

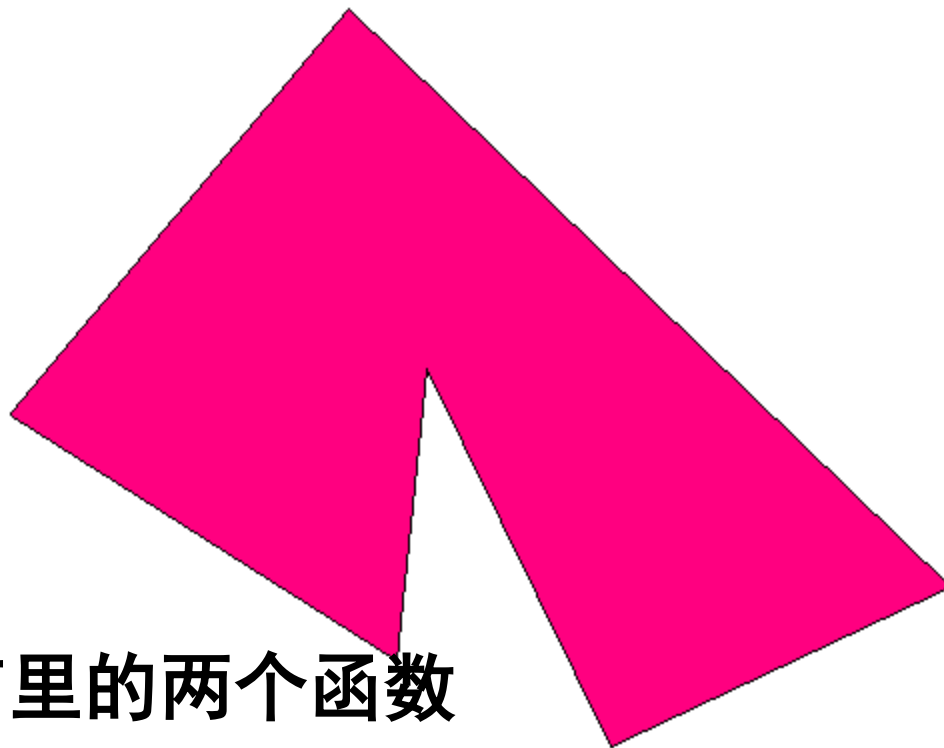


4.3 二维图形的填充

南京农业大学
谢忠红



4.3.1 种子填充算法



C语言里的两个函数

（那两个？完整的形式？）



如何实现图形的
填充呢？

本节内容：

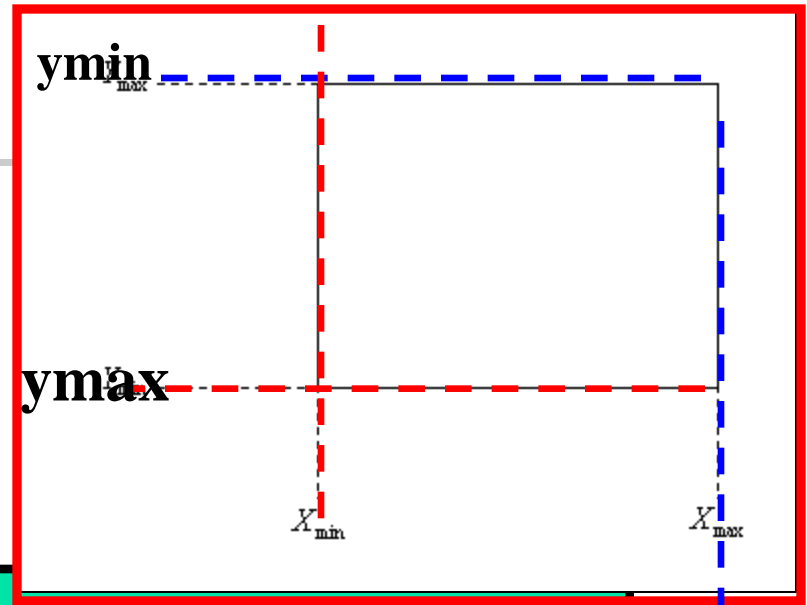
介绍实现图形填充的一些基本算法。



填充要解决
哪两个问题?

- 
- (1) 确定**哪些像素位于**填充图元的**内部**;
 - (2) 确定**以什么颜色**填充这些像素;

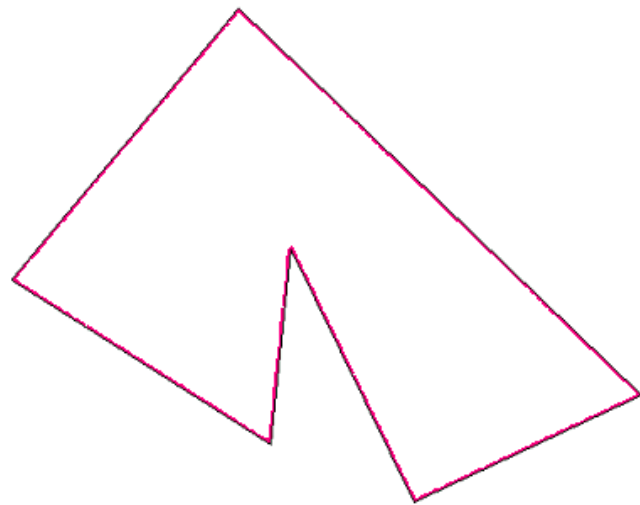
(1) 矩形的填充方法 (规则形状) (**fill_rec2**)



```
void FillRectangle(Rectangle *rect,int color)  
{ int x,y;  
  for(y = rect->ymin;y <= rect->ymax;y++)  
    for(x = rect->xmin;x <= rect->xmax;x++)  
      PutPixel(x,y,color);  
}    /*end of FillRectangle() */
```

(2) 逐点判断法

(不规则形状的多边形)

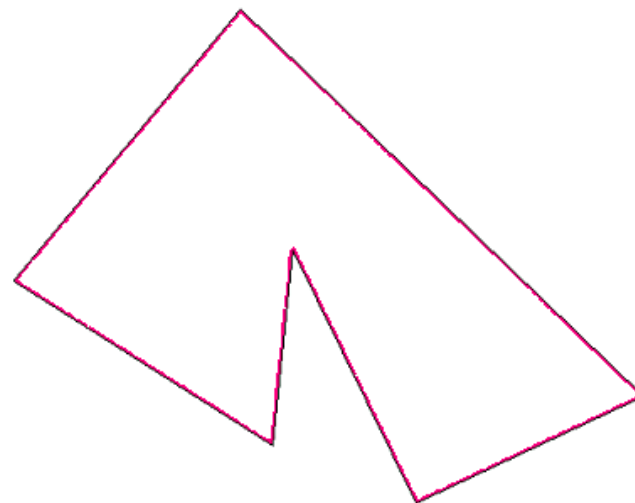


方法： 逐个判断绘图窗口中的像素，如果在多边形内，则给予相应的色彩，如果不在多边形内则赋予背景色彩。

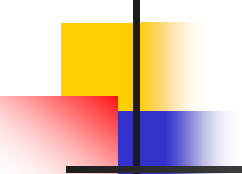
逐点判断法

定义一个多边形的结构体

```
#define MAX 100
typedef struct {
    int PolygonNum;
    Point vertexces[MAX]
} Polygon
```



```
// 多边形顶点个数
// 多边形顶点数组
// 多边形结构
```



```
void FillPolygonPbyP( Polygon *P, int Color)
{ int x,y;
  for(y = ymin;y <= ymax;y++)
    for(x = xmin;x <= xmax;x++)
      if(IsInside(P,x,y))           //在多边形内
        PutPixel(x,y,Color);
      else                         //在多边形外
        PutPixel(x,y,backgroundColor);
}
```




