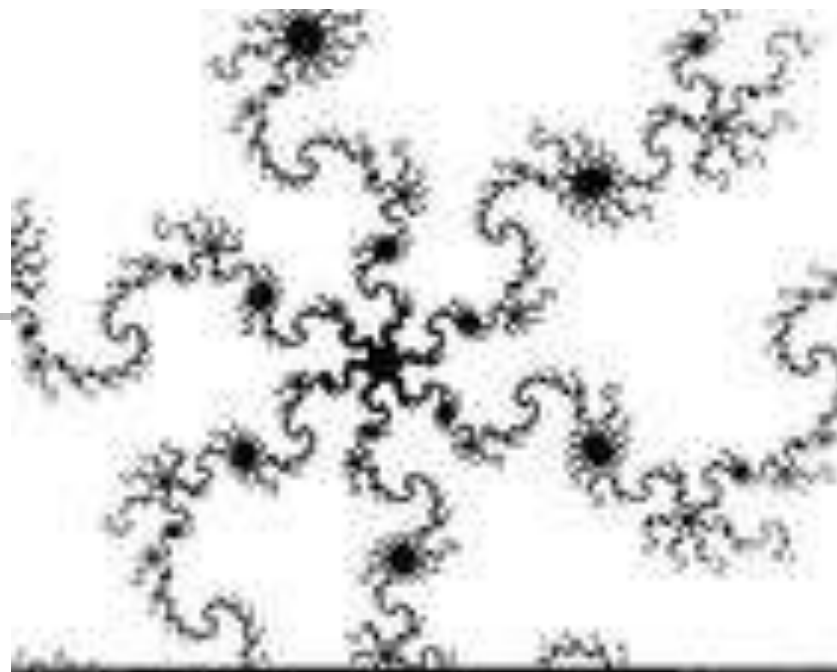
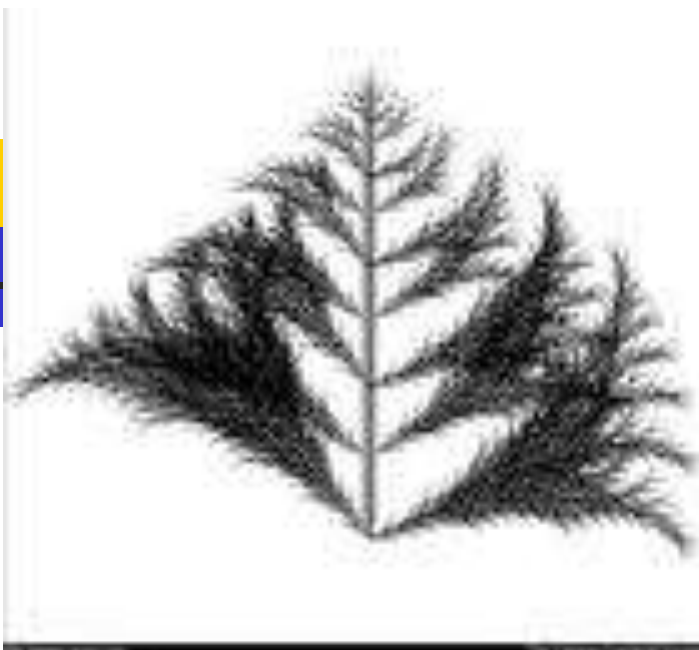
6.1.2 非传统造型技术