•引起一个问题，

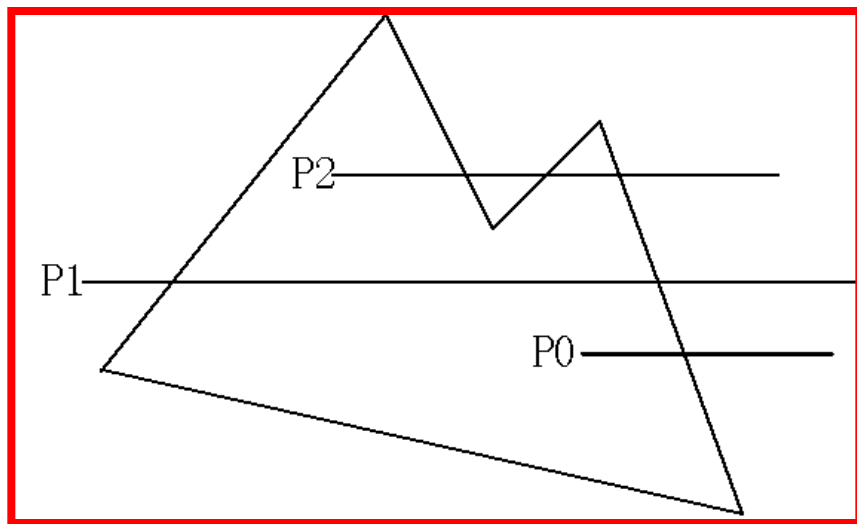
如何判断某个像素
是在多边形内
还是在多边形外呢？



■ 射线法判断像素点是否在多边形内还是外

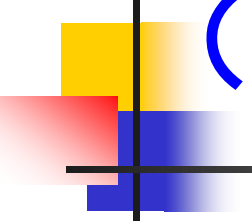
■ 步骤:

1. 从待判别点 v 发出射线
2. 求交点个数 k
3. K 的奇偶性决定了点与多边形的内外关系
(奇数个交点在内部偶数个交点在外面)



逐点判断法的优缺点

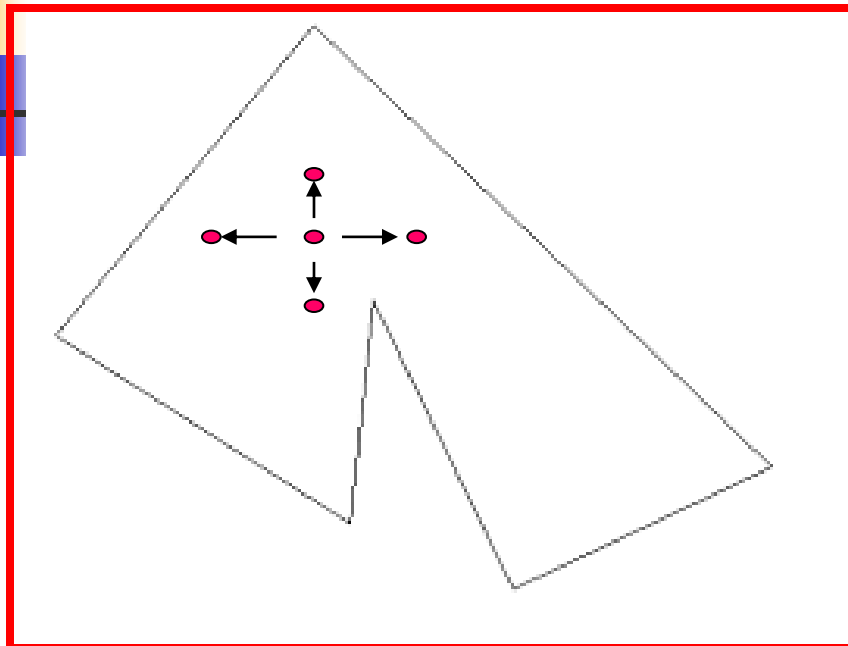
- 优点：程序简单，
- 缺点：速度太慢
- 主要原因：计算量太大，费时
- 由于该算法割断了各像素之间的联系，孤立地考察各像素与多边形的内外关系，使得几十万甚至几百万个像素都要一一判别，每次判别又要多次求交点，需要做大量的乘除运算，花费很多时间。



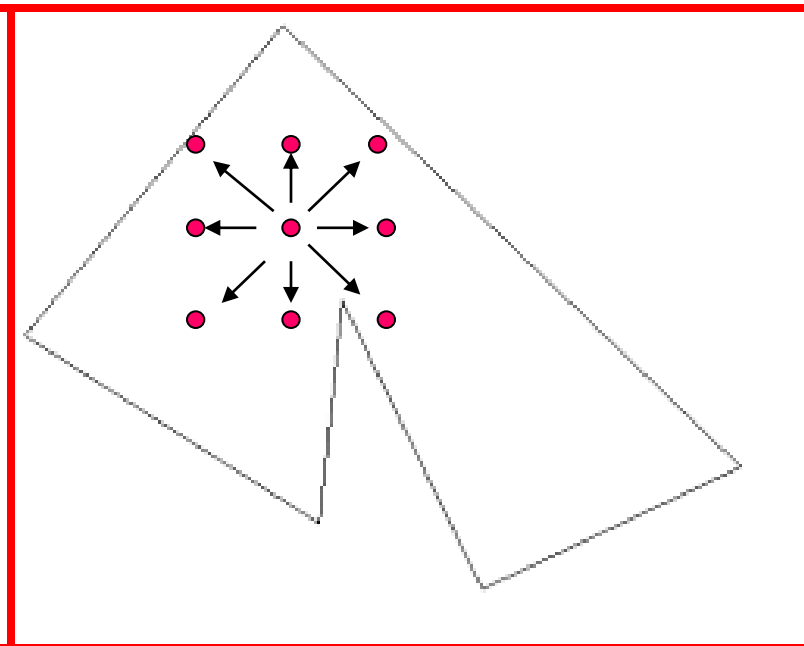
(3) 种子填充法

■ 基本思路:

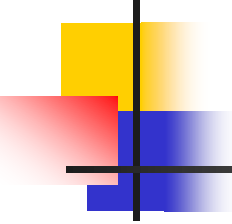
• 首先假设在多边形区域的内部，至少有一个像素（称为种子）是已知的。然后算法开始搜索与种子点相邻且位于区域内的其它像素。如果相邻点不在区域内，那么到达区域的边界；如果相邻点位于区域内，那么这一点就成为新的种子点，然后继续递归地搜索下去。



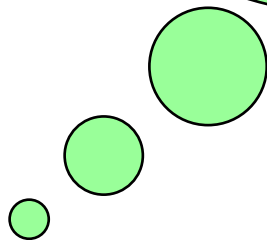
4-连通



8-连通



•使用种子填充方法如何实现圆心为 (OX, OY) 半径为 R 的圆的填充算法??

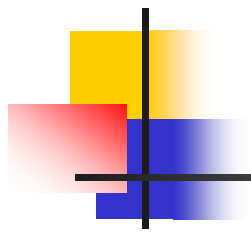


种子填充算法

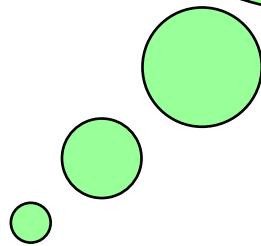
圆心 (OX, OY)

半径R的圆

```
void fill(int x,int y,int newcolor)
{ if( (x-OX)*(x-OX)+(y-OY)*(y-OY)<R*R )
    { if (getpixel(x,y)!=newcolor)
        { putpixel(x,y,newcolor) ;
          //设置(x,y)点为新颜色
          fill(x,y+1,newcolor) ;//上
          fill(x,y-1,newcolor) ;//下
          fill(x+1,y,newcolor);//左
          fill(x-1,y,newcolor);//右
        }
    }
}
```

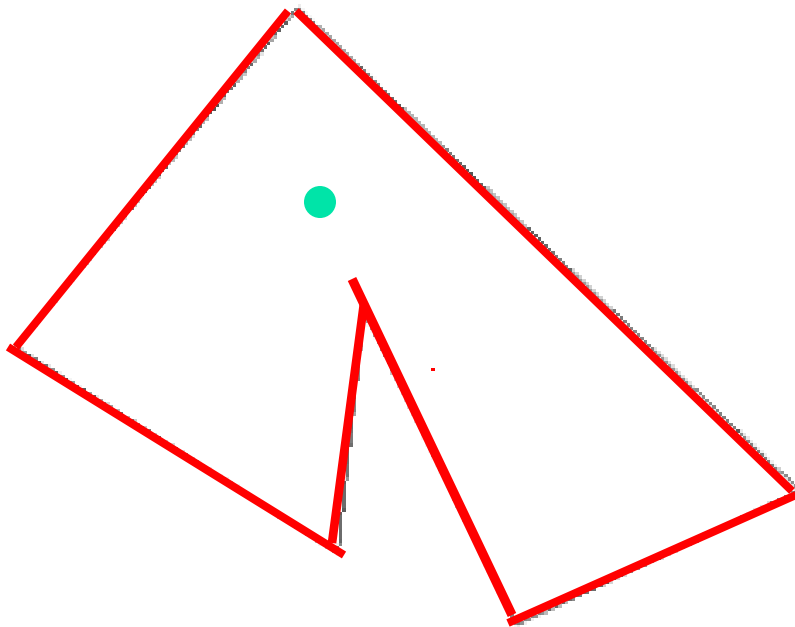


- 思考: 能否用种子填充
- 方法填充任意形状的
- 图形? 算法有何改变?

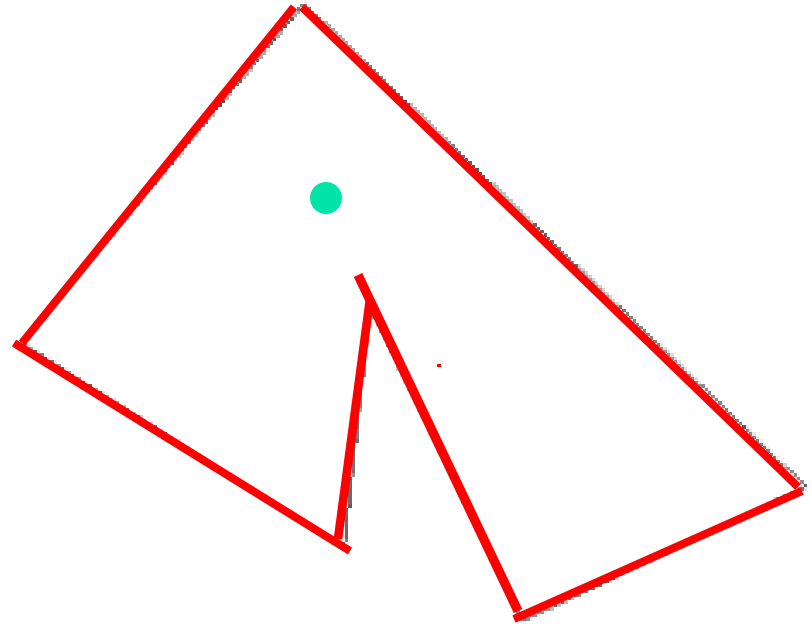


■ 难点分析:

- 形状任意时，我们无法确定边界的位置，



- 解决方法:
- 如果给定边界的颜色, 再给定种子点和填充色能否实现填充呢?



FloodFill(int x, int y , int bordercolor)

边界表示的4连通区域的递归填充算法

```
void MyFloodFill(int x,int y,int borderColor,  
                 int newColor)  
{ int color=getpixel(x,y);  
  if((color!=borderColor)&&(color!=newColor))  
  {PutPixel (x,y,newColor);  
   MyFloodFill(x,y+1,borderColor,newColor);  
   MyFloodFill(x,y-1,borderColor,newColor);  
   MyFloodFill(x-1,y,borderColor,newColor);  
   MyFloodFill(x+1,y,borderColor,newColor);  
  }  
}
```