■ 1 分形造型

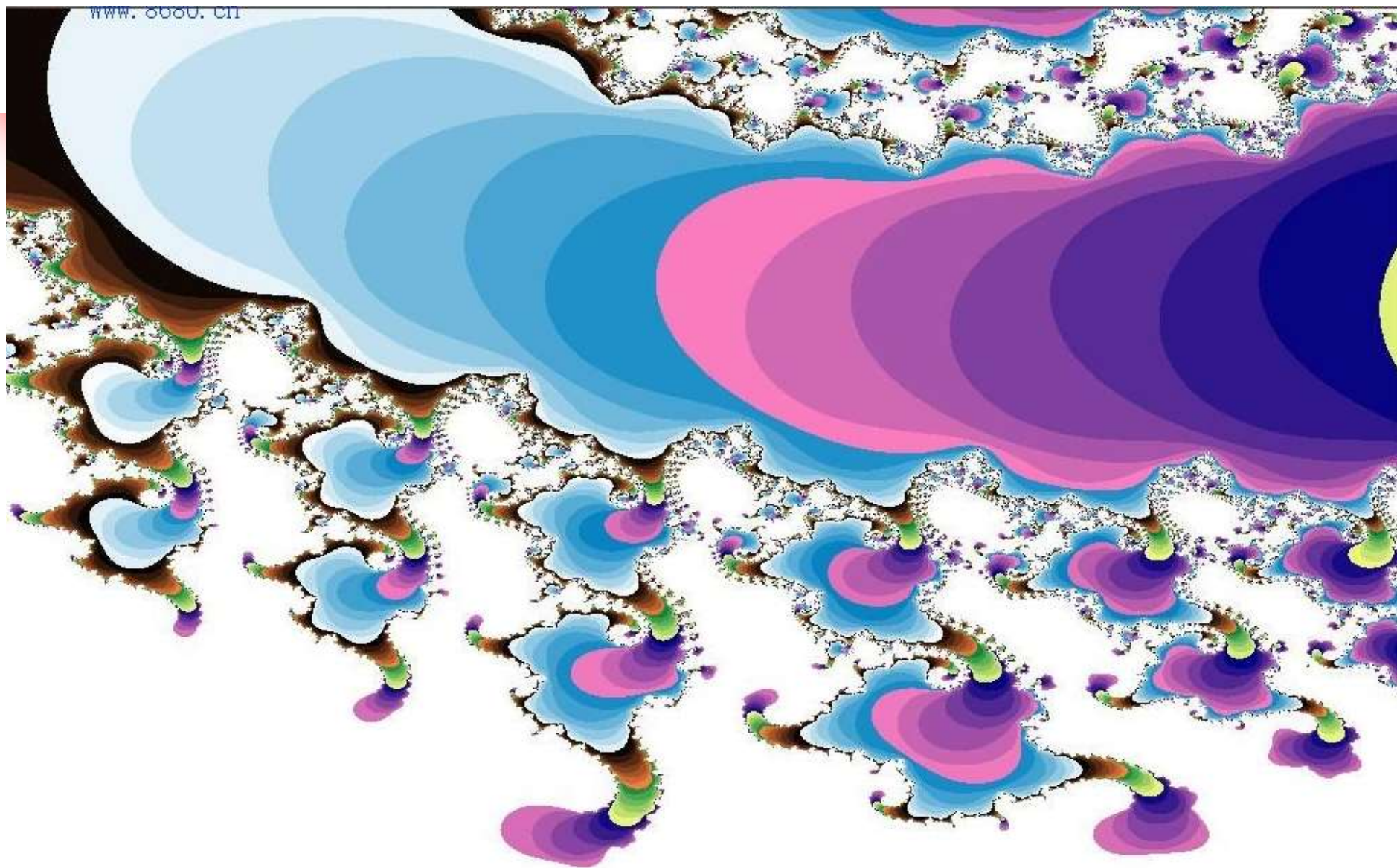
- 分形造型是利用分形几何学的自相似性，采用各种**模拟真实图形**的模型，使生成景象呈现出细节无穷回归性质的方法



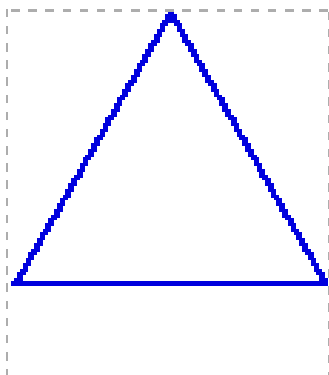
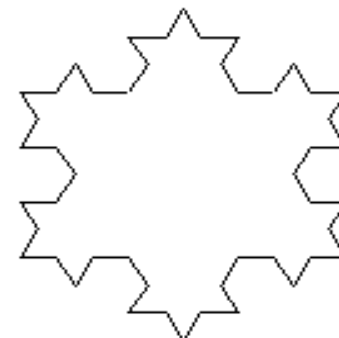
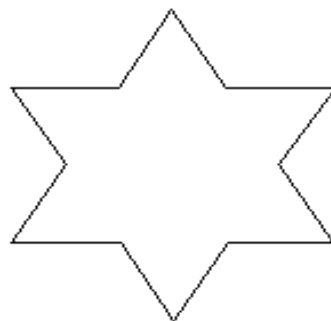
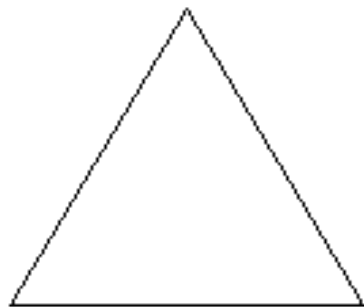




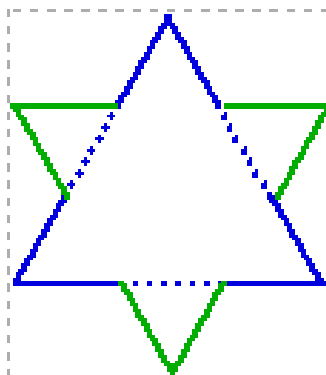




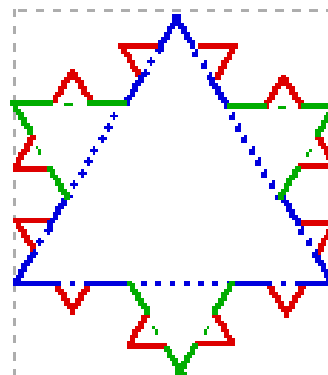
Koch雪花曲线



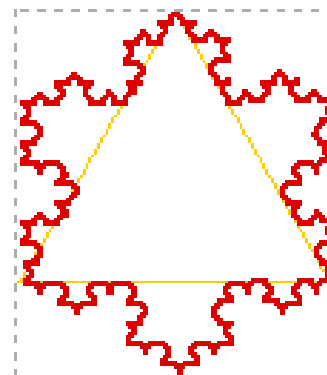
1.



2.



3.



4.