种子算法的优缺点

优点： 递归执行，算法简单，

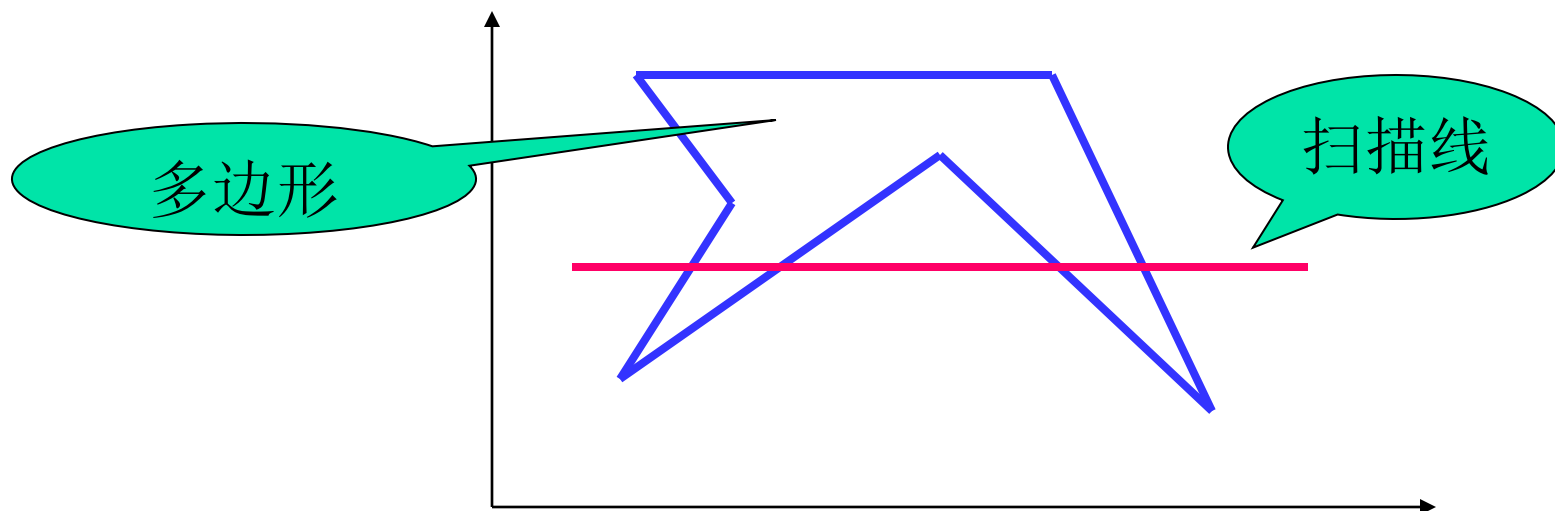
缺点： 效率不高，区域内每一像素都引起一次递归，进/出栈，费时费内存。

改进算法：减少递归次数，提高效率。

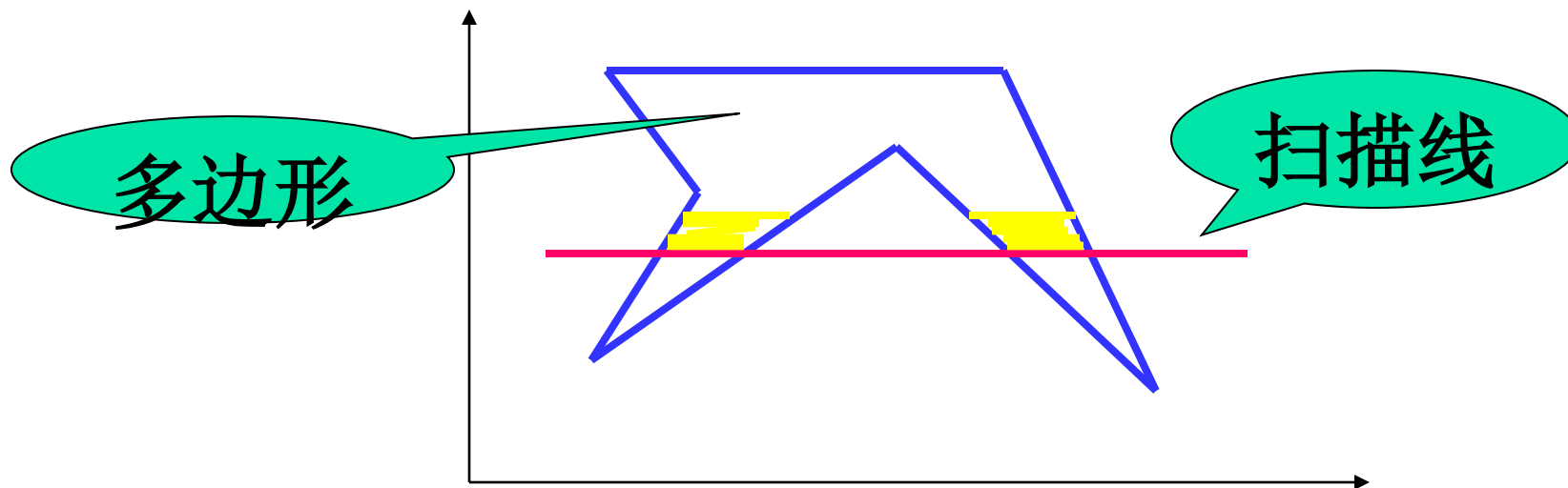
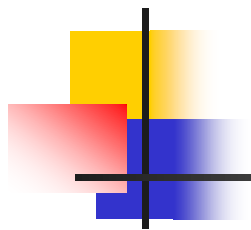
4.3.2 多边形扫描线填充原理

基本原理：

利用扫描线的连贯性，填充每一条扫描线位于多边形内部的区段。



例：



二、算法步骤

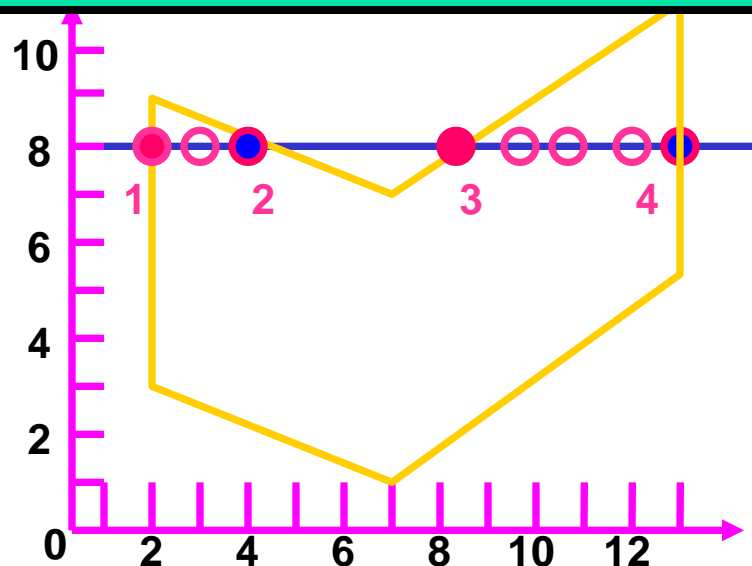
(1) 求出扫描线与多边形所有边的交点

(2) 交点排序(x坐标值以升序排列)

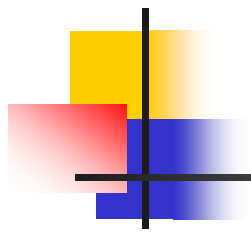
交点: $(2,8)(4,8)(8,8)(13,8)$

(3) 交点配对,

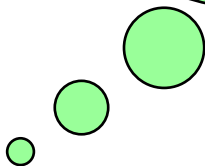
(4) 在配对的交点间填充区段。



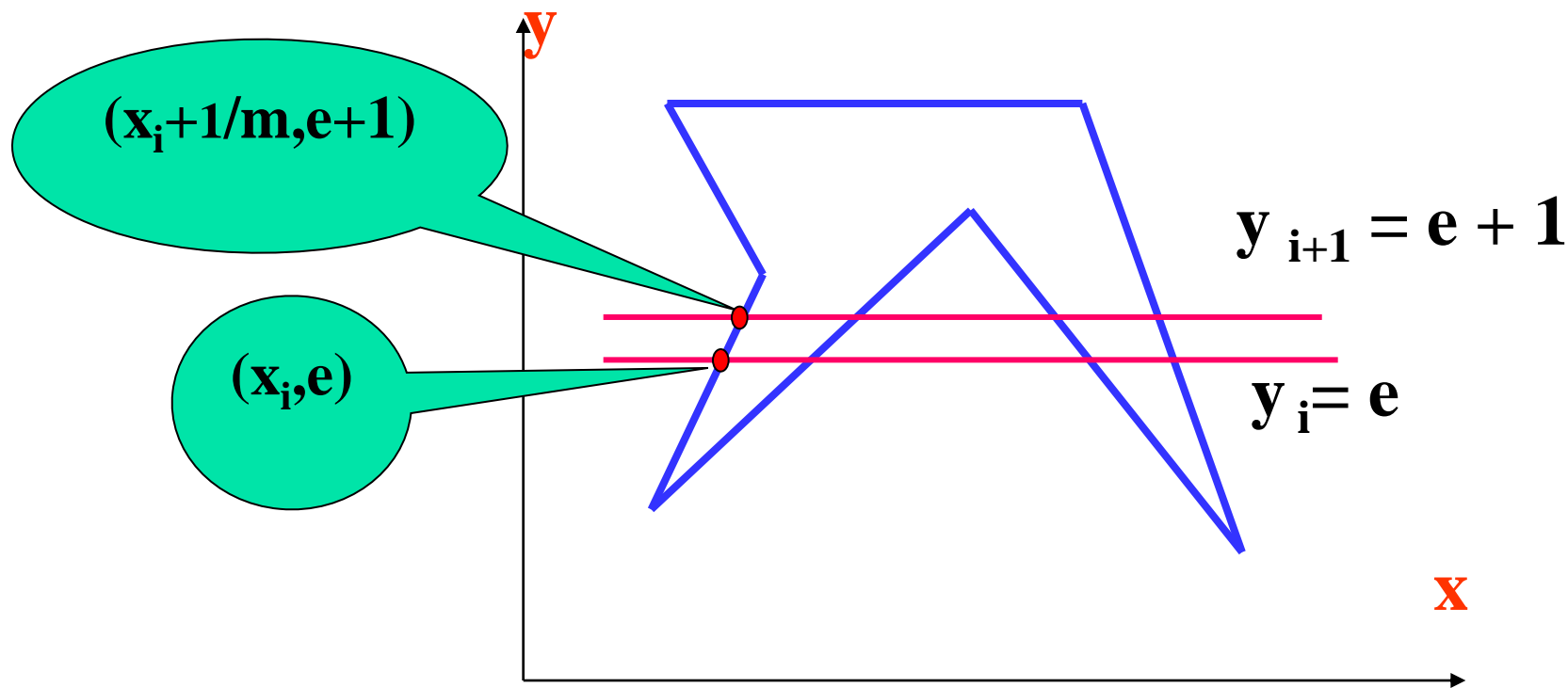
(一条扫描线与
多边形的边有偶
数个交点)



如何求的扫描线
和边的交点??

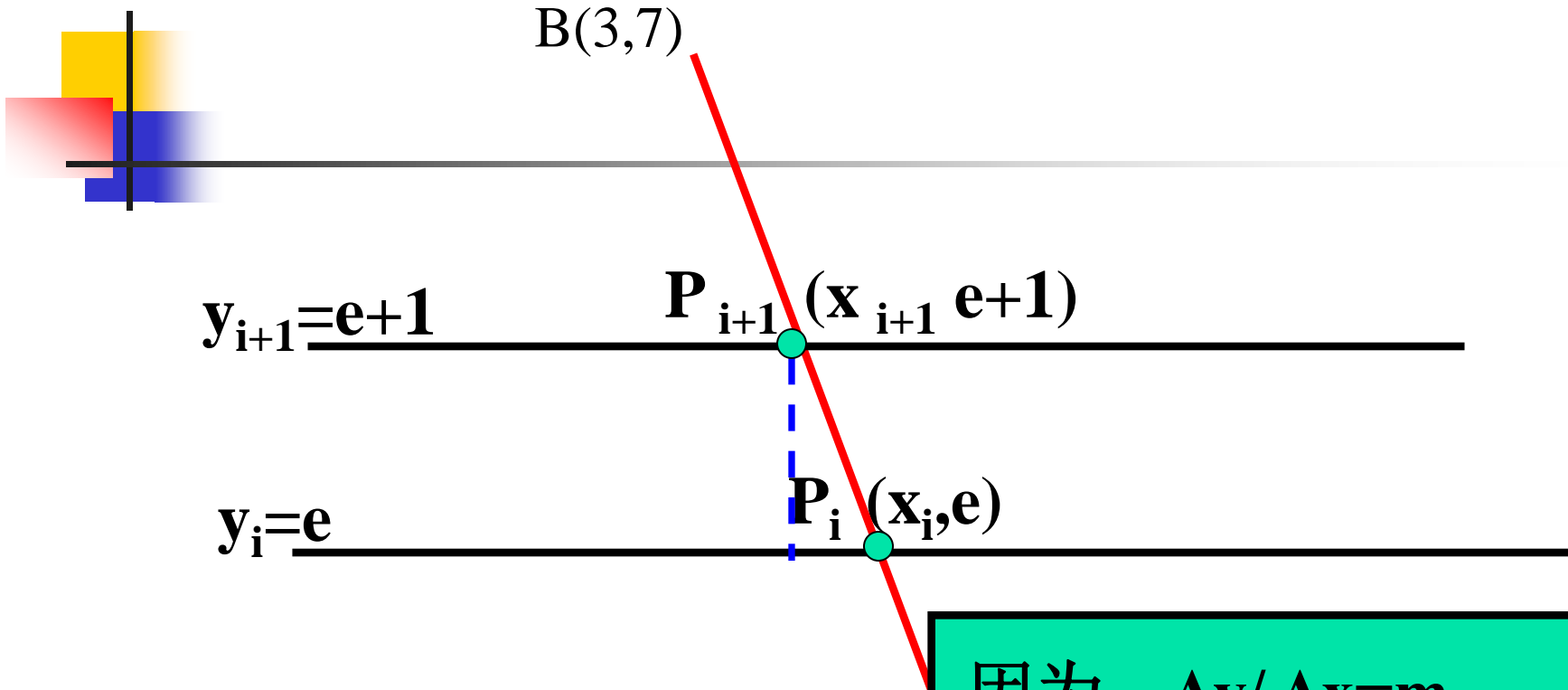


由扫描线 $y_i=e$ 和多边形的所有交点递推出扫描线 $y_{i+1}=d=e+1$ 与多边形各边的交点。



详细说明如下

■ 如何求出扫描线与边的交点呢？



$$y_{i+1} = y_i + 1$$

$$x_{i+1} = x_i + 1/m$$

因为: $\Delta y / \Delta x = m$

所以: $\Delta x = \Delta y * 1/m$

而 $\Delta y = 1$

$$\Delta x = 1/m$$

步骤

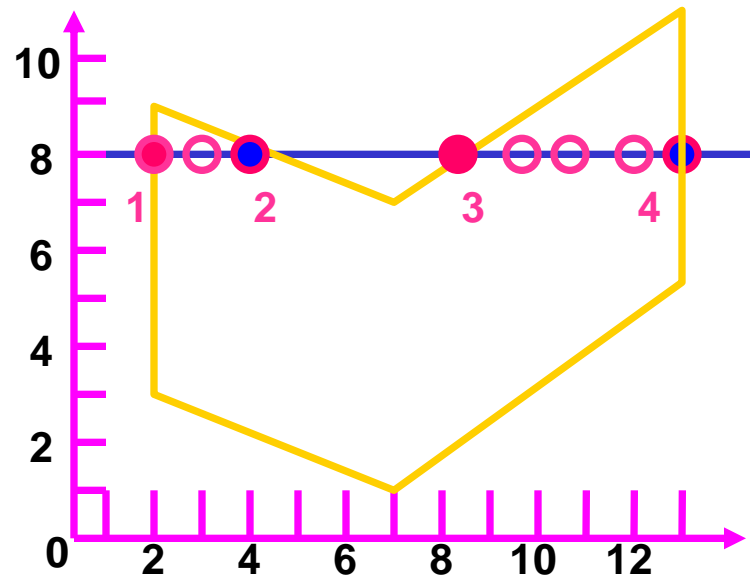
(1) 求出扫描线与多边形所有边的交点

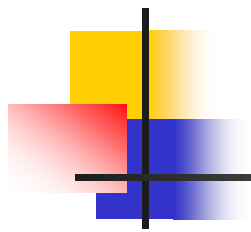
(2) 交点排序(x坐标值以升序排列)

交点: $(2,8)$ $(4,8)$ $(8,8)$ $(13,8)$

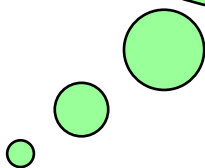
(3) 交点配对, 填充区段。

(4) 根据上条扫描线与多边形的交点求出下一条扫描线的交点后再排序填充, 以此类推





如何使用
计算机实现?



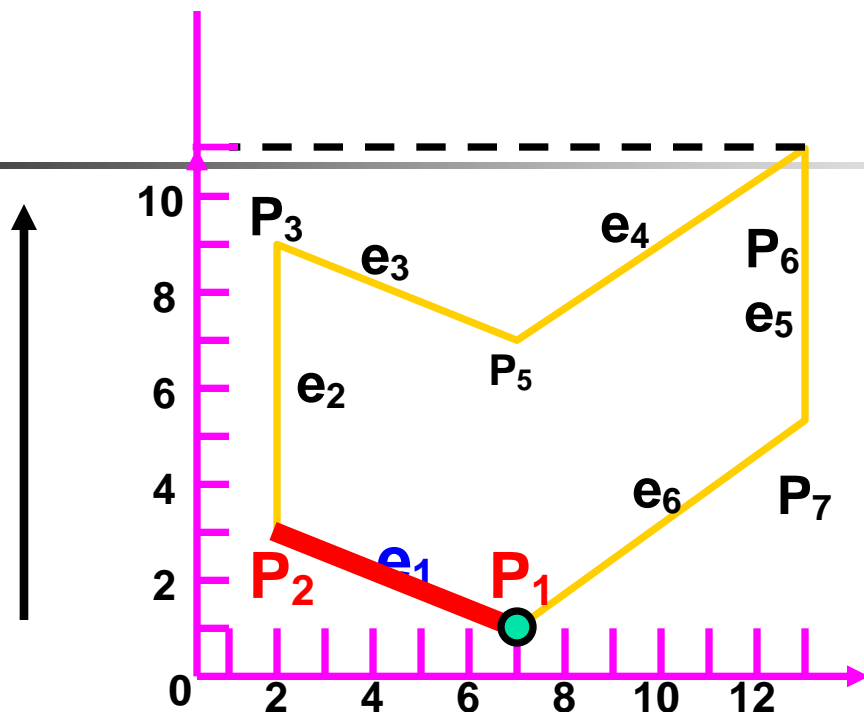


扫描转换多边形的扫描线算法

多边形
的表示

如何表示一条边？

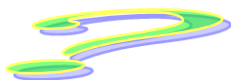
扫描
方向



下端点的x值

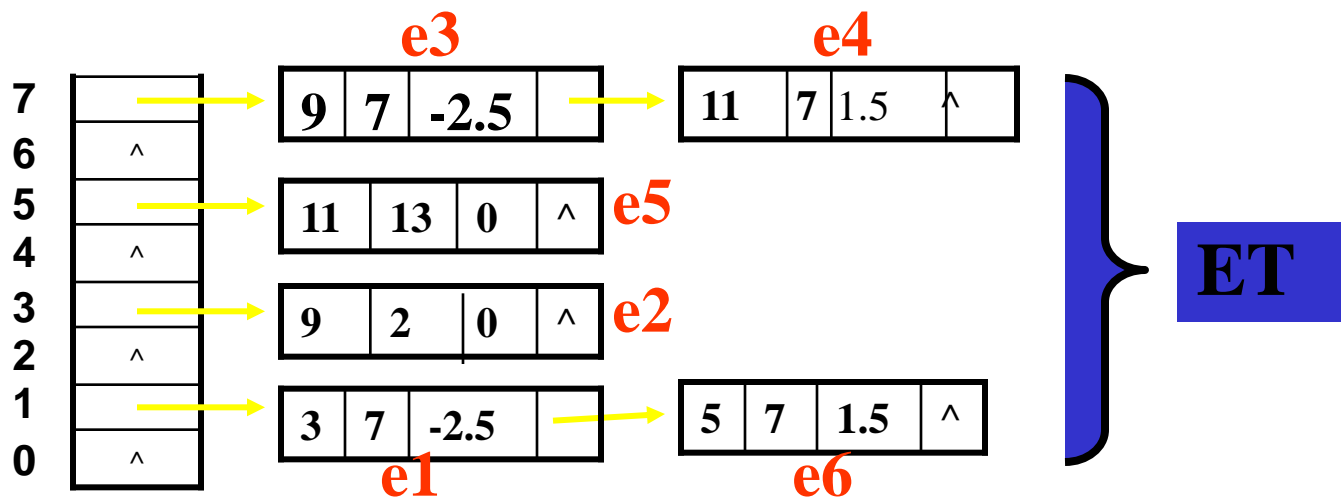
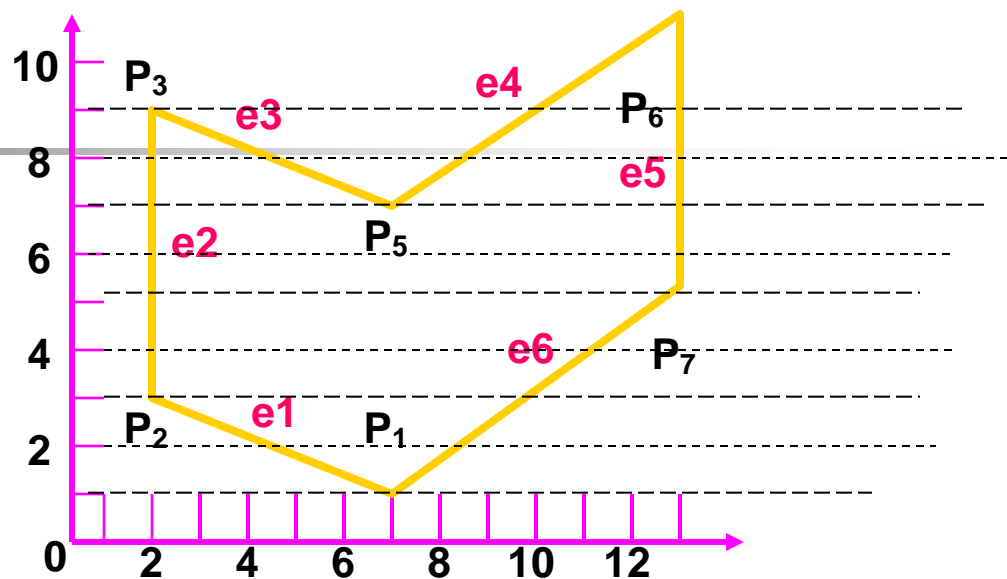
Y_{\max}	x_i	$1/m$
3	7	-2.5

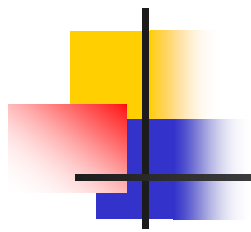
e_1



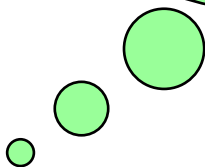
x_i 边最下面点的x的坐标why?

■ 如何把所有的边组织起来表示成一个多边形？





- 用什么数据
- 结构来表示呢？



数据结构-----边的分类表ET

基本元素为边结构，其定义为

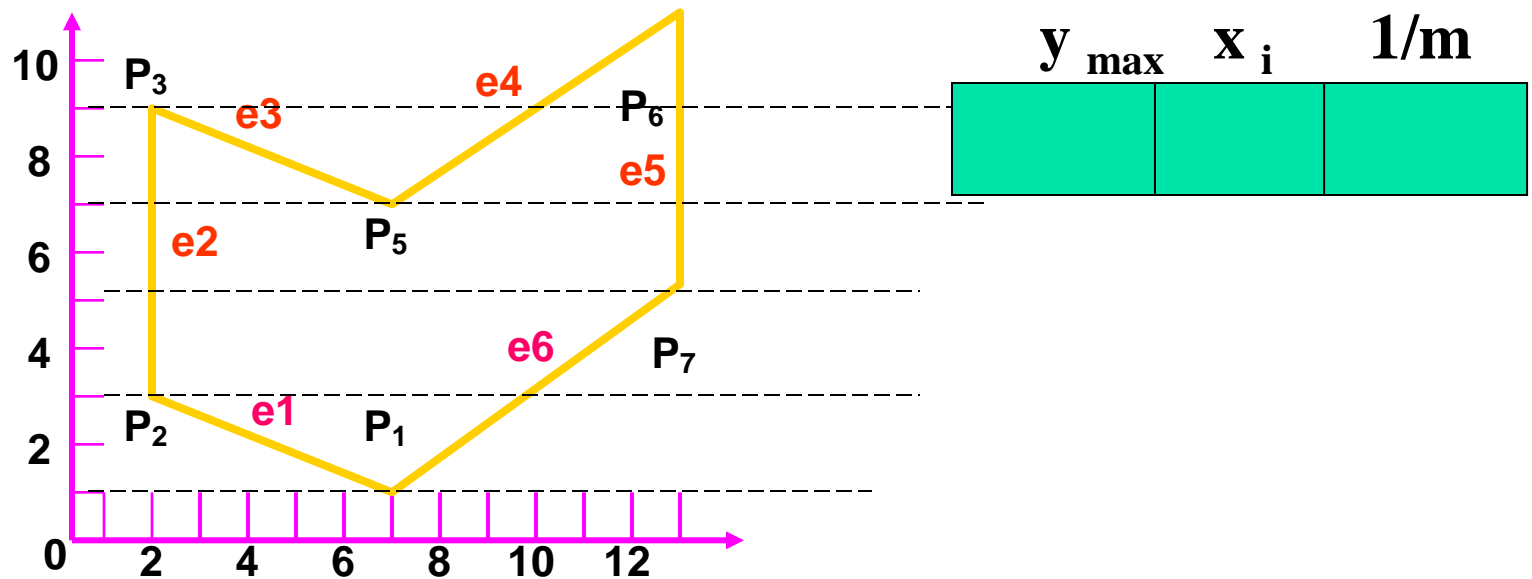
```
typedef struct
{
    int      ymax;
    float     x , deltax;
    struct Edge *nextEdge;
}Edge;
```

其中：

ymax:	边的上端点的y坐标；
x:	在ET中为边的下端点的x坐标，
deltax:	边的斜率的倒数 $1/m$ ；
nextEdge:	指向下一条边的指针。

建立一个**AET**活动边表

作用：表示当前扫描线与哪些边相交,并求出交点，在每对交点间填充指定的颜色。



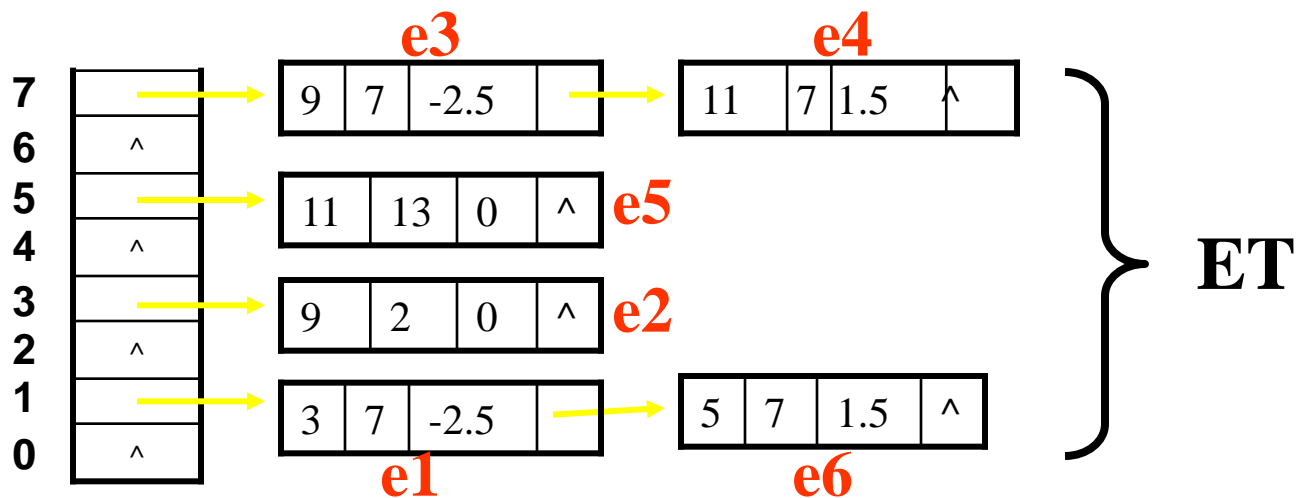
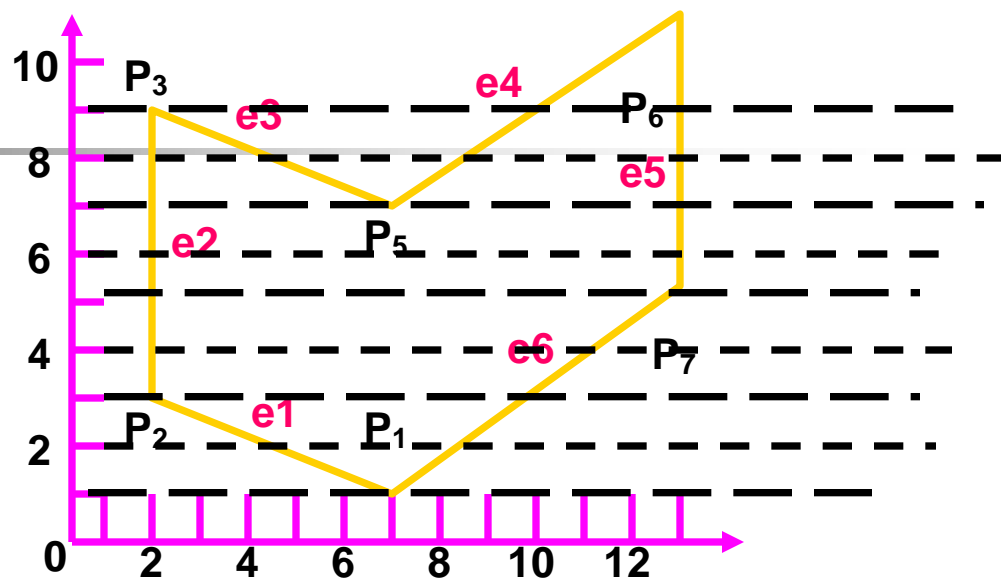
解题

■ 一个多边形的边表

边的表示形式:

y_{\max}	x_i	$1/m$
7	8	-1


AB



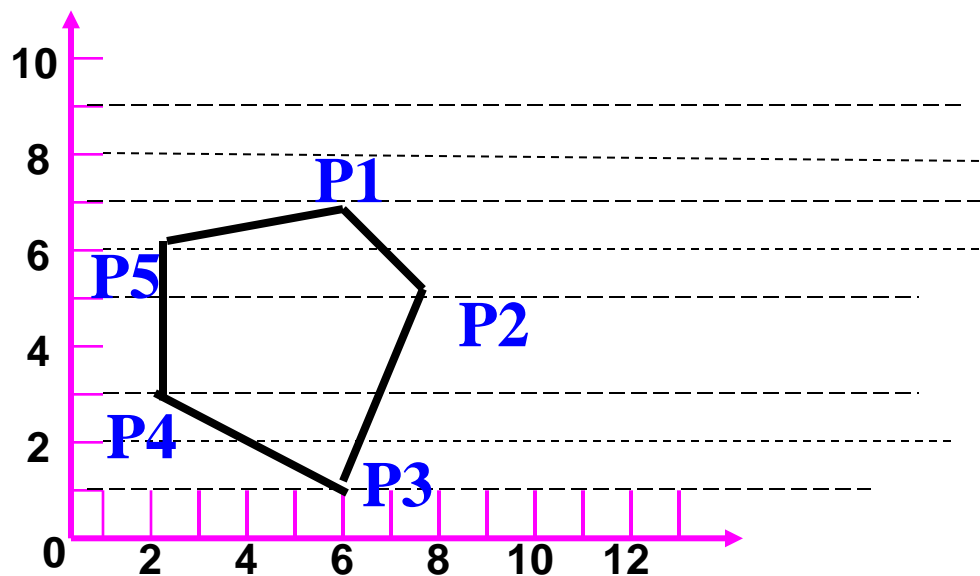


扫描线算法

1. 建立ET;
2. 将扫描线纵坐标 y 的初值置为ET中非空元素的最小序号;
3. 置AEL为空;
4. 执行下列步骤直至ET和AEL都为空:
 - (1) 如果ET中的第 y 行非空, 则将其中的所有边取出并插入AEL中;
 - (2) 如果有新的边插入AEL, 则对AEL中各边按 x 升序排序

- 
- (3) 对AEL中满足 $y=y_{\max}$ 边删去
 - (4) 根据上条扫描线与多边形的交点求出新的交点，并对AEL中的边两两配对，获得有效的填充区段，再填充；
 - (5) 将当前扫描线纵坐标 y 值递增1，即 $y=y+1$ ；
重复以上步骤直至AET表为空。

练习题




解题：请写出边表（ET）和活动边表



扫描线算法

1. 建立ET;
2. 将扫描线纵坐标 y 的初值置为ET中非空元素的最小序号;
3. 置AEL为空;
4. 执行下列步骤直至ET和AEL都为空:
 - (1) 如果ET中的第 y 类非空, 则将其中的所有边取出并插入AEL中;
 - (2) 如果有新的边插入AEL, 则对AEL中各边按 x 升序排序

- 
- (3) 对AEL中满足 $y=y_{\max}$ 边删去
 - (4) 根据上条扫描线与多边形的交点求出新的交点，并对AEL中的边两两配对，获得有效的填充区段，再填充；
 - (5) 将当前扫描线纵坐标 y 值递增1，即 $y=y+1$ ；
重复以上步骤直至AET表为空。

Setfillstyle(int pattern,int color)

FloodFill(int x,int y int bordercolor)

例:

setcolor(RED)

circle (100,100,80);

setfillstyle(SLASHFILL,YELLOW2)

floodfill(100,100,RED)

[返回](#)